Daniel Cremers
Bodo Rosenhahn
Alan L. Yuille
Frank R. Schmidt (Eds.)

LNCS 5604

Statistical and Geometrical Approaches to Visual Motion Analysis

International Dagstuhl Seminar
Dagstuhl Castle, Germany, July 2008
Revised Papers

Springer

Lecture Notes in Computer Science 5604

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Daniel Cremers Bodo Rosenhahn
Alan L. Yuille Frank R. Schmidt (Eds.)

Statistical and Geometrical Approaches to Visual Motion Analysis

International Dagstuhl Seminar
Dagstuhl Castle, Germany, July 13-18, 2008
Revised Papers

Springer

Volume Editors

Daniel Cremers
Universität Bonn, Institut für Informatik III
Römerstraße 164, 53117 Bonn, Germany
E-mail: dcremers@cs.uni-bonn.de

Bodo Rosenhahn
Leibniz Universität Hannover, Institut für Informationsverarbeitung
Appelstraße 9A, 30167 Hannover, Germany
E-mail: rosenhahn@tnt.uni-hannover.de

Alan L. Yuille
University of California, Department of Statistics and Psychology
8967 Math Sciences Building, Los Angeles, CA 90095-1554, USA
E-mail: yuille@stat.ucla.edu

Frank R. Schmidt
Universität Bonn, Institut für Informatik III
Römerstraße 164, 53117 Bonn, Germany
E-mail: schmidtf@cs.uni-bonn.de

Library of Congress Control Number: 2009930038

CR Subject Classification (1998): I.4.8, I.4, I.2.8-10, I.5, I.3.5, F.2.2

LNCS Sublibrary: SL 6 – Image Processing, Computer Vision, Pattern Recognition, and Graphics

ISSN 0302-9743
ISBN-10 3-642-03060-2 Springer Berlin Heidelberg New York
ISBN-13 978-3-642-03060-4 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12708987 06/3180 5 4 3 2 1 0

Preface

Motion analysis is central to both human and machine vision. It involves the interpretation of image data over time and is crucial for a range of motion tasks such as obstacle detection, depth estimation, video analysis, scene interpretation, video compression and other applications. Motion analysis is unsolved because it requires modeling of the complicated relationships between the observed image data and the motion of objects and motion patterns (e.g., falling rain) in the visual scene.

The Dagstuhl Seminar 08291 on *Statistical and Geometrical Approaches to Visual Motion Analysis* was held during July 13–18, 2008 at the International Conference and Research Center (IBFI), Schloss Dagstuhl, near Wadern in Germany. The workshop focused on critical aspects of motion analysis, including motion segmentation, the modeling of motion patterns and the different techniques used. These techniques include variational approaches, level set methods, probabilistic models, graph cut approaches, factorization techniques, and neural networks. All these techniques can be subsumed within statistical and geometrical frameworks. We further involved experts in the study of human and primate vision. Primate visual systems are extremely sophisticated at processing motion, thus there is much to be learnt from studying them. In particular, we discussed how to relate the computational models of primate visual systems to those developed for machine vision.

In total, 15 papers were accepted for these proceedings after the workshop. We were careful to ensure a high standard of quality for the accepted papers. All submissions were double-blind reviewed by at least two experts. The accepted papers reflect the state of the art in the field and cover various topics related to motion analysis. The papers in this volume are classified into four categories based on the topics *optic flow and extensions*, *human motion modeling*, *biological and statistical approaches* and *alternative approaches to motion analysis*.

We would like to thank the team at castle Dagstuhl for the professional support before, after and during the seminar. We are grateful to the participants of the Dagstuhl workshop for their active discussions and commitment during the seminar and for the remarkable efforts and the quality of timely delivered reviews. Apart from all the authors, we would like to also thank Christoph Garbe, Timo Kohlberger and Thomas Schoenemann for providing additional reviews.

The organization of this event would not have been possible without the effort and the enthusiasm of several people, and we thank all who contributed.

April 2009

Daniel Cremers
Bodo Rosenhahn
Alan Yuille
Frank R. Schmidt

Table of Contents

Optical Flow and Extensions

Human Motion Modeling

Biological and Statistical Approaches

Alternative Approaches to Motion Analysis

Discrete-Continuous Optimization for Optical Flow Estimation

Stefan Roth[1], Victor Lempitsky[2], and Carsten Rother[2]

[1] TU Darmstadt, Department of Computer Science, Darmstadt, Germany
[2] Microsoft Research Cambridge, UK

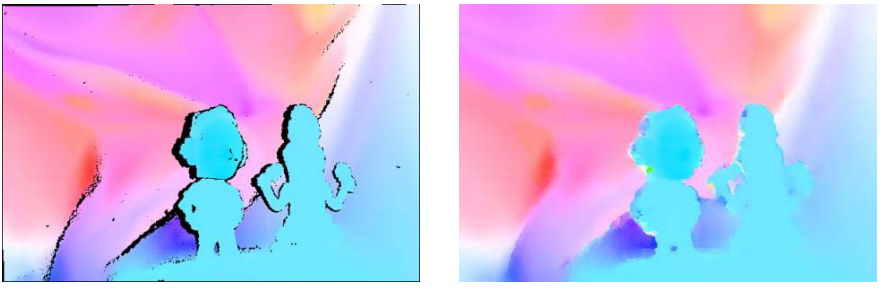
Abstract. The accurate estimation of optical flow is a challenging task, which is often posed as an energy minimization problem. Most top-performing methods approach this using continuous optimization algorithms. In many cases, the employed models are assumed to be convex to ensure tractability of the optimization problem. This is in contrast to the related problem of narrow-baseline stereo matching, where the top-performing methods employ powerful discrete optimization algorithms such as graph cuts and message-passing to optimize highly non-convex energies.

In this chapter, we demonstrate how similar non-convex energies can be formulated and optimized in the context of optical flow estimation using a combination of discrete and continuous techniques. Starting with a set of candidate solutions that are produced by either fast continuous flow estimation algorithms or sparse feature matching, the proposed method iteratively fuses these candidate solutions by the computation of minimum cuts on graphs. The obtained continuous-valued result is then further improved using local gradient descent. Experimentally, we demonstrate that the proposed energy is an accurate model and that the proposed discrete-continuous optimization scheme not only finds lower energy solutions than traditional discrete or continuous optimization techniques, but also leads to very accurate flow estimates.

1 Introduction

To this date, optical flow estimation remains a challenging problem, despite much research having focused on this problem and the significant progress made since the early works [1,2]. Two key challenges dominate recent research: First, the issue of choosing an appropriate and accurate computational model, and second, computing a good solution given a particular model. Most flow estimation methods, both historic and current, formulate optical flow estimation as an energy minimization problem. Many authors, including recently Papenberg *et al.* [3] suggested complex continuous optimization schemes for flow estimation. Despite their success, this and many other approaches are limited by the fact that the spatial regularity of flow is modeled as a convex function. While this assumption aids optimization, the estimated flow fields are somewhat smooth and lack very sharp discontinuities that exist in the true flow field, especially at motion

D. Cremers et al. (Eds.): Visual Motion Analysis, LNCS 5604, pp. 1–22, 2009.



(a) Ground truth (b) Estimated flow

Fig. 1. With a new, more accurate MRF energy and a powerful discrete-continuous optimization approach, the presented method is capable of accurate flow recovery from challenging image sequences such as "Mequon" from [6] (*hue = direction, saturation = magnitude, black in GT = "unknown"*)

boundaries. Black and Anandan [4] addressed this problem by introducing non-convex penalty functions based on robust statistics that are tolerant towards such outliers. While this formulation allowed for realistic discontinuities, the corresponding non-convex energy was hard to minimize. Despite the use of annealing, the optimization could often get trapped in poor local optima. Roth and Black [5] studied the spatial statistics of optical flow and found the derivative statistics to be very heavy-tailed. This validated the assumptions made by [4] and implied that the convex energies used by the majority of today's flow approaches only provide an approximation to the statistics of the flow. They also suggested a Markov random field (MRF) model motivated by these statistics, but due to the corresponding non-convex energies, inference has remained very challenging, and the flow fields estimated using a continuous optimization approach still suffer from smoothed discontinuities.

Maybe somewhat surprisingly, for narrow-baseline stereo matching these difficulties have been addressed quite a while ago, and the need for non-convex energy functions has not only been recognized but, in contrast to optical flow, also been widely addressed. Their use has been facilitated by modern discrete optimization algorithms, such as graph cuts [7] or message passing [8], which are often able to obtain nearly-global optima of such energies [9]. Most top-performing stereo techniques rely on discrete optimization for minimizing non-convex energies. Even though disparity estimation has a lot in common with optical flow estimation and can, in fact, be regarded as a particular constrained case of it, surprisingly little knowledge has been transferred to the optical flow problem. Only a few authors have attempted to use discrete optimization for optical flow [7,10,11,12,13], however, and only very recently state-of-the-art performance has been achieved [14,15].[1]

This work presents a new approach to the estimation of optical flow that combines powerful discrete with more traditional continuous optimization

[1] This chapter is an extended version of [14].

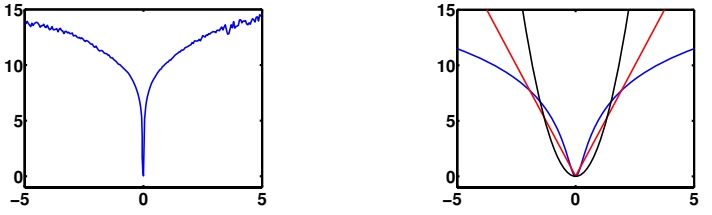


Fig. 2. *(Left)* Negative log-density of the x-derivative of the horizontal flow (from [5]). *(Right)* Charbonnier potentials (red, *c.f.* [16]) are not a good fit of these statistics, but better than quadratics (black). Here, we instead use the negative-log of a Student-t distribution (Lorentzian) (blue, *c.f.* [4]).

techniques. Similar to [4] and motivated by the statistics of optical flow [5], we employ a heavy-tailed spatial term as well as a robust data term, which permit modeling sharp discontinuities and occlusions in the flow. Similar to stereo methods, we rely on discrete optimization to minimize the resulting highly non-convex energy, but our optimization is not purely discrete. Instead, it relies on a set of continuous-valued proposal flow fields that are combined together using discrete optimization. Following [12], the optimal combination (*fusion*) of each pair of solutions with respect to the considered energy is computed from the minimum cut on a special graph [17,18,19]. Afterwards, the fused solution is locally improved using gradient descent. The suggested optimization scheme relies on discrete algorithms in order to avoid poor local minima of the energy, but operates with continuously-valued proposal solutions. This avoids the pitfalls of purely discrete approaches, namely the huge number of labels as well as discretization artifacts.

We follow two different approaches for computing proposal flow fields. The first relies on fast and simple flow algorithms such as the original Lucas-Kanade [2] and Horn-Schunck [1] methods. These proposals are standard spatially varying flow fields. The second approach relies on spatially constant proposals similar to the popular α-expansion algorithm [7]. To find these proposals, we determine dominant displacements by performing sparse feature matching between the frame pair based on SIFT features [20].

We first motivate our non-convex energy in Sec. 2, and then introduce our discrete-continuous optimization approach in Sec. 3. It allows for more efficient and robust optimization compared to hierarchical coarse-to-fine continuous frameworks, which are very often used for flow estimation. In Sec. 4 we show that our non-convex energy combined with our optimization procedure leads to high-quality flow results, superior to otherwise equivalent convex formulations. The algorithm developed here is the one of the top-ranking approaches on the Middlebury optical flow benchmark [6].

Compared to [14], we further investigate the relative performance of our framework to a more traditional α-expansion-like approach. Unlike [14], the α values are not chosen through uniform discretization of a 2D domain of admissible

motion vectors, but are computed via SIFT-matching (*c.f.* [21]). Such a non-uniform and image-adaptive choice of α values allows for a fairer comparison to α-expansion approaches. Moreover, we also give a number of additional details about the developed energy and algorithm, including performance measurements.

1.1 Background and Related Work

In the majority of approaches, optical flow estimation has been posed as a problem of energy minimization, or essentially equivalently as a problem of maximum a-posteriori estimation in a probabilistic model. The use of energy minimization dates back to the work of Horn and Schunck [1], where given a pair of images $I(x, y, t)$ and $I(x, y, t+1)$ the optical flow field (u, v) is computed by minimizing an energy of the form

$$E(u,v) = \int_{\Omega} D\left(I(x+u(x,y), y+v(x,y), t+1) - I(x,y,t)\right) + S\left(||\nabla u(x,y)||^2 + ||\nabla v(x,y)||^2\right) dx\, dy\,. \quad (1)$$

Here Ω denotes the image region, and (x, y) denotes the image coordinate. The first term, the so-called *data term*, embodies the *brightness constancy assumption*, which states that corresponding pixels in the two frames should have similar brightness. The second term, also called *spatial term*, imposes spatial regularity on the resulting optical flow field by penalizing spatially varying flow. This is necessary, because flow estimation is severely underconstrained and suffers, for example, from the aperture problem. Historically, [1] proposed using quadratic penalties for both data and spatial terms, and further linearized the brightness constancy assumption ($I_x u + I_y v + I_t = 0$, where $I_{x/y/t}$ is the derivative in $x/y/t$-direction). This makes the energy convex, and after spatial discretization quite easy to optimize.

Nevertheless, this simple formulation has a variety of problems. The linearized brightness constancy assumption allows estimating only very small motions, which has prompted the use of hierarchical coarse-to-fine estimation schemes, in which the solution from a coarser level is used to warp the frames toward each other. Interestingly, these can be interpreted as numerical schemes for approximately minimizing energies with non-linearized, thus non-convex, brightness constancy terms [3]. Another problem relates to the fact that a quadratic spatial term leads to overly smooth flow estimates lacking any discontinuities, as they naturally occur at motion boundaries. Even though robust non-convex penalty functions have been proposed [4] and are motivated by the flow statistics (see Fig. 2), they were hard to optimize with traditional continuous optimization approaches. Many subsequent methods thus often used slightly robust, but still convex penalty functions (*e.g.* [3,16]), which still lead to smoothed discontinuities. Another drawback is that they are missing a direct connection to the statistics of the data [5].

It thus still seems surprising that robust, non-convex penalties have rarely been used in conjunction with variational techniques, especially since data terms

are typically non-convex [3] no matter what the penalty function may be. Our results suggest that, given an otherwise unchanged regularization framework, non-convex penalties can lead to substantially better results as long as the optimization problem can be handled. In drawing comparisons to other techniques, it is important to note that convex formulations can still produce very accurate results. One has to keep in mind, however, that methods such as [3,16] use rather different regularizers (gradient magnitude-based ones) and hence are not directly comparable to the approach developed here. Nonetheless, we posit that such variational techniques would also benefit from non-convex penalties.

Some approaches recover more accurate motion discontinuities using explicit discontinuity modeling either using line processes [4] or explicit segmentation of the flow fields [22,23]. In both cases, the estimation of discontinuities is alternated with flow estimation, which also makes these approaches vulnerable to getting trapped in local minima.

A number of authors have formulated the problem of flow estimation using a Markov random field, in which case flow is usually computed using maximum a-posteriori (MAP) estimation [4,24,25]. Inference was performed using a number of approximative techniques, such as stochastic or deterministic relaxation, all of which are vulnerable to getting trapped in relatively poor local optima.

Recently, a number of attempts have been made to adapt discrete optimization methods for optical flow computation [7,10,11,12,13]. All of them, however, suffered from the problem of *label discretization*. While in stereo it is relatively easy to discretize the disparities, this is not the case for optical flow, where at each pixel a two-dimensional flow vector has to be described using discrete labels.

Fusion moves as are used here allow to combine continuous-valued proposal solutions and have shown to be helpful in a variety of contexts. For example, it has been shown [14,26] that they can be used to make graph-cut based optimization substantially more efficient. Moreover, dense disparity estimation using second-order priors has been shown to benefit from fusing spatially varying proposal solutions [27]. Finally, fusion moves have also been used in the context of spatially continuous variational formulations [28]. These methods, including the approach developed here, may also be related to layered-based correspondence methods such as [29]. These methods assume that the scene decomposes into a small set of layers with few parameters (such an assumption, however, rarely holds for real scenes). The energy, which is dependent both on the non-local layer parameters as well as on the pixel assignments to layers, is minimized by alternating discrete optimization updating pixel assignments and continuous optimization updating layer parameters. Despite the use of discrete algorithms, such optimization still often gets stuck in poor local minima.

2 Energy Formulation

Following a number of optical flow approaches (*e.g.* [4,24,25]) as well as a large body of work in stereo, we model the problem of optical flow estimation using pairwise Markov random fields. Flow estimation is performed by doing maximum

a-posteriori (MAP) inference. As we will see, this provides a good trade-off between the accuracy of the model as well as tractability of the inference, *i.e.*, optimization. The posterior probability of the flow field $\mathbf{f}$ given two images, I_0 and I_1, from a sequence is written as

$$p(\mathbf{f}|I^0, I^1) \;=\; \frac{1}{Z} \prod_{\mathbf{p}\in\Omega} \exp(-D_{\mathbf{p}}(f_{\mathbf{p}};\; I^0, I^1)) \cdot \prod_{(\mathbf{p},\mathbf{q})\in\mathcal{N}} \exp(-S_{\mathbf{p},\mathbf{q}}(f_{\mathbf{p}}, f_{\mathbf{q}}))\;, \quad (2)$$

where $f_{\mathbf{p}} = (u_{\mathbf{p}}, v_{\mathbf{p}})$ denotes the flow vector at pixel $\mathbf{p}$, the set $\mathcal{N}$ contains all pairs of adjacent pixels, and Z is a normalization constant. Before specifying the model in more detail, we obtain an equivalent energy function by taking the negative logarithm of the posterior and omitting constants:

$$E(\mathbf{f}) = \sum_{\mathbf{p}\in\Omega} D_{\mathbf{p}}(f_{\mathbf{p}};\; I^0, I^1) + \sum_{(\mathbf{p},\mathbf{q})\in\mathcal{N}} S_{\mathbf{p},\mathbf{q}}(f_{\mathbf{p}}, f_{\mathbf{q}})\;. \quad (3)$$

This can be viewed as a spatially discrete variant of Eq. (1).

Data term. The first term of the energy, the so-called data term, measures how well the flow field $\mathbf{f}$ describes the image observations. In particular, it models how well the corresponding pixels of I^0 and I^1 match. Traditionally, it is modeled based on the brightness (or color) constancy assumption, *e.g.*, $D_{\mathbf{p}}(f_{\mathbf{p}};\; I^0, I^1) = \rho_d(||I^1(\mathbf{p}+f_{\mathbf{p}}) - I^0(\mathbf{p})||)$. Note that we do not employ a linearized constraint as our optimization method does not require this. In our experimentation we found, however, that such simple color matching is heavily affected by illumination and exposure changes, in particular by shadows. To make the data term robust to these effects, we remove lower spatial frequencies from consideration:

$$H^i = I^i - G_\sigma * I^i, \qquad i \in 0, 1\;, \quad (4)$$

where G_σ is a Gaussian kernel with standard deviation σ. This strategy is somewhat similar to filtering the input images using gradient filters [3,30], which also removes low-frequency structure. Based on these filtered images we define

$$D_{\mathbf{p}}(f_{\mathbf{p}};\; I^0, I^1) = \rho_d(||H^1(\mathbf{p}+f_{\mathbf{p}}) - H^0(\mathbf{p})||)\;. \quad (5)$$

Here $||\cdot||$ denotes the Euclidean distance of the RGB color values. $H^1(\mathbf{p}+f_{\mathbf{p}})$ is computed using bicubic interpolation.

As suggested by the probabilistic interpretation from Eq. (2), the penalty function $\rho(\cdot)$ should model the negative log-probability of the color distance of pixels related through natural flow fields. The statistics of brightness constancy for optical flow have only been studied quite recently [30] and have reinforced the intuition that the presence of effects such as occlusions and specular reflections suggests that a robust treatment using heavy-tailed distributions is necessary to avoid penalizing large color changes unduly (cf. also [4]). We here use the Geman-McClure robust penalty function:

$$\rho_d(x) = \frac{x^2}{x^2+\mu^2}\;, \quad (6)$$

which has a similar shape as the truncated-quadratic penalty $f(x) = \min(ax^2, 1)$, which has been very successfully used in the stereo literature. The advantage of the Geman-McClure penalty is that it is differentiable everywhere. Note that more advanced data terms, such as learned ones that model the constancy of responses to a bank of linear filters [30] could be used here as well.

Spatial term. As is usual in a pairwise MRF formulation of flow, the spatial term penalizes changes in horizontal and vertical flow between adjacent pixels:

$$S_{\mathbf{p},\mathbf{q}} = \rho_{\mathbf{p},\mathbf{q}}\left(\frac{u_{\mathbf{p}} - u_{\mathbf{q}}}{||\mathbf{p} - \mathbf{q}||}\right) + \rho_{\mathbf{p},\mathbf{q}}\left(\frac{v_{\mathbf{p}} - v_{\mathbf{q}}}{||\mathbf{p} - \mathbf{q}||}\right) , \tag{7}$$

where $||\mathbf{p} - \mathbf{q}||$ is the Euclidean distance between the pixel centers of $\mathbf{p}$ and $\mathbf{q}$. As the flow differences between adjacent pixels closely approximate the spatial derivatives of the flow, we use the spatial statistics of optical flow [5] to motivate suitable penalty functions. Roth and Black [5] showed that the statistics of flow derivatives are very heavy-tailed and strongly resemble Student t-distributions. We therefore choose the penalty to be the (scaled) negative log of a Student-t distribution (see also Fig. 2):

$$\rho_{\mathbf{p},\mathbf{q}}(x) = \lambda_{\mathbf{p},\mathbf{q}} \log\left(1 + \tfrac{1}{2\nu^2}x^2\right) . \tag{8}$$

Motivated by the success of stereo approaches, we assume that the flow field discontinuities tend to coincide with image color discontinuities. Hence we make the smoothness weight $\lambda_{\mathbf{p},\mathbf{q}}$ spatially-dependent and set it to a lower value if the pixel values $I^0(\mathbf{p})$ and $I^0(\mathbf{q})$ are similar. We do not learn the parameters of the energy here (see Sec. 4 for details). Nonetheless, it is possible in principle to learn the parameters of our model either using maximum-likelihood [30] or by minimizing the flow error directly on the training set [31], but we leave this for future work.

While being of high fidelity, as we will see shortly, the MRF energy used here is more difficult to optimize than the energies used in recent popular optical flow methods such as [3,16]. While the non-linearized color constancy assumption already makes the objective non-convex [3], the penalty functions of data and spatial term in our formulation are robust, thus non-convex as well. Finally, the data term works with the high frequency content of images, which only adds to its non-linearity. Therefore, as we demonstrate in the experimental section, traditional continuous optimization schemes based on coarse-to-fine estimation and gradient descent often end up in poor local minima. Also, the developed energy is harder to optimize than many energies used in stereo matching, since the value at each pixel spans a potentially unbounded 2D rather than a bounded 1D domain, making it infeasible for purely discrete techniques to sample it densely enough. This suggests the use of a new, more powerful optimization scheme that combines the merits of discrete and continuous-valued approaches.

3 Energy Minimization

3.1 Graph Cut Methods for Energy Minimization

Over the last years, graph cut based methods have proven to be invaluable for the minimization of pairwise MRF energies in the case of discrete labels (*e.g.*, $x_{\mathbf{p}} \in \{0, 1, \dots N-1\}$), which take the form:

$$E(\mathbf{x}) = \sum_{\mathbf{p} \in \Omega} D_{\mathbf{p}}(x_{\mathbf{p}}) + \sum_{\mathbf{p},\mathbf{q} \in \mathcal{N}} S_{\mathbf{p},\mathbf{q}}(x_{\mathbf{p}}, x_{\mathbf{q}})\,. \tag{9}$$

These methods rely on the fact that in the case of MRFs with binary labels (N=2) finding the global minimum can be reduced to computing the minimal cut on a certain network graph [18,32]. The existing algorithms based on the maxflow/mincut duality find the minimal cut (and hence the global minimum of the MRF energy) very efficiently [33]. The graph construction proposed in [32,34], however, worked only for MRFs with a certain *submodularity* condition imposed on the pairwise terms. In the cases when these submodularity conditions are not met, a *partial* global optimum may still be computed via minimum cut on an extended graph [17,18,19]. Here, partiality implies that the label cannot be determined from the minimal cut for some of the nodes. However, the remaining (labeled) nodes are assigned the same label as they have in the global optimum. The number of nodes with unknown labels depends on the structure of the problem: connectivity, number of submodularity constraints violated, *etc.* [35].

Various ways have been suggested to extend graph cut based methods to MRFs with multiple labels (N>2) [7,12,36]. In particular, recently [12] suggested the *fusion move* approach to be used in this context. The fusion move considers two given N-valued labelings $\mathbf{x}^0$, $\mathbf{x}^1$ and introduces an auxiliary binary-valued labeling $\mathbf{y}$. The set of all possible auxiliary labelings naturally corresponds to the set of *fusions* of $\mathbf{x}^0$ and $\mathbf{x}^1$ (*i.e.*, labelings of the original MRF, where each node $\mathbf{p}$ receives either label $x^0_{\mathbf{p}}$ or label $x^1_{\mathbf{p}}$):

$$\mathbf{x}^f(\mathbf{y}) = (\mathbf{1} - \mathbf{y}) \cdot \mathbf{x}^0 + \mathbf{y} \cdot \mathbf{x}^1\ , \tag{10}$$

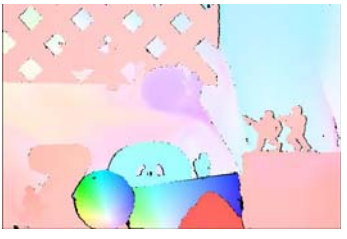
where the product is taken element-wise. This induces a binary-labeled MRF over the auxiliary variables with the energy defined as:

$$E^f(\mathbf{y}) = E(\mathbf{x}^f(\mathbf{y})) = \sum_{\mathbf{p} \in \Omega} d_{\mathbf{p}}(y_{\mathbf{p}}) + \sum_{\mathbf{p},\mathbf{q} \in \mathcal{N}} s_{\mathbf{p},\mathbf{q}}(y_{\mathbf{p}}, y_{\mathbf{q}}), \tag{11}$$
$$\text{where} \quad d_{\mathbf{p}}(i) = D_{\mathbf{p}}(x^i_{\mathbf{p}}), \quad s_{\mathbf{p},\mathbf{q}}(i,j) = S_{\mathbf{p},\mathbf{q}}(x^i_{\mathbf{p}}, x^j_{\mathbf{q}}).$$

The minimum of this binary-valued MRF can be computed via minimum cut on the extended graph. It corresponds to the fusion of $\mathbf{x}^0$ and $\mathbf{x}^1$ that is optimal with respect to the original energy from Eq. (9). Consequently, the energy of this optimal fusion will not be higher (and in most cases lower) than the energy of both $\mathbf{x}^0$ and $\mathbf{x}^1$. This crucial property of the fusion move algorithm can

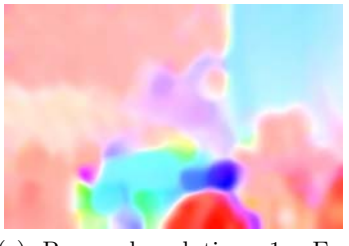
(a) First frame

(b) Second frame

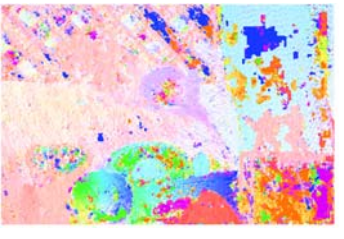


(c) Ground truth

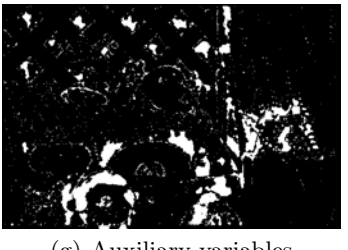
(d) Final solution, Energy=2041



(e) Proposal solution 1, Energy=7264



(f) Proposal solution 2, Energy=44859



(g) Auxiliary variables

(h) Fused solution, Energy=6022

Fig. 3. Results for the *Army* sequence from [6] (shown using the color encoding from [6]). The bottom two rows show **the first step of our discrete optimization** (where *"Lucas-Kanade meets Horn-Schunck"*). Here, a randomly chosen initial solution (e) (computed with Horn-Schunck) is fused with another randomly chosen proposal solution (f) (computed with Lucas-Kanade). The graph cut allows to compute the optimal fused solution (h) with much lower energy, which is passed on to the next iteration. The optimal auxiliary variables (g) show which regions are taken from Solution 1/e (black) and from Solution 2/f (white). In this example 99.998% of the nodes were labeled by the minimum cut on the extended graph.

be enforced even when the obtained optimal auxiliary labeling is not complete (in this case all the unknown labels are taken from the original solution that has a lower energy).

The fusion move generalizes the α-expansion move and the $\alpha\beta$-swap move proposed in [7]. Thus, the popular α-expansion algorithm may be regarded as subsequent fusions of the current labeling with different constant labelings. We will explore this special case later. The first technique (called FusionFlow) that we will describe in the following relies on discrete-continuous optimization that also uses fusion moves, here to combine proposal solutions. In particular, it relies on the fact that fusion moves can be applied even in the case of MRFs with continuous-valued labels such as in Eq. (3), which avoids discretization artifacts.

3.2 Discrete-Continuous Energy Minimization

Proposal solutions. The discrete part of our algorithm proceeds by first computing many different continuous-valued flow fields that serve as *proposal* solutions, and then iteratively fusing them with the current solution using graph cuts. After each fusion, different parts of the proposal solution are copied to the current solution so that the energy goes down (or stays equal). The success of the method thus depends on the availability of good proposal solutions.

It is important to note that the proposal solutions need not to be good across the whole image in order to be "useful". Instead, each solution may contribute to a particular region in the final solution, if it contains a reasonable flow estimate for that region, no matter how poor it is in other regions. This suggests the use of different flow computation methods with different strengths and weaknesses for computing the proposals. In our experiments, we used three kinds of the proposal solutions. Firstly, we used solutions obtained using the Lucas-Kanade (LK) method [2]. Due to the properties of the method, such solutions often contain good results for textured regions but are virtually useless in textureless regions. Secondly, we used solutions obtained using the Horn-Schunck (HS) method [1]. Such solutions often contain good results for regions with smooth motion, but motion discontinuities are always severely oversmoothed. Finally, we also used constant flow fields as proposals specifically for areas where neither the LK solutions nor the HS solutions contain useful flow estimates.

To obtain a rich set of proposal solutions, we use the LK and HS methods with various parameter settings. For HS we vary the strength of the regularization ($\lambda \in \{1, 3, 100\}$). Since both methods should be applied within a coarse-to-fine warping framework to overcome the limitations of the linearized data term (of the proposals, not of our energy), we also vary the number of levels in the coarse-to-fine hierarchy ($l \in \{1, \ldots, 5\}$). Finally, for all LK solutions and a few HS solutions we produce shifted copies (by shifting them $\pm 2^{l-1}$ and $\pm 2^{l}$ pixels in each direction). For the LK method, this corresponds to the use of a family of non-centralized windows and, hence, gives better chances of providing correct flow values near flow discontinuities, and as we found reduces the energy of the solution. These variations result in about 200 proposals (most of them, however, are shifted copies and

do not take much time to compute). 64 constant flow fields are also added to the set of proposals. The choice of the constants is discussed below.

It is important to note that other (potentially more efficient) approaches for obtaining proposal solutions may also be considered. We also experimented with proposal solutions by performing gradient descent from different starting points. We found this to lead to good results (given a sufficient number of minima), but did not pursue it in our final experiments due to its computational inefficiency, as Lukas-Kanade and Horn-Schunck algorithms permitted much faster computation of the proposal solution than more complex algorithms such as [16].

Discrete optimization. As described above, the proposal solutions are fused by computing the minimal cut on the extended graph. The process starts with the LK and HS proposal fields only. One of these proposal fields is randomly chosen as an initial solution. After that, the remaining LK and HS proposals are visited in random order, and each of them is fused with the current solution (as described in Sec. 3.1). An example of such a fusion (during the first iteration of the process) is shown in Fig. 3.

After all LK and HS solutions are visited, the motion vectors of the obtained fused solution are clustered into 64 clusters using the k-means algorithm. The centers of the clusters $\mathbf{c}_i \in \mathbb{R}^2$ are used to produce additional constant proposal flow fields $f_i(\mathbf{p}) \equiv \mathbf{c}_i$. Note that more sophisticated proposals dependent on the current solution may be suggested and our constant solutions are just one step in this direction.

The created constant proposals are added to pool of proposals and the fusion process continues until each proposal is visited twice more. At this point the procedure typically converges, *i.e.*, the obtained fused solution can no longer be changed by fusion with any of the proposals.

We should note that the number of unlabeled nodes during each fusion was always negligible (we never observed it exceeding 0.1% of the nodes). Each fusion is guaranteed not to increase the energy, and in practice the resulting solution always has an energy that is much smaller than the energy of the best proposal. We also observed that the order of fusions affected the result of the optimization only very insignificantly. Taking the *Army* sequence as an example and using the experimental settings discussed below, the energy of the final fused solution was 2435, while the lowest energy among all proposals was 6598, thus is significantly higher. Depending on the random order in which the proposals were visited, the variations of the obtained final energy were within 1% of each other.

Continuous optimization. After fusing flow fields using discrete optimization, we perform a continuous optimization step that helps "cleaning up" areas where the proposal solutions were not diverse enough, which for example may happen in relatively smooth areas. In order to perform continuous optimization, we analytically compute the gradient of the same energy we use in the discrete step, $\nabla_{\mathbf{f}} E(\mathbf{f})$, and use a standard conjugate gradient method [37] to perform local optimization starting from the final solution of the fusion process. The gradient of the spatial term is quite easily derived; computing the gradient of the data

term relies on the fact that $H^1(\mathbf{p} + f_\mathbf{p})$ is computed using bicubic interpolation, which allows computing the partial derivatives w.r.t. $u_\mathbf{p}$ and $v_\mathbf{p}$. We should note that the gradient bears resemblance to the discretization of the Euler-Lagrange equations for the objective used in [3].

Since the discrete optimization step avoids many of the poor local optima that are problematic for purely continuous optimization methods, the combination of discrete and continuous optimization leads to local minima with a substantially lower energy in most of our experiments.

4 Evaluation

To evaluate the presented approach and in particular to investigate the efficiency of the proposed energy minimization scheme in obtaining low-energy states of Eq. (3), we performed a number of experiments using the recent Middlebury optical flow benchmark dataset [6]. Since our method is applicable to color images and based on 2 frames, we use the 2-frame color versions of the datasets, and left the extension to multi-frame sequences for future work. In all experiments, we use an 8-neighborhood system for the spatial term and the following parameters, which were chosen using the training portion of the benchmark to give good performance on challenging real-world scenes: We use $\sigma = 1.5$ for the high-pass filter, $\mu = 16$, and $\nu = 0.2$ for the MRF potentials.

Inspired by many stereo algorithms (e.g. [7]), the regularization parameter $\lambda_{\mathbf{p},\mathbf{q}}$ is spatially varying in our approach (*i.e.*, depends on $\mathbf{p}$ and $\mathbf{q}$), which amounts to spatially-adaptive regularization. In that our goal is to encourage discontinuities in the optical flow to be aligned with discontinuities in the input images. Towards this end, we set

$$\lambda_{\mathbf{p},\mathbf{q}} = \begin{cases} 0.024, & \|I(\mathbf{p},t) - I(\mathbf{q},t)\| < 30 \\ 0.008, & \text{otherwise.} \end{cases} \tag{12}$$

This leads to stronger local regularization, if the absolute differences between the adjacent pixels $I(\mathbf{p},t)$ and $I(\mathbf{q},t)$ is small (*i.e.*, in smooth areas), and less regularization near discontinuities. Note that these parameters, like all of our parameters, are constant across all test sequences.

We evaluated the resulting method on 8 benchmarking sequences and found that it performed very competitively with respect to other methods in the benchmark, particularly on challenging real world scenes as shown in Fig. 1, Fig. 3, and Fig. 4. At the moment of finalization of this chapter, the developed method was ranked 4th in terms of the average angular error (AAE) among the methods already published, and 3rd in terms of the average end-point error out of 20 published methods in the benchmark. The only sequence where our method does not perform well is the Yosemite sequence. This is mainly due to the fact that our parameters were chosen to give good performance on real-world sequences. Nonetheless, increasing the smoothness weight by 16x without changing other parameters lowers the AAE on Yosemite from 4.55 to 2.33 degrees (see Fig. 5). Note that this behavior is consistent with most algorithms in the Middlebury benchmark.

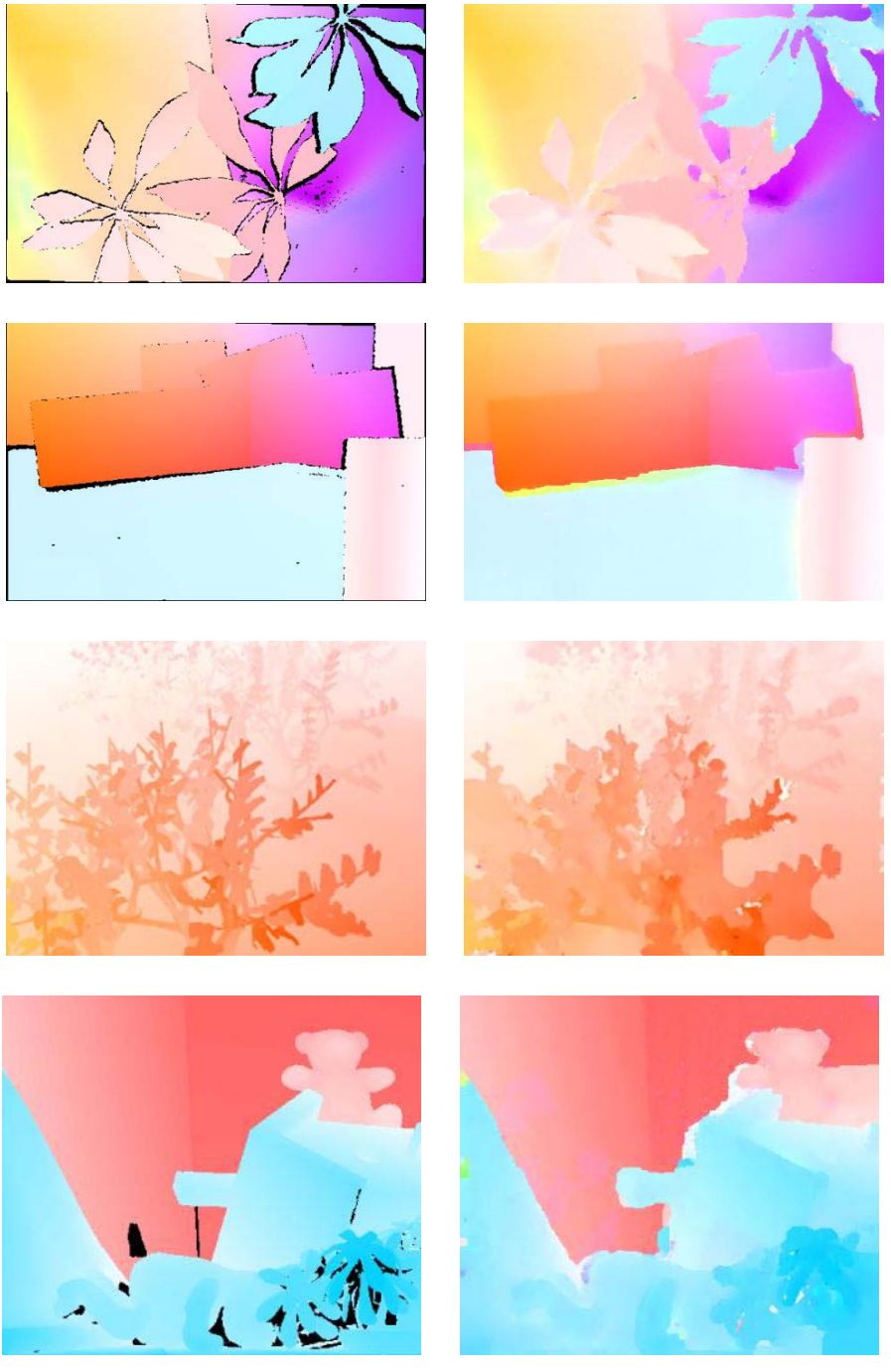
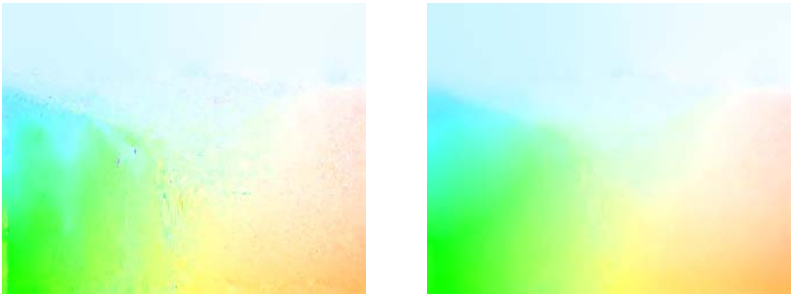


Fig. 4. Example results on the benchmark datasets (right) along with ground truth flows (left): Schefflera, Wooden, Grove, and Teddy (top to bottom)



(a) Standard parameters, AAE = 4.55 (b) Smoothness 16x, AAE = 2.33

Fig. 5. (a) Our method does not perform well on the synthetic Yosemite dataset with our standard parameter settings. (b) Increasing the smoothness parameter $\lambda_{\mathbf{p},\mathbf{q}}$ without changing any other parameters increases flow accuracy considerably.

Proposed energy vs. baseline energy. To evaluate the advantage of the proposed energy, we also considered a simple "baseline" energy that is quite similar to the objectives in popular continuous methods [3,16]. In particular, we again use an 8-neighborhood for the spatial term, but here with convex Charbonnier potentials ($Ch(x,\phi) = \phi^2\sqrt{1+x^2/\phi^2}$), which are similar to the absolute difference measure (see also Fig. 2). The trade-off weight λ was not adapted spatially. For the data term, we used gray-scale images as input from which we did not remove the low frequencies, and also relied on Charbonnier potentials. We optimized the energy using the approach developed here, and tuned the parameters of the baseline model using grid search on "RubberWhale" (see Fig. 6). As can be seen in Fig. 6a and c, the proposed energy clearly outperforms the baseline model visually and quantitatively, even on the sequence used to tune the baseline energy parameters. In particular, the robust spatial potentials employed here allow to recover sharp discontinuities, while at the same time recovering smooth flow fields in continuous areas (Fig. 6c). Also, the robust data potentials allowed to attain better performance in occlusion areas. Finally, ignoring low frequency image content substantially improved the results in areas with shadows, such as on "Schefflera", from which the baseline model suffers.

Our optimization vs. baseline optimization schemes. We also compared the proposed discrete-continuous optimization method with a baseline continuous and a baseline discrete optimization scheme, all on the same proposed energy. For baseline continuous optimization, we employed a hierarchical coarse-to-fine estimation framework with 5 levels (*c.f.* [4]), where at each pyramid level we used the gradient descent scheme described in Sec. 3.2. As a baseline discrete algorithm, we ran α-expansion (i.e. "conventional" graph cuts) [7]. To make the comparison more favorable for this baseline algorithm, we estimated the minimum and maximum horizontal and vertical flow from our high-quality solution (note that such an accurate estimate cannot be obtained directly from our

Table 1. Comparison of different optimization techniques: our full discrete-continuous, our discrete (*i.e.*, fusion of proposals without continuous improvement), baseline continuous, baseline discrete. Shown are the flow accuracy (average angular error) and the energy achieved. On the 8 test datasets [6], our optimization scheme consistently outperforms the two baseline algorithms.

	Army		Mequon		Schefflera		Wooden	
Optimization	AAE	$E(\mathbf{f})$	AAE	$E(\mathbf{f})$	AAE	$E(\mathbf{f})$	AAE	$E(\mathbf{f})$
Our (full)	**4.43**	**2041**	**2.47**	**3330**	**3.70**	**5778**	**3.68**	**1632**
Our (discrete)	4.97	2435	4.83	4375	5.14	7483	5.24	2180
Baseline continuous	7.97	4125	52.3	21417	36.1	24853	16.8	7172
Baseline discrete	5.61	3038	5.19	6209	5.36	8894	4.94	2782
	Grove		Urban		Yosemite		Teddy	
Optimization	AAE	$E(\mathbf{f})$	AAE	$E(\mathbf{f})$	AAE	$E(\mathbf{f})$	AAE	$E(\mathbf{f})$
Our (full)	4.06	**17580**	6.30	**5514**	4.55	**1266**	7.12	**9315**
Our (discrete)	**4.00**	21289	**6.27**	6568	**4.03**	1423	**6.68**	10450
Baseline continuous	64.0	78122	46.1	26517	23.2	4470	63.9	31289
Baseline discrete	9.03	44450	18.7	17770	5.67	1995	9.13	15283

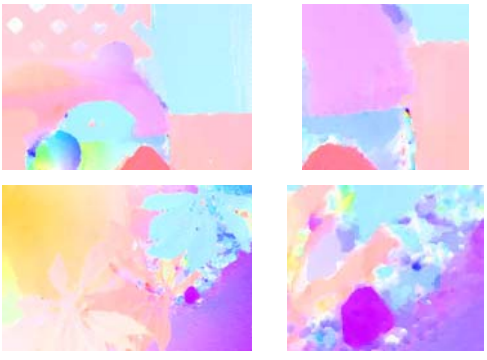
proposal solutions), and discretized this range uniformly using approximately 1000 labels (*i.e.*, 4 times more proposals than what is being used by the fusion approach).

We found that neither of the two baseline energy minimization schemes gave consistently good results for our energy across the benchmark datasets, neither in terms of the energy, nor in terms of flow accuracy (see Tab. 1). In particular, the continuous baseline algorithm failed on most of the datasets (see, *e.g.*, Fig. 6b). This suggests that the proposed energy may be simply too difficult for standard continuous optimization approaches, for example because of the non-convex potentials and the removal of low frequencies in the data term.

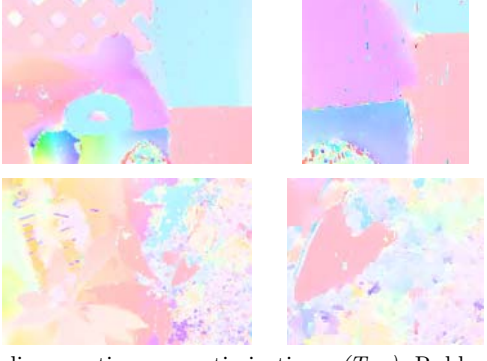
While the behavior of the continuous optimization could be improved, for example using deterministic annealing (*c.f.* [4]) or graduated non-convexity (*c.f.* [30]), such heuristics often do not work well across a wide range of datasets. The baseline discrete algorithm obviously suffered from the uniformity of discretization, especially for the datasets with large motion (see Fig. 7). Later on, we investigate the performance of the α-expansion approach with a more intelligent, non-uniform choice of constant labels.

How many proposals? In our experiments we focused on the accuracy of optical flow estimation, thus using extensive numbers of proposals and an exhaustive number of iterations within continuous optimization. As a result, our unoptimized MATLAB implementation takes more than an hour for processing a single frame pair.

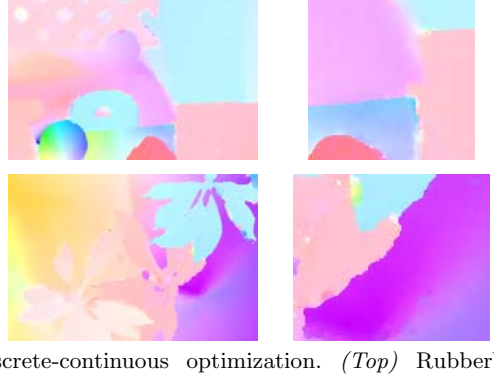
It is important, nevertheless, to emphasize that the flow fusion idea is useful in scenarios where speed matters. Fig. 3 shows one such example, where fusion of just two motion fields (each computed with real-time methods) improves the result over both of them. Note that one fusion takes only fractions of a second.



(a) Baseline energy, discrete-continuous optimization. *(Top)* RubberWhale: AAE = 5.169. *(Bottom)* Schefflera.



(b) Our energy, baseline continuous optimization. *(Top)* RubberWhale: E = 4149, AAE = 6.72. *(Bottom)* Schefflera: E = 24853.



(c) Our energy, discrete-continuous optimization. *(Top)* RubberWhale: E = 1861, AAE = 3.68. *(Bottom)* Schefflera: E = 5778.

Fig. 6. Results for different energies and optimization methods. Each part shows the estimated flow field as well as a detail of the result.

(a) "Urban" ground truth (b) Our discrete optimization (268 proposals) (c) α-expansion ($\approx$ 1000 proposals)

Fig. 7. For a dataset with large motion ("Urban"), the fusion of spatially-varying proposals (the discrete part of our optimization) gives substantially more accurate optical flow than α-expansion with uniform discretization (the baseline discrete optimization)

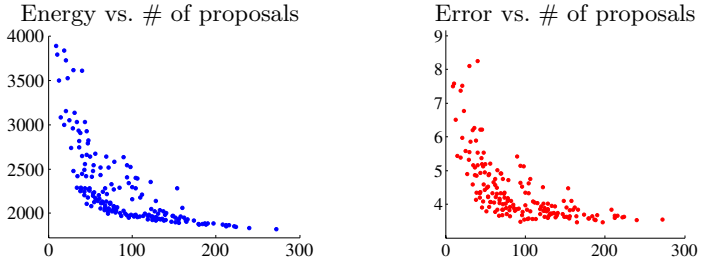


Fig. 8. The energy and the average angular error (AAE) after the discrete optimization step for different subsets of proposals for "Rubber Whale" [6]. The subsets were obtained by disabling different groups and sources of variation (e.g. disabling all Lucas-Kanade proposals, or disable shifting for Horn-Schunck proposals) compared to the full set. The x-coordinate of each dot corresponds to the number of proposals in the subset. The rightmost point on each plot corresponds to the full set of proposals. The plots suggest that sets of proposals that are 5 time smaller would do almost as well in our experiments.

Fig. 8 further explores the trade-off between the number of proposals and the quality of the solution for one of the sequences from [6]. It demonstrates that a five-fold reduction in the number of proposals leads to solutions that are only slightly worse than those computed with the full set of proposals. The possibility of the favourable trade-off between the number of proposals and the obtained energy is also demonstrated by Fig. 9, where a very fast decrease of the energy during the first fusions can be observed.

We also tried to fuse both the discrete and the discrete-continuous results with the known ground truth on "Rubber Whale" to determine the quality of our proposals. We found a mild energy reduction in the discrete case (from $E = 1821$ to 1756), but hardly any reduction in the discrete-continuous case (from $E = 1613$ to 1610). This shows that our proposals appear to be very appropriate, as even knowing the ground truth will not lower the energy much,

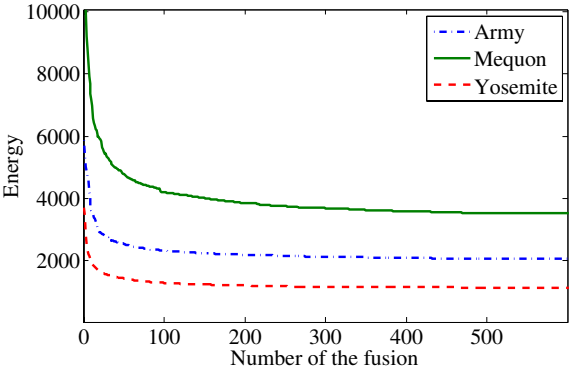


Fig. 9. The energy as a function of the number of fusions performed for the three of the test datasets. The energy drops quite rapidly in the beginning of the discrete optimization process.

Table 2. Spatially-varying proposals vs. constant SIFT-based proposals: The same number of spatially-varying proposals yields considerably lower energies than constant proposals, especially before the continuous-improvement stage. The difference becomes much smaller when the number of constant proposals is increased.

Datasets:	*Army*	*Mequon*	*Schefflera*	*Grove*	*Urban*	*Yosemite*	*Teddy*
268 spatially-varying proposals (LK+HS+constant)							
Discrete opt.	2435	4375	7483	21289	6568	1423	10450
Discrete-continuous opt.	**2041**	**3330**	5778	**17580**	**5514**	**1266**	**9315**
268 constant SIFT-based proposals							
Discrete opt.	3658	6457	9310	30539	7908	2318	11603
Discrete-continuous opt.	2218	3736	5926	21052	5886	1356	9669
Larger set of constant SIFT-based proposals							
Number of proposals	1052	808	704	2423	735	624	492
Discrete opt.	2615	4972	7239	20700	7277	1886	11000
Discrete-continuous opt.	2056	3403	**5586**	17701	5554	1302	9473

and that the continuous improvement step substantially improves the results toward the ground truth (especially in smoothly varying areas). Furthermore, this also suggests that the optimization presented here gets very close to the ground truth (as much as it is permitted by the model) and that further accuracy gains will require more involved models.

Are spatially-varying proposals necessary? As demonstrated above, our approach benefits from the combination of discrete and continuous optimization. It also extensively uses the fusion of spatially-varying proposals during the discrete optimization stage. We have also investigated whether a simpler, more traditional discrete optimization, working with constant proposals only, can give

comparable results, provided that the constant proposals are chosen in a smart way rather than by uniform discretization of the admissible motion range. Towards this end, we pursued the following approach: For the initial pair of images we performed SIFT matching to obtain several hundred of SIFT matches [20]. For each match $(\mathbf{F}^0, \mathbf{F}^1)$, where $\mathbf{F}^0$ and $\mathbf{F}^1$ are the matched feature points in the two frames I^0 and I^1, we create a constant proposal $f_{\mathbf{p}} \equiv \mathbf{F}^1 - \mathbf{F}^0$. We then performed the fusion of these proposals in three sweeps and improved the result with our continuous optimization (the discrete stage here is effectively the α-expansion procedure [7], except that we still used minimum cut on the extended graph for each fusion as the fusion energy can be non-submodular).

We performed this experiment with two different settings. In the first one, we picked the top 268 SIFT correspondences (thus, matching the number of spatially-varying proposals used in the earlier experiments). In the second setting, we set the SIFT match quality threshold to 0.9 and accepted all correspondences below it. We report the results for all datasets except "Wooden", where SIFT-matching is difficult (only 133 matches were obtained given this threshold).

The resulting energies are given in Tab. 2. As can be seen, for a fixed number of proposals, the spatially-varying set of proposals (HS+LK+constant) is giving considerably lower energies than the SIFT-based constant set of proposals. The difference, however, may be compensated to a large degree by taking an excessive number of constant SIFT-based proposals. Whichever strategy is more computationally efficient (smaller number of spatially-varying proposals or larger number of constant proposals) will depend on many factors, in particular on the cost of the computation of the spatially-varying proposals (Lucas-Kanade and Horn-Schunck in our case). However, this experiment as well as the experiments shown in Fig. 8 and 9 suggest that when the number of proposals (or graph-cut operations) is limited, spatially-varying approaches have an advantage.

Note that while we use SIFT descriptors to perform sparse matching between images in order to bootstrap our approach by generating constant proposals, it is also possible to use SIFT descriptors directly for dense matching of different scenes [21].

5 Conclusions

In this chapter we developed a new energy minimization approach for optical flow estimation that combines the advantages of discrete and continuous optimization. The power of the optimization method allowed us to leverage a complex, highly non-convex energy formulation, which is very challenging for traditional continuous optimization methods. The proposed energy formulation was motivated by the statistics of optical flow, borrows from the stereo literature, and is robust to brightness changes, such as in shadow regions. We suggested two different schemes for supplying the required proposal solutions, one using spatially-varying proposals from simple, standard flow algorithms (leading to the FusionFlow algorithm), and one using constant proposals from matching

SIFT features across frames. Experimentally, the approach demonstrates competitive performance on the Middlebury optical flow benchmark across a variety of complex real-world scenes.

Future work should consider whether more efficient proposal mechanisms can be developed that offer similar diversity with many fewer proposals to reduce runtime. With very few proposals, very fast flow estimation might even be possible, at least without continuous refinement. Moreover, it should prove interesting to further investigate if the method can benefit from dynamical proposals that are generated on the fly and adapted to the current solution rather than being generated ahead of time.

Acknowledgements

We would like to thank Pushmeet Kohli and Andrew Fitzgibbon for inspiring discussions and suggestions. We are also grateful to Yuri Boykov and Ramin Zabih for providing interesting and helpful suggestions. We would furthermore like to thank Vladimir Kolmogorov, Yuri Boykov, and Carl Rasmussen for publishing their optimization code packages, Sohaib Khan for publishing his Lucas-Kanade implementation, and Daniel Scharstein, Simon Baker, J.P. Lewis, Michael Black, and Rick Szeliski for creating and maintaining the Middlebury optical flow benchmark.

References

1. Horn, B.K.P., Schunck, B.G.: Determining optical flow. Artificial Intelligence 17(1-3), 185–203 (1981)
2. Lucas, B.D., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: IJCAI, April 1981, pp. 674–679 (1981)
3. Papenberg, N., Bruhn, A., Brox, T., Didas, S., Weickert, J.: Highly accurate optic flow computation with theoretically justified warping. IJCV 67(2), 141–158 (2006)
4. Black, M.J., Anandan, P.: The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields. CVIU 63(1), 75–104 (1996)
5. Roth, S., Black, M.J.: On the spatial statistics of optical flow. IJCV 74(1), 33–50 (2007)
6. Baker, S., Scharstein, D., Lewis, J.P., Roth, S., Black, M.J., Szeliski, R.: A database and evaluation methodology for optical flow. In: ICCV 2007 (2007), `http://vision.middlebury.edu/flow/`
7. Boykov, Y., Veksler, O., Zabih, R.: Fast approximate energy minimization via graph cuts. TPAMI 23(11), 1222–1239 (2001)
8. Sun, J., Zhen, N.N., Shum, H.Y.: Stereo matching using belief propagation. PAMI 25(7), 787–800 (2003)
9. Meltzer, T., Yanover, C., Weiss, Y.: Globally optimal solutions for energy minimization in stereo vision using reweighted belief propagation. In: ICCV 2005, vol. 1, pp. 428–435 (2005)
10. Felzenszwalb, P.F., Huttenlocher, D.P.: Efficient belief propagation for early vision. In: CVPR 2004, vol. 1, pp. 261–268 (2004)

11. Glocker, B., Komodakis, N., Paragios, N., Tziritas, G., Navab, N.: Inter and intra-modal deformable registration: Continuous deformations meet efficient optimal linear programming. In: Karssemeijer, N., Lelieveldt, B. (eds.) IPMI 2007. LNCS, vol. 4584, pp. 408–420. Springer, Heidelberg (2007)
12. Lempitsky, V., Rother, C., Blake, A.: LogCut - Efficient graph cut optimization for Markov random fields. In: ICCV 2007 (2007)
13. Shekhovtsov, A., Kovtun, I., Hlavac, V.: Efficient MRF deformation model for non-rigid image matching. In: CVPR 2007 (2007)
14. Lempitsky, V., Roth, S., Rother, C.: FusionFlow: Discrete-continuous optimization for optical flow estimation. In: CVPR 2008 (2008)
15. Glocker, B., Paragios, N., Komodakis, N., Tziritas, G., Navab, N.: Optical flow estimation with uncertainties through dynamic MRFs. In: CVPR 2008 (2008)
16. Bruhn, A., Weickert, J., Schnörr, C.: Lucas/Kanade meets Horn/Schunck: Combining local and global optic flow methods. IJCV 61(3), 211–231 (2005)
17. Boros, E., Hammer, P.L., Tavares, G.: Preprocessing of unconstrained quadratic binary optimization. Technical Report RUTCOR RRR (2006)
18. Boros, E., Hammer, P.L.: Pseudo-boolean optimization. Discrete Applied Mathematics 123(1-3), 155–225 (2002)
19. Kolmogorov, V., Rother, C.: Minimizing non-submodular functions with graph cuts — A review. TPAMI 29(7), 1274–1279 (2006)
20. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. IJCV 60(2), 91–110 (2004)
21. Liu, C., Yuen, J., Torralba, A.B., Sivic, J., Freeman, W.T.: SIFT flow: Dense correspondence across different scenes. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part III. LNCS, vol. 5304, pp. 28–42. Springer, Heidelberg (2008)
22. Brox, T., Bruhn, A., Weickert, J.: Variational motion segmentation with level sets. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006. LNCS, vol. 3951, pp. 471–483. Springer, Heidelberg (2006)
23. Mémin, É., Pérez, P.: Hierarchical estimation and segmentation of dense motion fields. IJCV 46(2), 129–155 (2002)
24. Heitz, F., Bouthemy, P.: Multimodal estimation of discontinuous optical flow using Markov random fields. TPAMI 15(12), 1217–1232 (1993)
25. Konrad, J., Dubois, E.: Multigrid Bayesian estimation of image motion fields using stochastic relaxation. In: ICCV 1988, pp. 354–362 (1988)
26. Lempitsky, V., Rother, C., Roth, S., Blake, A.: Fusion moves for Markov random field optimization. TPAMI (in revision)
27. Woodford, O.J., Torr, P.H.S., Reid, I.D., Fitzgibbon, A.W.: Global stereo reconstruction under second order smoothness priors. In: CVPR 2008 (2008)
28. Trobin, W., Pock, T., Cremers, D., Bischof, H.: Continuous energy minimization via repeated binary fusion. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part IV. LNCS, vol. 5305, pp. 677–690. Springer, Heidelberg (2008)
29. Birchfield, S., Natarjan, B., Tomasi, C.: Correspondence as energy-based segmentation. Image Vision Comp. 25(8), 1329–1340 (2007)
30. Sun, D., Roth, S., Lewis, J.P., Black, M.J.: Learning optical flow. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part III. LNCS, vol. 5304, pp. 83–97. Springer, Heidelberg (2008)
31. Li, Y., Huttenlocher, D.P.: Learning for optical flow using stochastic optimization. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part II. LNCS, vol. 5303, pp. 379–391. Springer, Heidelberg (2008)

32. Greig, D.M., Porteous, B.T., Seheult, A.H.: Exact MAP estimation for binary images. J. Roy. Stat. Soc. B 51(2), 271–279 (1989)
33. Boykov, Y., Kolmogorov, V.: An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. TPAMI 26(9), 1124–1137 (2004)
34. Kolmogorov, V., Zabih, R.: What energy functions can be minimized via graph cuts? TPAMI 24(2), 147–159 (2004)
35. Rother, C., Kolmogorov, V., Lempitsky, V., Szummer, M.: Optimizing binary MRFs via extended roof duality. In: CVPR 2007 (2007)
36. Ishikawa, H.: Exact optimization for Markov random fields with convex priors. TPAMI 25(10), 1333–1336 (2003)
37. Rasmussen, C.E.: `minimize.m` (September 2006), `http://www.kyb.tuebingen.mpg.de/bs/people/carl/code/minimize/`

An Improved Algorithm for TV-L^1 Optical Flow

Andreas Wedel[1,3], Thomas Pock[2], Christopher Zach[4], Horst Bischof[2], and Daniel Cremers[1]

[1] Computer Vision Group, University of Bonn
[2] Institute for Computer Graphics and Vision, TU Graz
[3] Daimler Group Research and Advanced Engineering, Sindelfingen
[4] Department of Computer Science, University of North Carolina at Chapel Hill

Abstract. A look at the Middlebury optical flow benchmark [5] reveals that nowadays variational methods yield the most accurate optical flow fields between two image frames. In this work we propose an improvement variant of the original duality based TV-L^1 optical flow algorithm in [31] and provide implementation details. This formulation can preserve discontinuities in the flow field by employing total variation (TV) regularization. Furthermore, it offers robustness against outliers by applying the robust L^1 norm in the data fidelity term.

Our contributions are as follows. First, we propose to perform a structure-texture decomposition of the input images to get rid of violations in the optical flow constraint due to illumination changes. Second, we propose to integrate a median filter into the numerical scheme to further increase the robustness to sampling artefacts in the image data. We experimentally show that very precise and robust estimation of optical flow can be achieved with a variational approach in real-time. The numerical scheme and the implementation are described in a detailed way, which enables reimplementation of this high-end method.

1 Introduction

The recovery of motion from images (see Figure 1) is a major task of biological and artificial vision systems. The objective of optical flow methods is to compute a flow field representing the motion of pixels in two consecutive image frames. Since optical flow is an highly ill-posed inverse problem, using pure intensity-based constraints results in an under-determined system of equations, which is known as the *aperture problem*. In order to solve this problem some kind of regularization is needed to obtain physically meaningful displacement fields.

In their seminal work [18], Horn and Schunck studied a variational formulation of the optical flow problem.

$$\min_{\boldsymbol{u}} \left\{ \int_{\Omega} |\nabla u_1|^2 + |\nabla u_2|^2 \, \mathrm{d}\Omega \; + \lambda \int_{\Omega} \left(I_1(\boldsymbol{x} + \boldsymbol{u}(\boldsymbol{x})) - I_0(\boldsymbol{x}) \right)^2 \mathrm{d}\Omega \right\} . \quad (1)$$

Here, I_0 and I_1 is the image pair, $\boldsymbol{u} = (u_1(\boldsymbol{x}), u_2(\boldsymbol{x}))^T$ is the two-dimensional displacement field and λ is a free parameter. The first term (regularization term) penalizes high variations in $\boldsymbol{u}$ to obtain smooth displacement fields. The second term (data term) is also known as the optical flow constraint. It assumes, that the intensity values of $I_0(\boldsymbol{x})$ do not change during its motion to $I_1(\boldsymbol{x}+\boldsymbol{u}(\boldsymbol{x}))$. The free parameter λ weighs between

D. Cremers et al. (Eds.): Visual Motion Analysis, LNCS 5604, pp. 23–45, 2009.

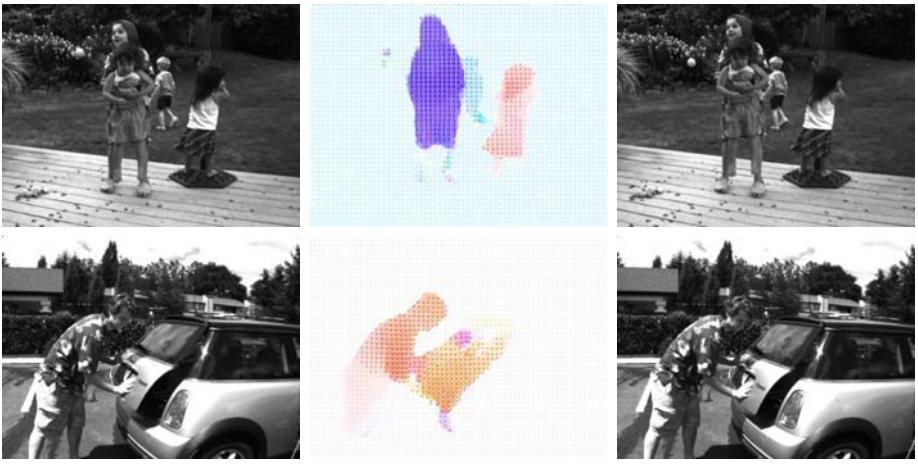


Fig. 1. Optical flow for the *backyard* and *mini cooper* scene of the Middlebury optical flow benchmark. Optical flow captures the dynamics of a scene by estimating the motion of every pixel between two frames of an image sequence. The displacement of every pixel is shown as displacement vectors on top of the commonly used flow color scheme (see Figure 7).

the data fidelity term and the regularization force. Generally speaking, $\boldsymbol{u}$ registers the pixels of the source image I_0 onto the pixels of the target image I_1.

Since the Horn-Schunck model penalizes deviations in a quadratic way, it has two major limitations. It does not allow for discontinuities in the displacement field, and it does not handle outliers in the data term robustly. To overcome these limitations, several models including robust error norms and higher order data terms have been proposed. Since discontinuities in the optical flow appear often in conjunction with high image gradients, several authors replace the homogeneous regularization in the Horn-Schunck model with an anisotropic diffusion approach [21,29]. Others substitute the squared penalty functions in the Horn-Schunck model with more robust variants. Black and Anandan [7] apply estimators from robust statistics and obtain a robust and discontinuity preserving formulation for the optical flow energy. Aubert et al. [3] analyze energy functionals for optical flow incorporating an L^1 data fidelity term and a general class of discontinuity preserving regularization forces. Papenberg et al. [22] employ a differentiable approximation of the TV (resp. L^1) norm and formulate a nested iteration scheme to compute the displacement field.

Most approaches for optical flow computation replace the nonlinear intensity profile $I_1(\boldsymbol{x} + \boldsymbol{u}(\boldsymbol{x}))$ by a first order Taylor approximation to linearize the problem locally. Since such approximation is only valid for small displacements, additional techniques are required to determine the optical flow correctly for large displacements. Scale-space approaches [1] and coarse-to-fine warping (e.g. [2,9,19]) provide solutions to optical flow estimation with large displacements.

In several applications, such as autonomous robot navigation, it is necessary to calculate displacement fields in real-time. Real-time optical flow techniques typically consider only the data fidelity term to generate displacement fields [12,25]. One of the

first variational approaches to compute the optical flow in real-time was presented by Bruhn et al. [10,11]. In their work a highly efficient multi-grid approach is employed to obtain real-time or near real-time performance. The aim of their approach is very similar to our objective: obtaining robust and discontinuity preserving solutions for optical flow with highly efficient implementations. Nevertheless, we utilize a completely different solution strategy, namely a duality based TV-L^1 optical flow algorithm introduced in [31]. In the following section, we reproduce this approach before we present some improvements to increase the robustness and flow accuracy.

2 TV-L^1 Optical Flow [31]

In the basic setting two image frames I_0 and $I_1 : (\Omega \subseteq \mathbb{R}^2) \to \mathbb{R}$ are given. The objective is to find the disparity map $\boldsymbol{u} : \Omega \to \mathbb{R}^2$, which minimizes an image-based error criterion together with a regularization force. In this work we focus on the plain intensity difference between pixels as the image similarity score. Hence, the target disparity map $\boldsymbol{u}$ is the minimizer of

$$\int_{\Omega} \Big\{ \lambda \phi \left(I_0(\boldsymbol{x}) - I_1(\boldsymbol{x} + \boldsymbol{u}(\boldsymbol{x})) \right) + \psi(\boldsymbol{u}, \nabla \boldsymbol{u}, \ldots) \Big\} \, d\boldsymbol{x}, \tag{2}$$

where $\phi \left(I_0(\boldsymbol{x}) - I_1(\boldsymbol{x} + \boldsymbol{u}(\boldsymbol{x})) \right)$ is the image data fidelity, and $\psi(\boldsymbol{u}, \nabla \boldsymbol{u}, \ldots)$ depicts the regularization term. The parameter λ weighs between the data fidelity and the regularization force. Selecting $\phi(x) = x^2$ and $\psi(\nabla \boldsymbol{u}) = |\nabla \boldsymbol{u}|^2$ results in the Horn-Schunck model [18].

The choice of $\phi(x) = |x|$ and $\psi(\nabla \boldsymbol{u}) = |\nabla \boldsymbol{u}|$ yields to the following functional consisting of an L^1 data penalty term and total variation regularization:

$$E = \int_{\Omega} \Big\{ \lambda |I_0(\boldsymbol{x}) - I_1(\boldsymbol{x} + \boldsymbol{u}(\boldsymbol{x}))| + |\nabla \boldsymbol{u}| \Big\} \, d\boldsymbol{x}. \tag{3}$$

Although Eq. 3 seems to be simple, it offers computational difficulties. The main reason is that both, the regularization term and the data term, are not continuously differentiable. One approach is to replace $\phi(x) = |x|$ and $\psi(\nabla \boldsymbol{u}) = |\nabla \boldsymbol{u}|$ with differentiable approximations, $\phi_\varepsilon(x^2) = \sqrt{x^2 + \varepsilon^2}$ and $\psi_\varepsilon(\nabla \boldsymbol{u}) = \sqrt{|\nabla \boldsymbol{u}|^2 + \varepsilon^2}$, and to apply a numerical optimization technique on this slightly modified functional (e.g. [15,9]).

In [13] Chambolle proposed an efficient and exact numerical scheme to solve the Rudin-Osher-Fatemi energy [23] for total variation based image denoising. In the following we show how this approach was adopted in [31] to the optical flow case, yielding a different approach to solve Eq. 3.

2.1 The 1D Stereo Case

In this section we restrict the disparities to be non-zero only in the horizontal direction, e.g. a normalized stereo image pair is provided. Hence, $\boldsymbol{u}(\boldsymbol{x})$ reduces to a scalar $u(\boldsymbol{x})$, and we use the (sloppy) notation $\boldsymbol{x} + u(\boldsymbol{x})$ for $\boldsymbol{x} + (u(\boldsymbol{x}), 0)^T$. The following derivation is based on [4], but adapted to the stereo/optical flow setting. At first, we linearize image I_1 near $\boldsymbol{x} + u_0$, i.e.

$$I_1(\boldsymbol{x} + u) = I_1(\boldsymbol{x} + u_0) + (u - u_0) \, I_1^x(\boldsymbol{x} + u_0),$$

where u_0 is a given disparity map and I_1^x is the derivative of the image intensity I_1 wrt. the x-direction. Using the first order Taylor approximation for I_1 means, that the following procedure needs to be embedded into an iterative warping approach to compensate for image nonlinearities. Additionally, a multi-level approach is employed to allow large disparities between the images.

For fixed u_0 and using the linear approximation for I_1, the TV-L^1 functional (Eq. 3) now reads as:

$$E = \int_\Omega \left\{ \lambda |u\, I_1^x + I_1(\boldsymbol{x} + u_0) - u_0\, I_1^x - I_0| + |\nabla u| \right\} d\boldsymbol{x}. \tag{4}$$

In the following, we denote the current residual $I_1(\boldsymbol{x} + u_0) + (u - u_0)\, I_1^x - I_0$ by $\rho(u, u_0, \boldsymbol{x})$ (or just $\rho(u)$ by omitting the explicit dependency on u_0 and $\boldsymbol{x}$). Moreover, we introduce an auxiliary variable v and propose to minimize the following convex approximation of Eq. 4:

$$E_\theta = \int_\Omega \left\{ |\nabla u| + \frac{1}{2\theta}(u - v)^2 + \lambda |\rho(v)| \right\} d\boldsymbol{x}\ , \tag{5}$$

where θ is a small constant, such that v is a close approximation of u. This convex minimization problem can be optimized by alternating steps updating either u or v in every iteration:

1. For v being fixed, solve

$$\min_u \int_\Omega \left\{ |\nabla u| + \frac{1}{2\theta}(u - v)^2 \right\} d\boldsymbol{x}. \tag{6}$$

   This is the total variation based image denoising model of Rudin, Osher and Fatemi [23].
2. For u being fixed, solve

$$\min_v \int_\Omega \left\{ \frac{1}{2\theta}(u - v)^2 + \lambda\, |\rho(v)| \right\} d\boldsymbol{x}. \tag{7}$$

   This minimization problem can be solved point-wise, since it does not depend on spatial derivatives of v.

An efficient solution for the first step (Eq. 6) is based on gradient descent and subsequent re-projection using the dual-ROF model [14]. It is based on a dual formulation of Eq. 6 and yields a globally convergent iterative scheme. Since this algorithm is an essential part of our method, we reproduce the relevant results from [14]:

Proposition 1. *The solution of Eq. (6) is given by*

$$u = v + \theta\, \mathbf{div}\, \boldsymbol{p}. \tag{8}$$

The dual variable $\boldsymbol{p} = [p_1, p_2]$ *is defined iteratively by*

$$\tilde{\boldsymbol{p}}^{n+1} = \boldsymbol{p} + \frac{\tau}{\theta} \left(\nabla \left(v + \theta\, \mathbf{div}\, \boldsymbol{p}^n \right) \right) \text{ and} \tag{9}$$

$$\boldsymbol{p}^{n+1} = \frac{\tilde{\boldsymbol{p}}^{n+1}}{\max\left\{1, |\tilde{\boldsymbol{p}}^{n+1}|\right\}} \tag{10}$$

where $\boldsymbol{p}^0 = \mathbf{0}$ *and the time step* $\tau \leq 1/4$.

Proposition 2. *The solution of the minimization task in Eq. 7 is given by the following thresholding step:*

$$v = u + \begin{cases} \lambda\theta\, I_1^x & \text{if} \quad \rho(u) < -\lambda\theta\,(I_1^x)^2 \\ -\lambda\theta\, I_1^x & \text{if} \quad \rho(u) > \lambda\theta\,(I_1^x)^2 \\ -\rho(u)/I_1^x & \text{if} \quad |\rho(u)| \leq \lambda\theta\,(I_1^x)^2. \end{cases} \tag{11}$$

This means, that the image residual $\rho(v)$ is allowed to vanish, if the required step from u to v is sufficiently small. Otherwise, v makes a bounded step from u, such that the magnitude of the residual decreases. The proposition above can be shown directly by analyzing the three possible cases, $\rho(v) > 0$ (inducing $v = u - \lambda\theta\, I_1^x$), $\rho(v) < 0$ ($v = u + \lambda\theta\, I_1^x$) and $\rho(v) = 0$ ($v = u - \rho(u)/I_1^x$).

2.2 Generalization to Higher Dimensions

In this section we extend the method introduced in the previous section to optical flow estimation, i.e. a N-dimensional displacement map $\boldsymbol{u}$ is determined from two given N-D images I_0 and I_1. The first order image residual $\rho(\boldsymbol{u}, \boldsymbol{u}_0, \boldsymbol{x})$ wrt. a given disparity map $\boldsymbol{u}_0$ is now $I_1(\boldsymbol{x}+\boldsymbol{u}_0) + \langle \nabla I_1,\, \boldsymbol{u} - \boldsymbol{u}_0 \rangle - I_0(\boldsymbol{x})$. Additionally, we write u_d for the d-th component of $\boldsymbol{u}$ ($d \in \{1, \ldots, N\}$).

The generalization of Eq. 5 to more dimensions is the following energy:

$$E_\theta = \int_\Omega \left\{ \sum_d |\nabla u_d| + \sum_d \frac{1}{2\theta}(u_d - v_d)^2 + \lambda |\rho(\boldsymbol{v})| \right\} d\boldsymbol{x}. \tag{12}$$

Similar to the stereo setting, minimizing this energy can be performed by alternating optimization steps:

1. For every d and fixed v_d, solve

$$\min_{u_d} \int_\Omega \left\{ |\nabla u_d| + \frac{1}{2\theta}(u_d - v_d)^2 \right\} d\boldsymbol{x}. \tag{13}$$

   This minimization problem is identical to Eq. 6 and can be solved by the same procedure. Note, that the dual variables are introduced for every dimension, e.g. Eq. 8 now reads as

$$u_d = v_d - \theta\, \mathbf{div}\, \boldsymbol{p}_d. \tag{14}$$

2. For $\boldsymbol{u}$ being fixed, solve

$$\min_{\boldsymbol{v}} \sum_d \frac{1}{2\theta}(u_d - v_d)^2 + \lambda\, |\rho(\boldsymbol{v})|\,. \tag{15}$$

The following proposition generalizes the thresholding step from Proposition 2 to higher dimensions:

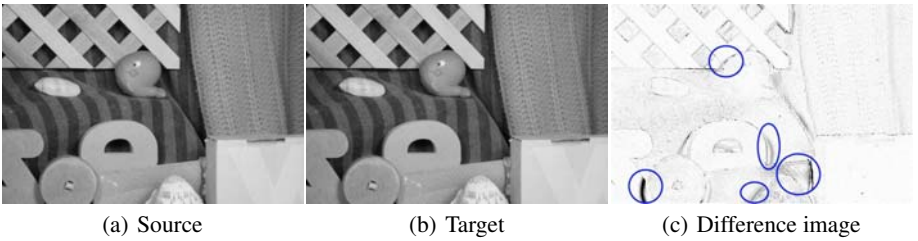
Proposition 3. *The solution of the minimization task in Eq. 15 is given by the following thresholding step:*

$$\boldsymbol{v} = \boldsymbol{u} + \begin{cases} \lambda\theta\,\nabla I_1 & \text{if} \quad \rho(\boldsymbol{u}) < -\lambda\theta\,|\nabla I_1|^2 \\ -\lambda\theta\,\nabla I_1 & \text{if} \quad \rho(\boldsymbol{u}) > \lambda\theta\,|\nabla I_1|^2 \\ -\rho(\boldsymbol{u})\,\nabla I_1/|\nabla I_1|^2 & \text{if} \quad |\rho(\boldsymbol{u})| \leq \lambda\theta\,|\nabla I_1|^2. \end{cases} \tag{16}$$

This proposition essentially states, that the N-dimensional optimization problem can be reduced to a one-dimensional thresholding step, since $\boldsymbol{v}$ always lies on the line $l^{\perp}$ going through $\boldsymbol{u}$ with direction ∇I_1 (for every $\boldsymbol{x}$). This can be seen as follows: The first part in Eq. 15, $\sum_d (u_d - v_d)^2/2\theta$, is basically the squared distance of $\boldsymbol{v}$ to $\boldsymbol{u}$, and the second part, $\lambda\,|\rho(\boldsymbol{v})|$, is the unsigned distance to the line $l : \rho(\boldsymbol{w}) = 0$, i.e. $I_1(\boldsymbol{x}+\boldsymbol{u}_0) + \langle \nabla I_1,\, \boldsymbol{w} - \boldsymbol{u}_0 \rangle - I_0(\boldsymbol{x}) = 0$. If we consider all $\boldsymbol{v}_\mu$ with a fixed distance μ to $\boldsymbol{u}$, then the functional in Eq. 15 is minimized for the $\boldsymbol{v}_\mu$ closest to the line l (with minimal normal distance). This is also valid for the true minimizer, hence the optimum for Eq. 15 is on $l^{\perp}$. In addition, the one-dimensional thresholding step in gradient direction can be applied (Proposition 2), resulting in the presented scheme.

3 Increasing Robustness to Illumination Changes

The image data fidelity term $\phi\,(I_0(\boldsymbol{x}) - I_1(\boldsymbol{x} + \boldsymbol{u}(\boldsymbol{x})))$ states that the intensity values of $I_0(\boldsymbol{x})$ do not change during its motion to $I_1(\boldsymbol{x} + \boldsymbol{u}(\boldsymbol{x}))$. For many sequences this constraint is violated due to sensor noise, illumination changes, reflections, and shadows. Thus, real scenes generally show artifacts that violate the optical flow constraint. Figure 2 shows an example, where the ground truth flow is used to register two images from the Middlebury optical flow benchmark data base [5]. Although the two images are registered at the best using the ground truth flow, the intensity difference image between the source image and the registered target image reveals the violations of the optical flow constraint. Some of these regions, showing artifacts of shadow and shading reflections, are marked by blue circles in the intensity difference image.



(a) Source (b) Target (c) Difference image

Fig. 2. The source and target images of the *rubber-whale* sequence in the Middlebury optical flow benchmark have been registered using the ground truth optical flow. Still, intensity value differences are visible due to sensor noise, reflections, and shadows. The intensity difference image is encoded from white (no intensity value difference) to black (10% intensity value difference). Pixels which are visible in a single image due to occlusion are shown in white.

A physical model of brightness changes was presented in [17], where brightness change and motion is estimated simultaneous; shading artefacts however have not been addressed. In [30] and [20] the authors used photometric invariants to cope with brightness changes, which requires color images. A common approach in literature to tackle illumination changes is to use image gradients besides, or instead of, the plain image intensity values in the data term [9]. This implies that multiple data fidelity terms have to be used and images are differentiated twice, which is known to be noisy.

Here we propose a structure-texture decomposition similar to the approach used in [26] to model the intensity value artifacts due to shading reflections and shadows. The basic idea behind this splitting technique is that an image can be regarded as a composition of a structural part, corresponding to the main large objects in the image, and a textural part, containing fine scale-details [4]. See Figure 3 for an example of such a structure-texture decomposition, also known as cartoon-texture decomposition. The expectation is, that shadows show up only in the structural part which includes the main large objects.

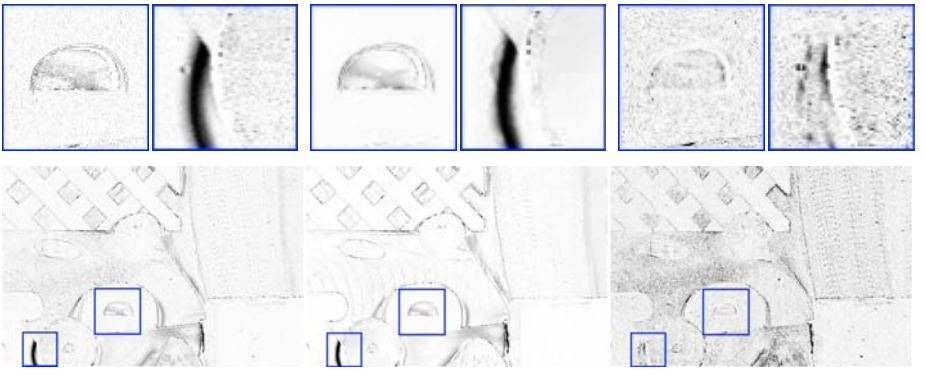
(a) Original (b) Structure part (c) Texture part

Fig. 3. The original image is decomposed into a structural part, corresponding to the main large objects in the image, and a textural part, containing fine-scale details. All images are scaled into the same intensity value range after decomposition.

The structure-texture decomposition is accomplished using the total variation based image denoising model of Rudin, Osher and Fatemi [23]. For the intensity value image $I(\boldsymbol{x})$, the structural part is given as the solution of

$$\min_{I_S} \int_{\Omega} \left\{ |\nabla I_S| + \frac{1}{2\theta}(I_S - I)^2 \right\} d\boldsymbol{x}. \tag{17}$$

The textural part $I_T(\boldsymbol{x})$ is then computed as the difference between the original image and its denoised version, $I_T(\boldsymbol{x}) = I(\boldsymbol{x}) - I_S(\boldsymbol{x})$. Figure 4 shows the intensity difference images between the source image and the registered target image using the ground truth flow for the original image and its decomposed parts. For most parts the artifacts due to shadow and shading reflections show up in the original image and the structural part. The intensity value difference using the textural part, which contains fine-scale details, is noisier than the intensity value difference in the structural part. These intensity value differences are mainly due to sensor noise and sampling artifacts while shadow and shading reflection artifacts have been almost completely removed. This is best visible in the area of the punched hole of the rotated D-shaped object.



(a) Original (b) Structure part (c) Texture part

Fig. 4. Intensity difference images between the source image and the registered target image using ground truth optical flow for the original image and its structure-texture decomposed versions (intensity coding as in Figure 2). Note the presence of shading reflection and shadow artifacts in the original image and in the structure image.

This observation leads to the assumption that the computation of optical flow using the textural part of the image is not perturbed by shadow and shading reflection artifacts, which cover large image regions. To prove this assumption experimentally, we use a blended version of the textural part, $I_T(\alpha, \boldsymbol{x}) = I(\boldsymbol{x}) - \alpha I_S(\boldsymbol{x})$, as input for the optical flow computation. Figure 5 shows the accuracy for optical flow computation using a fixed parameter set and varying the blending factor α. The plot reveals that for larger values of α the accuracy of the optical flow is 30% better than using a small value for α. This confirms the assumption that removing large perturbations due to shadow and shading reflections yields better optical flow estimates. In the experiments we set $\alpha = 0.95$ and compute the image decomposition as follows:

The original source and target images are scaled into the range $[-1, 1]$ before computing the structure part. We use $\lambda_{ROF} = 0.125$ and 100 iterations of the re-projection step presented in Proposition 1 to solve Eq. (17). In the CPU implementation, the resulting source and target texture images are also equivalently scaled into the range $[-1, 1]$ prior to optical flow computation.

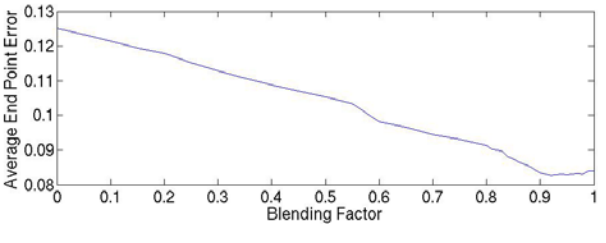


Fig. 5. The plot shows the optical flow accuracy, measured as the average end point error, using different α values for the blending of the textural part of the image (same image pair as Figure 3). The improvement using the textural part for the optical flow computation becomes visible.

4 Implementation

This section gives details on the employed numerical procedure and on the implementation for the proposed TV-L^1 optical flow approach. Although the discussion in Section 2.2 is valid for any image dimension $N \geq 2$, the discussion in this Section is specifically tailored for the case $N = 2$.

4.1 Numerical Scheme

The generally non-convex energy functional for optical flow (Eq. 3) becomes a convex minimization problem after linearization of the image intensities (Eq. 4). But this linearization is only valid for small displacements. Hence, the energy minimization procedure is embedded into a coarse-to-fine approach to avoid convergence to unfavorable local minima. We employ image pyramids with a down-sampling factor of 2 for this purpose. The resulting numerical scheme is summarized in Algorithm 1.

Algorithm 1. Numerical scheme of the TV-L^1 optical flow. In the numerical scheme, a super-scripted ^{L} denotes the pyramid level.

Input: Two intensity images I_0 and I_1
Output: Flow field $\boldsymbol{u}$ from I_0 to I_1

Preprocess the input images; (Sec. 3)
for $L = 0$ *to* max_level **do**
  Calculate restricted pyramid images ${}^L I_0$ and ${}^L I_1$;
end
Initialize ${}^L\boldsymbol{u} = 0$, ${}^L\boldsymbol{p} = 0$, and $L = max\_level$;
while $L \geq 0$ **do**
  for *W = 0 to max_warps* **do**
    Re-sample coefficients of ρ using ${}^L I_0$, ${}^L I_1$, and ${}^L\boldsymbol{u}$; (Warping)
    for $Out = 0$ *to* $max\_outer\_iterations$ **do**
      Solve for ${}^L\boldsymbol{v}$ via thresholding; (Eq. 16)
      for $In = 0$ *to* $max\_inner\_iterations$ **do**
        Perform one iteration step to solve for ${}^L\boldsymbol{u}$; (Prop. 1)
      end
      Median filter ${}^L\boldsymbol{u}$;
    end
  end
  if $L > 0$ **then**
    Prolongate ${}^L\boldsymbol{u}$ and ${}^L\boldsymbol{p}$ to next pyramid level $L - 1$;
  end
end

Beginning with the coarsest level, we solve Eq. 3 at each level of the pyramid and propagate the solution to the next finer level. This solution is further used to compute the coefficients of the linear residual function ρ by sampling I_0 and I_1 using the corresponding pyramid levels. Thus, the warping step for I_1 takes place every time, the solution is propagated across pyramid levels. We use additional warps on each level to get more accurate results. Avoiding poor local minima is not the only advantage of the coarse-to-fine approach. It turns out, that the filling-in process induced by the regularization occurring in texture-less region is substantially accelerated by a hierarchical scheme as well. In the following subsections the single steps of the numerical scheme are outlined and implementation details are provided.

4.2 Pyramid Restriction and Prolongation

The pyramid restriction and prolongation operations for image intensities, flow vectors, and the dual variable $\boldsymbol{p}$ are quite different. While gray values can simply be averaged, flow vectors need to be scaled with the scaling factor between the pyramid levels to yield valid displacement vectors on every pyramid level. In our case we employ image pyramids with a down-sampling factor of 2.

The restriction operator, which is used for the intensity images is a combination of a low pass 5×5 binomial filter and subsequent down-sampling [24]. That is, odd rows and columns are removed from the image (note, that such procedure does require the size of the input image to be a power of 2 times the size of the lowest resolved image). The mask used is

$$\frac{1}{16}\begin{bmatrix}1\\4\\6\\4\\1\end{bmatrix} \times \frac{1}{16}\begin{bmatrix}1\,4\,6\,4\,1\end{bmatrix} = \frac{1}{256}\begin{bmatrix}1 & 4 & 6 & 4 & 1\\ 4 & 16 & 24 & 16 & 4\\ 6 & 24 & 36 & 24 & 6\\ 4 & 16 & 24 & 16 & 4\\ 1 & 4 & 6 & 4 & 1\end{bmatrix} . \tag{18}$$

The prolongation operator up-samples the image, that is, inserts odd zero rows and columns, and then applies the 5×5 binomial filter multiplied by 4 to it. Here we have to differentiate between up-sampling of flow vectors $\boldsymbol{u}$, which have to be multiplied by a factor of 2 and up-sampling of the dual variable $\boldsymbol{p}$.

The dual variable $\boldsymbol{p}$ is not multiplied by a factor. Instead, Dirichlet boundary conditions are enforced by first setting the border of the dual variable to 0 and then up-sampling the dual variable.

4.3 Outer Iteraion: Re-sampling the Data Term Coefficients via Warping

Similarly to the 1D stereo case (Sec. 2.1), the image I_1 is linearized using the first order Taylor approximation near $\boldsymbol{x} + \boldsymbol{u}_0$, where $\boldsymbol{u}_0$ is a given optical flow map:

$$I_1(\boldsymbol{x}+\boldsymbol{u}) = I_1(\boldsymbol{x}+\boldsymbol{u}_0) + (\boldsymbol{u}+\boldsymbol{u}_0)\nabla I_1(\boldsymbol{x}+\boldsymbol{u}_0)\ . \tag{19}$$

The data fidelity term $\rho(\boldsymbol{u})$ now reads

$$\rho(\boldsymbol{u}) = \boldsymbol{u}\nabla I_1(\boldsymbol{x}+\boldsymbol{u}_0) + \underbrace{I_1(\boldsymbol{x}+\boldsymbol{u}_0) - \boldsymbol{u}_0\nabla I_1(\boldsymbol{x}+\boldsymbol{u}_0) - I_0(\boldsymbol{x})}_{c}\,, \tag{20}$$

where the right part, denoted by c, is independent of $\boldsymbol{u}$, and hence fixed. We use bi-cubic look-up to calculate the intensity value $I_1(\boldsymbol{x}+\boldsymbol{u}_0)$ and the derivatives of I_1 (bi-linear lookup on the GPU). The derivatives on the input images are approximated using the five-point stencil $\frac{1}{12}\begin{bmatrix}-1 & 8 & 0 & -8 & 1\end{bmatrix}$. If the bi-cubic look-up falls onto or outside the original image boundary, a value of 0 is returned.

Assuming that $\boldsymbol{u}_0$ is a good approximation for $\boldsymbol{u}$, the optical flow constraint states that $I_0(\boldsymbol{x}) \approx I_1(\boldsymbol{x}+\boldsymbol{u}_0)$. Taking this further onto image derivatives, we obtain that $\nabla I_0(\boldsymbol{x})$ is a good approximation for $\nabla I_1(\boldsymbol{x}+\boldsymbol{u}_0)$. Note, that replacing $\nabla I_1(\boldsymbol{x}+\boldsymbol{u}_0)$ with $\nabla I_0(\boldsymbol{x})$ implies that no bi-cubic look-up for the image gradients has to be employed and the computation time can be sped up. However, it turns out that using blended versions of the derivatives larger flow vectors can be matched and hence better results are achieved. Figure 6 shows the accuracy for blended versions of the derivative $\nabla I = (1-\beta)\nabla I_1(\boldsymbol{x}+\boldsymbol{u}_0) + \beta\nabla I_0(\boldsymbol{x})$ keeping all other parameters fix. Values for β around 0.5 show the best results in terms of optical flow accuracy. This can be explained by the fact that both images contribute to the gradient, increasing the redundancy. In our experiments we use a fixed value of $\beta = 0.4$. A similar approach has been proposed for symmetric KLT tracking by Birchfield in [6].

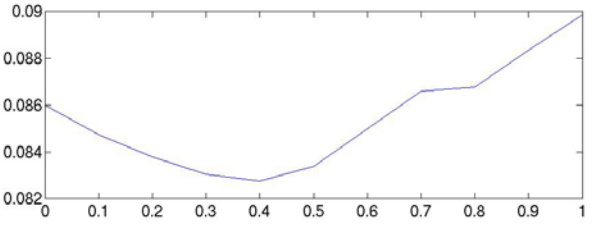


Fig. 6. The plot shows the optical flow accuracy, measured as the average end point error, using different β values for the blending of the gradients from image I_0 and I_1 (same image pair as Figure 3). The improvement using a blended version of the gradients for the optical flow computation becomes visible.

4.4 Inner Iteration: Minimization Procedure of u and v

Within every outer iteration (Proposition 3 followed by Proposition 1), a given number of fixed-point scheme steps (inner iterations) are perfomed to update all $\boldsymbol{p}_d$ (and therefore $\boldsymbol{u}$, Proposition 1), followed by a median filtering of $\boldsymbol{u}$.

The implementation of Proposition 1 uses backward differences to approximate $\mathbf{div}\, \boldsymbol{p}$ and forward differences for the numerical gradient computation in order to have mutually adjoint operators [13].

The discrete version of the forward difference gradient $(\nabla u)_{i,j} = ((\nabla u)^1_{i,j}, (\nabla u)^2_{i,j})$ at pixel position (i, j) for a data field of width N and height M is defined as

$$(\nabla u)^1_{i,j} = \begin{cases} u_{i+1,j} - u_{i,j} & \text{if } i < N \\ 0 & \text{if } i = N \end{cases} \tag{21}$$

and

$$(\nabla u)^2_{i,j} = \begin{cases} u_{i,j+1} - u_{i,j} & \text{if } j < M \\ 0 & \text{if } j = M \end{cases} . \tag{22}$$

The discrete version of the backward differences divergence operator is

$$(\mathbf{div}\, \boldsymbol{p})_{i,j} = \begin{cases} p^1_{i,j} - p^1_{i-1,j} & \text{if } 1 < i < N \\ p^1_{i,j} & \text{if } i = 1 \\ -p^1_{i-1,j} & \text{if } i = N \end{cases} + \begin{cases} p^2_{i,j} - p^2_{i,j-1} & \text{if } 1 < j < M \\ p^2_{i,j} & \text{if } j = 1 \\ -p^2_{i,j-1} & \text{if } j = M \end{cases} . \tag{23}$$

The iterative re-projection scheme to update $\boldsymbol{u}$ using the a quadratic coupling term with $\boldsymbol{v}$ essentially assumes the differences between $\boldsymbol{v}$ and $\boldsymbol{u}$ to be Gaussian. After updating $\boldsymbol{u}$, we still find that the solution contains outliers. With the median filtering of $\boldsymbol{u}$ we discard these outliers successfully. The median filter employed is a 3×3 median filter, which can be efficiently implemented [16].

4.5 Acceleration by Graphics Processing Units

Numerical methods working on regular grids, e.g. rectangular image domains, can be effectively accelerated by modern graphics processing units (GPUs). We employ the huge computational power and the parallel processing capabilities of GPUs to obtain a fully accelerated implementation of our optical flow approach. The GPU-based procedure is essentially a straightforward CUDA implementation of the numerical scheme in Algorithm 1. We currently use a fixed but tunable number of warps and iterations on each level in our implementations. Results using both, the CPU version and the GPU-based optical flow can be found in the next section.

5 Results

In this section we provide three sets of results. The first set quantitatively evaluates the accuracy increase for the proposed improved optical flow algorithm on the Middlebury flow benchmark. The benchmark provides a training data set where the ground truth optical flow is known and an evaluation set used for a comparison against other algorithms in literature. For visualization of the flow vectors we used the color coding scheme proposed in [5] (See also Figure 7).

The second set evaluates real scenes, taken from a moving vehicle. It demonstrates the performance of the improved optical flow algorithm under different illumination conditions and under large image motion.

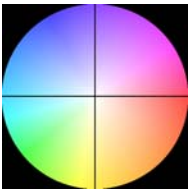


Fig. 7. Color coding of the flow vectors: Direction is coded by hue, length is coded by saturation

Table 1. Evaluation results on the Middlebury training data. The evaluation is splittet into real-time *Performance* results and the results of the proposed TV-L^1-improved algorithm, employing additional warps and bi-cubic lookup. The table shows the average end point error of the estimated flow fields. Parameters have been carefully chosen for algorithm comparison (see text for parameters and run-time).

		Dimetrodon	Grove2	Grove3	Hydrangea	RubberWhale	Urban2	Urban3	Venus
Performance	P(GPU)	0.259	0.189	0.757	0.258	0.218	0.652	1.069	0.482
	P	0.236	0.190	0.803	0.240	0.302	0.598	0.897	0.486
	P-MF(GPU)	0.224	0.173	0.671	0.251	0.183	0.508	0.889	0.433
	P-MF	0.202	0.161	0.666	0.236	0.161	0.468	0.679	0.428
	P-I_T-MF(GPU)	0.186	0.200	0.743	0.186	0.118	0.487	1.026	0.314
	P-I_T-MF	**0.171**	0.191	0.730	0.173	0.109	0.390	0.812	0.311
	TV-L^1-improved	0.190	**0.154**	**0.665**	**0.147**	**0.092**	**0.319**	**0.630**	**0.260**

Table 2. Run-time comparison for the **Performance** section in Table 1. Using the parallel power of a GPU yields performance gain at the cost of accuracy loss. The run-time is measured on the *Grove3* test image (640×480 px).

	Algorithm	Processor	Avg. Accuracy	Run-time
Performance	P(GPU)	NVidia® GeForce® GTX 285	0.486	0.039 [sec]
	P	Intel® Core™2 Extreme 3.0GHz	0.468	0.720 [sec]
	P-MF(GPU)	NVidia® GeForce® GTX 285	0.416	0.055 [sec]
	P-MF	Intel® Core™2 Extreme 3.0GHz	0.375	0.915 [sec]
	P-I_T-MF(GPU)	NVidia® GeForce® GTX 285	0.408	0.061 [sec]
	P-I_T-MF	Intel® Core™2 Extreme 3.0GHz	0.361	1.288 [sec]

In the third set of results we show optical flow results of our core real-time implementation (no texture images and no median filter) on a graphics card to evaluate our algorithm in indoor scenes.

5.1 Evaluation on the Middlebury Benchmark

The performance section in Table 1 compares real-time capable implementations for optical flow. Both, the TV-L^1 optical flow algorithm and the image decomposition described in Section 3, employ the Rudin-Osher-Fatemi denoising algorithm. This denoising step can be efficiently implemented on modern graphics cards, putting up with small accuracy losses: For parallel processing, the iterative denoising (Proposition 1) is executed on sub-blocks of the image in parallel, where boundary artifacts may occur. Hence, high accuracy is exchanged versus run-time performance (see Table 2). The measured timings do *not* include the image uploads to video memory and the final visualization of the obtained displacement field. These times are included in the timings of the optical flow algorithm for video sequences in Section 5.3.

In all three algorithm settings, P, P-MF, and P-I_T-MF, the linearized optical flow constraints (19) is used as data term. The number of outer iterations is set to one. In the plain version, algorithm P, 5 inner iterations are used in every warping step. The number of refinement warps on every pyramid level was set to 25. The parameter settings are $\lambda = 25$ and $\theta = 0.2$. Gray value look-up is bi-linear, as this can be done without additional costs on modern graphics cards. The image gradient is computed via central derivatives from the average of both input images.

The P-MF algorithm extends the basic algorithm by an additional Median filter step, hence 5 iterations of the Proposition 1, followed by a median filter step, are performed for each warp. The Median filter makes the whole scheme more robust against outliers. For this reason the influence of the data term, weighted by λ can be increased to $\lambda = 50$. All other parameters are kept fix.

In the third algorithm, P-I_T-MF, the textural part of the image is used, as described in Section 3. Note that *for real-time purposes* the input images are only scaled into the range $[-1, 1]$ once, prior to texture extraction, by using the maximum gray value. For real-time computation the min/max computation in the texture image is quite time-consuming on a GPU. Again the increase of accuracy at the cost of a longer execution time can be seen in the quantitative evaluation. It is interesting to note that only the flow fields for the *real scenes* within the test set benefit from the image decomposition. The optical flow for the *rendered scenes*, Grove and Urban, is actually worse. This is not surprising as texture-extraction removes some structure information in the images; such procedure is only beneficial if the images contain illumination artifacts. Because this is the fact for all natural scenes (which are for obvious reasons more interesting and challenging), in the remaining experiments the texture-structure decomposition is performed inherently.

In the settings for the proposed *TV-L1-improved* algorithm we set the focus on accuracy. For this, we use 35 warps, 5 outer iterations, and 1 inner iteration. We set $\lambda = 30$ and $\theta = 0.25$, and use bi-cubic lookup as well as five-point differences for the gradients.

Average angle error	avg. rank	Army (Hidden texture) GT im0 im1			Mequon (Hidden texture) GT im0 im1			Schefflera (Hidden texture) GT im0 im1			Wooden (Hidden texture) GT im0 im1			Grove (Synthetic) GT im0 im1			Urban (Synthetic) GT im0 im1			Yosemite (Synthetic) GT im0 im1			Teddy (Stereo) GT im0 im1		
		all	disc	untext	all	disc	untext	all	disc	untext	all	disc	untext	all	disc	untext	all	disc	untext	all	disc	untext	all	disc	untext
TV-L1-improved [19]	3.8	3.36 1	**9.63** 1	**2.62** 1	2.82 2	10.7 3	2.23 2	6.50 4	15.8 4	2.73 2	3.80 3	21.3 6	**1.76** 1	3.34 1	4.38 2	**2.39** 1	5.97 3	18.1 7	5.67 6	3.57 11	4.92 12	3.43 11	4.01 3	9.84 3	3.44 2
F-TV-L1 [18]	5.2	5.44 8	12.5 6	5.69 10	5.46 9	15.0 9	4.03 9	7.40 9	16.3 6	3.42 6	5.08 7	23.3 10	2.81 6	3.42 2	**4.34** 1	3.03 2	4.05 2	**15.1** 1	**3.18** 1	2.43 6	3.92 6	1.07 5	3.90 2	9.35 2	**2.61** 1
Brox et al. [7]	6.2	4.80 7	14.4 10	4.29 8	4.05 7	13.5 6	3.71 7	6.63 6	16.0 5	7.26 9	5.22 8	22.7 9	3.22 8	4.56 11	6.09 16	3.40 3	3.97 1	17.9 5	3.41 2	2.07 3	3.76 4	1.18 2	5.14 4	11.9 6	4.28 4
Fusion [9]	6.5	4.43 5	13.7 8	4.08 6	2.47 1	**8.91** 1	2.24 3	3.70 1	**9.68** 1	3.12 4	3.68 1	19.8 3	2.54 5	4.26 8	5.16 9	4.31 11	6.32 5	16.8 3	6.15 7	4.55 15	5.78 15	3.10 10	7.12 11	13.6 11	7.86 13
Dynamic MRF [10]	7.1	4.58 6	12.4 5	4.14 7	3.25 4	13.9 7	2.27 4	6.02 3	16.0 7	**2.36** 1	4.39 5	22.6 8	2.51 4	3.61 3	4.55 4	3.46 4	6.01 9	22.2 15	6.70 8	2.41 5	3.40 2	3.69 12	9.26 16	17.0 15	10.2 16
SegOF [13]	7.2	5.85 9	13.5 7	3.98 5	7.40 10	14.9 8	8.13 13	8.55 11	17.3 10	9.01 10	6.50 12	**18.1** 1	5.14 12	3.90 7	4.53 3	4.81 14	6.57 6	21.7 13	6.81 10	1.65 1	3.49 3	**1.08** 1	3.71 1	**9.23** 1	3.63 3
CBF [15]	7.3	3.95 3	10.1 2	3.44 4	3.70 6	10.6 2	3.85 8	5.64 2	13.5 2	3.34 5	3.71 2	21.5 7	1.99 2	4.36 9	5.50 10	3.55 6	11.3 16	19.1 9	9.05 15	6.79 17	7.37 18	11.6 17	5.50 5	11.8 5	5.66 6
GraphCuts [17]	8.4	6.25 10	14.3 9	5.53 9	8.60 12	20.1 13	6.61 10	7.91 10	15.4 3	10.9 11	4.88 6	19.0 2	3.05 7	3.78 4	4.71 7	3.94 8	8.74 12	16.4 2	5.39 4	4.04 13	4.87 10	4.85 15	6.35 8	12.2 7	6.05 9

(a) Average angle error on the Middlebury optical flow benchmark.

Average end-point error	avg. rank	Army (Hidden texture) GT im0 im1			Mequon (Hidden texture) GT im0 im1			Schefflera (Hidden texture) GT im0 im1			Wooden (Hidden texture) GT im0 im1			Grove (Synthetic) GT im0 im1			Urban (Synthetic) GT im0 im1			Yosemite (Synthetic) GT im0 im1			Teddy (Stereo) GT im0 im1		
		all	disc	untext	all	disc	untext	all	disc	untext	all	disc	untext	all	disc	untext	all	disc	untext	all	disc	untext	all	disc	untext
TV-L1-improved [19]	3.7	**0.09** 1	**0.26** 1	**0.07** 1	0.20 2	0.71 2	0.16 2	0.53 6	1.18 6	0.22 2	0.21 3	1.24 7	**0.11** 1	**0.90** 1	1.31 2	**0.72** 1	1.51 6	1.93 4	0.84 3	0.18 11	0.17 8	0.31 11	0.73 3	1.62 3	0.87 3
Fusion [9]	5.1	0.11 4	0.34 5	0.10 5	**0.19** 1	**0.69** 1	0.16 2	**0.29** 1	**0.66** 1	0.23 3	**0.20** 1	1.19 5	0.14 4	1.07 5	1.42 5	1.22 9	1.35 3	**1.49** 1	0.86 5	0.20 12	0.20 15	0.26 9	1.07 8	2.07 9	1.39 9
F-TV-L1 [18]	5.3	0.14 8	0.35 7	0.14 9	0.34 9	0.98 8	0.26 7	0.59 10	1.19 7	0.26 6	0.27 7	1.36 11	0.16 6	**0.90** 1	**1.30** 1	0.76 3	**0.54** 1	1.62 2	**0.36** 1	0.13 6	0.15 7	0.20 7	**0.68** 1	1.56 2	**0.66** 1
Brox et al. [7]	6.0	0.12 6	0.37 9	0.11 7	0.31 8	0.97 7	0.28 8	0.48 4	1.11 4	0.48 9	0.28 8	1.28 8	0.18 8	1.13 8	1.57 11	1.11 7	1.02 2	2.02 6	0.60 2	0.10 2	0.13 3	**0.11** 1	0.93 5	2.00 7	1.07 5
Dynamic MRF [10]	6.5	0.12 6	0.34 5	0.11 7	0.22 3	0.89 6	0.16 2	0.44 2	1.13 5	**0.20** 1	0.24 5	1.29 9	0.14 4	1.11 6	1.52 8	1.13 8	1.54 6	2.37 15	0.93 6	0.13 6	0.12 2	0.31 11	1.27 11	2.33 13	1.66 11
CBF [15]	6.6	0.10 2	0.28 2	0.09 3	0.29 5	0.79 3	0.29 9	0.45 3	0.98 2	0.24 5	0.21 3	1.22 6	0.13 3	0.96 3	1.39 4	0.74 2	2.32 16	2.19 9	1.29 15	0.34 17	0.27 18	0.85 17	0.86 4	1.78 4	1.06 4
SegOF [13]	7.2	0.15 9	0.36 8	0.10 5	0.57 10	1.16 10	0.59 14	0.68 11	1.24 9	0.64 10	0.32 11	**0.86** 1	0.26 11	1.18 11	1.50 7	1.47 13	1.63 9	2.09 6	0.96 7	**0.08** 1	0.13 3	0.12 2	0.70 2	**1.50** 1	0.69 2
Second-order prior [11]	7.6	0.10 2	0.30 3	0.08 2	0.22 3	0.85 4	0.15 1	0.57 9	1.28 11	0.23 3	0.20 1	1.14 3	0.11 1	1.13 8	1.55 10	1.03 6	2.52 17	2.45 16	1.25 13	0.42 18	0.25 17	1.09 19	0.98 6	1.92 6	1.07 5

(b) Average end point error on Middlebury optical flow benchmark.

Fig. 8. Error measurements on the Middlebury optical flow benchmark as on October 22nd 2008. The proposed method (*TV-L1-improved*) outperforms other current state-of-the-art methods for optical flow on the Middlebury optical flow benchmark in both measurement categories, angle error and end point error.

The result on the test data set are shown in the last row of Table 1. Run-time on the *Grove3* sequence was 3.46 seconds. Figure 8 shows the benchmark results. Currently, as on October 22nd 2008, there are results of 19 different optical flow algorithms in the benchmark. Our method outperforms all approaches in terms of angle error and end point error. Figures 14 and 15 show the obtained results for all eight evaluation sequences. For most part, the remaining flow errors are due to occlusion artifacts.

5.2 Traffic Scenes

The computation of optical flow is important to understand the dynamics of a scene. We evaluated our optical flow in different scenarios under different illumination conditions (night, day, shadow). Images are taken from a moving vehicle where the camera monitors the road course ahead.

The first experiment in Figure 9 shows the optical flow computation on an image sequence with a person running from the right into the driving corridor. Due to illumination changes in the image (compare the sky region for example) and severe vignetting artifacts (images intensity decreases circular from the image middle), standard optical flow computation fails. Using the proposed structure-texture decomposition, a valid flow estimation is still possible. Note the reflection (note: this is not a *shading reflection*) of the moving person on the engine hood which is only visible in the

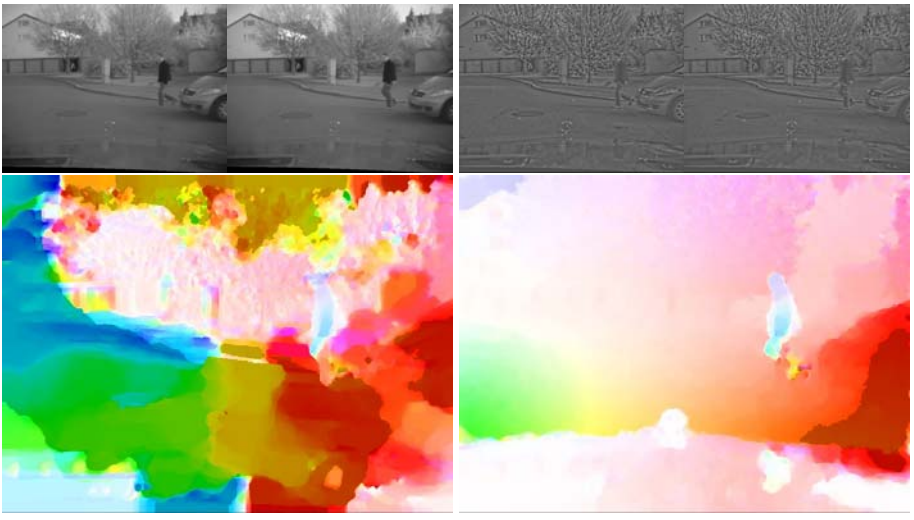


Fig. 9. Optical flow computation with illumination changes. Due to illumination changes, the optical flow constraint in the two input images (*upper left images*) is violated and flow computation on the pixel intensities (*left*) fails. Using the structure-texture decomposed images (*upper right images*), a valid flow estimation is still possible (*right side*). The optical flow color is saturated for flow vector length above $15px$.

structure-texture decomposed images. Artifacts due to vignetting and illumination change are not visible in the structure-texture decomposed images. This demonstrates the increase in robustness for the optical flow computation under illumination changes using the proposed decomposition of the input images.

A second example in Figure 10 shows a scene at night with reflections on the ground plane. In the intensity images the scene is very dark and not much structure is visible. The structure-texture decomposed images reveal much more about the scene. Note, that this information is also included in the intensity image but most structure in the original images is visible in the cloud region. The figure shows the optical flow using the decomposed images. Note the correct flow estimation of the street light on the left side.

The next two examples demonstrate the accurate optical flow computation for large displacements. In Figure 11 the image is taken while driving under a bridge on a country road. Note, that the shadow edge of the bridge is visible in the original images but not in the decomposed image. The large flow vectors on the reflector post are correctly matched. Only in the vicinity of the car optical flow is perturbed due to missing texture on the road surface.

Figure 12 shows a scene with shadows on the road. The structure-texture decomposed image reveals the structure on the road surface better then the original intensity

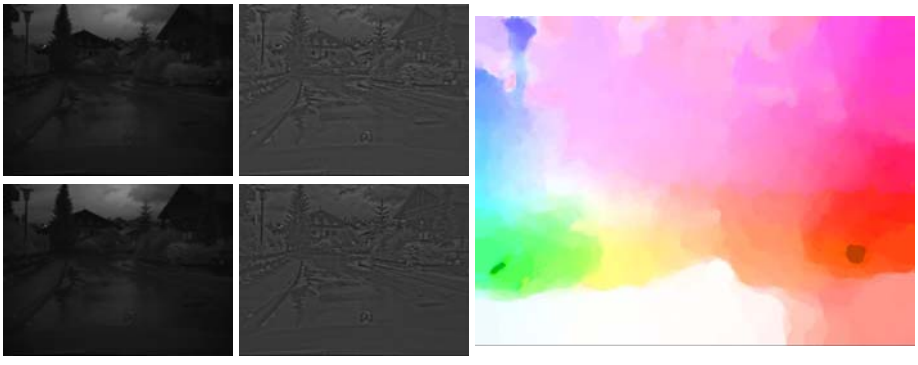


Fig. 10. Computation of optical flow for a night scene. The left images are the original intensity images. The middle images are the structure-texture decomposed images used for optical flow computation. The optical flow result is shown on the right, where flow vectors with length above 10px are saturated in the color coding.

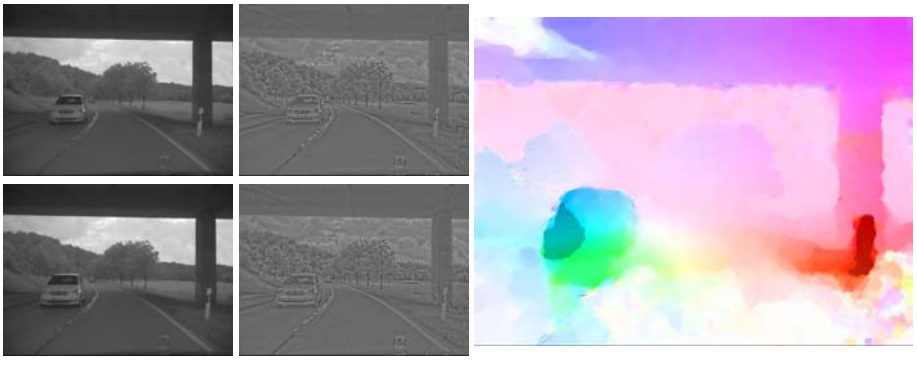


Fig. 11. The scene shows the computation of optical flow with large displacement vectors. The original input images are shown on the *left*. The *middle* images are the blended structure-texture images. Flow vectors above 20px are color-saturated in the optical flow color image.

images. We have used different scales for the optical flow color scheme to demonstrate the accuracy of our optical flow algorithm. Although nothing about epipolar geometry is used in the flow algorithm (as opposed to e. g. [27]), the effect of expansion (and hence depth) corresponding to flow length becomes visible. Note, that optical flow for the reflection posts is correctly estimated even for flow length above 8px. Optical flow is correctly estimated for the road surface up to 30px. The shadows in the scene have no negative impact on the flow calculation. The engine hood behaves like a mirror and optical flow on the engine hood is perturbed due to reflections. Although the optical flow for the engine hood is very much different for flow vectors on the road surface, this has no negative impact on the estimation of the optical flow for the road surface. Note the accurate flow discontinuity boundary along the engine hood.

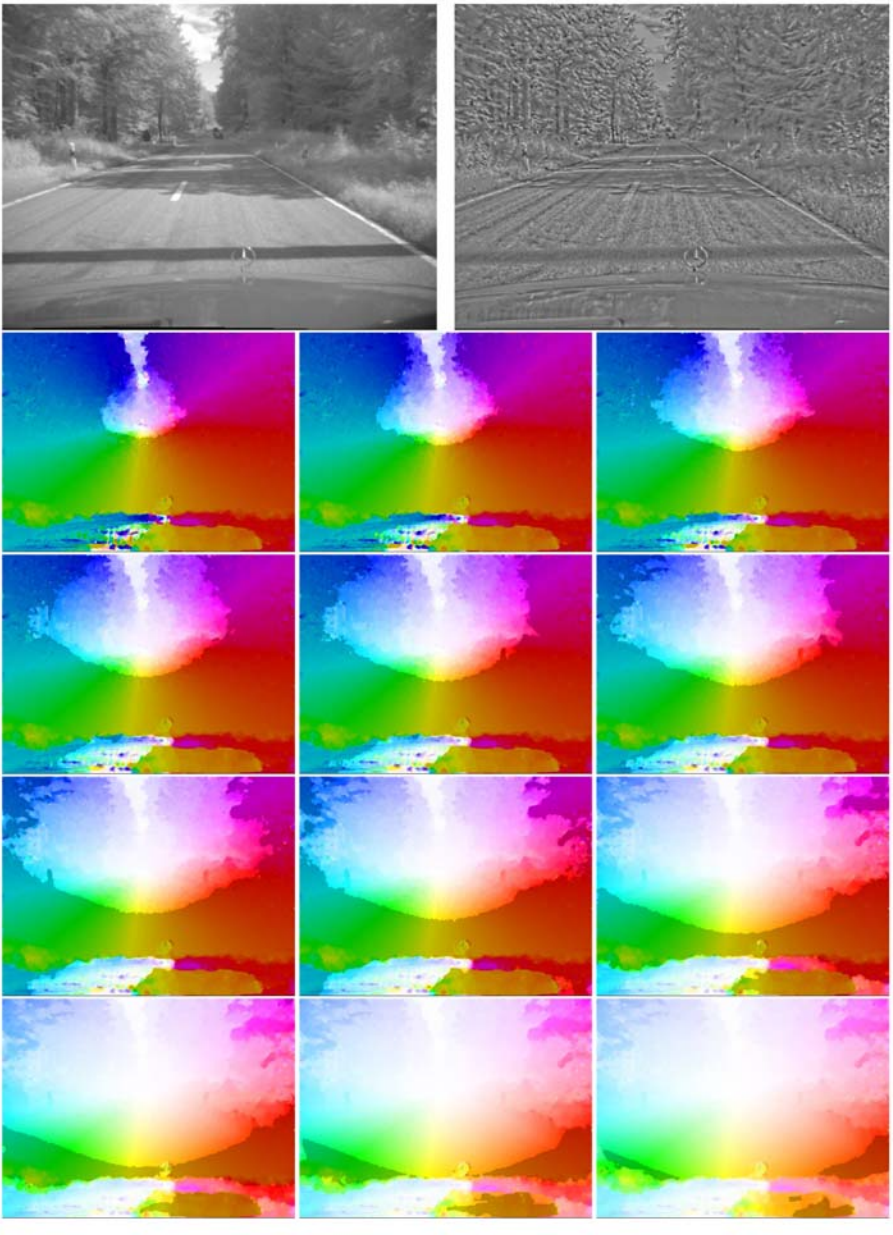


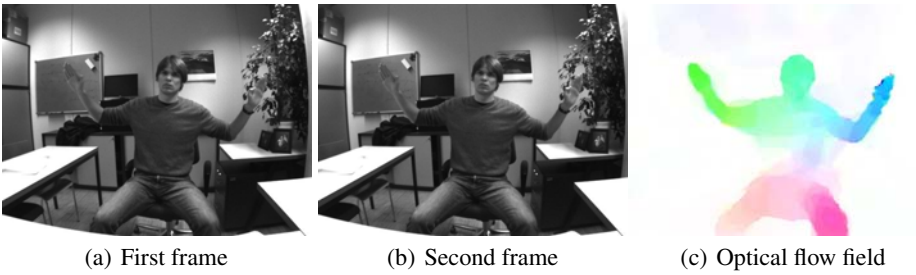
Fig. 12. Optical flow field for the scene depicted in the upper left with the original and structure-texture image. The flow is saturated for flow vector length above $1, 2, 3, 4, 5, 6, 8, 10, 15, 20, 25, 30$ pixels from left to right.

5.3 Real-Time Optical Flow

We provide timing results for our optical flow approach depicted in Table 3. We used a standard personal computer equipped with a 2.13 GHz Core2-Duo CPU, 2 GB of main memory and a NVidia GTX280 graphics card. The computer runs a 64-bit Linux operating system and we used a recent NVidia graphics driver. The timings in Table 3 are given in frames per second for the depicted fixed number of outer iterations on each level of the image pyramid. We used one warp on each level, the number of fixed point steps was set to 5. The measured timings include the image uploads to video memory and the final visualization of the obtained displacement field. The timing results indicate, that real-time performance of 32 frames per second can be achieved at a resolution of 512×512 pixels. Frames from a live video demo application are shown in Figure 13, which continuously reads images from a firewire camera and visualizes the optical flow for consecutive frames. Note that the entire algorithm (including the building of the image pyramids) is executed on the GPU. The only part of the host computer is to upload the images on the GPU. In [11] Bruhn et al. obtained a performance of about 12 frames per second for 160×120 images and a variational model very similar to our TV-L^1 model. From this we see that our approach is about 26 times faster. However we should also note that their approach is computed on the CPU.

Table 3. Observed frame rates at different image resolutions and with varying number of outer iterations on our tested hardware

Graphics Card: NVidia GTX280			
Image resolution	25 Iterations	50 Iterations	100 Iterations
128×128	153	81	42
256×256	82	44	23
512×512	32	17	9



(a) First frame (b) Second frame (c) Optical flow field

Fig. 13. Captured frames and generated optical flow field using our live video application. The image resolution is 640×480.The performance is about 30 frames per second in this setting.

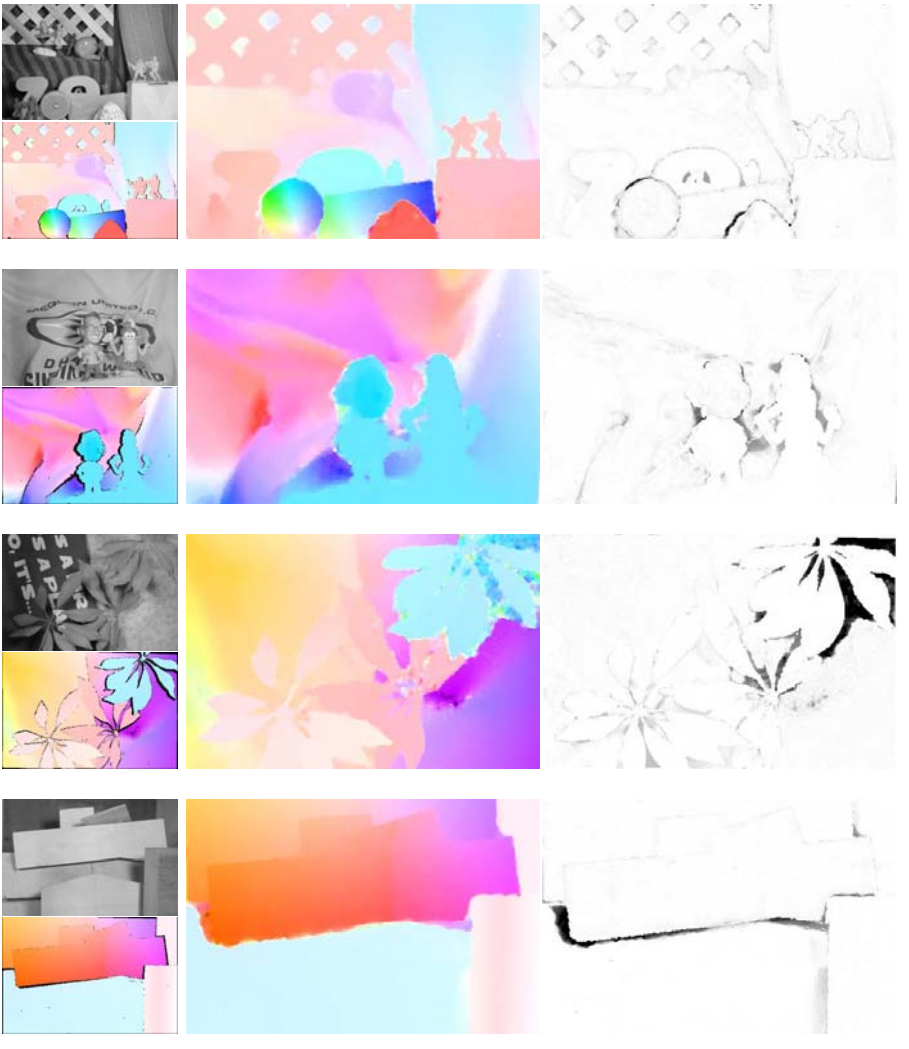


Fig. 14. Optical flow results for the *army*, *mequon*, *schefflera*, and *wooden* sequence of the Middlebury flow benchmark. The left images show the first input image and the ground truth flow. The middle image shows the optical flow using the proposed algorithm. The right image shows the end point error of the flow vector, where black corresponds to large errors.

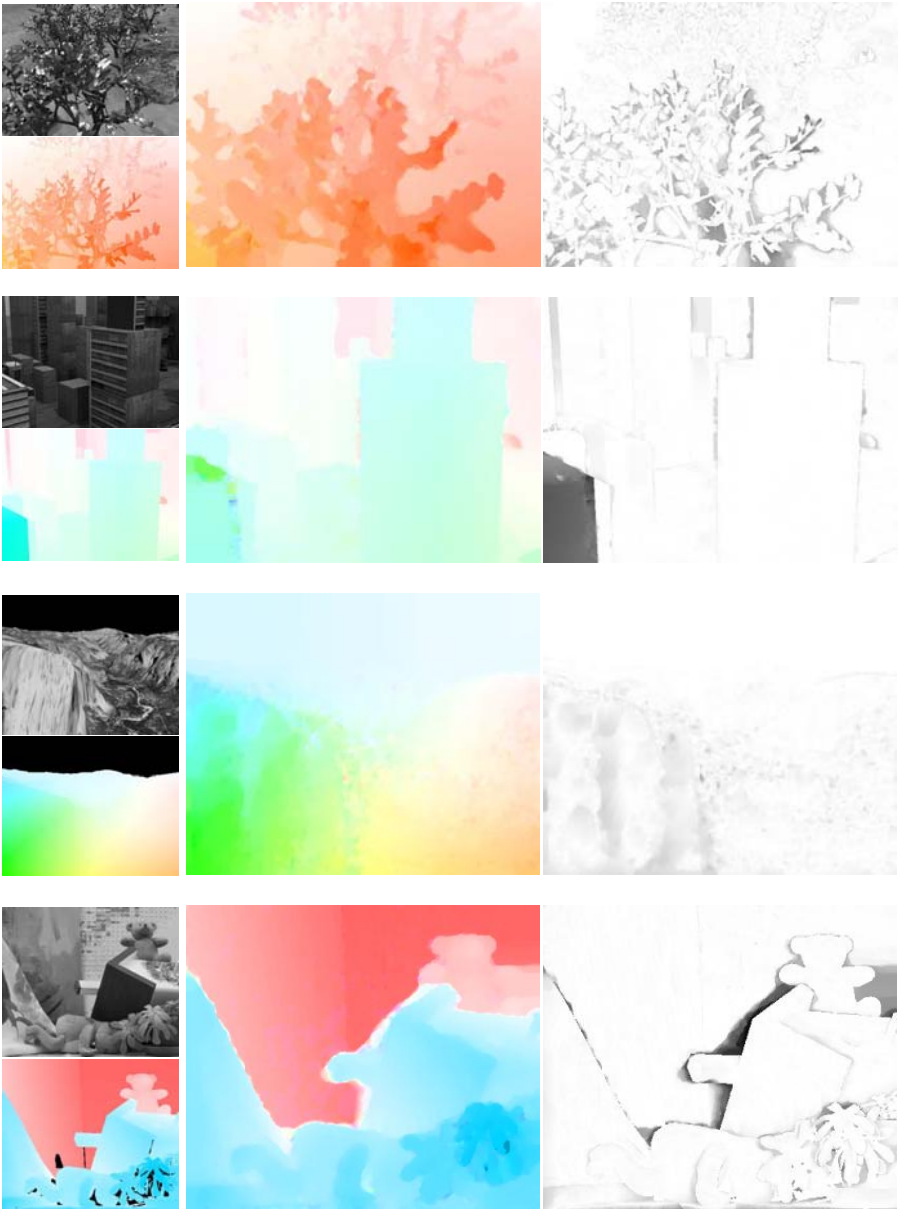


Fig. 15. (continued) Optical flow results for the *grove*, *urban*, *yosemite*, and *teddy* sequence of the Middlebury flow benchmark. The left images show the first input image and the ground truth flow. The middle image shows the optical flow using the proposed algorithm. The right image shows the end point error of the flow vector, where black corresponds to large errors.

6 Conclusion and Future Research

We have proposed an improved algorithm for the TV-L^1 optical flow method of [31]. We improved the core algorithm using blended versions of the image gradients and a median filter to reject flow outliers. In this paper, the numerical scheme was outlined which can efficiently be implemented on either CPUs or modern graphics processing units. We gave implementation details and showed that the proposed improvements increase the accuracy of the optical flow estimation.

We additionally proposed to use a blended version of the structure-texture decomposition, originally proposed for optical flow computation in [26]. The decomposition of the input image into its structural and textural parts allows to minimize illumination artifacts due to shadows and shading reflections. We showed that this leads to more accurate results in the optical flow computation.

Our improved algorithm for TV-L^1 optical flow was evaluated on the Middlebury optical flow benchmark, showing state-of-the-art performance. Our proposed algorithm is solely based on the image intensities in terms of gray values. Future work includes the extension of our approach to handle color images as well.

An interesting research area is the extension of the proposed optical flow algorithms to use multiple data terms. One direct application is the computation of scene flow, incorporating three data terms [28]. We are currently investigating extensions of the proposed optical flow method to adopt it to this stereo scene flow case.

The edge preserving nature of total variation can be enhanced, if a suitable weighted TV-norm/active contour model is applied [8]. Future work will address the incorporation of such feature for stereo and optical flow estimation.

References

1. Alvarez, L., Weickert, J., Sánchez, J.: A scale-space approach to nonlocal optical flow calculations. In: Proceedings of the Second International Conference on Scale-Space Theories in Computer Vision, pp. 235–246 (1999)
2. Anandan, P.: A computational framework and an algorithm for the measurement of visual motion. Int. J. Comput. Vision 2, 283–310 (1989)
3. Aubert, G., Deriche, R., Kornprobst, P.: Computing optical flow via variational techniques. SIAM J. Appl. Math. 60(1), 156–182 (1999)
4. Aujol, J.-F., Gilboa, G., Chan, T., Osher, S.: Structure-texture image decomposition–modeling, algorithms, and parameter selection. Int. J. Comput. Vision 67(1), 111–136 (2006)
5. Baker, S., Scharstein, D., Lewis, J.P., Roth, S., Black, M.J., Szeliski, R.: A Database and Evaluation Methodology for Optical Flow. In: ICCV 1993, pp. 1–8 (2007)
6. Birchfield, S.: Derivation of Kanade-Lucas-Tomasi Tracking Equation. Technical Report (1997)
7. Black, M.J., Anandan, P.: A framework for the robust estimation of optical flow. In: ICCV 1993, pp. 231–236 (1993)
8. Bresson, X., Esedoglu, S., Vandergheynst, P., Thiran, J., Osher, S.: Fast Global Minimization of the Active Contour/Snake Model. Journal of Mathematical Imaging and Vision (2007)
9. Brox, T., Bruhn, A., Papenberg, N., Weickert, J.: High accuracy optical flow estimation based on a theory for warping. In: Pajdla, T., Matas, J(G.) (eds.) ECCV 2004. LNCS, vol. 3024, pp. 25–36. Springer, Heidelberg (2004)

10. Bruhn, A., Weickert, J., Feddern, C., Kohlberger, T., Schnörr, C.: Variational optical flow computation in real time. IEEE Transactions on Image Processing 14(5), 608–615 (2005)
11. Bruhn, A., Weickert, J., Kohlberger, T., Schnörr, C.: A multigrid platform for real-time motion computation with discontinuity-preserving variational methods. Int. J. Comput. Vision 70(3), 257–277 (2006)
12. Camus, T.A.: Real-time quantized optical flow. Journal of Real-Time Imaging 3, 71–86 (1997); Special Issue on Real-Time Motion Analysis
13. Chambolle, A.: An algorithm for total variation minimization and applications. Journal of Mathematical Imaging and Vision 20(1–2), 89–97 (2004)
14. Chambolle, A.: Total variation minimization and a class of binary MRF models. Energy Minimization Methods in Computer Vision and Pattern Recognition, 136–152 (2005)
15. Chan, T.F., Golub, G.H., Mulet, P.: A nonlinear primal-dual method for total variation-based image restoration. In: ICAOS 1996, Paris, vol. 219, pp. 241–252 (1996)
16. Devillard, N.: Fast median search: an ANSI C implementation (1998), http://www.eso.org/ndevilla/median/ (visited October 2008)
17. Haussecker, H., Fleet, D.J.: Estimating Optical Flow with Physical Models of Brightness Variation. IEEE Transactions on Pattern Analysis and Machine Intelligence 23(6), 661–674 (2001)
18. Horn, B.K.P., Schunck, B.G.: Determining optical flow. Artificial Intelligence 17, 185–203 (1981)
19. Mémin, E., Pérez, P.: Hierarchical estimation and segmentation of dense motion fields. Int. J. Comput. Vision 46(2), 129–155 (2002)
20. Mileva, Y., Bruhn, A., Weickert, J.: Illumination-robust Variational Optical Flow with Photometric Invariants. In: Hamprecht, F.A., Schnörr, C., Jähne, B. (eds.) DAGM 2007. LNCS, vol. 4713, pp. 152–162. Springer, Heidelberg (2007)
21. Nagel, H.-H., Enkelmann, W.: An investigation of smoothness constraints for the estimation of displacement vector fields from image sequences. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI) 8, 565–593 (1986)
22. Papenberg, N., Bruhn, A., Brox, T., Didas, S., Weickert, J.: Highly accurate optic flow computation with theoretically justified warping. Int. J. Comput. Vision, 141–158 (2006)
23. Rudin, L.I., Osher, S., Fatemi, E.: Nonlinear total variation based noise removal algorithms. Physica D 60, 259–268 (1992)
24. Stewart, E.: Intel Integrated Performance Primitives: How to Optimize Software Applications Using Intel IPP. Intel Press (2004) ISBN 0971786135
25. Strzodka, R., Garbe, C.: Real-time motion estimation and visualization on graphics cards. In: IEEE Visualization 2004, pp. 545–552 (2004)
26. Trobin, W., Pock, T., Cremers, D., Bischof, H.: An Unbiased Second-Order Prior for High-Accuracy Motion Estimation. In: Rigoll, G. (ed.) DAGM 2008. LNCS, vol. 5096, pp. 396–405. Springer, Heidelberg (2008)
27. Wedel, A., Pock, T., Braun, J., Franke, U., Cremers, D.: Duality TV-L1 flow with fundamental matrix prior. Image and Vision Computing New Zealand (November 2008)
28. Wedel, A., Rabe, C., Vaudrey, T., Brox, T., Franke, U., Cremers, D.: Efficient Dense Scene Flow from Sparse or Dense Stereo Data. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part I. LNCS, vol. 5302, pp. 739–751. Springer, Heidelberg (2008)
29. Weickert, J., Brox, T.: Diffusion and regularization of vector- and matrix-valued images. Inverse Problems, Image Analysis and Medical Imaging. Contemporary Mathematics 313, 251–268 (2002)
30. van de Weijer, J., Gevers, T.: Robust Pptical Flow from Photometric Invariants. In: International Conference on Image Processing (2004)
31. Zach, C., Pock, T., Bischof, H.: A Duality Based Approach for Realtime TV-L1 Optical Flow. In: Hamprecht, F.A., Schnörr, C., Jähne, B. (eds.) DAGM 2007. LNCS, vol. 4713, pp. 214–223. Springer, Heidelberg (2007)

An Evaluation Approach for Scene Flow with Decoupled Motion and Position

Andreas Wedel[1,2], Tobi Vaudrey[3], Annemarie Meissner[1],
Clemens Rabe[1], Thomas Brox[4], Uwe Franke[1], and Daniel Cremers[2]

[1] Daimler Group Research
[2] University of Bonn
[3] University of Auckland
[4] U.C. Berkeley (University of California)

Abstract. This chapter presents a technique for estimating the three-dimensional displacement vector field that describes the motion of each visible scene point. This displacement field consists of the change in image position, also known as optical flow, and the change in disparity and is called the scene flow. The technique presented uses two consecutive image pairs from a stereo sequence. The main contribution is to decouple the disparity (3D position) and the scene flow (optical flow and disparity change) estimation steps. Thus we present a technique to estimate a dense scene flow field using a variational approach from the gray value images and a given stereo disparity map.

Although the two subproblems disparity and scene flow estimation are decoupled, we enforce the scene flow to yield consistent displacement vectors in all four stereo images involved at two time instances. The decoupling strategy has two benefits: Firstly, we are independent in choosing a disparity estimation technique, which can yield either sparse or dense correspondences, and secondly, we can achieve frame rates of 5 fps on standard consumer hardware. This approach is then expanded to real-world displacements, and two metrics are presented that define likelihoods of movement with respect to the background. Furthermore, an evaluation approach is presented to compare scene flow algorithms on long image sequences, using synthetic data as ground truth.

1 Introduction

One of the most important features to extract in image sequences from a dynamic environment is the motion of objects within the scene. Humans perform this using a process called visual kinisthesia, this encompasses both the perception of movement of objects in the scene and also the observers own movement. Perceiving this using vision based methods can prove to be difficult and is an ill-posed problem. Images from a single camera are not well constrained. Only two dimensional motion can be estimated from sequential images, this is referred to as optical flow. This two-dimensional motion is the true three-dimensional scene motion, projected onto the image plane. During this projection process, one dimension is lost and cannot be recovered without additional constraints or information. Hence, one may say that motion computation using a single camera is not well constrained.

D. Cremers et al. (Eds.): Visual Motion Analysis, LNCS 5604, pp. 46–69, 2009.

This is a different story, once a stereo camera system is available. The distance estimate from the triangulation of stereo correspondences provides the necessary additional information needed to reconstruct the three-dimensional scene motion. Then, ambiguities only arise

... if the camera motion is not known (in particular the camera is not stationary). Then the motion of the scene involves two primary parts; the motion of dynamic objects within the scene and the motion of the static background from the motion of the camera.

... around areas with missing structure in a local neighborhood (i. e. this leads to the aperture problem).

The ambiguity between motion induced by camera motion and dynamic objects can be solved if the ego-motion of the camera is known. A common way to deal with missing structure and to achieve dense estimates is, similarly to the two-dimensional optical flow estimation, the use of variational approaches that incorporate a smoothing function.

For 3D motion, the use of a variational approach has been used to achieve dense estimates [8]. However, only visible points can be tracked, so we refer to *dense scene flow* as the 3D image displacement field for 3D points that can be seen by both cameras (i.e., omitting occlusions). This defines scene flow as displacement and position estimates in image coordinates (pixels). The scene flow is then translated into real-world coordinates to become the motion of objects in the scene. See Figure 1 for an example, where the motion of a preceding vehicle becomes visible in the 3D scene flow.

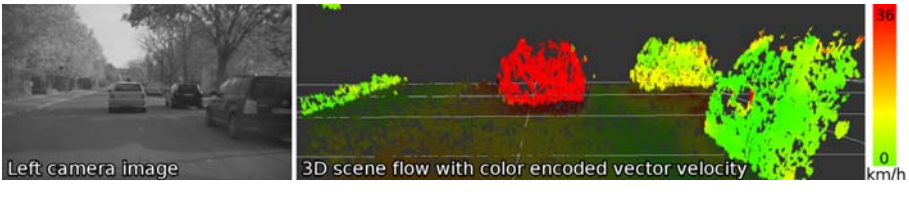


Fig. 1. Scene flow example. Despite similar distance from the viewer, the moving car (red) can be clearly distinguished from the parked vehicles (green).

1.1 Related Work

2D motion vectors are obtained by optical flow estimation techniques. There are dense as well as sparse techniques. Sparse optical flow techniques, such as KLT tracking [18], usually perform some kind of feature tracking and are preferred in time-critical applications, due to computational benefits. Dense optical flow is mostly provided by variational models based on the method of Horn and Schunck [7]. Local variational optimisation is used to minimise an energy function that assumes constant pixel intensities and a smooth flow field. The basic framework of Horn and Schunck has been improved over time to cope with discontinuities in the flow field, and to obtain robust solutions with the presence of outliers in image intensities [11]. Furthermore, larger displacements can

be estimated thanks to image warping and non-linearised model equations [2,11]. Currently, variational techniques yield the most accurate optical flow in the literature [22]. Real-time methods have been proposed in [3,24].

Scene flow involves an additional disparity estimation problem, as well as the task to estimate the change of disparity over time. The work in [13] introduced scene flow as a joint motion and disparity estimation method. The succeeding works in [8,12,25] presented energy minimisation frameworks including regularisation constraints to provide dense scene flow. Other dense scene flow algorithms have been presented in multiple camera set-ups [14,21]. However, these allow for non-consistent flow fields in single image pairs.

None of the above approaches run in real-time, giving best performances in the scale of minutes. Real-time scene flow algorithms, such as the one presented in [15], provide only sparse results both for the disparity and the displacement estimates. The work in [9] presents a probabilistic scene flow algorithm with computation times in the range of seconds, but yielding only integer pixel-accurate results. In contrast, the method we present in the following provides sub-pixel accurate scene flow close to real-time for reasonable image sizes.

1.2 Contribution

Combining both the position estimation and motion estimation into one framework, and has been the common approach for scene flow (e.g. [25,13,8]). In this chapter, we propose to decouple the motion estimation from the position estimation, while still maintaining the disparity constraints. The decoupling of depth (disparity) and motion (optical flow and disparity change) estimation might look unfavorable at a first glance; but it has at least two important advantages:

Firstly, the challenges in motion estimation and disparity estimation are quite different. With disparity estimation, thanks to the epipolar constraint, only a scalar field needs to be estimated. This enables the use of (semi-) global optimization methods, such as dynamic programming or graph-cuts, to establish point correspondences. Optical flow estimation, on the other hand, requires the estimation of a vector field, which rules out such (semi-) global optimization strategies. Additionally, motion vectors are usually smaller in magnitude than disparities. With optical flow, occlusion handling is less important than the sub-pixel accuracy provided by variational methods.

> *Splitting scene flow computation into the estimation sub-problems,* disparity *and* optical flow with disparity change*, allows one to choose the optimal technique for each task.*

At this point it is worth noting that, although the problems of disparity estimation and motion estimation are separated, the here presented method still involves a coupling of these two tasks, as the optical flow is enforced to be consistent with the computed disparities.

Secondly, the two sub-problems can be solved more efficiently than the joint problem. This allows for real-time computation of scene flow, with a frame rate of 5 fps on QVGA images (320×240 pixel, assuming the disparity map is given).

The splitting approach to scene flow is about 500 times faster compared to recent techniques for joint scene flow computation.

Nevertheless, an accuracy that is at least as good as the joint estimation method is achieved on test sequences. Furthermore, in combination with a dense disparity map the scene flow field and its corresponding world flow field are dense.

This chapter is organized as follows. In Section 2, we present how to calculate the scene flow from two consecutive stereo image pairs. The scene flow can then be used to calculate likelihoods that voxels are moving; we present two such approaches for statistical analysis in Section 3.

In our final section, an evaluation method is presented as a useful way to compare and contrast scene flow algorithms, using a 400 frames long, synthetically generated stereo image sequence. Along with this evaluation method, results are presented on real-world data to validate the usefulness of the algorithm.

2 Formulation of Scene Flow

This section explains the formulation of the scene flow algorithm. It identifies how the decoupling of position and motion is put to use effectively, while still maintaining the stereo disparity constraint in both stereo image pairs.

Figure 2 outlines the approach. As seen from this figure, the stereo images pair is needed for both, the previous and current time frame. The derived approach also requires the disparity map from the previous time frame. This information is passed to the scene flow algorithm for processing, to produce the optical flow and the change in disparity between the image pairs.

2.1 Stereo Computation

Our algorithm requires a pre-computed disparity map. Current state-of-the-art algorithms (e.g., see Middlebury [16]) require normal stereo epipolar geometry, such that

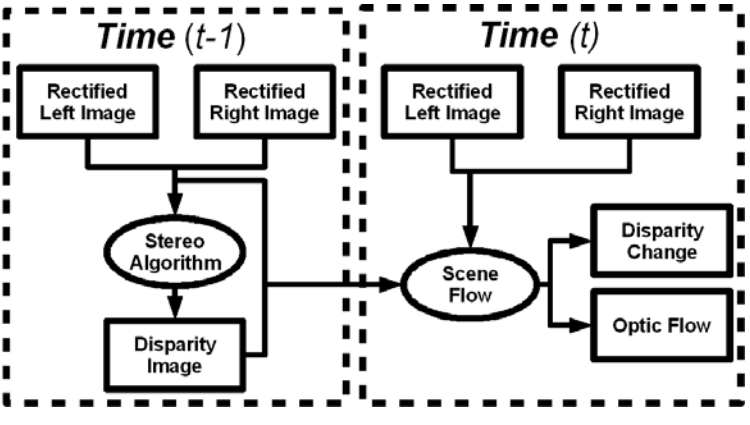


Fig. 2. Outline of our novel scene flow algorithm

pixel rows y for the left and right images coincide. In practice, this is achieved by undistorting the camera images and warping the images via rectification (using the relative extrinsic configuration of the two involved cameras) e.g., as outlined in Hartley and Zissermann [5].

In addition, the principle points of both images are rearranged, such that they lie on the same image coordinates $[x_0, y_0]^\top$ (columns, rows).

A world point ($[X, Y, Z]^\top$ lateral, vertical and depth respectively) is projected into the cameras images, yielding $[x, y]^\top$ in the left image and $[x+d, y]^\top$ in the right image, according to:

$$\begin{pmatrix} x \\ y \\ d \end{pmatrix} = \frac{1}{Z} \begin{pmatrix} X f_x \\ Y f_y \\ -b f_x \end{pmatrix} + \begin{pmatrix} x_0 \\ y_0 \\ 0 \end{pmatrix} \tag{1.1}$$

with the focal lengths f_x and f_y (in pixels) for the x and y direction and the baseline distance between the two camera projection centres b (in metres). The disparity value d therefore encodes the difference in the x-coordinate of an image correspondence between the left and right image. With known camera intrinsic parameters, the position of a world point can easily be recovered from an (x, y, d) measurement using Equation (1.1).

The goal of the stereo correspondence algorithm is to estimate the disparity d, for every non-occluded pixel in the left image. This is accomplished by local methods (using a small matching window from the left image to the right image) or global methods (incorporating some global energy minimisation). The disparity information can then be used to reconstruct the 3D scene.

The scene flow algorithm presented in this chapter has the flexibility to be able to use any disparity map as input. Dense or sparse algorithms are handled effectively due to the variational nature of the approach.

Hierarchical correlation algorithms, yielding sparse sub-pixel accuracy disparity maps, are commonly used due to their real-time capability. We use a typical implementation described in [4], which allows disparity computation at about 100 Hz. In addition to this, an implementation based on the algorithm described in [17] is used. It computes sparse pixel-discrete disparity maps using Census based hash tables, and is massively parallel so available in hardware (FPGA or ASIC) without extra computational cost.

Using globally consistent energy minimisation techniques, it becomes possible to compute dense disparity maps, which yield a disparity value for every non-occluded pixel. We use semi-global matching with mutual information [6]. The algorithm is implemented on dedicated hardware and runs at 30 Hz on images with a resolution of 640×480 pixels.

To demonstrate that the derived scene flow algorithm is able to use different disparity estimation techniques as input, Section 4 shows results with the aforementioned sparse and dense stereo algorithms.

2.2 Stereo Motion Constraints

The basic outline of our algorithm is shown in Figure 3. We use two consecutive pairs of stereo images at time $t-1$ and t. Similar to the optical flow field, the scene flow

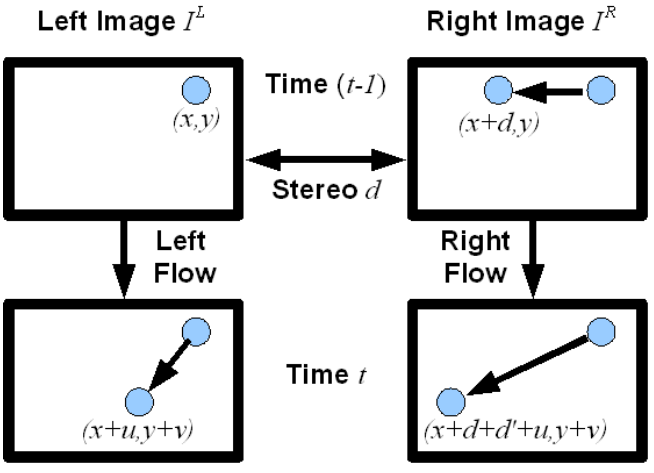


Fig. 3. Motion and disparity constraints employed in our scene flow algorithm

field (three-dimensional world-coordinate flow field) is projected into the image space $[u, v, d']^\top$. u and v is the change in image x and y (in pixels) respectively, and d' is the change in disparity (in pixels).

The three-dimensional scene flow field can only be reconstructed, when both the image position position $[x, y, d]^\top$ and its temporal change $[u, v, d']^\top$ are known. d is estimated using an arbitrary stereo algorithm, see Section 2.1. The disparity change and the two-dimensional optical flow field have to be estimated from the stereo image pairs.

The first equation that we derive is from the left half of Figure 3. This equation is the normal optical flow intensity constraint, i.e., the intensity should be the same in both images for the same world point in the scene. Let $I(x, y, t)^L$ be the intensity value of the left image, at pixel position $[x, y]^\top$ and time t. This leads to the following constraint:

$$I(x, y, t-1)^L = I(x+u, y+v, t)^L \tag{1.2}$$

The flow in the right hand image can also be derived using the same principle (See right half of Figure 3). Let $I(x, y, t)^R$ be the intensity of the right image, at pixel position $[x, y]^\top$ and time t. Due to rectification, we know that flow in the left image and right image will have the same y component, this leads to the difference being only in the x component of the equation. This leads to:

$$I(x+d, y, t-1)^R = I(x+d+d'+u, y+v, t)^R \tag{1.3}$$

highlighting that the flow in the x component is only different by the disparity d and the disparity change d'.

Calculating optical flow in the left and right image separately, we could derive the disparity change $d' = u^R - u^L$, where u^R and u^L denote the estimated flow fields in the left and right image, respectively. However, to estimate the disparity change more accurate, consistency of the left and right image at time t is enforced. More precisely,

the gray values of corresponding pixels in the stereo image pair at time t should be equal (as seen in both bottom halve of the diagram in Figure 3). This yields the third constraint,

$$I(x+u, y+v, t)^L = I(x+d+d'+u, y+v, t)^R \tag{1.4}$$

Figure 4 shows a summary of the above equations. If we rearrange the above equations, it results in:

$$\begin{aligned} E_{LF} &:= I(x+u, y+v, t)^L - I(x, y, t-1)^L = 0 \\ E_{RF} &:= I(x+d+d'+u, y+v, t)^R - I(x+d, y, t-1)^R = 0 \\ E_{DF} &:= I(x+d+d'+u, y+v, t)^R - I(x+u, y+v, t)^L = 0 \end{aligned} \tag{1.5}$$

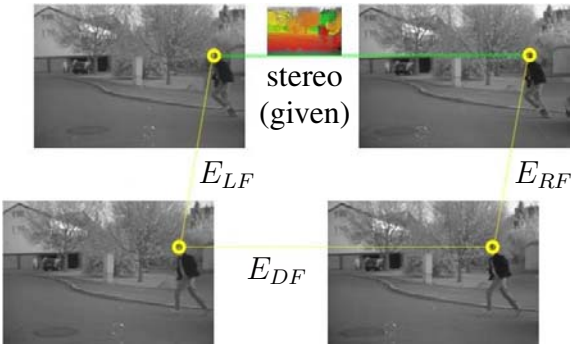


Fig. 4. The scene flow equations (1.5) are illustrated for two stereo image pairs

Occlusion handling is an important aspect in scene flow estimation. It increases computational costs, but clearly improves results. Regarding the influence of occlusion handling on disparity estimation and displacement estimation, we find that disparities are generally much larger than the magnitude of the optical flow and the disparity change, which is only a few pixels. Accordingly, occlusion handling is much more decisive for disparity estimation than for motion estimation.

From a practical point of view, the effort in occlusion handling should be set in comparison to the gained accuracy in the scene flow result. Hence, we do not explicitly model occlusion handling for the scene flow and simply discard occluded areas (using left-right consistency checks) and areas with no information identified by the stereo estimation algorithm. More precisely, for pixels with no valid disparity value, Equations 1.3 and 1.4 will not be evaluated. Such procedure implicitly enables the use of sparse disparity maps.

2.3 Energy Equations for Scene Flow

Scene flow estimates according to the constraints formulated in Section 2 are computed in a variational framework by minimising an energy functional consisting of a data term (derived from constraints) and a smoothness term that enforces smooth and dense

scene flow. By integrating over the image domain Ω we obtain an energy for scene flow, defined as:

$$E_{\mathrm{SF}} = \int_{\Omega} \Big(E_{\mathrm{D}} + E_{\mathrm{S}} \Big) dx\, dy \tag{1.6}$$

where E_{D} and E_{S} are the data and smoothness contributions respectively. By using the constraints from Equation 1.5 we obtain the following data term:

$$E_{\mathrm{D}} = \Psi\left(E_{LF}^2\right) + c(x,y)\,\Psi\left(E_{RF}^2\right) + c(x,y)\,\Psi\left(E_{DF}^2\right) \tag{1.7}$$

where $\Psi(s^2) = \sqrt{s^2 + \varepsilon}$ ($\varepsilon = 0.0001$) denotes a robust function that compensates for outliers [2] and the function $c(x,y)$ returns 0 if there is no disparity known at $[x,y]^\top$ (due to occlusion or sparse stereo method), or 1 otherwise.

The smoothness term penalises local deviations in the scene flow components and employs the same robust function as the data term in order to deal with discontinuities in the scene flow field:

$$E_{\mathrm{S}} = \lambda\,\Psi\left(|\nabla u|^2\right) + \lambda\,\Psi\left(|\nabla v|^2\right) + \gamma\,\Psi\left(|\nabla d'|^2\right) \tag{1.8}$$

where $\nabla = (\partial/\partial x, \partial/\partial y)^\top$. The parameters λ and γ regulate the importance of the smoothness constraint, weighting for optic flow and disparity change respectively. Interestingly, due to the fill-in effect of the above regularisation, the proposed variational formulation provides dense image flow estimates $[u, v, d']^\top$, even if the disparity d is non-dense.

2.4 Minimisation of the Energy

For minimising the above energy we compute its Euler-Lagrange equations:

$$\begin{aligned}\Psi'(E_{LF}^2)E_{LF}I_x^L + c\Psi'(E_{RF}^2)E_{RF}I_x^R + c\Psi'(E_{DF}^2)E_{DF}I_x^R \\ -\lambda\,\mathrm{div}\left(\nabla u\, E'_{\mathrm{S}}\right) = 0\end{aligned} \tag{1.9}$$

$$\begin{aligned}\Psi'(E_{LF}^2)E_{LF}I_y^L + c\Psi'(E_{RF}^2)E_{RF}I_y^R + c\Psi'(E_{DF}^2)E_{DF}I_y^R \\ -\lambda\,\mathrm{div}\left(\nabla v\, E'_{\mathrm{S}}\right) = 0\end{aligned} \tag{1.10}$$

$$c\Psi'(E_{RF}^2)E_{RF}I_x^r + c\Psi'(E_{DF}^2)E_{DF}I_x^r - \gamma\,\mathrm{div}\left(\nabla d'\, E'_{\mathrm{S}}\right) = 0 \tag{1.11}$$

where $\Psi'(s^2)$ denotes the derivative of Ψ with respect to s^2. Partial derivatives of $I(x+u, y+v, t)^L$ and $I(x_d + d' + u, y+v, t)^R$ are denoted by subscripts x and y. For simplicity, $c = c(x,y)$. Also, $E'_{\mathrm{S}} = E_{\mathrm{S}}$ with all Ψ functionals being replaced with Ψ'. These equations are non-linear in the unknowns, so we stick to the strategy of two nested fixed point iteration loops as suggested in [2]. This comes down to a warping scheme as also employed in [11]. The basic idea is to have an outer fixed point iteration loop that contains the linearisation of the E_{LF}, E_{RF}, and E_{DF}.

In each iteration, an increment of the unknowns is estimated and the second image is then warped according to the new estimate. The warping is combined with a coarse-to-fine strategy, where we work with down-sampled images that are successively refined with the number of iterations. Since we are interested in real-time estimates, we use only 4 scales with 2 outer fixed point iterations at each scale.

In the present case, we have the following linearisation, where k denotes the iteration index. We start the iterations with $[u^0, v^0, d'^0]^\top = [0, 0, 0]^\top$ for all $[x, y]^\top$:

$$I(x + u^k + \delta u^k, y + v^k + \delta v^k, t)^L \approx I(x + u^k, y + v^k, t)^L + \delta u^k I_x^L + \delta v^k I_y^L \quad (1.12)$$

$$\begin{aligned} &I(x + d + d'^k + \delta d'^k + u^k + \delta u^k, y + v^k + \delta v^k, t)^R \\ &\quad \approx I(x + d + d'^k + u^k, y + v^k, t)^R + \delta u^k I_{(x+d)}^R + \delta d'^k I_{(x+d)}^R + \delta v^k I_y^R \end{aligned} \quad (1.13)$$

From these expressions we can derive linearised versions of E_{LF}, E_{RF}, and E_{DF}. The remaining non-linearity in the Euler-Lagrange equations is due to the robust function. In the inner fixed point iteration loop the Ψ' expressions are kept constant and are recomputed after each iteration. This finally leads to the following linear equations:

$$\begin{aligned} &\Psi'((E_{LF}^{k+1})^2)(E_{LF}^k + I_x^{L,k}\delta u^k + I_y^{L,k}\delta v^k)I_x^{L,k} \\ &+ c\Psi'((E_{RF}^{k+1})^2)(E_{RF}^k + I_x^{R,k}(\delta u^k + \delta d'^k) + I_y^{R,k}\delta v^k)I_x^{R,k} \\ &\qquad - \lambda \operatorname{div}\left(E_{\mathrm{S}}^{',k+1}\nabla(u^k + \delta u^k)\right) = 0 \end{aligned} \quad (1.14)$$

$$\begin{aligned} &\Psi'((E_{LF}^{k+1})^2)(E_{LF}^k + I_x^{L,k}\delta u^k + I_y^{L,k}\delta v^k)I_y^{l,k} \\ &+ c\Psi'((E_{RF}^{k+1})^2)(E_{RF}^k + I_x^{R,k}(\delta u^k + \delta d'^k) + I_y^{R,k}\delta v^k)I_y^{R,k} \\ &\qquad - \lambda \operatorname{div}\left(E_{\mathrm{S}}^{',k+1}\nabla(v^k + \delta v^k)\right) = 0 \end{aligned} \quad (1.15)$$

$$\begin{aligned} &c\Psi'((E_{RF}^{k+1})^2)(E_{RF}^k + I_x^{R,k}(\delta u^k + \delta d'^k) + I_y^{R,k}\delta v^k)I_x^{R,k} \\ &\qquad + c\Psi'((E_{DF}^{k+1})^2)(E_{DF}^k + I_x^{R,k}\delta d'^k)I_x^{R,k} \\ &\qquad\qquad - \gamma \operatorname{div}\left(E_{\mathrm{S}}^{',k+1}\nabla(d'^k + \delta d'^k)\right) = 0 \end{aligned} \quad (1.16)$$

where

$$E_{\mathrm{S}}^{',k+1} = \lambda|\nabla u^{k+1}|^2 + \lambda|\nabla v^{k+1}|^2 + \gamma|\nabla d'^{k+1}|^2 \quad (1.17)$$

and the iteration index was omitted from inner fixed point iteration loop to keep the notation uncluttered. Expressions with iteration index $k + 1$ are computed using the current increments $\delta u^k, \delta v^k, \delta d'^k$. Some terms from the original Euler-Lagrange equations have vanished due to the use of $I(x_d, y, t)^R = I(x, y, t)^L$ from the linearised

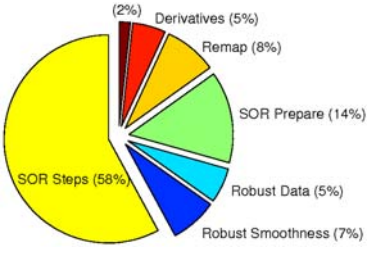


Image Size (pixels)	SOR Steps per Warp	Warps per Level	Total Time
640×480	45	2	975 ms
512×512	20	3	756 ms
320×240	45	2	205 ms

Fig. 5. Break down of computational time for our algorithm (3.0GHz Intel®Core™2). The pie graph shows the time distribution for the 640×480 images. The real-time applicability of the algorithm for image sizes of (320×240) is indicated in the table.

third constraint (Equation 1.4). After discretisation, the corresponding linear system is solved via successive over-relaxation. It is worth noting that, for efficiency reasons, it is advantageous to update the Ψ' after a few iterations of SOR. The shares of computation time taken by the different operations are shown in Figure 5.

3 Analysing Scene Flow

This section proposes methods for evaluating our scene flow algorithm. This involves first taking our scene flow (image coordinates) estimate, then estimating three-dimensional scene flow (real-world coordinates). We also propose two metrics for estimating confidence of moving points within the scene.

3.1 From Image Scene Flow to 3D Scene Flow

We have now derived the image scene flow as a combined estimation of optical flow and disparity change. Using this information, we can compute two world points that define the start and end point of the 3D scene flow. These equations are derived from the inverse of Equation 1.1 (f_x,f_y, and b are defined there as well).

$$X_{t-1} = (x - x_0)\frac{b}{d}, \quad Y_{t-1} = (y - y_0)\frac{f_y}{f_x}\frac{b}{d}, \quad Z_{t-1} = \frac{f_x b}{d} \tag{1.18}$$

and

$$X_t = (x + u - x_0)\frac{b}{d + d'}, \quad Y_t = (y + v - y_0)\frac{f_y}{f_x}\frac{b}{d + d'}, \quad Z_t = \frac{f_x b}{d + d'} \tag{1.19}$$

Obviously, the 3D scene flow $[X', Y', Z']^\top$ is the difference between these two world points. For simplicity we will assume that $f_y = f_x$. This yields:

$$\begin{pmatrix} X' \\ Y' \\ Z' \end{pmatrix} = \begin{pmatrix} X_t - X_{t-1} \\ Y_t - Y_{t-1} \\ Z_t - Z_{t-1} \end{pmatrix} = b \begin{pmatrix} \frac{x-x_0}{d} - \frac{x+u-x_0}{d+d'} \\ \frac{y-y_0}{d} - \frac{y+v-y_0}{d+d'} \\ \frac{f_x}{d} - \frac{f_x}{d+d'} \end{pmatrix} \tag{1.20}$$

For follow-on calculations (e.g., speed, detection of moving objects, segmentation of objects, integration, etc.) we need the accuracy of the scene flow vector. In the following subsections, we will define two metrics that estimate the likelihood that a pixel is moving, i.e., not static. The metrics provided in this section are ideas for follow-on evaluations, but the evaluations themselves are outside the scope of this chapter.

3.2 Motion Metric

First, we define the uncertainty of our measurements. These can be represented by standard deviations as σ_d, σ_u, σ_v, and $\sigma_{d'}$ (subscript denoting variable of standard deviation), one would assume that $\sigma_u = \sigma_v$ and the disparity estimate to be less accurate than the flow. We do not explicitly assume this and derive the covariance matrix $\mathbf{\Sigma_I}$ for the 3D scene flow using error propagation:

$$\mathbf{\Sigma_I} = \mathbf{J}\ \mathrm{diag}(\sigma_d^2, \sigma_u^2, \sigma_v^2, \sigma_{d'}^2)\ \mathbf{J}^\top \tag{1.21}$$

where

$$\mathbf{J} = \begin{pmatrix} \frac{\partial X'}{\partial d} & \frac{\partial X'}{\partial u} & \frac{\partial X'}{\partial v} & \frac{\partial X'}{\partial d'} \\ \frac{\partial Y'}{\partial d} & \frac{\partial Y'}{\partial u} & \frac{\partial Y'}{\partial v} & \frac{\partial Y'}{\partial d'} \\ \frac{\partial Z'}{\partial d} & \frac{\partial Z'}{\partial u} & \frac{\partial Z'}{\partial v} & \frac{\partial Z'}{\partial d'} \end{pmatrix} = b \begin{pmatrix} \left(\frac{(x+u-x_0)}{(d+d')^2} - \frac{(x-x_0)}{d^2}\right) & \frac{-1}{d+d'} & 0 & \frac{(x+u-x_0)}{(d+d')^2} \\ \left(\frac{(y+v-y_0)}{(d+d')^2} - \frac{(y-y_0)}{d^2}\right) & 0 & \frac{-1}{d+d'} & \frac{(y+v-y_0)}{(d+d')^2} \\ \left(\frac{f_x}{(d+d')^2} - \frac{f_x}{d^2}\right) & 0 & 0 & \frac{f_x}{(d+d')^2} \end{pmatrix} \tag{1.22}$$

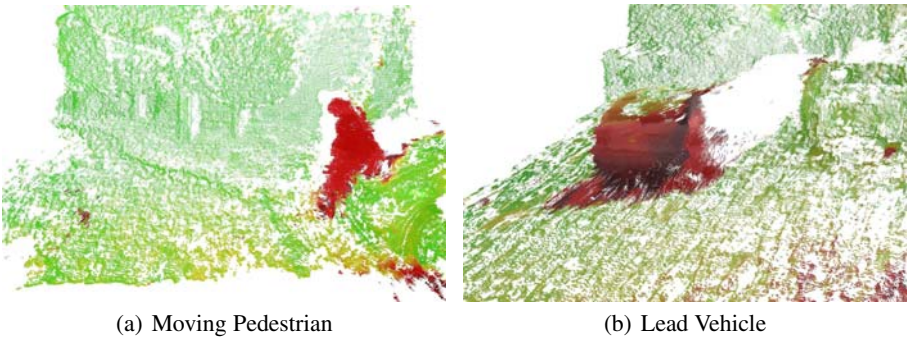
This error propagation holds true, as long as the distribution is zero-mean and scales by the standard deviation (e.g., Gaussian and Laplacian). From this model, one can see that the disparity measurement has the highest influence on the covariance (as it is either by itself or quadratically weighted in the equations). Furthermore, the larger the disparity, the more precise the measurement; as $d \to \infty$ all $\sigma \to 0$.

The derivation above only holds true for stationary cameras. Assume the motion of the camera (in our case vehicle) is given from either inertial sensors or an ego-motion estimation method (e.g., [1]). This motion is composed of a rotation $\mathbf{R}$ (matrix composed by the combination of rotations about the X, Y and Z axis) about the origin of the camera coordinate system and a translation $\mathbf{T} = [T_X, T_Y, T_Z]^\top$. The total motion vector $\mathbf{M}$ is calculated as:

$$\mathbf{M} = \begin{pmatrix} M_X \\ M_Y \\ M_Z \end{pmatrix} = \begin{pmatrix} X_t \\ Y_t \\ Z_t \end{pmatrix} - \mathbf{R} \begin{pmatrix} X_{t-1} \\ Y_{t-1} \\ Z_{t-1} \end{pmatrix} + \mathbf{T} \tag{1.23}$$

Again the motion is known with a certain accuracy. For simplicity we assume the rotational parts to be small, which holds true for most vehicle applications. This approximates the rotation matrix by a unary matrix for the error propagation calculation. We denote the standard deviations of the translations as a three-dimensional covariance matrix $\mathbf{\Sigma_T}$. The total translation vector vector $\mathbf{M}$ now has the covariance matrix $\mathbf{\Sigma_M} = \mathbf{\Sigma_I} + \mathbf{\Sigma_T}$.

Now one can compute the likelihood of a flow vector to be moving, hence belonging to a moving object. Assuming a stationary world and a Gaussian error propagation, one



(a) Moving Pedestrian (b) Lead Vehicle

Fig. 6. Results using the Manhalobis distance metric Q. (a) shows a pedestrian running from behind a vehicle. (b) shows a lead vehicle driving forward. Colour encoding is Q, i.e., the hypothesis that the point is moving, green $\leftrightarrow$ red $\equiv$ low $\leftrightarrow$ high.

expects a standard normal distribution with mean 0 and covariance matrix $\mathbf{\Sigma_M}$. Deviations from this assumption are found by testing this null hypothesis or the goodness of fit. This can be done by evaluating the Mahalanobis distance [10]:

$$Q = \sqrt{\mathbf{M}^\top \mathbf{\Sigma_M^{-1}} \mathbf{M}} \tag{1.24}$$

The squared Mahalanobis distance Q is χ^2 distributed and outliers are found by thresholding, using the assumed quantiles of the χ^2 distribution. For example, the 95% quantile of a distribution with three degrees of freedom is 7.81, the 95% quantile lies at 11.34. Hence a point is moving with a probability of 99% if the Mahalanobis distance is above 11.34. This again holds only if the measurement variances are correct. Figure 6 demonstrates results using this metric. In both images, it is easy to identify what parts of the scene are static, and which parts are moving. The movement metric Q only identifies the probability of a point being stationary, it does not provide any speed estimates. Note that this metric computes a value at every scene point. Another two examples of results obtained using this metric can be seen in Figure 7.

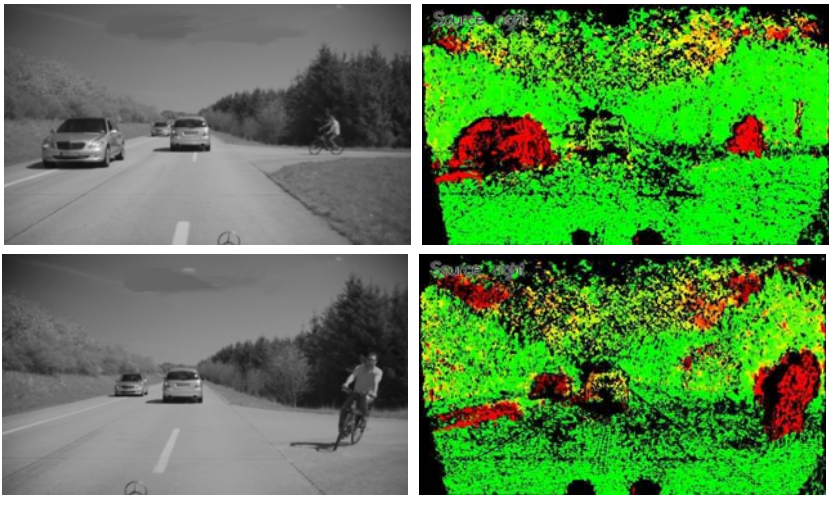
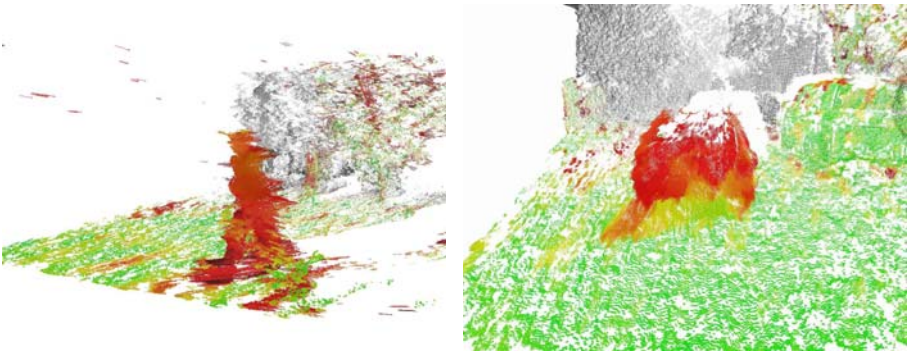


Fig. 7. Two examples of the motion metric defined in Section 3.1. Left images show the original image, right images show the motion metric results, green $\Leftrightarrow$ red represents low $\Leftrightarrow$ high likelihood that point is moving.

3.3 Speed Metric

The Q metric omitted any information about speed. To estimate the speed S the L^2-norm (length) of the displacement vector is calculated.

$$S = |\mathbf{M}| \tag{1.25}$$



(a) Moving Pedestrian (b) Lead Vehicle

Fig. 8. Results using speed S and its standard deviation σ_S. (a) shows the pedestrian running with a speed of 3.75 m/s. (b) shows a lead vehicle driving forward with a speed of 12.5 m/s. Colour encoding is S, green $\leftrightarrow$ red $\equiv$ stationary $\leftrightarrow$ moving. σ_S is encoded using saturation, points in the distance are therefore grey or black.

The problem is that points at large distances are always estimated as moving. This is because a small disparity change yield large displacements in 3D (see Equation (1.20)). If inaccuracies are used in the length computation one can still not derive speed information. One way around the problem is to give a lenient variance of the speed measurement σ_S^2. An approach to estimate this variance is to calculate the *spectral norm* of the covariance matrix. This involves computing the eigenvalues of the squared matrix, then taking the square root of the maximum eigenvalue.

$$\sigma_S^2 = \|\mathbf{\Sigma_M}\| = \sqrt{\lambda_{\max}\left(\mathbf{\Sigma_M^\top \Sigma_M}\right)} \tag{1.26}$$

Using this we now have a speed S and associated variance σ_S^2. Using these metrics leads to the examples in Figure 8. In this figure, it is easy to identify the speed of moving targets, and also how confident we are of the speed measurement. The pedestrian in Figure 8(a) had a displacement of 15 cm with a frame rate of 25 Hz, i.e., 3.75 m/s. The vehicle in 8(b) had a displacement of 50cm, i.e., 12.5 m/s. In both examples only the information with high confidence is taken into account, so moving objects are easily identified.

From the metrics provided in this section, we now have a likelihood that the object is moving Q, estimated speed of the object S and the accuracy of the speed σ_S^2.

4 Evaluation

The first two subsections here present a summary of the original studies of our scene flow algorithm from [23]. The third subsection presents an evaluation approach for long stereo sequences, along with some sample results. Real-world results of the algorithm are provided at the end of the evaluation section.

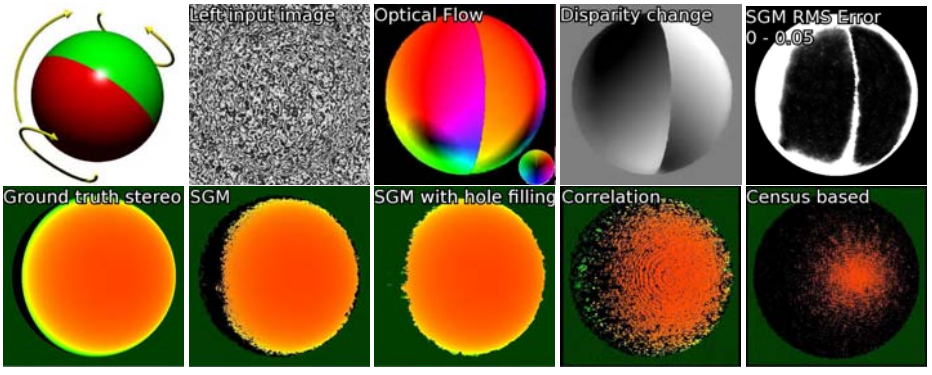


Fig. 9. Ground truth test: *rotating sphere*. Quantitative results are shown in Table 1. **Top:** The left image shows the movement of the sphere. Optical flow and disparity change are computed on the basis of SGM stereo [6]. Colour encodes the direction of the optical flow (key in bottom right), intensity its magnitude. Disparity change is encoded from black (increasing) to white (decreasing). Bright parts of the RMS figure indicate high $RMS_{u,v,d'}$ error values of the computed scene flow. **Bottom:** disparity images are colour encoded green to orange (low to high). Black areas indicate missing disparity estimates or occluded areas.

Table 1. Root mean square (pixels) and average angular error (degrees) for scene flow of the *rotating sphere* sequence. Various stereo algorithms are used as input for our scene flow estimation, generating varying results.

Stereo Algorithm	RMS_d (density)	Without occluded areas			With occluded areas		
		$RMS_{u,v}$	$RMS_{u,v,d'}$	$AAE_{u,v}$	$RMS_{u,v}$	$RMS_{u,v,d'}$	$AAE_{u,v}$
Ground truth	0 (100%)	0.31	0.56	0.91	0.65	2.40	1.40
SGM [6]	2.9 (87%)	0.34	0.63	1.04	0.66	2.45	1.50
Correlation [4]	2.6 (43%)	0.33	0.73	1.02	0.65	2.50	1.52
Census based [17]	7.8 (16%)	0.32	1.14	1.01	0.65	2.68	1.43
Hug.-Dev. [8]	3.8 (100%)	0.37	0.83	1.24	0.69	2.51	1.75
Fill-SGM	10.9 (100%)	0.45	0.76	1.99	0.77	2.55	2.76

4.1 Rotating Sphere

To assess the quality of our scene flow algorithm, it was tested on synthetic sequences, where the ground truth is known[1].

The first ground truth experiment is the *rotating sphere* sequence from [8] depicted in Figure 9. In this sequence the spotty sphere rotates around its y-axis to the left, while the two hemispheres of the sphere rotate in opposing vertical directions. The resolution is 512×512 pixels.

We tested the scene flow method together with four different stereo algorithms: semi-global matching (SGM [6]), SGM with hole filling (favours smaller disparities),

[1] The authors thank Huguet and Devernay for providing their *sphere scene*.

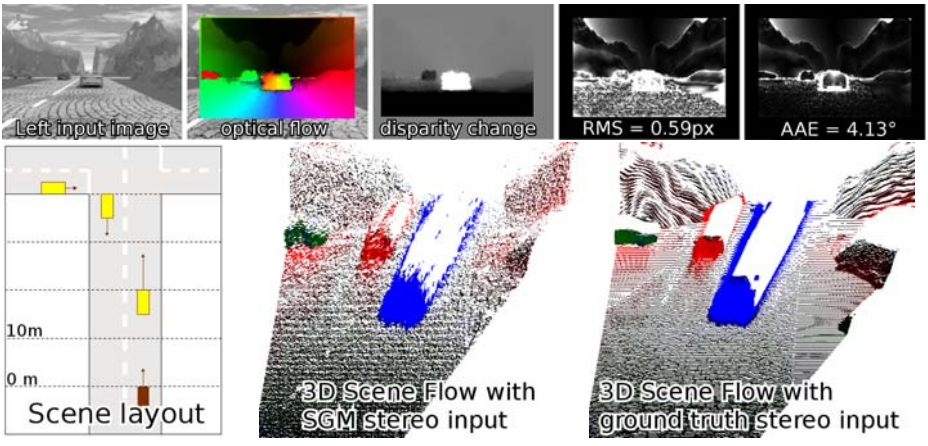


Fig. 10. Povray-rendered traffic scene (Frame 11). **Top:** Colour encodes direction (border = direction key) and intensity the magnitude of the optical flow vectors. Brighter areas in the error images denote larger errors. For comparison, running the code from [8] generates an RMS error of 0.91px and AAE of 6.83°. **Bottom right:** 3D views of the scene flow vectors. Colour encodes their direction and brightness their magnitude (black = stationary). The results from the scene are clipped at a distance of 100m. Accurate results are obtained even at greater distances.

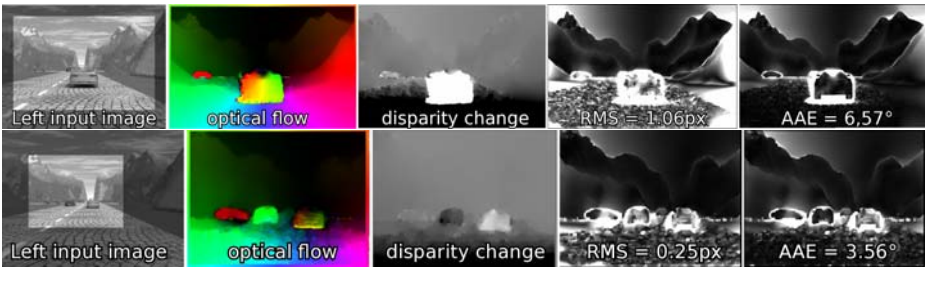


Fig. 11. More frames from the traffic scene in Figure 10. The top row highlights the problems such as transparency of the windshield, reflectance, and moving shadows. The bottom row demonstrates that we still maintain accuracy at distances of 50 m.

correlation pyramid stereo [4], and an integer accurate census-based stereo algorithm [17]. The ground truth disparity was also used for comparison, i.e., using the ground truth as the input disparity for our algorithm.

For each stereo algorithm, we calculated the absolute angular error (AAE) and the root mean square (RMS) error

$$RMS_{u,v,d,d'} = \sqrt{\frac{1}{n}\sum_{\Omega} \|(u_i, v_i, d_i, d'_i)^\top - (u_i^*, v_i^*, d_i^*, d_i'^*)^\top\|^2} \tag{1.27}$$

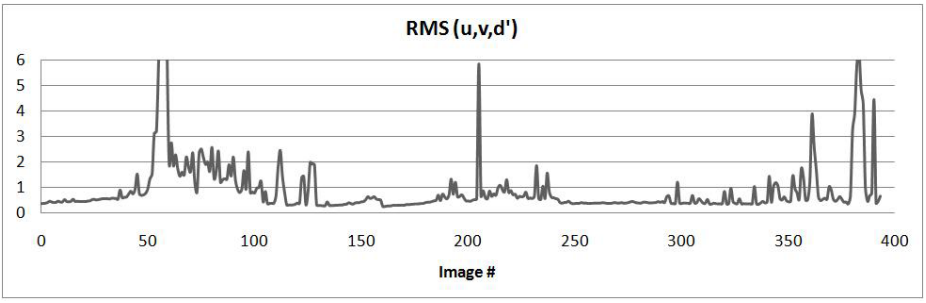


Fig. 12. RMS error evaluation over the entire sequence

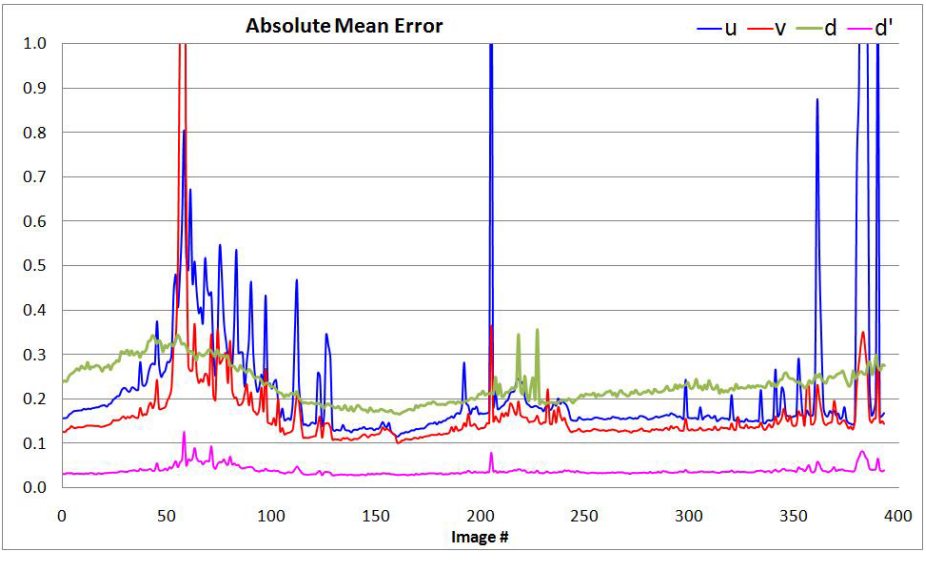


Fig. 13. Mean error evaluation over entire sequence

where a superscript * denotes the ground truth solution and n is the number of pixels. If there is no disparity measure d (either by sparse algorithm or occlusion) then a value of 0 is used (so still contributes to error). In our notation for RMS, if a subscript is omitted, then both the respective ground truth and estimated value are set to zero.

$$AAE_{u,v} = \frac{1}{n} \sum_{\Omega} \arctan \left(\frac{uv^* - u^*v}{uu^* + vv^*} \right) \tag{1.28}$$

as used in [8]. The errors were calculated in using two image domains Ω: firstly, calculating statistics over all non-occluded areas, and secondly calculating over the whole sphere. As in [8], pixels from the background were not included in the statistics.

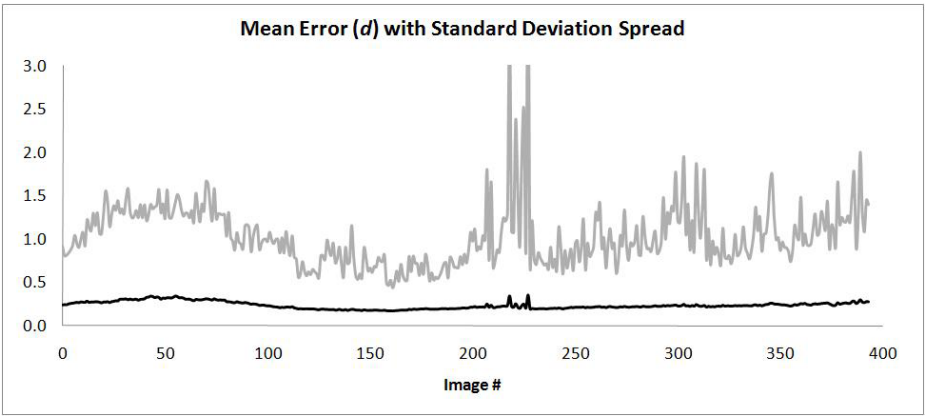


Fig. 14. An example graph using the mean error and variance from Section 4.3. This graph shows the results for disparity d. Mean error of d (i.e., $\mu(d)$) is the dark coloured line. The light line is 1 standard deviation from the mean (i.e., $\mu(d) + \sigma(d)$). The graphs for flow u,v, and d', are in Figure 15.

The resulting summary can be seen in Table 1. We achieve lower errors than the Huguet and Devernay method, even using sparse correlation stereo. Particularly, the RMS error of the scene flow is much smaller and we are still considerably faster. This is explained by the higher flexibility in choosing the disparity estimation method. The table shows that SGM with hole filling yields inferior results than the other stereo methods. This is due to false disparity measurements in the occluded area. It is better to feed the sparse measurements of SGM to the variational framework, which yields dense estimates as well, but with higher accuracy. SGM was chosen as the best method and is used in the remainder of the results section; it is also available on dedicated hardware without any extra computational cost.

4.2 Povray Traffic Scene 1

In a second ground truth example we use a Povray-rendered traffic scene [20], which is available online publicly for comparison [19]. The scene layout shown in Figure 10. We calculated the $RMS_{u,v,d'}$ error and the 3D angular error defined by:

$$AAE_{3D} = \frac{1}{n} \sum_{\Omega} \arccos \left(\frac{uu^* + vv^* + d'd'^* + 1}{\sqrt{(u^2 + v^2 + d'^2 + 1)\left((u^*)^2 + (v^*)^2 + (d'^*)^2 + 1\right)}} \right)$$

where * defines a ground truth value.

Results are shown in Figures 10 and 11. They compare favourably to the results obtained when running the code from [8]. The average $RMS_{u,v,d'}$ error for the whole sequence (subregion as in Figure 10) was 0.64 px and the 3D angular error was $3.0°$.

4.3 Evaluation Approach Using Stereo Synthetic Data

In this subsection the Povray Traffic Scene 2 is used to analyse the output from our scene flow approach. It is a more complex driving scene, involving hills, trees, and realistic physics; it consists of 400 sequential stereo image pairs. An example picture is shown in Figure 18. This scene is not yet publicly available, but will be available publicly in the future [19].

This data set contains 400 frames, so an evaluation approach has been devised to handle such a large dataset. For each image pair at time t, the following is calculated:

- $RMS_{u,v,d'}$ from the subsection above.
- Error at each pixel, i.e., difference between estimate and ground truth.
- Absolute mean error for u,v,d', and d.
- Variance of the error for u,v,d', and d.

The mean and variance of the error are defined as:

$$\mu(a) = \frac{1}{n} \sum_{\Omega} |a - a^*| \tag{1.29}$$

$$\sigma^2(a) = \frac{1}{n} \left(\sum_{\Omega} (a - a^*)^2 \right) - (\mu(a))^2 \tag{1.30}$$

respectively, where a represents u,v,d', or d, and * denotes ground truth value.

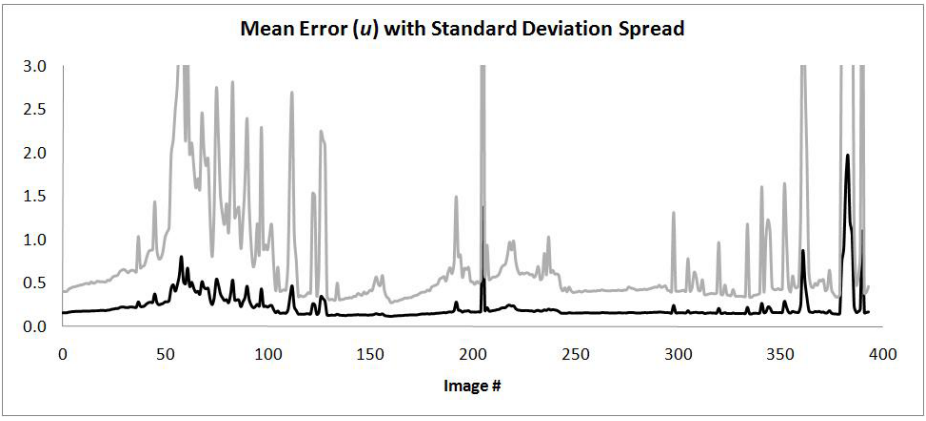
We have provided an example using our scene flow algorithm in Figure 18. From this figure it can be seen that the major errors are on object boundaries, and the errors in d' are the lowest in magnitude. Another single frame example is shown in Figure 19.

The example results for the entire image sequence can be seen in the graphs for Figures 12, 13, 14, and 15. In Figure 13, it can be seen that the error for u and v are consistently about the same error as for d (except in a few frames where the error is dramatically higher). From these graphs, frames can be identified which are causing problems for an algorithm, and the algorihm can be improved iteratively. The errors and standard deviation for the flow $[u, v, d']^\top$ (Figure 15) are of similar shape, yet different magnitudes (compared to Figure 14). This is expected as they are solved using the variational energy minimisation and smoothed in the unified framework. The sequences are provided for public use and will allow comparisons of scene flow algorithms in future works.

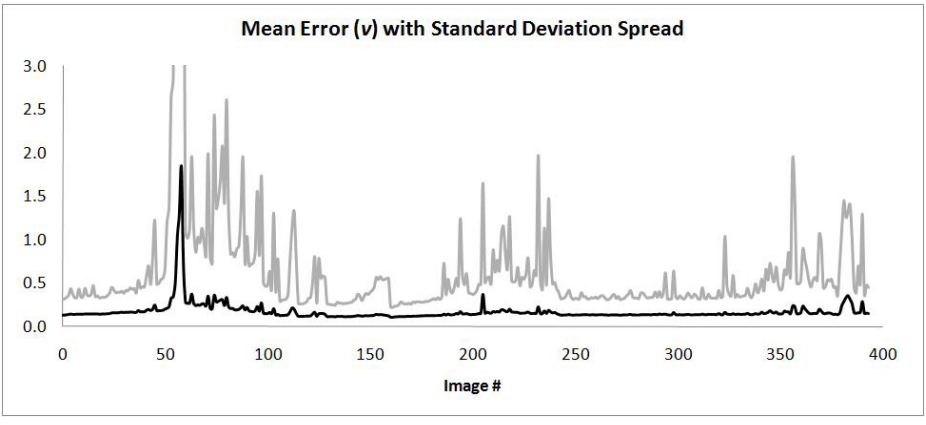
4.4 Real-World Scenes

Figure 16 and Figure 17 show scene flow results in real-world scenes with a moving camera. Ego-motion of the camera is known from vehicle odometry and compensated in the depicted results.

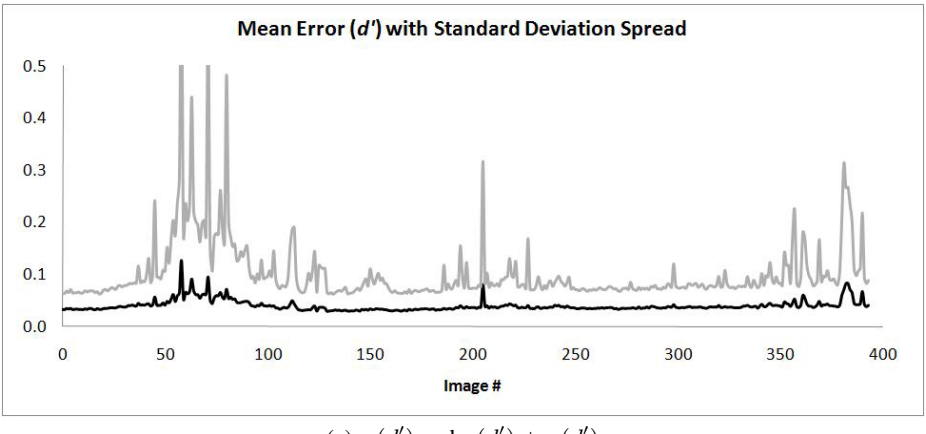
Figure 16 shows an image from a sequence where the ego-vehicle is driving past a bicyclist. The depicted scene flow shows that most parts of the scene, including the vehicle stopping at the traffic lights, are correctly estimated as stationary. Only the bicyclist is moving and its motion is accurately estimated.



(a) $\mu(u)$ and $\mu(u) + \sigma(u)$



(b) $\mu(v)$ and $\mu(v) + \sigma(v)$



(c) $\mu(d')$ and $\mu(d') + \sigma(d')$

Fig. 15. The graphs for flow u,v, and d'. Colour coding and explanation as in Figure 14.

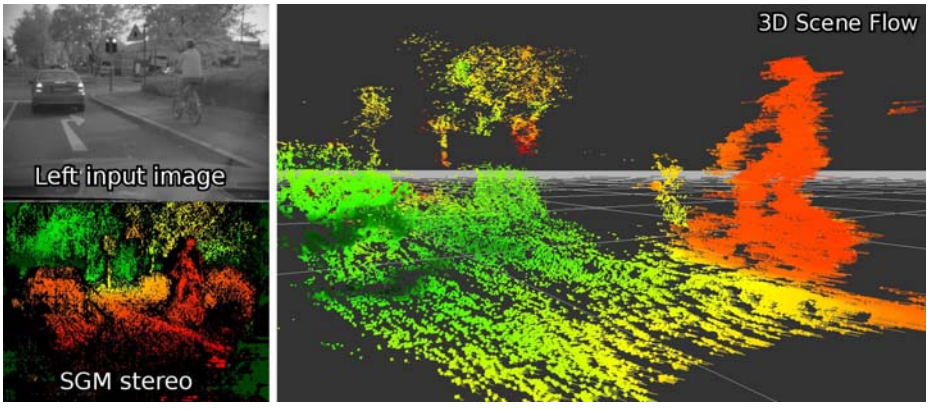


Fig. 16. Dense scene flow in a traffic scene. The colour in the lower left image encodes distance from red to green (close to far); the colour in the scene flow image (right) shows vector lengths after ego-motion compensation (green to red = 0 to $0.4m/s$). Only the cyclist is moving.

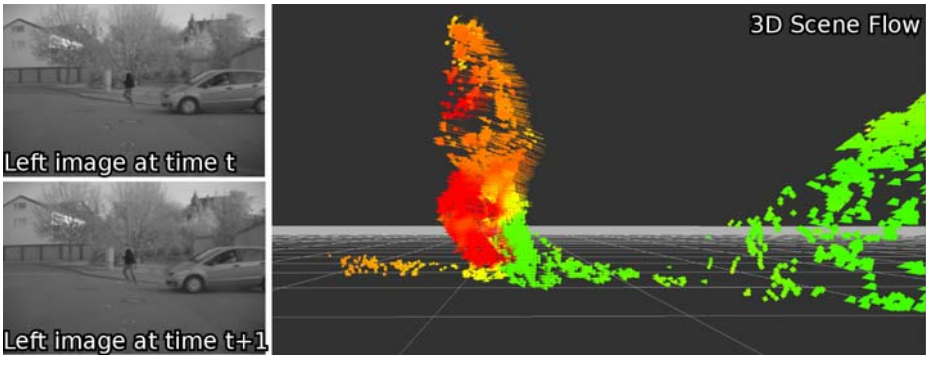
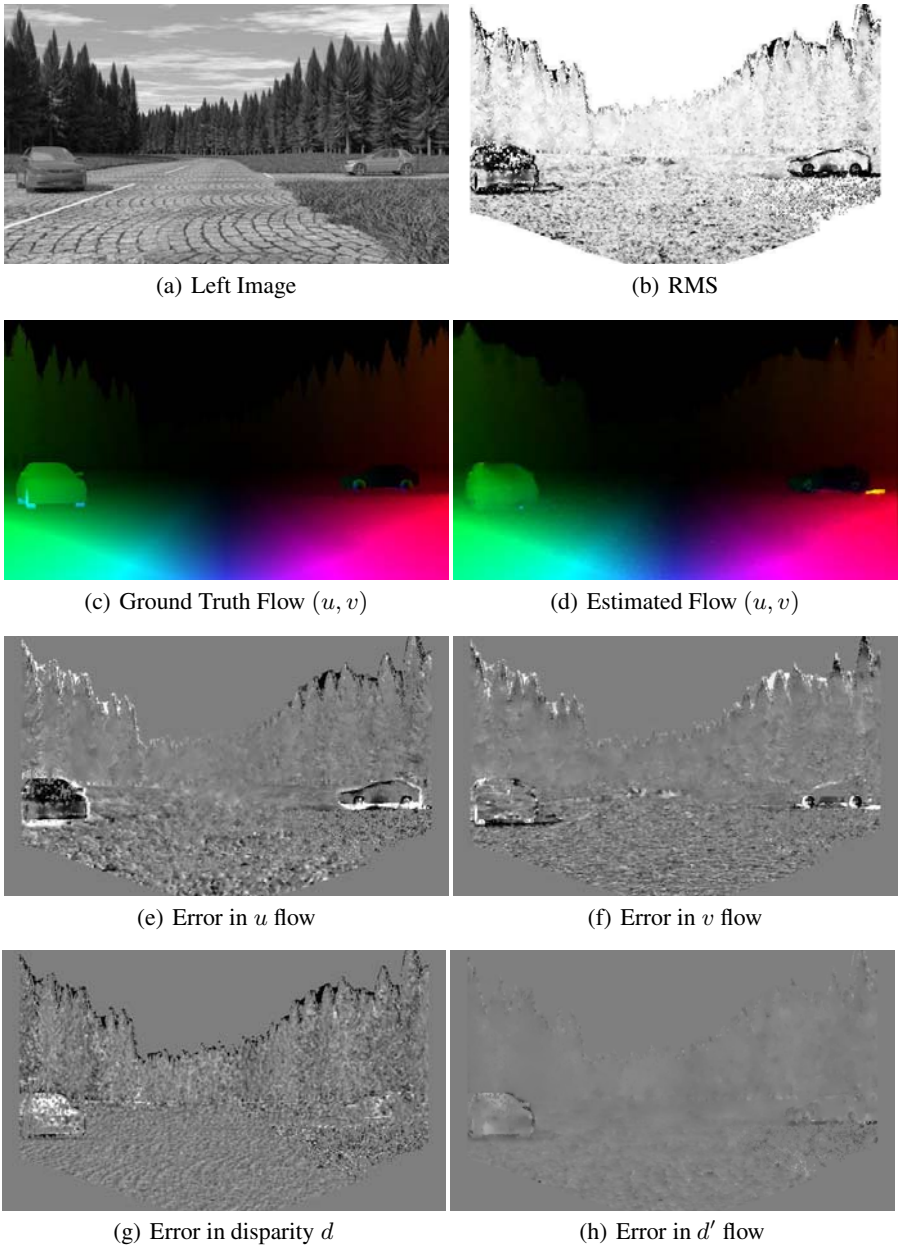


Fig. 17. Scene with a person running from behind a parked vehicle. The colour encoding is as in Figure 16.

Figure 17 shows results from a scene where a person runs from behind a parked vehicle. The ego-vehicle is driving forward at 30 km/h and turning to the left. The measurements on the ground plane and in the background are not shown to focus visual attention on the person. The results show that points on the parked vehicle are estimated as stationary, where as points on the person are registered as moving. The accurate motion results can be well observed for the person's legs, where the different velocities of each leg are well estimated.

More real-world results are seen in Figure 7. Here, the motion metric is used to determine likelihood that points are moving. From this result, it shows a high correctness and should provide good results for follow on uses, such as segmentation. This is outside the scope of this chapter but is in scope of future work.



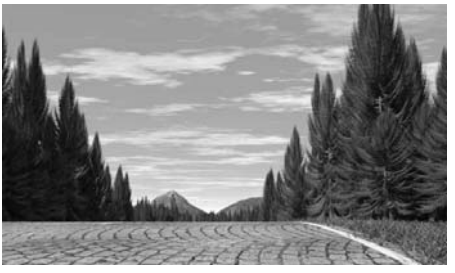
(a) Left Image (b) RMS

(c) Ground Truth Flow (u, v) (d) Estimated Flow (u, v)

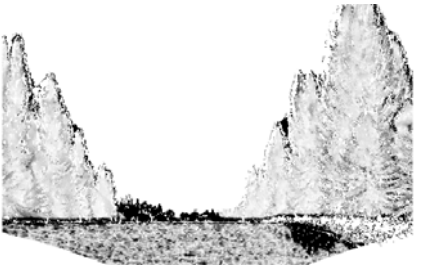
(e) Error in u flow (f) Error in v flow

(g) Error in disparity d (h) Error in d' flow

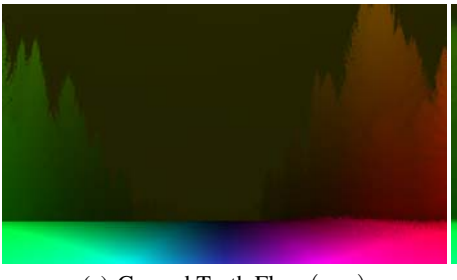
Fig. 18. Povray-rendered traffic scene 2 (Frame 215). RMS encoded low to high as light to dark brightness. Flow colour encoded as in Figure 9. Error images encoded with brightness, negative = dark, positive = light, zero = mid-grey value (e.g., infinite background). Points at infinity and occlusions in RMS and error images are shown as a zero value in each colour scale.



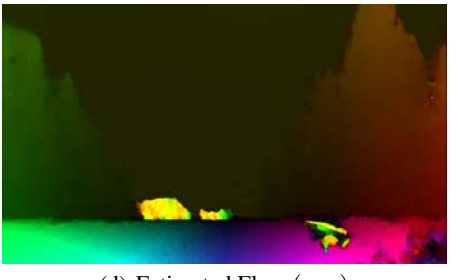
(a) Left Image



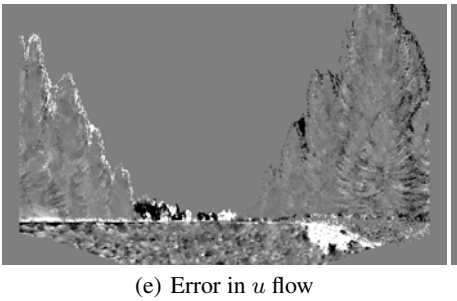
(b) RMS



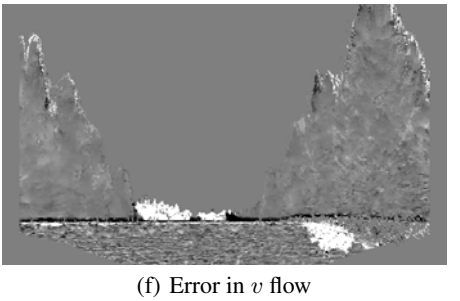
(c) Ground Truth Flow (u, v)



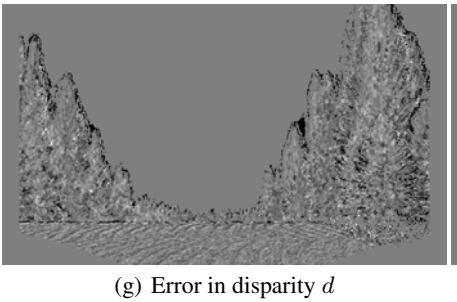
(d) Estimated Flow (u, v)



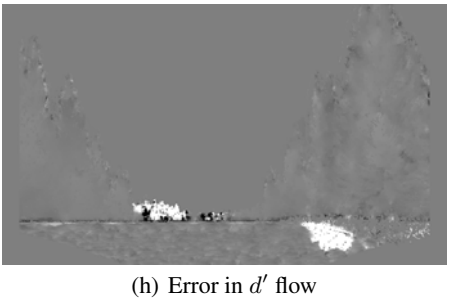
(e) Error in u flow



(f) Error in v flow



(g) Error in disparity d



(h) Error in d' flow

Fig. 19. Povray-rendered traffic scene 2 (Frame 58). Encoding as Figure 19.

5 Conclusions

We presented a variational framework for dense scene flow estimation, which is based on a decoupling of the disparity estimation from the motion estimation, while enforcing consistent disparity between times. We showed that this strategy has two main advantages: Firstly, we can choose optimal methods for estimating both disparity and scene flow. In particular, we can combine occlusion handling and semi-global (combinatorial) optimisation for disparity estimation with dense, sub-pixel accurate scene flow estimation. Secondly, for the first time, we obtain dense scene flow results very efficiently in real-time.

We then derived the image scene flow to 3D scene flow calculations. From this information, we also introduced two likelihood metrics (motion and speed) to estimate if points in the scene are moving or static. Examples were given to visually show how these will benefit follow on processes.

In our evaluation, we present both synthetic and real-world image results. Along with this evaluation is a methodology to evaluate scene flow algorithms to one another using long synthetic image sequences. This information can be gathered over an entire sequence and further statistical inference can be made.

In future work, we would like to use follow-on processes, such as segmentation, to evaluate our motion and speed metrics. Furthermore, we would like to take advantage of the statistical inference for long image sequences. With this information we can then compare and contrast different scene flow algorithms.

References

1. Badino, H.: A robust approach for ego-motion estimation using a mobile stereo platform. In: Jähne, B., Mester, R., Barth, E., Scharr, H. (eds.) IWCM 2004. LNCS, vol. 3417, pp. 198–208. Springer, Heidelberg (2007)
2. Brox, T., Bruhn, A., Papenberg, N., Weickert, J.: High accuracy optical flow estimation based on a theory for warping. In: Pajdla, T., Matas, J(G.) (eds.) ECCV 2004. LNCS, vol. 3024, pp. 25–36. Springer, Heidelberg (2004)
3. Bruhn, A., Weickert, J., Kohlberger, T., Schnörr, C.: Discontinuity preserving computation of variational optic flow in real-time. In: ScaleSpace 2005, pp. 279–290 (2005)
4. Franke, U., Joos, A.: Real-time stereo vision for urban traffic scene understanding. In: Proc. IEEE Intelligent Vehicles Symposium, Dearborn, pp. 273–278 (2000)
5. Hartley, R.I., Zisserman, A.: Multiple View Geometry in Computer Vision. Cambridge University Press, Cambridge (2000)
6. Hirschmüller, H.: Stereo processing by semiglobal matching and mutual information. IEEE Pattern Analysis and Machine Intelligence (PAMI) 30(2), 328–341 (2008)
7. Horn, B., Schunck, B.: Determining optical flow. Artificial Intelligence 17, 185–203 (1981)
8. Huguet, F., Devernay, F.: A variational method for scene flow estimation from stereo sequences. In: IEEE Int. Conf. on Computer Vision (ICCV) (October 2007)
9. Isard, M., MacCormick, J.: Dense motion and disparity estimation via loopy belief propagation. In: Narayanan, P.J., Nayar, S.K., Shum, H.-Y. (eds.) ACCV 2006. LNCS, vol. 3852, pp. 32–41. Springer, Heidelberg (2006)
10. Mahalanobis, P.C.: On the generalised distance in statistics. In: Proc. of the National Institute of Science of India, vol. 12, pp. 49–55 (1936)

11. Mémin, E., Pérez, P.: Dense estimation and object-based segmentation of the optical flow with robust techniques. IEEE Trans. on Image Processing 7(5), 703–719 (1998)
12. Min, D., Sohn, K.: Edge-preserving simultaneous joint motion-disparity estimation. In: Proc. Int. Conf. on Pattern Recognition (ICPR), Washington, DC, USA, 2006, pp. 74–77. IEEE Computer Society Press, Los Alamitos (2006)
13. Patras, I., Hendriks, E., Tziritas, G.: A joint motion/disparity estimation method for the construction of stereo interpolated images in stereoscopic image sequences. In: Proc. 3rd Annual Conf. of the Advanced School for Computing and Imaging, Heijen, The Netherlands (1997)
14. Pons, J.-P., Keriven, R., Faugeras, O.: Multi-view stereo reconstruction and scene flow estimation with a global image-based matching score. Int. J. Computer Vision 72(2), 179–193 (2007)
15. Rabe, C., Franke, U., Gehrig, S.: Fast detection of moving objects in complex scenarios. In: Proc. IEEE Intelligent Vehicles Symposium, June 2007, pp. 398–403 (2007)
16. Scharstein, D., Szeliski, R.: A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. In: Proc. IEEE Int. Conf. on Computer Vision (ICCV), pp. 7–42 (2002)
17. Stein, F.: Efficient computation of optical flow using the census transform. In: Rasmussen, C.E., Bülthoff, H.H., Schölkopf, B., Giese, M.A. (eds.) DAGM 2004. LNCS, vol. 3175, pp. 79–86. Springer, Heidelberg (2004)
18. Tomasi, C., Kanade, T.: Detection and tracking of point features. Technical Report CMU-CS-91-132, Carnegie Mellon University (April 1991)
19. University of Auckland. enpeda. Image Sequence Analysis Test Site (EISATS) (2008), http://www.mi.auckland.ac.nz/EISATS/
20. Vaudrey, T., Rabe, C., Klette, R., Milburn, J.: Differences between stereo and motion behaviour on synthetic and real-world stereo sequences. In: Proc. Int. Conf. Image and Vision Computing New Zealand (IVCNZ) (2008)
21. Vedula, S., Baker, S., Rander, P., Collins, R., Kanade, T.: Three-dimensional scene flow. IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI) 27(3), 475–480 (2005)
22. Wedel, A., Pock, T., Zach, C., Bischof, H., Cremers, D.: An improved algorithm for TV-L1 optical flow. In: Cremers, D., Rosenhann, B., Yuille, A., Schmidt, F.R. (eds.) Visual Motion Analysis 2008. LNCS, vol. 5604, pp. 23–45. Springer, Heidelberg (2009)
23. Wedel, A., Rabe, C., Vaudrey, T., Brox, T., Franke, U., Cremers, D.: Efficient dense scene flow from sparse or dense stereo data. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part I. LNCS, vol. 5302, pp. 739–751. Springer, Heidelberg (2008)
24. Zach, C., Pock, T., Bischof, H.: A duality based approach for realtime tv-L^1 optical flow. In: Hamprecht, F.A., Schnörr, C., Jähne, B. (eds.) DAGM 2007. LNCS, vol. 4713, pp. 214–223. Springer, Heidelberg (2007)
25. Zhang, Y., Kambhamettu, C.: On 3d scene flow and structure estimation. In: Proc. IEEE Conf. in Computer Vision and Pattern Recognition, vol. 2, p. 778. IEEE Computer Society Press, Los Alamitos (2001)

An Affine Optical Flow Model for Dynamic Surface Reconstruction

Tobias Schuchert and Hanno Scharr

Institute for Chemistry and Dynamics of the Geosphere, ICG-3
Forschungszentrum Jülich GmbH, 52425 Jülich, Germany
{t.schuchert,h.scharr}@fz-juelich.de

Abstract. In this paper we develop a differential model for simultaneous estimation of geometry and dynamics of a surface patch. To do so we combine a surface patch model in local 3D coordinates, a pinhole camera grid model and a brightness change model analogous to the brightness constancy constraint equation for optical flow. It turns out to be an extension of the well known affine optical flow model to higher dimensional data sets. Each of the translational and affine components of the optical flow is a term consisting of a mixture of surface patch parameters like its depth, slope, velocities etc. We evaluate the model by comparing estimation results using a simple local estimation scheme to available ground-truth. This simple estimation scheme already allows to get results in the same accuracy range one can achieve using range flow, i.e. a model for the estimation of 3D velocities of a surface point given a measured surface geometry. Consequently the new model allows direct estimation of additional surface parameters range flow is not capable of, without loss of accuracy in other parameters. What is more, it allows to design estimators coupling shape and motion estimation which may yield increased accuracy and/or robustness in the future.

1 Introduction

In this paper we develop an affine optical-flow-like differential model for motion and shape reconstruction using image sequences of a camera grid. The model is valid for instantaneously moving cameras observing moving surfaces. This is unlike previous work (e.g. [1,19,22,38]) where either an observed surface moves, or a camera, but not both.

Motion estimation as well as disparity estimation are standard tasks for optical flow algorithms as well known from early publications (e.g. [17,23] and many more). The model introduced here combines the two cases. It is an extension of a model first introduced in [28] and further refined and extended in [30,32]. It allows simultaneous local estimation of 3D motion, 3D position and surface normals in a spatio-temporal affine optical-flow-like model. The standard affine optical flow model (cmp. e.g. [13]).

$$\nabla I \left[\begin{pmatrix} u_x \\ u_y \end{pmatrix} + \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} \right] + I_t \mathrm{d}t = 0 \tag{1.1}$$

D. Cremers et al. (Eds.): Visual Motion Analysis, LNCS 5604, pp. 70–90, 2009.

defines parameters in image coordinates, i.e. flow parameters. In this contribution the meaning of these parameters $u_.$ and $a_{..}$ will be explained using 3D coordinates and parameters of the imaged moving surface patch.

1.1 Approach

An image sequence can be interpreted as data in a 3D x-y-t-space. A brightness change constraint equation in this space defines a linear model for the changes of grey values due to apparent local object motion. This apparent object motion is called optical flow or displacement vector field. Assuming the data comes from a single fixed camera displacements can be explained by object motion. Assuming a moving camera looking at a fixed scene displacements (then usually called disparities) are anti-proportional to local depth. This is known as structure from camera motion (e.g. [20]).

The basic idea for the new model presented here is to interpret the camera position $\mathbf{s} = (s_x, s_y)$ as additional dimension(s) of the data. Hence all image sequences acquired by a 1D camera grid can be combined to sample a 4D-Volume in x-y-s_x-t-space. If a 2D camera grid is used (as e.g. in [24]) data forms a 5D-Volume in x-y-s_x-s_y-t-space. We define brightness constancy in this space as vanishing total differential of the intensity data. A camera model is used to project a dynamic surface patch into the image. This yields equations for x- and y-position of the patch as a function of camera position and time. Plugging the linearized total differentials of x and y into the total differential of the intensity results in the sought for model. It boils down to be an affine optical flow model with 3 dimensions (s_x, s_y, t) behaving like the time dimension in a usual affine optical flow model. A detailed derivation can be found in Section 2.

In order to evaluate the model we set up a local parameter estimation procedure (Section 3). We use the structure tensor method (e.g. [3]) to estimate flow parameters. The structure tensor implements local total least squares estimation ideally suited for estimation when Gaussian noise present. No robust statistics or regularization terms formulating prior knowledge are applied. Such terms may conceal model errors and are therefore less suitable for model evaluation. Obviously the structure tensor approach is not the best estimator in situations where prior knowledge really is available, e.g. when dealing with a specific fixed application. In this paper we are *not* aiming at a *best method* for an application, but at *model evaluation*. Therefore a simple local estimator is the best choice for our goal. The flow parameters in the model stand for mixed motion-, normals- and disparity-parameters that have to be disentangled. Especially when 2D camera grids are used multiple independent measurements are present in each model equation. These not only have to be disentangled, but also recombined respecting their error estimates.

All experiments (Section 4) use synthetic data with ground truth available. For systematic evaluation of accuracies we use sinusoidal patterns projected to moving surfaces without using approximations. By doing so we generate input grey value data in 32bit-float accuracy suppressing otherwise unavoidable quantization noise. For more realistic scenes we use simple geometric structures

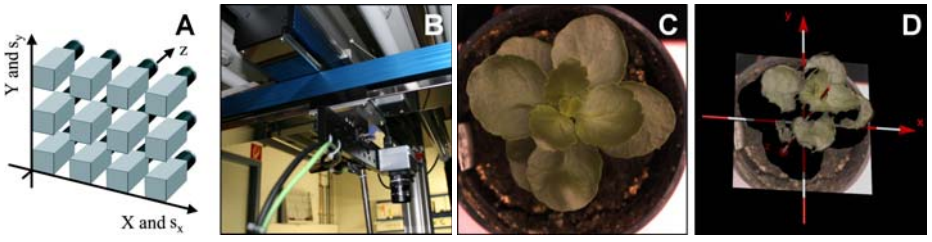


Fig. 1. Setup for our target application plant growth analysis. A camera grid (**A**) is simulated using a single camera on a moving stage (**B**). From images (**C**) taken from vertical view a 3D reconstruction (**D**) is calculated.

moving in a known way rendered by POV-Ray [7]. For these scenes we also have ground truth available, but suffer from quantization noise as grey value range is 8bit-integer. We compare motion results to range flow [33] in order to give an intuition of the accuracies achievable using the simple structure tensor for estimation. In contrast to our model, range flow needs depth information as input and is able to estimate 3D translation only. In its standard form it can not handle further geometric parameters and can not deal with inputs from multiple cameras. Object intensities can be incorporated into range flow estimates [31] as well as in multi-camera optical-flow-like models [32].

Although we do not aim at presenting a *method* yet, we have a target application in mind when designing our experiments: plant growth analysis, especially leaf growth. For imaging we use a single camera on an x-y-moving stage instead of a real camera grid. This allows us to use very small baselines, much smaller than camera dimensions. Obviously images are not taken at exactly the same point in time, which could be accounted for in the discretization of derivative kernels applied to the images. We neglect this effect here, because plants move slow enough such that time intervals between two images at the same camera position are much larger than the overall acquisition time for all camera images representing 'one' point in time.

1.2 Contribution

The current paper is an extension of a series of papers [28,30,32]. Our contributions here are the following:

- Derivation of temporal affine flow parameters (Δt-terms).
- Shift of representation from Lagrangian to Eulerian view necessary when introducing Δt-terms.
- A projective correction term for average neighbor distances in 3D based on local pixel coordinates.
- A thorough model evaluation focusing on the influence of the novel Δt-terms.

1.3 Other Related Work

There is rich literature on optical flow estimation techniques (see e.g. [2,4,5,16,26]) and many extensions have been developed. There are extensions towards affine motion estimation [10,11], scene flow [36] and brightness changes [8,15]. Special filters [9,12,18] have been studied. There already exist frameworks for simultaneous motion and stereo analysis (e.g. [6,35,37]). Also stereo for curved surfaces still finds attention [21].

1.4 Paper Organization

We start by deriving the model (Section 2) followed by a section on parameter estimation in this model (Section 3). Section 4 then evaluates the model. The paper finishes with summary and conclusions (Section 5).

2 Derivation of the BCCE

In this section we derive a constraint equation describing local brightness changes in the data. To do so we project a geometric model of a moving surface patch onto a camera sensor plane using a pinhole camera model. This geometric description of moving projected surface elements is then combined with a brightness constancy assumption forming the sought for constraint equation.

2.1 Surface Patch Model

For each pixel at a given point in time we want to estimate from a local neighborhood depth, motion and surface normals of a 3D patch imaged at that location. Thus we use a surface patch valid for the whole neighborhood as object/motion model. This surface element has its center at world coordinate position (X_0, Y_0, Z_0) and is modeled as a function of time t and local world coordinates $(\Delta X, \Delta Y)$ with X- and Y-slopes Z_X and Z_Y. It moves with velocity (U_X, U_Y, U_Z):

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} X_0 + U_X \Delta t + \Delta X \\ Y_0 + U_Y \Delta t + \Delta Y \\ Z_0 + U_Z \Delta t + Z_X \Delta X + Z_Y \Delta Y \end{pmatrix} \tag{1.2}$$

The surface normal is then $(-Z_X, -Z_Y, 1)$. Velocity (U_X, U_Y, U_Z) could be further modeled using translation and rotation for rigid patches or using an affine transform for deformable patches. We omit this step here for simplicity.

2.2 Camera Model

We use a pinhole camera at world coordinate position $(S_X, S_Y, 0)$, looking into Z-direction (cmp. Figure 2)

$$\begin{pmatrix} x \\ y \end{pmatrix} = \frac{f}{Z} \begin{pmatrix} X - S_X \\ Y - S_Y \end{pmatrix} \tag{1.3}$$

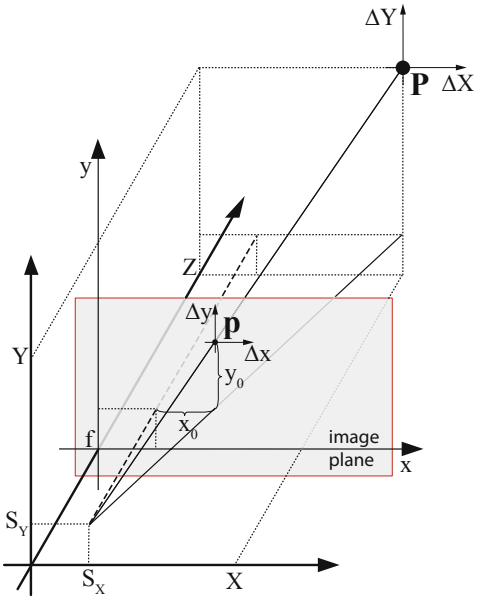


Fig. 2. Pinhole projection of a point P using a camera with center at (S_X, S_Y)

where x and y are sensor coordinates and camera position space is sampled equidistantly using a camera grid.

A local neighborhood for parameter estimation is defined using neighboring pixels. However, we are interested in the local neighborhood given on the 3D surface patch by means of ΔX and ΔY. We therefore need to derive a relation between local 3D coordinates ΔX and ΔY and local sensor coordinates Δx and Δy. To do so we observe that a point (X_0, Y_0, Z_0) is projected onto the sensor by:

$$\begin{pmatrix} X_{x_0,y_0} \\ Y_{x_0,y_0} \\ Z_{x_0,y_0} \end{pmatrix} = \frac{Z_{0,0}}{f} \begin{pmatrix} x_0 \\ y_0 \\ f \end{pmatrix} \frac{1}{1 - Z_X \frac{x_0}{f} - Z_Y \frac{y_0}{f}} \tag{1.4}$$

with depth $Z_{0,0}$ at the principal point $(x, y) = (0, 0)$ (cmp. [14]), and (x_0, y_0) being the pixel coordinates where (X_0, Y_0, Z_0) is projected to. The difference of two points on the same 3D plane is

$$\begin{pmatrix} \Delta X \\ \Delta Y \\ \Delta Z \end{pmatrix} = \begin{pmatrix} X_{x_1,y_1} - X_{x_0,y_0} \\ Y_{x_1,y_1} - Y_{x_0,y_0} \\ Z_{x_1,y_1} - Z_{x_0,y_0} \end{pmatrix} = \frac{Z_0}{f c_{x_1,y_1}} \begin{pmatrix} \left(1 - Z_Y \frac{y_0}{f}\right) \Delta x + Z_Y \frac{x_0}{f} \Delta y \\ \left(1 - Z_X \frac{x_0}{f}\right) \Delta y + Z_X \frac{y_0}{f} \Delta x \\ Z_X \Delta x + Z_Y \Delta y \end{pmatrix} \tag{1.5}$$

with

$$c_{x,y} = 1 - Z_X \frac{x}{f} - Z_Y \frac{y}{f} \tag{1.6}$$

and $\Delta x = x_1 - x_0$, $\Delta y = y_1 - y_0$.

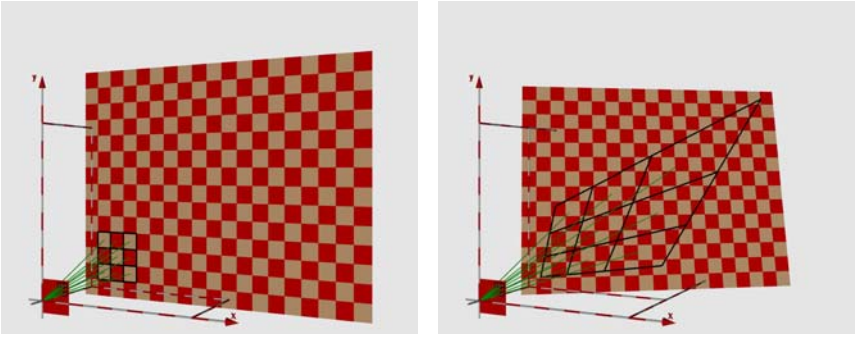


Fig. 3. Pinhole projection of a patch. Left: fronto-parallel surface, right: Surface not fronto-parallel. We selected huge pixel sizes to demonstrate the effect.

Correction factor $c_{x,y}$ depends on the local position (x, y) of a pixel within the local neighborhood. However in our parameter estimation scheme, we derive a single set of image-based parameters for a whole patch. Consequently we need to use a single correction factor c per patch independent of positions within the patch. The best value to use would be

$$< c >= \left(\frac{1}{N} \sum_{i=1}^{N} \frac{1}{c_{x_i,y_i}} w_i \right)^{-1} \tag{1.7}$$

where N is the number of pixels in the local neighborhood and w_i are (Gaussian) weights, i.e. the diagonal entries of the matrix $\mathbf{W}$ from Equation 1.23. However, using this equation would be cumbersome, as we want to use c when resolving for Z_X and Z_Y. In our calculations we simply use $c = c_{x_0,y_0}$ instead, i.e. the value of c at the patch center. The error $|c_{x_0,y_0} - < c > |$ we thereby accept is almost always in the range of $1e-6$ for typical observed slopes $Z_X << 10$ and $Z_Y << 10$, focal length $f \approx 25$mm, pixel sizes well below 10μm, and standard deviations of the Gaussian weights $w_i \lessapprox 10$ pixels.

2.3 Brightness Change Model

Using a camera grid means sampling (s_x, s_y)-space using several cameras each of which acquires an image sequence. We combine all of these sequences into one 5D data set sampling the continuous intensity function $I(x, y, s_x, s_y, t)$, i.e. intensity lives in a 5D space. We assume that the acquired brightness of a surface element is constant under camera translation, meaning we are looking at a Lambertian surface. In addition this brightness shall be constant in time, i.e. we need temporally constant illumination, which can be relaxed easily using the approach shown in [15]. We see that brightness does not change with one temporal and two spatial coordinates. In other words there is a 3D manifold in our

5D space in which I does not change. Thus the total differential $\mathrm{d}I$ vanishes in this manifold. The brightness model therefore is

$$\mathrm{d}I = I_x \mathrm{d}x + I_y \mathrm{d}y + I_{s_x} \mathrm{d}s_x + I_{s_y} \mathrm{d}s_y + I_t \mathrm{d}t = 0 \tag{1.8}$$

We use the notation $I_* = \frac{\partial I}{\partial *}$ for derivatives of the image intensities I. Please note that all derivatives and differentials in this equation have physical units. Image coordinates x and y are given in pixel, i.e. the physical length of a sensor element on the camera chip, typically several μm. Camera coordinates s_x and s_y are given in camera to camera displacements. And time t is given in "frames", i.e. the inverse of the temporal acquisition rate. This is important when interpreting parameters estimated from the model equations.

2.4 Combination of the 3 Models

In order to derive a single linear equation from the above models, we first project the moving surface element (Equation 1.2) to the sensor plane using a pinhole camera (Equation 1.3)

$$\begin{pmatrix} x \\ y \end{pmatrix} = \frac{f}{Z} \begin{pmatrix} X_0 + U_X \Delta t + \Delta X - S_X \\ Y_0 + U_Y \Delta t + \Delta Y - S_Y \end{pmatrix} \tag{1.9}$$

Consequently we can calculate the differentials $\mathrm{d}x$ and $\mathrm{d}y$ for a *fixed surface location* (i.e. for constant ΔX and ΔY)

$$\begin{pmatrix} \mathrm{d}x \\ \mathrm{d}y \end{pmatrix} = \frac{f}{Z} \begin{pmatrix} (U_X - U_Z \frac{x}{f})\mathrm{d}t - \mathrm{d}s_x \\ (U_Y - U_Z \frac{y}{f})\mathrm{d}t - \mathrm{d}s_y \end{pmatrix} \tag{1.10}$$

This equation is nonlinear in the sought for 3D parameters. We simplify it approximating f/Z via its Taylor expansion. From Equation 1.2 we know that $Z = Z_0 + U_Z \Delta t + Z_X \Delta X + Z_Y \Delta Y$ and thus

$$\frac{f}{Z} \approx \frac{f}{Z_0} - \frac{Z_X}{Z_0 c} \Delta x - \frac{Z_Y}{Z_0 c} \Delta y - \frac{f U_Z}{Z_0^2} \Delta t \tag{1.11}$$

Plugging Equation 1.11 into Equation 1.10, using local coordinates $x = x_0 + \Delta x$ and $y = y_0 + \Delta y$, sorting by differentials and Δ-terms, and ignoring higher order Δ-terms we get

$$\begin{aligned} dx = & \frac{f}{Z} \left[\left(U_X - \frac{x}{f} U_Z \right) dt - ds_x \right] \\ = & \frac{f}{Z_0} \left(U_X - \frac{x_0}{f} U_Z \right) dt \\ & - \left[\frac{Z_X}{Z_0 c} \left(U_X - \frac{x_0}{f} U_Z \right) + \frac{U_Z}{Z_0} \right] \Delta x dt \\ & - \left[\frac{Z_Y}{Z_0 c} \left(U_X - \frac{x_0}{f} U_Z \right) \right] \Delta y dt \\ & - \left[\frac{f U_Z}{Z_0^2} \left(U_X - \frac{x_0}{f} U_Z \right) \right] \Delta t dt \\ & - \frac{f}{Z_0} ds_x + \frac{Z_X}{Z_0 c} \Delta x ds_x + \frac{Z_Y}{Z_0 c} \Delta y ds_x + \frac{f U_Z}{Z_0^2} \Delta t ds_x \end{aligned} \tag{1.12}$$

$$\begin{aligned} dy &= \tfrac{f}{Z}\left[\left(U_Y - \tfrac{y}{f}U_Z\right)dt - ds_y\right] \\ &= \tfrac{f}{Z_0}\left(U_Y - \tfrac{y_0}{f}U_Z\right)dt \\ &- \left[\tfrac{Z_X}{Z_0 c}\left(U_Y - \tfrac{y_0}{f}U_Z\right)\right]\Delta x dt \\ &- \left[\tfrac{Z_Y}{Z_0 c}\left(U_Y - \tfrac{y_0}{f}U_Z\right) + \tfrac{U_Z}{Z_0}\right]\Delta y dt \\ &- \left[\tfrac{fU_Z}{Z_0^2}\left(U_Y - \tfrac{y_0}{f}U_Z\right)\right]\Delta t dt \\ &- \tfrac{f}{Z_0}ds_y + \tfrac{Z_X}{Z_0 c}\Delta x ds_y + \tfrac{Z_Y}{Z_0 c}\Delta y ds_y + \tfrac{fU_Z}{Z_0^2}\Delta t ds_y \end{aligned} \tag{1.13}$$

We may now rename the mixed 3D parameters in Equation 1.12 with

$$\begin{aligned} dx = {} & u_x dt + a_{11}\Delta x dt + a_{12}\Delta y dt + a_{13}\Delta t dt \\ & + \nu ds_x + b_1\Delta x ds_x + b_2\Delta y ds_x + b_3\Delta t ds_x \end{aligned} \tag{1.14}$$

and Equation 1.13 with

$$\begin{aligned} dy = {} & u_y dt + a_{21}\Delta x dt + a_{22}\Delta y dt + a_{23}\Delta t dt \\ & + \nu ds_y + b_1\Delta x ds_y + b_2\Delta y ds_y + b_3\Delta t ds_y \end{aligned} \tag{1.15}$$

Substituting Equations 1.14 and 1.15 into the brightness change model (Equation 1.8 we get the sought for affine-optical-flow-like model for dynamic surface reconstruction

$$\begin{aligned} & \nabla I \left[\begin{pmatrix} u_x \mathrm{d}t + \nu \mathrm{d}s_x \\ u_y \mathrm{d}t + \nu \mathrm{d}s_y \end{pmatrix} + \begin{pmatrix} a_{11}\mathrm{d}t + b_1 \mathrm{d}s_x & a_{12}\mathrm{d}t + b_2 \mathrm{d}s_x & a_{13}\mathrm{d}t + b_3 \mathrm{d}s_x \\ a_{21}\mathrm{d}t + b_1 \mathrm{d}s_y & a_{22}\mathrm{d}t + b_2 \mathrm{d}s_y & a_{23}\mathrm{d}t + b_3 \mathrm{d}s_y \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \\ \Delta t \end{pmatrix} \right] \\ & + I_{s_x}\mathrm{d}s_x + I_{s_y}\mathrm{d}s_y + I_t \mathrm{d}t = 0 \end{aligned} \tag{1.16}$$

We observe that the components of this model are known from the standard affine model for optical flow (Equation 1.1), i.e. translations u_x, u_y, and affine components a_{11}, a_{12}, a_{21}, and a_{22} in the well known form [13]. In addition the model contains temporal affine parameters a_{13} and a_{23} that are not due to acceleration – we have not modeled acceleration in Equation 1.2 – but due to projected motion in depth direction. Further there are disparity ν and its affine parameters b_1, b_2 and b_3 representing the projections of slopes Z_X, Z_Y, and depth velocity U_Z. They may be grouped into two affine optical flow like submodels for depth estimation.

For a 2D camera grid both depth-submodels, i.e. the submodel with terms containing $\mathrm{d}s_x$ and the one containing terms with $\mathrm{d}s_y$ are present. When only a 1D camera grid is available, say oriented in s_x-direction, then terms containing $\mathrm{d}s_y$ can not be evaluated and are removed by setting $\mathrm{d}s_y = 0$. In the remainder of this paper we call models operating on data of a 2D camera grid a *2D model*, the ones referring to a 1D camera grid *1D models*.

The model (Equation 1.16) is derived for *fixed surface location*. However we evaluate the model at *fixed sensor position*. As a consequence we will see in the next section that in term b_3 (and the respective factors in a_{13} and a_{23}) velocity U_Z must be exchanged by the projection corrected partial derivative of the depth field Z_t.

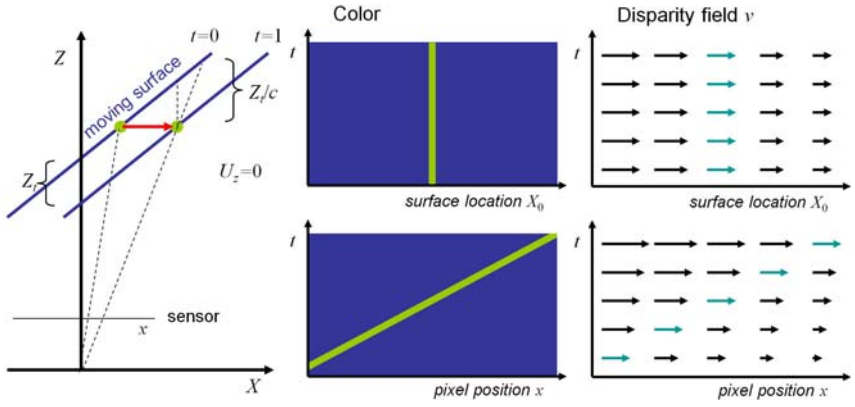


Fig. 4. Eulerian versus Lagrangian view. Left: A blue surface moves from left to right. The green point indicated the observed surface location $X(t)$. Velocity in x-direction is $U_X = \text{const}$ and in depth direction $U_Z = 0$. Middle, top: Color observed in Lagrangian view moving with the surface point. Middle, bottom: Color observed in Eulerian view at fixed pixel position. Right: Lagrangian and Eulerian view of the observed disparity field. Note that while $U_Z = 0$ disparity changes in temporal direction in Eulerian view (right, bottom). This disparity change corresponds to the observed depth change Z_t/c (left).

2.5 Changing from Lagrangian to Eulerian View

So far all parameters in Equations 1.12 and 1.13 are given for fixed locations $X(t)$ on the moving surface patch. However, the camera data we want to model is acquired at fixed pixel positions $x = fX(t_0)/Z(t_0)$, not moving with the patch. This is no problem as long as our model describes the status of the patch at a fixed point in time t_0, e.g. $U_Z(x, t_0) = U_Z(X(t_0), t_0)$, as usual for (affine) optical flow models not modeling the motion field in temporal direction. In the model introduced here, we do model the disparity field in time direction. This is due to the Taylor expansion in Equation 1.11. Consequently the term $\frac{fU_Z(X(t),t)}{Z_0^2}\Delta t|_{t=t_0}$, i.e. term b_3 in Equation 1.12 and the respective factors in a_{13} and a_{23} become

$$\frac{fU_Z(X(t),t)}{Z_0^2}\Delta t\Big|_{t=t_0} \rightarrow \frac{f(Z_t(x,t))}{Z_0^2 c}\Delta t\Big|_{t=t_0} \tag{1.17}$$

where the partial temporal derivative $Z_t(x,t)$ of the depth field $Z(x,t)$ is connected with U_Z via the range flow constraint [33]

$$U_Z(X(t),t)|_{t=t_0} = (Z_t(x,t) + Z_X(x,t)U_X(x,t) + Z_Y(x,t)U_Y(x,t))|_{t=t_0} \tag{1.18}$$

The correction factor c is due to camera projection, cmp. Figure 4.

Such changes of view-point are well known in fluid dynamics, where sensors are either moving with the flowing fluid (Lagrangian view) or are anchored at

fixed locations (Eulerian view). Temporal changes of the sensor signal in the Lagrangian view are modeled using total derivatives, in Eulerian view using partial derivatives. This is in complete consistence with Equation 1.18.

2.6 Local Equations for Flow Components

After this change of view-point we can now state the parameter equations, i.e. the equations connecting flow components with surface patch parameters. Flow parameters coming with $\mathrm{d}t$ are

$$\begin{aligned}
u_x &= \frac{f}{Z_0}\left(U_X - \frac{x_0}{f}U_Z\right) & u_y &= \frac{f}{Z_0}\left(U_Y - \frac{y_0}{f}U_Z\right)\\
a_{11} &= -\frac{Z_X}{Z_0 c}\left(U_X - \frac{x_0}{f}U_Z\right) - \frac{U_Z}{Z_0} & a_{21} &= -\frac{Z_X}{Z_0 c}\left(U_Y - \frac{y_0}{f}U_Z\right)\\
a_{12} &= -\frac{Z_Y}{Z_0 c}\left(U_X - \frac{x_0}{f}U_Z\right) & a_{22} &= -\frac{Z_Y}{Z_0 c}\left(U_Y - \frac{y_0}{f}U_Z\right) - \frac{U_Z}{Z_0}\\
a_{13} &= -\frac{f}{Z_0}\frac{Z_t}{Z_0 c}\left(U_X - \frac{x_0}{f}U_Z\right) & a_{23} &= -\frac{f}{Z_0}\frac{Z_t}{Z_0 c}\left(U_Y - \frac{y_0}{f}U_Z\right)
\end{aligned} \tag{1.19}$$

The ones coming with $\mathrm{d}s_x$ or $\mathrm{d}s_y$ are

$$\begin{aligned}
\nu &= -\frac{f}{Z_0} & b_3 &= \frac{f}{Z_0}\frac{Z_t}{Z_0 c}\\
b_1 &= \frac{Z_X}{Z_0 c} & b_2 &= \frac{Z_Y}{Z_0 c}
\end{aligned} \tag{1.20}$$

All patch parameters are evaluated at a fixed pixel position and point in time. We may use the range flow constraint (Equation 1.18) to calculate U_Z from the derivatives of Z and motion in U_X and U_Y direction.

3 Parameter Estimation

As far as we know there is no estimator available to solve Equation 1.12 and Equation 1.13 for the parameters directly. In this section we present a way to solve for the parameters using two standard least squares estimators. In Section 3.1 we briefly review estimation of the affine flow parameter vectors based on the structure tensor method. Section 3.2 shows disentangling of depth and motion parameters using linear combination of the solution vectors. Finally in Section 3.2 motion parameters are estimated via least squares.

3.1 Revision of the Structure Tensor

In this parameter estimation method a model with linear parameters p_j has to be given in the form $\mathbf{d}^{\mathrm{T}}\mathbf{p} = 0$ with data depending vector $\mathbf{d}$ and parameter vector $\mathbf{p}$. Equation 1.16 is of this form when identifying

$$\begin{aligned}
\mathbf{d} &= (I_x,\, I_y,\, I_x\Delta x,\, I_x\Delta y,\, I_x\Delta t,\, I_y\Delta x,\, I_y\Delta y,\, I_y\Delta t,\, I_{s_x},\, I_{s_y},\, I_t)^T\\
\mathbf{p} &= (u_x\mathrm{d}t + \nu\mathrm{d}s_x,\, u_y\mathrm{d}t + \nu\mathrm{d}s_y,\, a_{11}\mathrm{d}t + b_1\mathrm{d}s_x,\, a_{12}\mathrm{d}t + b_2\mathrm{d}s_x,\, a_{13}\mathrm{d}t + b_3\mathrm{d}s_x,\\
&\quad a_{21}\mathrm{d}t + b_1\mathrm{d}s_y,\, a_{22}\mathrm{d}t + b_2\mathrm{d}s_y,\, a_{23}\mathrm{d}t + b_3\mathrm{d}s_y,\, \mathrm{d}s_x,\, \mathrm{d}s_y,\, \mathrm{d}t)^T
\end{aligned} \tag{1.21}$$

In our case we have one equation for each 4D or 5D pixel. In order to construct an over-determined system of equations, one uses the assumption, that within a small neighborhood Ω of pixels i all equations are approximately solved by the same set of parameters $\mathbf{p}$, i.e.

$$\mathbf{d}_i^{\mathrm{T}}\mathbf{p} = e_i \text{ for all pixels } i \text{ in } \Omega \tag{1.22}$$

with errors e_i which have to be minimized by the sought for solution $\tilde{\mathbf{p}}$. Using a matrix D composed of the vectors $\mathbf{d}_i$ via $\mathbf{D}_{ij} = (\mathbf{d}_i)_j$ Equation 1.22 becomes $\mathbf{Dp} = \mathbf{e}$. We minimize $\mathbf{e}$ in a weighted 2-norm

$$||\mathbf{e}|| = ||\mathbf{Dp}|| = \mathbf{p}^{\mathrm{T}}\mathbf{D}^{\mathrm{T}}\mathbf{WDp} =: \mathbf{p}^{\mathrm{T}}\mathbf{Jp} \stackrel{!}{=} \min \tag{1.23}$$

where $\mathbf{W}$ is a diagonal matrix containing the weights. In our case Gaussian weights are used (see Section 4). The matrix $\mathbf{J} = \mathbf{D}^{\mathrm{T}}\mathbf{WD}$ is the so called structure tensor. The space of solutions $\tilde{\mathbf{p}}$ is spanned by the eigenvector(s) to the smallest eigenvalue(s) of $\mathbf{J}$. We call this space the null-space of $\mathbf{J}$ as the smallest eigenvalue(s) are 0 if the model is fulfilled perfectly, i.e. $\mathbf{e} = \mathbf{0}$. For standard optical flow the null-space is 1D, here it is 2D or 3D, depending on the model used. If there is not enough variation in the data, the so called aperture problem occurs, meaning the null-space has a higher dimension as indicated by the model. Here, we assume that enough data variation is present. Handling of other cases can be deduced from literature (e.g. [34]).

Solving for Flow Components. For our models the solution space of 1.23 is spanned by eigenvectors $\tilde{\mathbf{p}}_\mathbf{i}$ of $\mathbf{J}$ (where $i \in \{1, 2\}$ using a 1D camera grid or $i \in \{1, 2, 3\}$ using a 2D camera grid). The components $(\tilde{p}_i)_j$ of these eigenvectors always represent two flow components (cmp. 1.21), one depending on time and coming together with $\mathrm{d}t$ and the other depending on a camera displacement direction ($\mathrm{d}s_x$ or $\mathrm{d}s_y$).

We solve for time depending flow parameters (u_x, u_y, a_{11}, a_{12}, a_{13}, a_{21}, a_{22}, a_{23}) by linearly combining solution vectors $\tilde{\mathbf{p}}_\mathbf{i}$ such that $\mathrm{d}t = 1$ and $\mathrm{d}s_x = \mathrm{d}s_y = 0$, where $\mathrm{d}s_x$, $\mathrm{d}s_y$ and $\mathrm{d}t$ are components 9 to 11, respectively. Components 1 to 8 of the resulting vector then correspond to u_x, u_y, a_{11}, a_{12}, a_{13}, a_{21}, a_{22}, and a_{23} respectively.

When using a 1D camera grid $\mathrm{d}s_y$ already vanishes and we only need to linearly combine solution vectors $\tilde{\mathbf{p}}_\mathbf{i}$ such that $\mathrm{d}t = 0$ and $\mathrm{d}s_x = 1$. Depth related flow parameters ν, b_1, b_2, b_3 then correspond to components 1, 3, 4, and 5, respectively.

When using a 2D camera grid, 2 linear independent eigenvectors with $\mathrm{d}t = 0$ can be derived. One with $\mathrm{d}s_x = 1$ and $\mathrm{d}s_y = 0$, and the other with $\mathrm{d}s_x = 0$ and $\mathrm{d}s_y = 1$. Thus we could estimate v, b_1, b_2 and b_3 from either of the 2 eigenvectors, but unfortunately the two estimates for each flow parameter are neither identical, nor independent. This becomes clear considering the following example. Assume our cameras are looking at a striped pattern oriented in y-direction, i.e. parallel to s_y. Then a 1D camera grid shifting along x-direction will easily determine depth from disparity ν, but a camera grid shifting along

y-direction suffers from the aperture problem and can not determine depth. A 2D camera grid also can only resolve depth from its s_x-movement, but not from s_y-movement. In this example the two estimates are independent. Now think of a striped pattern not oriented along x- or y-direction. The two ν-estimates are now coupled and we need to combine them according to their error covariance matrix $\mathbf{C}$. We do so by calculating $\mathbf{C}$ following Nestares and Fleet [25], and rotating the representation of s_x-s_y-space such that $\mathbf{C}$ becomes diagonal. Then the resulting single estimate for ν is a sum of the independent estimates weighted by the inverse of their individual errors

$$\nu = \frac{\nu_1 C_{11}^{-0.5} + \nu_2 C_{22}^{-0.5}}{C_{11}^{-0.5} + C_{22}^{-0.5}} \tag{1.24}$$

where ν_1 and ν_2 are the two independent disparity estimates and C_{11} and C_{22} are the diagonal entries of the diagonalised covariance matrix $\mathbf{C}$.

3.2 Disentangling Surface Patch Parameters

Disentangling the flow parameters into parameters describing the local surface patch means solving the nonlinear system of Equations 1.19 and 1.20 together with the range flow constraint (Equation 1.18). We have 7 unknowns (U_X, U_Y, U_Z, Z_0, Z_X, Z_Y, Z_t) and one equation per estimated flow component, i.e. 12, and the range flow constraint. We have an overdetermined system of equations and consequently some choices how to solve it.

In this contribution we want to be able to compare our results with the range flow method [33]. Range flow needs a spatio-temporal depth map $Z_0(x,t)$ as input. Consequently we solve for depth related parameters Z_0, Z_X, Z_Y first and then resolve motion components U_X, U_Y, and U_Z. We may or may not solve for and use Z_t as we are not explicitly interested in it.

The first step is solving for depth and normals. From the estimates for the flow components ν, b_1, b_2 and b_3 (cmp. Section 3.1) and Equations 1.20 and 1.6 we first derive Z_0 using $Z_0 = f/\nu$ and then solve for Z_X, Z_Y, Z_t, and c. Focal length f has to be known e.g. from a calibration step.

The second step is solving for motion components. To do so we substitute Equations 1.20 into Equations 1.19 and get

$$\begin{aligned}
u_x &= -\nu\left(U_X - \tfrac{x_0}{f}U_Z\right) & u_y &= -\nu\left(U_Y - \tfrac{y_0}{f}U_Z\right)\\
a_{11} &= -b_1\left(U_X - \tfrac{x_0}{f}U_Z\right) + \tfrac{\nu U_Z}{f} & a_{21} &= -b_1\left(U_Y - \tfrac{y_0}{f}U_Z\right)\\
a_{12} &= -b_2\left(U_X - \tfrac{x_0}{f}U_Z\right) & a_{22} &= -b_2\left(U_Y - \tfrac{y_0}{f}U_Z\right) + \tfrac{\nu U_Z}{f}\\
a_{13} &= -b_3\left(U_X - \tfrac{x_0}{f}U_Z\right) & a_{23} &= -b_3\left(U_Y - \tfrac{y_0}{f}U_Z\right)
\end{aligned} \tag{1.25}$$

There are several ways to solve for U_X, U_Y, and U_Z

Full affine model. Having all flow parameters at hand, we may solve this system of 8 equations for U_X, U_Y, and U_Z in least squares sense or, better, in a weighted least squares sense respecting errors in flow components. This has not yet been investigated and is beyond the scope of this paper.

Standard affine model. Using a standard affine optical flow model without Δt-terms, we ignore equations with a_{13} and a_{23} and use the remaining 6 equations. When the modeled surface patch is fronto-parallel, i.e. $Z_X = Z_Y = 0$, we see that U_Z is estimated from the affine components a_{11} and a_{22}, i.e. divergence terms, whereas U_X and U_Y are estimated from the translational components of the affine optical flow.

Range flow from camera motion. Apart from the motion information in the temporal affine flow components (Equations 1.19) we may use the range flow constraint (Equation 1.18). Combining this constraint with the first two equations of Equation 1.25, i.e standard optical flow yields three equations for the three unknowns U_X, U_Y, and U_Z. This system of equations can be solved straight forward.

Range flow from depth field. An even simpler flow model may be estimated when not using b_3. Then no temporal affine terms need to be calculated. However then Z_t in the range flow constraint needs to be replaced by $\partial_t Z_0$ the partial temporal derivative of the estimated depth Z_0. This means similar to range flow then depth has to be estimated in advance. As before we get three equations and three unknowns and solve straight forward.

Obviously these estimation procedures may be mixed. As an example we simultaneously use range flow from camera motion and the standard affine model.

4 Experiments

We use synthetic sinusoidal sequences for systematic error analysis of the presented models. In a second experiment we compare the models using more realistic data, i.e. a translating cube rendered with POV-Ray [7]. In Section 3.2 we presented different ways to solve for 3D motion. Especially terms determining U_Z are switched on and of in the different model equations. For evaluation we compare the following models

- Model 1 uses only the temporal parameter vector, i.e. estimation of U_Z from divergence of intensity data,
- Model 2 uses the affine temporal component b_3 ($\Delta t \mathrm{d}s_x$-term) for estimation of U_Z,
- Model 3 uses the partial temporal derivative of the depth field $\partial_t Z_0$ instead of Z_t from the affine temporal component b_3,
- Model 4 is a combination of model 1 and 2, i.e. we use information from divergence and the affine temporal component b_3, equally weighted.

For reference we compare these models to range flow [33]. We only show experiments using a 1D camera grid, because results for a 2D grid were quite comparable and did not give extra insights.

Derivatives occurring in the model are discretized with optimized separable 5-tab filter sets presented in [27]. Systematic discretization errors are therefore minimal, but of course influence presented results.

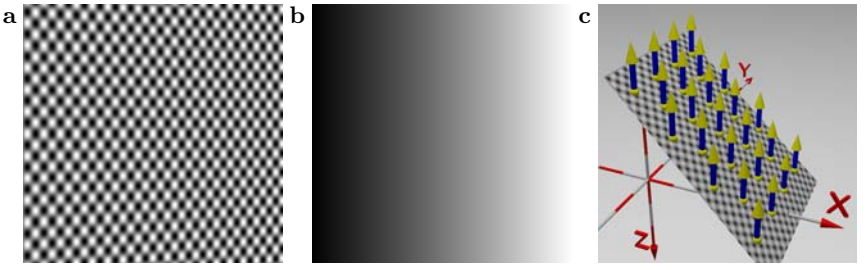


Fig. 5. (a) Central image of a sinusoidal sequence, (b) grey value coded depth map and (c) rendered patch in 3D with ground truth velocities

4.1 Sinusoidal Pattern

For systematic error analysis we modeled a surface patch with a sinusoidal pattern The intensity values of the sequence are in the range $[-1; 1]$. Figure 5 shows a rendered surface patch with slopes $Z_X = 1$ and $Z_Y = 0$, the grey value coded depth map for Z_0 ($Z_0 = 100$ mm for the central pixel) and the rendered patch with velocities $U_X = 0.00733$ mm/frame, $U_Y = -0.00367$ mm/frame and $U_Z = -0.6$ mm/frame. We imply optical flow values well below 1 pixel/frame to avoid errors coming from structure tensor estimation. We again want to stress that the estimation scheme is kept as simple as possible in order to show *model* effects. A coarse to fine scheme of course would allow for larger object movements.

The synthetic sensor contains 401×401 pixels of size 0.0044 mm^2. The focal length f of the projective camera is set to 12 mm. This pixel size and focal length are realistic in our lab setups.

We compare performance of the introduced models for surface patches with different inclinations and velocities. Furthermore we investigate the effect of additive gaussian noise with standard deviation $\sigma_n = 0.0032$ and of two baselines with $d_1 = 0.01$ mm ('near' baseline) and $d_2 = 0.5$ mm ('wide' baseline). Near baseline needs no preprocessing of the input sequences as optical flow between camera positions is below 1 pixel. For wide baseline we shift images in camera displacement direction towards the image of the central camera to get smallest image motion for the central pixel (here we have a preshift of 14 pixel). Because of this preprocessing we cropped the effective image size to 301×301 pixel in order to avoid border effects. These small baselines are selected such that estimator effects are minimized and model effects dominate the experiments. E.g. the wider the baseline, the smaller the maximum surface slope Z_X a stereo method can handle is. This is visible in Figure 7. Simulating a camera grid with a single camera on a moving stage (cmp. Figure 1), such small baselines are accessible in real experiments.

We use the optimized $5 \times 5 \times 5$ filter sets proposed by [29]. The 3D structure tensor weighting matrix $\mathbf{W}$ (cmp. Equation 1.23) has a spatio-temporal size

of 65×65 pixel and 5 frames and standard deviation of $\sigma_W = 19$ in space and $\sigma_W = 1$ in time direction. We use the same parameters for all sequences. Handling large neighborhoods is more challenging for a local model than using small neighborhoods, as errors due to missing (or even wrong) higher order terms influence results more.

For our analysis we compare the standard deviation of the mean absolute value of the relative error

$$Err_U = \frac{1}{N} \sum_{i}^{N} \frac{|U_{estimated} - U_{reference}|}{|U_{reference}|}. \tag{1.26}$$

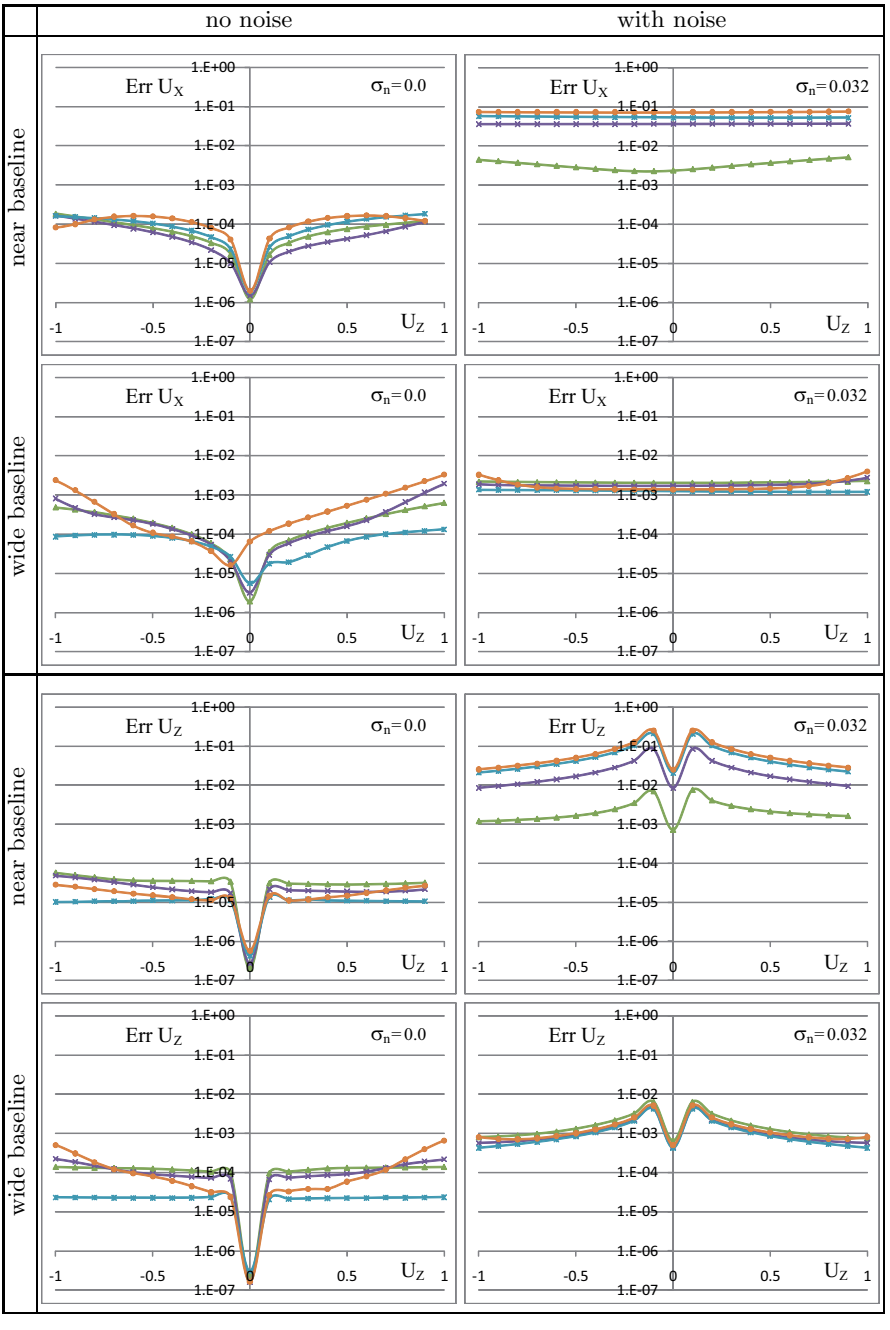
over all pixel N at a minimum distance of 70 pixel from the nearest image border. Errors are depicted for U_X and U_Z, i.e. the parameters range flow can also handle (errors of U_Y showed same characteristics as of U_X). We show their developing with respect to changing depth velocity U_Z (Figure 6) and surface inclination Z_X (Figure 7). Four different experiments are depicted: noise-free (left) and noisy (right) image sequences for both near and wide baselines. The figures show errors of U_X (top) and U_Z (bottom) for increasing U_Z and Z_X, respectively.

In Figure 6 we observe the following:

- Generally, when no depth motion is present, i.e. $U_Z = 0$, all models perform one to two orders of magnitude better than with motions in the range of 1/1000 of the average depth.
- Non-surprisingly errors are generally smaller for noise free data and when using wide baseline setups.
- For noise free data all novel models show errors similar to range flow, the difference is almost always less than one order of magnitude. For U_X model 4 performs worst and always worse than range flow. Errors of at least one of the other novel models are smaller than or equal to the ones of range flow in all investigated cases.
- In the wide baseline case with noise all models perform the same. Errors of U_X are approximately independent from U_Z for noisy sequences while for noise-free data errors increase for increasing U_Z.
- In the near baseline case with noise model 1 clearly outperforms all other models by 1 to 1.5 orders of magnitude. It shows approximately the same performance as in the wide baseline case.

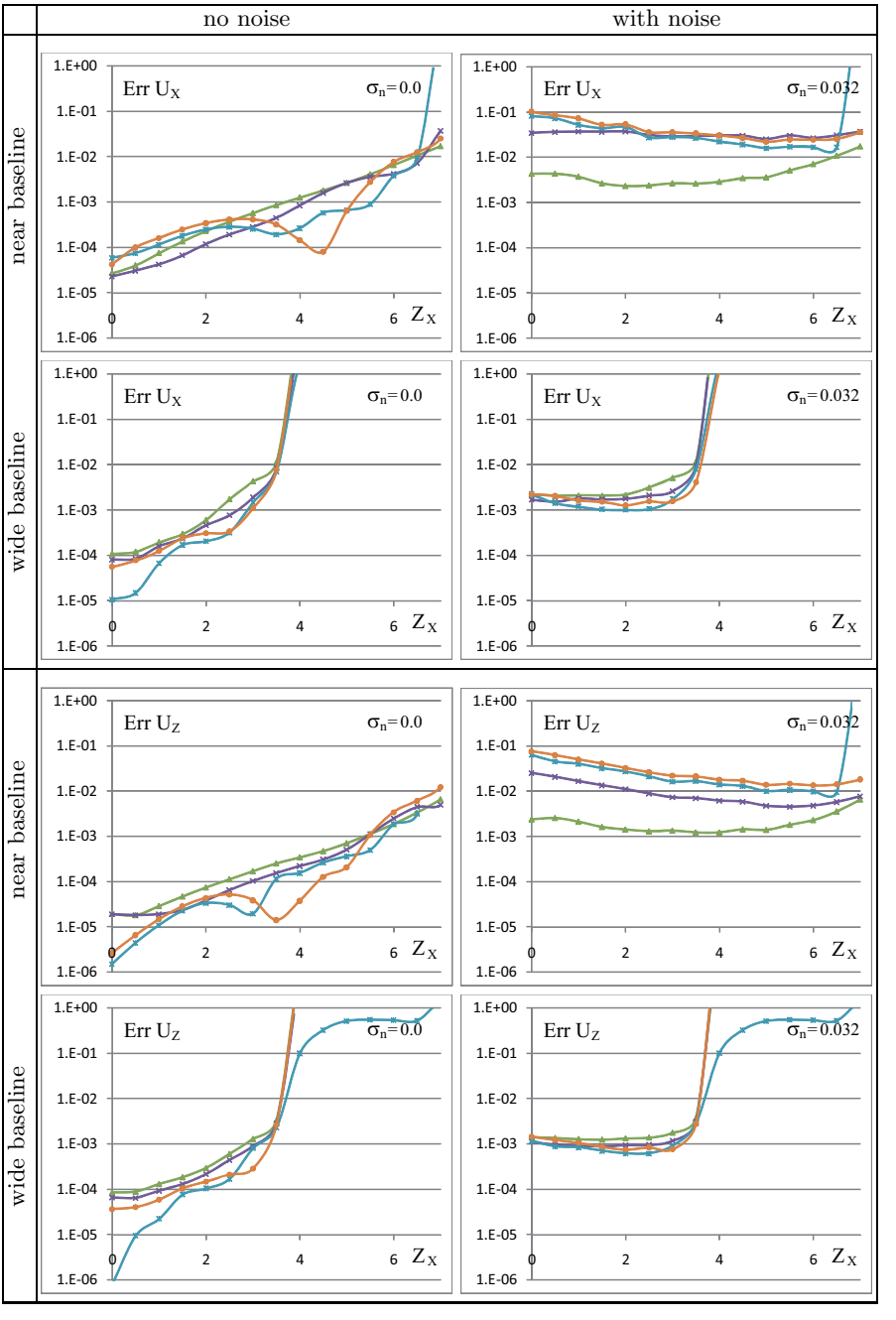
In Figure 7 we observe the following:

- In the noise free case, all models perform best for fronto-parallel surfaces, i.e. $Z_X = 0$.
- In the wide baseline case estimation breaks down for steep surfaces, i.e. when $Z_X \gtrapprox 4$. This is not the case for near baseline setups.
- As with increasing U_Z in the wide baseline case with noise all models perform the same. Errors of U_X and U_Z are approximately independent from Z_X for noisy sequences while for noise-free data errors increase for increasing Z_X.



Model 1 ▲, model 2 ●, model 3 ✱, model 4 ✕, range flow ✱.

Fig. 6. Standard deviation of Err_U of U_X (top box) and U_Z (bottom box) versus U_Z



Model 1, model 2, model 3, model 4, range flow.

Fig. 7. Standard deviation of Err_U of U_X (top box) and U_Z (bottom box) versus Z_X

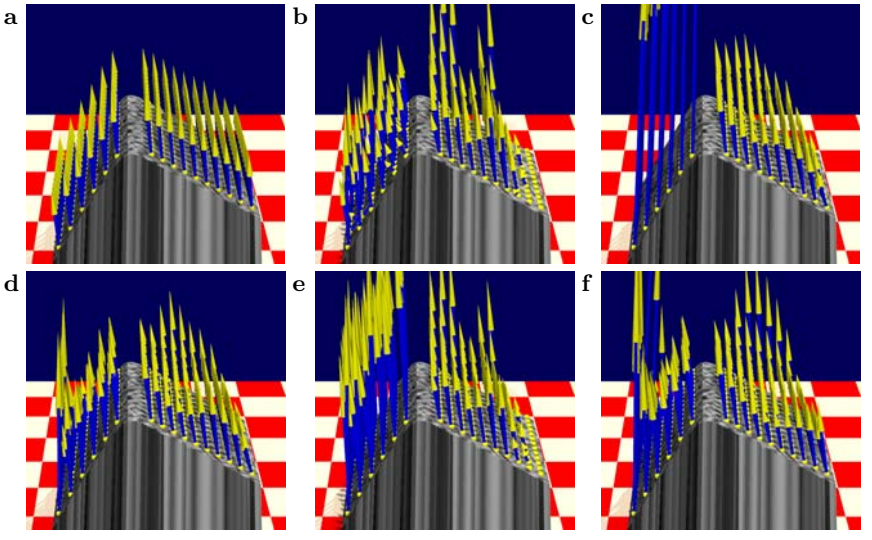


Fig. 8. (a) Velocity and depth estimates using Model 3 with no amplification. In all other images errors are amplified by 100 in U_X-U_Y-direction and 50 in U_Z-direction for better visibility. Model 1 (b), model 2 (c), model 3 (d), model 4 (e) and range flow (f).

– Also as with increasing U_Z in the near baseline case with noise model 1 clearly outperforms all other models by 1 to 1.5 orders of magnitude. Also here it shows approximately the same performance as in the wide baseline case.

We conclude that the novel models are accurate. Discretization and parameter estimation are relevant and need to be further investigated. As a method the novel models 1 to 3 are as good as range flow in different application scenarios. Especially when noise plays a role in the acquired data, model 1 together with the structure tensor is the currently best performing method. Furthermore we see that near baseline allows reliable motion estimates for slopes up to $Z_X = 7$ whereas for wide baseline the maximum slope is $Z_X = 4$.

4.2 Synthetic Cube

The synthetic cube sequence allows us to compare the models on more realistic data with ground truth available. The cube is moving with $U_X = -0.2$ mm/frame, $U_Y = 0$ mm/frame and $U_Z = -2$ mm/frame and is covered with a constant noise pattern in order to make local estimation reliable. With more realistic textures estimation effects may have more influence disturbing model evaluation. Illumination remains constant to fulfill the brightness constancy constraint.

The weighting matrix $\mathbf{W}$ is the same as for the sinusoidal sequences. The multi-camera image sequence is generated for 5 cameras with a displacement of 5 mm. The baseline results in a preshift of 14 pixel.

Figure 8 shows velocity and depth estimates of the proposed models ray-traced using POV-Ray [7]. Figure 8a shows estimation results of model 1. Motion vectors are visually indistinguishable from the ones of the ground truth. We are interested in high accuracy of the estimated motion. In Figures 8b-f errors are amplified by 100 in $U_X - U_Y$-direction and 50 in U_Z-direction for better comparison of the models. Figures 8b-e show amplified results of models 1 to 4, respectively and Figure 8f shows amplified results for range flow. Velocity estimates for model 1 and model 4 (Figures 8b and e, respectively) are distorted on both sides of the cube. Model 2 (Figure 8c) yields most reliable estimates on the right side of the cube whereas on the left, steeper side the estimates are quite bad. This is due to inaccurate estimation of the affine component b_3 (i.e. $\Delta t \mathrm{d}s_x$-term). For the steep side estimates by model 3 and range flow are better apart from estimates at the border of the cube.

5 Summary and Conclusions

We derived a novel model for simultaneous estimation of surface depth and slopes as well as 3D motion. Model parameters can be estimated in different ways and an optimal estimator using all terms simultaneously is not yet available. We therefore tested the performance of different term combinations or (sub-)models (named Model 1 to 4 in Section 4) using a simple total least squares estimator and synthetic data. Motion estimates delivered by these sub-models are compared to range flow as reference. Generally speaking the models are in the same accuracy range as range flow, meaning modeling of the local neighborhood is accurate. Wide baseline setups generally yield better results than near baseline setups. However, when steeply inclined surfaces need to be handled experiments on sinusoidal patterns show that near baseline setups are preferable. For less inclined surfaces and when data suffers from noise model 1 then with a near baseline setup still yields the same accuracy as with wide baselines. This makes model 1 the first choice in noisy scenarios.

In the wide baseline, low noise scenario range flow and model 3 perform the same and best in the sinusoidal pattern experiment. This is the scenario we chose for the cube experiment. Model 3 performs a little bit better than range flow, but not significantly.

Overall we conclude that the novel model is accurate even when using a large neighborhood of 65×65 pixel. What still needs to be developed in order to construct a *method* suitable for our target application is a robust, possibly variational estimator using all terms of the model.

References

1. Adiv, G.: Determining 3-d motion and structure from optical flow generated by several moving objects. IEEE Trans. on Pattern Analysis and Machine Intelligence 7(4), 384–401 (1985)

2. Barron, J., Fleet, D., Beauchemin, S.: Performance of optical flow techniques. International Journal of Computer Vision 12(1), 43–77 (1994)
3. Bigün, J., Granlund, G.H.: Optimal orientation detection of linear symmetry. In: International Conference On Computer Vision, pp. 433–438 (1987)
4. Black, M., Fleet, D., Yacoob, Y.: Robustly estimating changes in image appearence. Computer Vision and Image Understanding 7(1), 8–31 (2000)
5. Bruhn, A., Weickert, J., Schnörr, C.: Lucas/kanade meets horn/schunck: Combining local and global optic flow methods. International Journal of Computer Vision 61(3), 211–231 (2005)
6. Carceroni, R., Kutulakos, K.: Multi-view 3d shape and motion recovery on the spatio-temporal curve manifold. In: International Conference On Computer Vision, vol. (1), pp. 520–527 (1999)
7. Cason, C.: Persistence of vision ray tracer (POV-Ray), version 3.6, Windows (2005)
8. Denney, T.S.J., Prince, J.L.: Optimal brightness functions for optical flow estimation of deformable motion. IEEE Trans. Im. Proc. 3(2), 178–191 (1994)
9. Farid, H., Simoncelli, E.P.: Optimally rotation-equivariant directional derivative kernels. In: 7th Int'l Conf. Computer Analysis of Images and Patterns, Kiel (1997)
10. Farnebäck, G.: Fast and accurate motion est. using orient. tensors and param. motion models. In: International Conference on Pattern Recognition, pp. 135–139 (2000)
11. Fleet, D., Black, M., Yacoob, Y., Jepson, A.: Design and use of linear models for image motion analysis. International Journal of Computer Vision 36(3), 171–193 (2000)
12. Fleet, D., Langley, K.: Recursive filters for optical flow. Pattern Analysis and Machine Intelligence 17(1), 61–67 (1995)
13. Fleet, D., Weiss, Y.: Optical flow estimation. In: Mathematical models for Computer Vision: The Handbook. Springer, Heidelberg (2005)
14. Hartley, R.I., Zisserman, A.: Multiple View Geometry in Computer Vision, 2nd edn. Cambridge University Press, Cambridge (2004)
15. Haußecker, H., Fleet, D.: Computing optical flow with physical models of brightness variation. IEEE Transactions on Pattern Analysis and Machine Intelligence 23(6), 661–673 (2001)
16. Haußecker, H., Spies, H.: Motion. In: Jähne, B., Haußecker, H., Geißler, P. (eds.) Handbook of Computer Vision and Applications. Academic Press, London (1999)
17. Horn, B., Schunck, B.: Determining optical flow. Artificial Intelligence 17, 185–204 (1981)
18. Jähne, B., Scharr, H., Körkel, S.: Principles of filter design. In: Handbook of Computer Vision and Applications. Academic Press, London (1999)
19. Kanatani, K.: Structure from motion without correspondence: general principle. In: Proc. Image Understanding Workshop, pp. 10711–10716 (1985)
20. Matthies, L.H., Szeliski, R., Kanade, T.: Kalman filter-based algorithms for estimating depth from image sequences. International Journal of Computer Vision 3, 209–236 (1989)
21. Li, G., Zucker, S.: Differential geometric consistency extends stereo to curved surfaces. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006. LNCS, vol. 3953, pp. 44–57. Springer, Heidelberg (2006)
22. Longuet-Higgins, H., Prazdny, K.: The interpretation of a moving retinal image. In: Proceedings of The Royal Society of London B, vol. 208, pp. 385–397 (1980)
23. Lucas, B., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: Proc. Seventh International Joint Conference on Artificial Intelligence, Vancouver, Canada, August 1981, pp. 674–679 (1981)

24. Nakamura, Y., Matsuura, T., Satoh, K., Ohta, Y.: Occlusion detectable stereo–occlusion patterns in camera matrix. In: International Conference on Computer Vision and Pattern Recognition, pp. 371–378 (1996)
25. Nestares, O., Fleet, D., Heeger, D.: Likelihood functions and confidence bounds for total-least-squares problems. In: IEEE Conference on Computer Vision and Pattern Recognition, Hilton Head, South Carolina, vol. I, pp. 523–530 (2000)
26. Papenberg, N., Bruhn, A., Brox, T., Didas, S., Weickert, J.: Highly accurate optic flow computation with theoretically justified warping. International Journal of Computer Vision 67(2), 141–158 (2006)
27. Scharr, H.: Optimal Operators in Digital Image Processing. PhD thesis, Interdisciplinary Center for Scientific Computing, University of Heidelberg, Germany (2000)
28. Scharr, H.: Towards a multi-camera generalization of brightness constancy. In: Jähne, B., Mester, R., Barth, E., Scharr, H. (eds.) IWCM 2004. LNCS, vol. 3417, pp. 78–90. Springer, Heidelberg (2007)
29. Scharr, H.: Optimal filters for extended optical flow. In: Jähne, B., Mester, R., Barth, E., Scharr, H. (eds.) IWCM 2004. LNCS, vol. 3417, pp. 14–29. Springer, Heidelberg (2007)
30. Scharr, H., Schuchert, T.: Simultaeous estimation of depth, motion and slopes using a camera grid. In: Kobbelt, T.A.L., Kuhlen, T., Westermann, R. (eds.) Vision Modeling and Visualization 2006, Aachen, pp. 81–88 (2006)
31. Schuchert, T., Aach, T., Scharr, H.: Range flow for varying illumination. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part I. LNCS, vol. 5302, pp. 509–522. Springer, Heidelberg (2008)
32. Schuchert, T., Scharr, H.: Simultaneous estimation of surface motion, depth and slopes under changing illumination. In: Hamprecht, F.A., Schnörr, C., Jähne, B. (eds.) DAGM 2007. LNCS, vol. 4713, pp. 184–193. Springer, Heidelberg (2007)
33. Spies, H., Jähne, B., Barron, J.: Range Flow Estimation. Computer Vision and Image Understanding 85(3), 209–231 (2002)
34. Spies, H., Jähne, B.: A general framework for image sequence analysis. In: Fachtagung Informationstechnik, pp. 125–132 (2001), http://citeseerx.ist.psu.edu/viewdoc/summary?, doi:10.1.1.21.1678
35. Szeliski, R.: A multi-view approach to motion and stereo. In: International Conference on Computer Vision and Pattern Recognition (1999)
36. Vedula, S., Baker, S., Rander, P., Collins, R., Kanade, T.: Threedimensional scene flow. In: International Conference On Computer Vision 1999, pp. 722–729 (1999)
37. Vedula, S., Baker, S., Seitz, S., Collins, R., Kanade, T.: Shape and motion carving in 6d. In: International Conference on Computer Vision and Pattern Recognition 2000, pp. 592–598 (2000)
38. Waxman, A., Kamgar Parsi, B., Subbarao, M.: Closed-form solutions to image flow equations for 3d structure and motion. International Journal on Computer Vision 1(3), 239–258 (1987)

Deinterlacing with Motion-Compensated Anisotropic Diffusion

Matthias Ghodstinat, Andrés Bruhn, and Joachim Weickert

Mathematical Image Analysis Group
Faculty of Mathematics and Computer Science, Building E1.1
Saarland University, 66041 Saarbrücken, Germany
{ghodstinat,bruhn,weickert}@mia.uni-saarland.de

Abstract. We present a novel deinterlacing scheme that makes consequent use of discontinuity-preserving partial differential equations (PDEs). It combines the accuracy of recent variational motion estimation techniques with the directional interpolation qualities of anisotropic diffusion filters. Our algorithm proceeds in three steps: First, we interpolate the interlaced images by means of a spatial edge enhancing diffusion process (EED). Then we apply the variational optic flow technique of Brox et al. (2004) in order to obtain a precise interframe registration. Finally we use a spatiotemporal generalisation of EED for motion-compensated inpainting of the missing data in the original sequence. Experiments demonstrate that the proposed method outperforms not only classical deinterlacing schemes, but also a recent PDE-based approach.

1 Introduction

Since the beginning of television more than 70 years ago, interlacing is the predominant sampling technique for recording and transmitting video. Although it was originally developed to increase the frame rate of television sets based on cathode ray tubes, it forms nowadays the basis for all analogue broadcast systems (PAL, SECAM, NTSC). Interlacing is based on a simple idea: Instead of considering the complete image domain of a video sequence, alternatingly only the even-indexed and the odd-indexed lines of an image are stored, respectively. While such a proceeding reduces the vertical resolution, it allows to double the effective frame rate at the same time. This in turn improves the temporal smoothness of the video and thus prevents from large area flickering due to fast motion. Although interlacing proved to be a good compromise between quality and bandwidth consumption, it became obsolete in the era of high definition television (HDTV). Today's digital devices such as plasma screens or LCD panels even require scanline converters to display video streams that are originally interlaced. Thus the problem arises how interlaced video data can be converted to progressive image content, i.e. to video sequences that offer the full vertical resolution. This task that requires the filling-in of missing information at even and odd lines of subsequent video frames is referred to as *deinterlacing* [3,31,32]. It is closely related to *image* or *video inpainting* [2,10,11] where

D. Cremers et al. (Eds.): Visual Motion Analysis, LNCS 5604, pp. 91–106, 2009.

missing information at arbitrary locations has to be restored and *video superresolution* [13,21,31,36] that aims at improving both spatial and vertical resolution by combing information from several consecutive frames of an image sequence.

Deinterlacing has been researched for about 20 years and various algorithms have been proposed in the literature to tackle this problem. Depending on their strategy, these algorithms can be classified into four types of methods [2,22]: The simplest methods are *non-adaptive linear techniques*. Such methods fill in missing information by simply repeating or averaging lines in vertical or temporal direction [3,24,32]. *Directional interpolation methods* follow a slightly different strategy. In order to allow the preservation of small image details they adapt the interpolation process to the orientation of the local image structure [3,31,37]. Thus they succeed to improve the quality in heavily textured image regions, where sufficient information from the spatial neighbourhood is available. A third class of methods is given by so-called *motion adaptive algorithms*. These techniques respect the motion of objects during the interpolation process by locally switching between spatial and temporal information [2,8,17,22,28]. The type of information that is actually used for filling in the missing information is thereby determined by analyzing the local motion situation. In general, this is done by means of a simple motion detector. The fourth and most advanced class of deinterlacing techniques are so-called *motion compensated approaches*. In contrast to their motion adaptive counterparts these techniques make use of a real motion estimator. Thus they are able to correct the image sequence by the occurring motion either before or during the actual interpolation step [1,4,10,23,29].

Due to their ability to consider motion information in the deinterlacing process, motion adaptive and motion compensated approaches give the best results out of these four classes [2,22,23,27]. However, both strategies can hardly be compared, since they have quite complementary advantages and shortcomings: While motion adaptive techniques can provide very accurate results, they are known to have severe problems with large displacements [22]. In this case they switch back to pure spatial interpolation and do not exploit the full temporal information. Motion compensated methods, on the other hand, can handle large displacements, but may suffer from inaccuracies of their motion estimators [4].

The goal of this paper is thus to combine the advantages of both strategies within a single algorithm. Close in spirit to the work in [26] that proposes a successive refinement of the deinterlacing results, we propose a three step method that makes consequent use of discontinuity-preserving partial differential equations (PDEs). While the variational optic flow technique of Brox *et al.* [5] is used to provide accurate displacement fields for the motion compensation step, edge-enhancing anisotropic diffusion filters (EED) [34] are applied before and after the correction of the image sequence to perform pure spatial and motion adaptive spatiotemporal interpolation, respectively. Quality benchmarks with four different scenes show a very good performance of the combined method: It not only allows to deinterlace images with small and large motion, it even preserves small details in both cases. Comparisons to classical interlacing methods and a recent PDE-based technique demonstrate the clear superiority of our approach. Please

note that our algorithm is intended to serve as an offline tool. Nevertheless, we point out ways to speed up the computation such that real-time performance can be achieved.

Our paper is organised as follows. In Section 2 we briefly present the main idea of our algorithm and motivate the use of its different components. While Section 3 discusses how to apply spatial edge enhancing anisotropic filtering in the context of image inpainting, Section 4 is dedicated to variational methods for motion estimation and motion compensation. In Section 5 the motion adaptive part of our algorithm is explained. Here, the spatial model for edge enhancing anisotropic inpainting is generalised to the spatiotemporal domain. Finally, Section 6 gives a detailed evaluation of our approach as well as comparisons to other methods from the literature. A summary in Section 7 concludes the paper.

2 A Novel Three Step Algorithm

Let us consider an interlaced RGB image sequence $\mathbf{f}(\mathbf{x}, t)$, where $\mathbf{f} = (f_1, f_2, f_3)^\top$ stands for the different colour channels, $\mathbf{x} = (x, y)^\top$ denotes the location within a rectangular image domain Ω and $t \geq 0$ denotes time. Let us furthermore specify the subsets of odd-indexed and even-indexed lines in the image domain as Ω_o and Ω_e with $\Omega_\mathrm{o} \cup \Omega_\mathrm{e} = \Omega$. Then, assuming w.l.o.g. that the first frame only contains odd-indexed lines, the deinterlacing problem comes down to recovering $\mathbf{f}(\mathbf{x}, t)$ for $\mathbf{x} \in \Omega_\mathrm{e}$ if t is even and $\mathbf{f}(\mathbf{x}, t)$ for $\mathbf{x} \in \Omega_\mathrm{o}$ if t is odd. In order to solve this problem we propose the following PDE-based strategy that is based on three consecutive steps:

(1) *Spatial Deinterlacing.* In order to allow the estimation of full resolution displacement fields in our motion compensation step (Step 2), we have to deinterlace our input image sequence first. However, since the images are not yet motion compensated, the use of temporal information is not recommendable. Thus, we restrict ourselves in our first step to a pure *spatial* interpolation process that is anisotropic [35]. It is based on the 2-D edge enhancing diffusion (EED) scheme [34] that already proved its favourable performance in the context of image interpolation for image (de)compression [15,16].

(2) *Motion Estimation and Compensation.* In order to keep displacements small and thus the accuracy high, only blocks of three consecutive frames are considered in our second step. Its goal is to register the first and the last frame of each block onto the central one. Thus, even in the context of large displacements, the temporal information becomes spatially aligned such that it can be easily used for interpolation. For computing the required displacement fields, we make use of the variational optic flow approach of Brox *et al.* [5]. This discontinuity-preserving method is among the most precise techniques for motion estimation in terms of error measures.

(3) *Motion Compensated Anisotropic Diffusion.* In the final step of our algorithm we recompute the originally missing lines of each frame using *spatiotemporal*

information. To this end, we consider the temporally aligned blocks from Step 2 and apply a spatiotemporal and thus motion adaptive variant of the EED-based interpolation process from Step 1. Due to the anisotropic nature of EED, this allows to recover details that have not been aligned correctly in the motion compensation step.

After we have explained the basic outline of our new deinterlacing scheme, let us now discuss its three basic steps in more detail.

3 Spatial Deinterlacing

Following our strategy from the previous section, we start by deinterlacing all frames separately. This can be seen as a special instance of our original problem for a fixed time t, where the set of known data points Ω_+ is either given by Ω_e or Ω_o. Our goal is now to find for each frame $\mathbf{f}(\mathbf{x}, t)$ a deinterlaced version $\mathbf{u}(\mathbf{x}, t)$ that is smooth in $\Omega \setminus \Omega_+$ and identical to $\mathbf{f}$ in Ω_+. In order to solve this task, we propose to use a spatial deinterlacing process based on edge enhancing anisotropic diffusion (EED) [34]. Such schemes have already been applied successfully in the context of image inpainting, where they provided excellent interpolation results [15,16]. Given an interlaced RGB image at time $t = t_0$, EED computes the corresponding deinterlaced result as steady state of the three coupled nonlinear evolution equations

$$\begin{aligned} \partial_\tau u_i = \operatorname{div}\big(D(\nabla u_1, \nabla u_2, \nabla u_3)\, \nabla u_i\big) \quad & \text{in } \Omega \times (0, \infty) \\ & \text{for } i = 1, 2, 3 \end{aligned} \tag{1}$$

with Neumann (reflecting) boundary conditions across image boundaries,

$$\partial_n \mathbf{u} = \mathbf{0} \text{ on } \partial\Omega\,, \tag{2}$$

Dirichlet boundary conditions that prescribe the solution at given lines,

$$\mathbf{u}(\mathbf{x}, t_0, \cdot) = \mathbf{f}(\mathbf{x}, t_0) \text{ in } \Omega_+\,, \tag{3}$$

and the initial condition that the original image is used where lines are available:

$$\mathbf{u}(\mathbf{x}, t_0, 0) = \begin{cases} \mathbf{f}(\mathbf{x}, t_0) & \text{in } \Omega_+ \\ \mathbf{0} & \text{in } \Omega \setminus \Omega_+ \end{cases}\,. \tag{4}$$

Here, u_i stands for the different RGB channels, $\nabla u_i = (u_{ix}, u_{iy})^\top$ denotes their spatial gradient, τ serves as evolution time which is a pure numerical parameter, and the 2×2 matrix $D(\nabla u_1, \nabla u_2, \nabla u_3)$ represents a diffusion tensor that couples the different colour channels. This tensor that steers the diffusion process is constructed from the eigenvectors $\mathbf{v}_1$, $\mathbf{v}_2$ and eigenvalues λ_1, λ_2 of the joint structure tensor [12,14]

$$J_{\rho,\sigma}(u_1, u_2, u_3) = K_\rho * \sum_{i=1}^{3} (\nabla u_i^\sigma)(\nabla u_i^\sigma)^\top\,. \tag{5}$$

In this context, $K_\rho *$ denotes convolution with a Gaussian K of standard deviation ρ and ∇u_i^σ indicates that the channel u_i has been presmoothed with a Gaussian of standard deviation σ prior to differentiation. Assuming the eigenvalues to be ordered in a decreasing manner, i.e. $\lambda_1 \geq \lambda_2$, the EED diffusion tensor finally reads

$$D(\nabla u_1, \nabla u_2, \nabla u_3) = (\mathbf{v}_1 \mid \mathbf{v}_2) \begin{pmatrix} \Psi_1(\lambda_1) & 0 \\ 0 & \Psi_2(\lambda_2) \end{pmatrix} \begin{pmatrix} \mathbf{v}_1^\top \\ \mathbf{v}_2^\top \end{pmatrix} \tag{6}$$

where the $\mid$ operator merges neighbouring vectors into a matrix and the functions applied to the first and to the second eigenvalue are given by

$$\Psi_1(s) = \frac{1}{\sqrt{1 + \frac{s}{\epsilon^2}}}\,, \qquad \Psi_2(s) = 1\,. \tag{7}$$

with ϵ being a contrast parameter. While the Charbonnier diffusivity [9] chosen for $\Psi_1(s)$ inhibits the diffusion process across the most dominant image structure, the second function $\Psi_2(s)$ allows smoothing along it. This in turn gives us the desired anisotropic edge-preserving behaviour that is characteristic for EED.

In order to compute the steady state, i.e. $\tau \to \infty$, of the coupled evolution equations given by (1)-(6), we used an explicit scheme based on finite difference approximations. Typical run times for such an implementation are in the order of about 3 seconds for images of size 300×300. Recent parallelisations of more efficient numerical schemes on the Cell Processor of Sony's Playstation 3 even achieve up to 34 frames per second [25], thereby dealing with an almost twice as large and less regular inpainting domain Ω_-. Taking those differences into account, run times of about 20 milliseconds can be expected for the spatial deinterlacing step, if modern parallel architectures are used.

4 Motion Estimation and Compensation

After we have deinterlaced at least three consecutive frames of the original image, we can continue with our second step: the motion estimation and compensation. To this end, we consider blocks of three consecutive frames, and register the first and the last frame onto the central one:

$$\mathbf{f}(\mathbf{x}, t_0 - 1) \quad \xrightarrow{\mathbf{w}^-} \quad \mathbf{f}(\mathbf{x}, t_0) \quad \xleftarrow{\mathbf{w}^+} \quad \mathbf{f}(\mathbf{x}, t_0 + 1)$$

Although it is possible to use more than three frames in this step, one should be aware that this results in larger displacements which in turn would deteriorate the quality of the motion estimation.

In order to compute the two motion fields $\mathbf{w}^+ = (v^+, w^+)^\top$ and $\mathbf{w}^- = (v^-, w^-)$ we make use of the highly precise variational optic flow technique of

Brox *et al.* [5]. For RGB images, this technique computes the displacement field between two consecutive frames as minimiser of the energy functional

$$E(\mathbf{w}^{\pm}) = \int_{\Omega} \psi\Big(\underbrace{\sum_{i=1}^{3} \left| f_i(\mathbf{x}+\mathbf{w}^{\pm}, t_0 \pm 1) - f_i(\mathbf{x}, t_0) \right|^2}_{\text{colour constancy}} + \gamma \underbrace{\sum_{i=1}^{3} \left| \nabla f_i(\mathbf{x}+\mathbf{w}^{\pm}, t_0 \pm 1) - \nabla f_i(\mathbf{x}, t_0) \right|^2}_{\text{colour gradient constancy}} \Big) \, d\mathbf{x} + \alpha \int_{\Omega} \psi\big(\underbrace{|\nabla v^{\pm}|^2 + |\nabla w^{\pm}|^2}_{\text{spatial smoothness}} \big) \, d\mathbf{x} \,. \tag{8}$$

While the first expression in the data term models the assumption that the colour of objects remains constant over time, the second one renders the approach more robust against varying illumination. This is achieved by assuming constancy of the spatial image gradient of the different colour channels given by $\nabla f_i = (f_{ix}, f_{iy})^\top$. The weighting between the two assumptions is realised with a positive scalar γ. In order to allow for a correct estimation of large displacements, both assumptions are used in their original nonlinear form. Please note that this is extremely important in the context of deinterlacing, since the typical motion is in the order of several pixels. Finally, deviations from both the data and the smoothness term are penalised in a non-quadratic way via a robust function ψ. This improves the performance of the approach with respect to outliers and noise in the case of the data term and preserves motion boundaries by modelling a piecewise smooth flow field in the case of the smoothness term. For both purposes the regularised L_1-norm is used, which, for the smoothness term, comes down to the total variation (TV) regulariser [30] given by $\psi(s^2) = \sqrt{s^2 + \epsilon^2}$. The regularisation parameter ϵ is set to 10^{-3}. In this context, one should note that without the preservation of motion boundaries, blurry flow information would lead to a filling-in of wrong temporal data at the corresponding image locations. This in turn would result in a significant deterioration of object boundaries in the deinterlaced image both with respect to quality and localisation.

The minimisation of this energy functional is done via its associated Euler-Lagrange equations. These equations are given by the following coupled system of nonlinear elliptic PDEs:

$$0 = \psi'(...)\Big(\sum_{i=1}^{3} \gamma_i \, \big(f_i(\mathbf{x}+\mathbf{w}^{\pm}, t_0 \pm 1) - f_i(\mathbf{x}, t_0)\big) \frac{\partial}{\partial x} f_i(\mathbf{x}+\mathbf{w}^{\pm}, t_0 \pm 1) \Big) + \alpha \, \mathrm{div}\left(\psi'\left(|\nabla u^{\pm}|^2 + |\nabla v^{\pm}|^2\right) \nabla u^{\pm}\right) , \tag{9}$$

$$0 = \psi'(...)\Big(\sum_{i=1}^{3}\gamma_i\,\big(f_i(\mathbf{x}+\mathbf{w}^{\pm},t_0\pm1)-f_i(\mathbf{x},t_0)\big)\frac{\partial}{\partial y}f_i(\mathbf{x}+\mathbf{w}^{\pm},t_0\pm1)\Big)$$
$$+\,\alpha\,\mathrm{div}\left(\psi'\left(|\nabla u^{\pm}|^2+|\nabla v^{\pm}|^2\right)\nabla v^{\pm}\right)\,, \tag{10}$$

where $\psi'_D(...)$ abbreviates the factor in front of the data term that actually reads

$$\psi'(...) = \psi'\Big(\quad\sum_{i=1}^{3}\left|f_i(\mathbf{x}+\mathbf{w}^{\pm},t_0\pm1)-f_i(\mathbf{x},t_0)\right|^2$$
$$+\,\gamma\,\sum_{i=1}^{3}\left|\nabla f_i(\mathbf{x}+\mathbf{w}^{\pm},t_0\pm1)-\nabla f_i(\mathbf{x},t_0)\right|^2\Big)\,. \tag{11}$$

As suggested in [5], we implemented a coarse-to-fine warping strategy based on two nested fixed point iterations. This yields typical run times of about 3 seconds for computing both the forward and the backward flow field for images of size 300×300. However, also in this case the computations can be accelerated significantly: Either one can use more sophisticated numerical schemes such as the nonlinear multigrid methods presented in [6,7] or can consider recent implementations on parallel architectures such as graphics hardware [18,33] or the Cell processor [19,20]. While efficient numerics already allow to reduce the computational effort to less than a second, highly parallel implementations promise run times of about 30 millisecond for both flow fields. Thus real-time seems also to be possible for the second step of our deinterlacing algorithm.

After the two displacement fields have been computed, all frames are registered towards the central one. To this end, we use a backward warping strategy based on bilinear interpolation. Alternatively, also higher order splines can be used, but first results did not show significant improvements in terms of quality. From a computational viewpoint, this effort is neglectable.

5 Motion Compensated Anisotropic Diffusion

In the third and last step of our algorithm we apply a spatiotemporal anisotropic diffusion process to *recompute* the originally missing lines of the central frame of each registered image block. To this end, we consider a spatiotemporal generalisation of the EED-based deinterlacing algorithm from Section 3. It is defined on the image domain of the complete block and reads

$$\partial_\tau u_i = \mathrm{div}\big(D(\nabla_t u_1,\nabla_t u_2,\nabla_t u_3)\,\nabla_t u_i\big) \quad \text{in } \Omega\times[t_0-1,t_0+1,]\times(0,\infty)$$
$$\text{for } i=1,2,3\,. \tag{12}$$

Here, $\nabla_t = (u_{ix}, u_{iy}, u_{it})^\top$ denotes the spatiotemporal gradient of u_i which allows to consider temporal information in contrast to the spatial one that we

used in Step 1. Consequently, also the joint structure tensor is extended to the spatiotemporal domain

$$J_{\rho,\sigma}(u_1, u_2, u_3) = K_\rho * \sum_{i=1}^{3} (\nabla u_i^\sigma)(\nabla u_i^\sigma)^\top \; . \tag{13}$$

Finally, the joint diffusion tensor is adapted correspondingly and reads now

$$D(\nabla u_1, \nabla u_2, \nabla u_3) = (\mathbf{v}_1 \mid \mathbf{v}_2 \mid \mathbf{v}_3) \begin{pmatrix} \Psi_1(\lambda_1) & 0 & 0 \\ 0 & \Psi_2(\lambda_2) & 0 \\ 0 & 0 & \Psi_3(\lambda_3) \end{pmatrix} \begin{pmatrix} \mathbf{v}_1^\top \\ \mathbf{v}_2^\top \\ \mathbf{v}_3^\top \end{pmatrix} , \tag{14}$$

where the functions of the three eigenvalues are defined as

$$\Psi_1(s) = \frac{1}{\sqrt{1 + \frac{s}{\epsilon^2}}} , \qquad \Psi_2(s) = 1 , \qquad \Psi_3(s) = 1 . \tag{15}$$

This yields an anisotropic diffusion process that smoothes only along the plane perpendicular to the local gradient of the evolving image, but not across it.

Actually, this process is fully *motion adaptive*: By inhibiting the diffusion in gradient direction, it stops the filling-in of information along that direction which is least consistent with respect to the colour value of the current location. This in turn prevents the algorithm from using meaningless temporal information, if the estimated displacement field did locally not allow for a correct interframe registration. Moreover, since the directions are not necessarily axis-aligned, the combination of compensation and adaptation allows to correct small misalignments if the registration did not perfectly match.

As in Step 1, this process is discretised by means of a simple explicit scheme using finite differences and the solution for $\tau \to \infty$ is computed. However, since only the originally missing lines of the central frame are recomputed, it is hardly more expensive from a computational viewpoint than the pure spatial interpolation process. The main difference comes from the additional number of neighbours required for the discretisation of the anisotropic diffusion process in 3-D. Typical run times are in the order of 4 seconds for images of size 300×300. Considering the additional effort compared to the 2-D case, run times of about 40 milliseconds per frame seem to be possible on parallel hardware. Thus also for this third step real-time is well within our computational reach.

6 Experiments

In order to evaluate the quality of our approach, we have created four interlaced test sequences with increasing difficulty: The *House*, the *Duck*, the *City* and the *Run* sequence (see Figure 1). These sequences have been cropped from video sequences of the European Broadcasting Union (EBU)[1] and then interlaced afterwards. This offers the advantage that the ground truth is known and

[1] `ftp://ftp.ebu.ch/en/technical/hdtv/test_sequences.php`

Fig. 1. Cropped and interlaced test image sequences created from sequences of the European Broadcasting Union (300 × 300 pixels). *Top Left:* House. *Top Right:* Duck. *Bottom Left:* City. *Bottom Right:* Run.

the deinterlacing quality can be determined in terms of error measures. In our case this was done via the *average absolute colour error*

$$E_{\mathrm{AACE}} = \frac{1}{3N} \sum_{i=1}^{3} \sum_{p=1}^{N} |f_{i,p}^{\mathrm{truth}} - f_{i,p}^{\mathrm{deinter}}| \ , \tag{16}$$

where N is the number of pixels and f_c^{truth} and f_c^{deinter} denote the RGB channels of the ground truth and the deinterlaced image, respectively. For comparing the performance of our method, we have implemented three classical techniques: field doubling, line averaging, and spatiotemporal median filtering. Additionally we have considered a recent PDE-based approach based on motion adaptive total variation that is among the best techniques for deinterlacing [22]. Here, the original implementation of the authors was used. Finally, we also included

Table 1. Parameter settings for the spatial and the spatiotemporal deinterlacing step

Sequence	2-D EED (Step 1)			MCEED (Step 3)		
	σ	ϵ	ρ	σ	ϵ	ρ
House	0.6	1.8	0.0	0.5	0.7	4.0
Duck	0.6	1.6	0.0	0.7	0.8	5.0
City	0.6	1.7	0.0	0.5	0.7	4.0
Run	0.7	2.1	0.0	0.5	0.8	4.0

Table 2. Comparison to interlacing methods from the literature. Deviations from the ground truth are measured by the average absolute colour error. RGB values are in the range $[0, 255]$. Superscripts next to error values denote the rank of a method.

Technique	Increasing Motion →			
	House	Duck	City	Run
Field Doubling [31]	2.91^{3}	4.05^{2}	6.89^{6}	13.23^{6}
Line Averaging [31]	4.22^{6}	5.92^{6}	3.70^{3}	5.61^{3}
Our Method (2-D EED)	4.06^{5}	5.87^{5}	3.66^{2}	5.59^{2}
Spatiotemporal Median [3]	2.99^{4}	4.38^{4}	4.27^{5}	7.01^{5}
Total Variation [22]	2.89^{2}	4.15^{3}	3.77^{4}	5.85^{4}
Our Method (MCEED)	$\mathbf{2.58}^{1}$	$\mathbf{3.48}^{1}$	$\mathbf{3.29}^{1}$	$\mathbf{4.48}^{1}$

the results from pure spatial edge-enhancing anisotropic diffusion (2-D EED) that serves as first step in our algorithm. In all cases the parameters have been optimised with respect to the AACE. While the parameters for the optical flow computation were set fixed to $\alpha = 20$ and $\gamma = 5$, the parameters for the spatial and spatiotemporal inpainting steps are listed in Table 1. As one can see, they only vary slightly throughout all sequences. Typical runtimes for our algorithm are in the order of 10 seconds for images of size 300×300. However, as pointed out before, efficient implementations on parallel hardware such as graphics cards or the Cell processor promise frame rates of about 10 deinterlaced frames per second (adding up the expected time for all three steps).

Figure 2 shows by the example of the two sequences with largest displacements (*City, Run*) that our method gives excellent deinterlacing results. As one can see from the corresponding error values in Table 2, our method thereby clearly outperforms all other techniques. In particular for the *Run* scene, where the largest motion is present, the superiority of our method becomes obvious: While pure spatial interpolation methods (line averaging, 2-D EED) perform relatively good – at least they do not introduce wrong temporal information – motion adaptive techniques such as median filtering or the total variation approach show some problems. Also field doubling as pure temporal interpolation method gives very bad results. Only our motion compensated edge-enhancing diffusion technique (MCEED) is capable of improving the spatial results by additionally considering temporal information. Also for images with smaller motion, our method

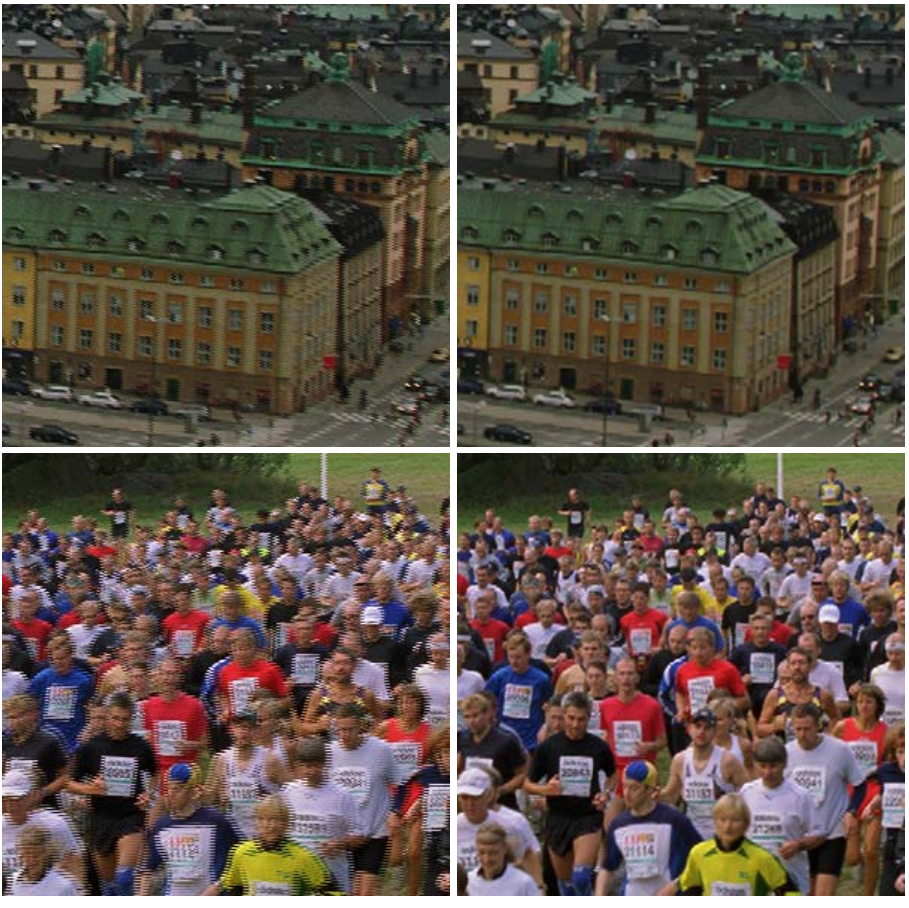


Fig. 2. Results for the *City* and the *Run* sequence. *Left Column:* Interlaced Images. *Right Column:* Deinterlaced images with MCEED (our method).

performs favourably. Although, in this situation, the motion adaptive algorithms are better than the spatial ones, they cannot compete with our results that show again the lowest errors.

In order to allow for a visual comparison of the deinterlacing quality of all methods, we show a representative detail from the results for the *Run* scene in Figure 3. It depicts a running man and requires the preservation of many small scale features. How challenging the deinterlacing of this image actually is can be seen from the classical field doubling method that simply merges two consecutive half-frames (fields). Since it uses the temporal information in the worst possible way, its poor results are a good indicator for the difficulty of the scene. Evidently, our motion compensated approach gives the sharpest results of all methods: Features such as the slightly open mouth, the structure of the

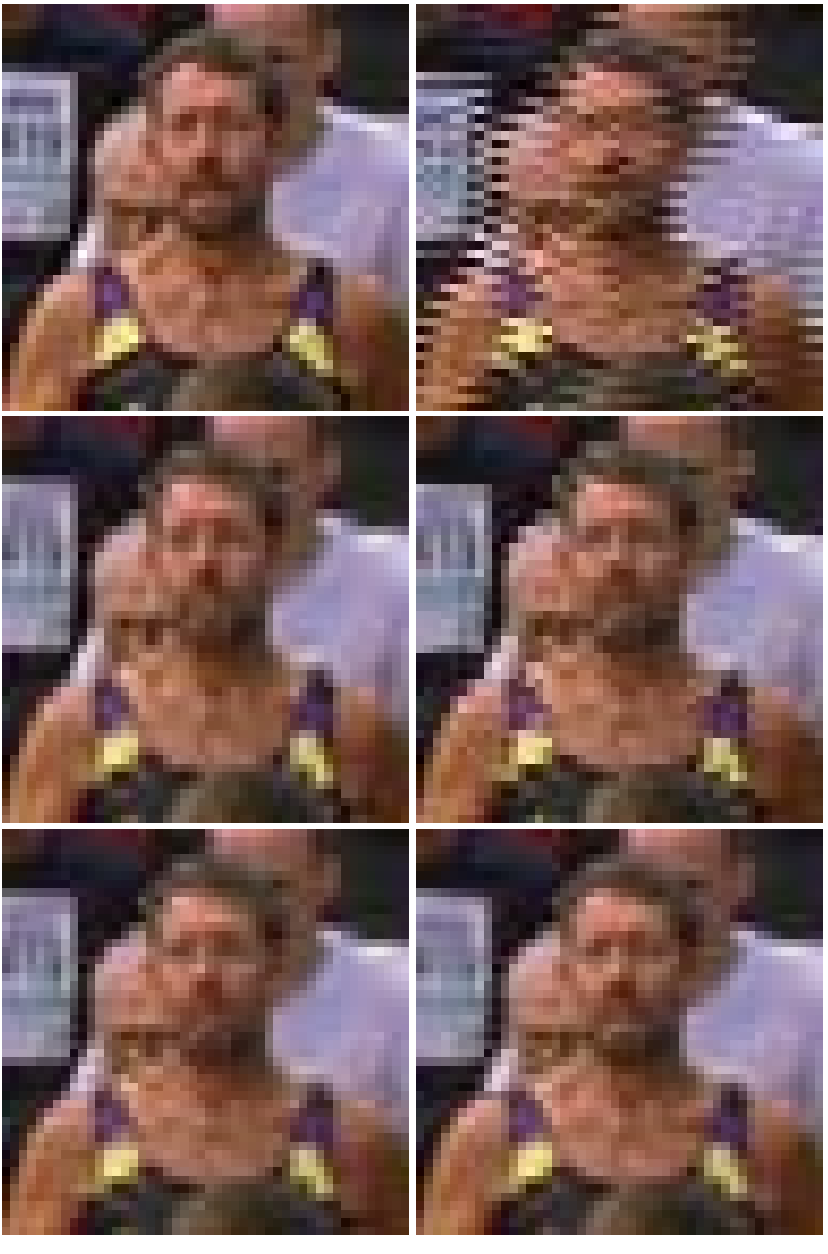


Fig. 3. Detail comparison of the *Run* sequence for different deinterlacing methods (48×48 pixels). *Top Left:* Truth. *Top Right:* Field doubling. *Center Left:* Line averaging. *Center Right:* Median filtering. *Bottom Left:* Motion adaptive total variation [22]. *Bottom Right:* MCEED (our method).

Fig. 4. Detail comparison for the main building in the *House* sequence (60×30 pixels). *Left:* Truth. *Center:* Motion adaptive total variation [22]. *Right:* MCEED (our method).

Fig. 5. Detail comparison for a building in the upper left corner of the *City* sequence (60×30 pixels). *Left:* Truth. *Center:* Spatial EED (our method, only Step 1). *Right:* MCEED (our method).

beard and the numerous details at the shoulders are recovered with a much higher quality than by any other method. Obviously, this is a direct consequence of the use of discontinuity-preserving concepts in *all* parts of our algorithm. It makes explicit that selecting the components carefully and adjusting them to each other is an important task when designing a combined algorithm.

Two more examples for the discontinuity-preserving deinterlacing property of our method are given in Figures 4 and 5. Figure 4 shows a detail of the deinterlaced *House* image and compares it to the result of the motion adaptive total variation technique. Since the motion in this scene is rather low, one would expect good results from both methods. However, only our technique allows to recover the small horizontal structures due to the excellent interpolation properties of the spatiotemporal edge-enhancing diffusion filter. A similar observation can be made in Figure 5, where a detail of the deinterlaced *City* image is depicted. Here, we compare our method against a pure spatial edge-enhancing diffusion filtering that also offers a very good interpolation quality. Although the 2-D EED algorithm gives the second best results for this scene, it cannot reconstruct the horizontal lines properly. Once again, only our approach allows to fill in the necessary information. In contrast to the previous example, where only the edge-preservation was important, the motion compensation plays an equally important role for this scene. By correctly aligning the subsequent images, the missing lines can be recovered correctly from the temporal information. Using spatial information alone this problem could not have been solved. This shows that high quality deinterlacing is possible for scenes with large displacements if motion adaptive and motion compensated approaches are combined.

7 Summary and Conclusions

In this paper we have demonstrated that motion compensated and motion adaptive approaches can benefit from each other. To this end, we developed a combined three strep strategy that makes use of recent PDE-based approaches for motion estimation and image inpainting. Thereby we focused on such techniques that are capable of preserving discontinuities in the computational process. In the experimental section, the advantages of our new method became explicit: In contrast to other approaches that performed well either for scenes with large or for scenes with small motion, it was able to achieve high quality results in both cases. Moreover, due to the discontinuity-preserving nature of its single components, it allowed to recover small details in the deinterlacing process that could not be restored by other schemes. This shows that both contributions are equally important for the success of our algorithm: Without combining motion compensated and motion adaptive techniques, our method would not work for small and large motion, while without preserving discontinuities, important features would get lost during the deinterlacing process.

Our ongoing work addresses the implementation of efficient numerical schemes on recent parallel hardware. Moreover, it focuses on the application of such schemes to images in full HDTV resolution (1920×1280). Evidently, this requires additional speedups that have to be achieved both on the algorithmic as well as on the parallelisation side. In this context, also simplified models may help to bridge the gap towards real-time HDTV deinterlacing. This, however, is topic of our future work.

Acknowledgements

We thank Sune Keller, Francois Lauze and Mads Nielsen for providing their deinterlacing algorithm for testing.

References

1. Almog, A., Levi, A., Bruckstein, A.M.: Spatial de-interlacing using dynamic time warping. In: Proc. 2005 IEEE International Conference on Image Processing, Genova, Italy, vol. 2, pp. 1010–1013 (2005)
2. Ballester, C., Bertalmio, B., Caselles, V., Garrido, L., Marques, A., Ranchin, F.: An inpainting-based deinterlacing method. IEEE Transactions on Image Processing 16(10), 2476–2491 (2007)
3. Bellers, E.B., de Haan, G.: A Key Technology for Scanline Conversion. Elsevier, Amsterdam (2000)
4. Biswas, M., Kumar, S., Nguyen, T.: Performance analysis of motion-compensated de-interlacing systems. IEEE Transactions on Image Processing 15(9), 2596–2609 (2006)
5. Brox, T., Bruhn, A., Papenberg, N., Weickertw, J.: High accuracy optic flow estimation based on a theory for warping. In: Pajdla, T., Matas, J(G.) (eds.) ECCV 2004. LNCS, vol. 3024, pp. 25–36. Springer, Heidelberg (2004)

6. Bruhn, A., Weickert, J.: Towards ultimate motion estimation: Combining highest accuracy with real-time performance. In: Proc. Tenth International Conference on Computer Vision, Beijing, China, vol. 1, pp. 749–755. IEEE Computer Society Press, Los Alamitos (2005)
7. Bruhn, A., Weickert, J., Kohlberger, T., Schnörr, C.: A multigrid platform for real-time motion computation with discontinuity-preserving variational methods. International Journal of Computer Vision 70(3), 257–277 (2006)
8. Capodiferro, L.: Interlaced to progressive conversion by median filtering. In: Chiarglione, L. (ed.) Proc. 3rd International Workshop on HDTV, vol. 3, pp. 677–684 (1989)
9. Charbonnier, P., Blanc-Féraud, L., Aubert, G., Barlaud, M.: Two deterministic half-quadratic regularization algorithms for computed imaging. In: Proc. 1994 IEEE International Conference on Image Processing, Austin, TX, pp. 168–172. IEEE Computer Society Press, Los Alamitos (1994)
10. Cocquerez, J.P., Chanas, L., Blanc-Talon, J.: Simultaneous inpainting and motion estimation of highly degraded video-sequences. In: Bigun, J., Gustavsson, T. (eds.) SCIA 2003. LNCS, vol. 2749, pp. 685–692. Springer, Heidelberg (2003)
11. D'Amore, L., Marcellino, L., Murli, A.: Image sequence inpainting: a numerical approach for blotch detection and removal via motion estimation. Journal of Computational and Applied Mathematics 2, 396–413 (2007)
12. Di Zenzo, S.: A note on the gradient of a multi-image. Computer Vision, Graphics and Image Processing 33, 116–125 (1986)
13. Elad, M., Feuer, A.: Superresolution restoration of an image sequence: adpative filering approach. IEEE Transactions on Pattern Analysis and Machine Intelligence 21, 817–834 (1999)
14. Förstner, W., Gülch, E.: A fast operator for detection and precise location of distinct points, corners and centres of circular features. In: Proc. ISPRS Intercommission Conference on Fast Processing of Photogrammetric Data, Interlaken, Switzerland, June 1987, pp. 281–305 (1987)
15. Galić, I., Weickert, J., Welk, M., Bruhn, A., Belyaev, A., Seidel, H.-P.: Towards PDE-based image compression. In: Paragios, N., Faugeras, O., Chan, T., Schnörr, C. (eds.) VLSM 2005. LNCS, vol. 3752, pp. 37–48. Springer, Heidelberg (2005)
16. Galić, I., Weickert, J., Welk, M., Bruhn, A., Belyaev, A., Seidel, H.-P.: Image compressions with anisotropic diffusion. Journal of Mathematical Imaging and Vision 31, 255–269 (2008)
17. Gillies, D., Plantholt, M., Westerkamp, D.: Motion adaptive field rate upconversion algorithms for 900 lines/100 hz/2:1 displays. IEEE Transactions on Consumer Electronics 36, 149–160 (1990)
18. Grossauer, H., Thoman, P.: GPU-based multigrid: Real-time performance in high resolution nonlinear image processing. In: Gasteratos, A., Vincze, M., Tsotsos, J.K. (eds.) ICVS 2008. LNCS, vol. 5008, pp. 141–150. Springer, Heidelberg (2008)
19. Gwosdek, P., Bruhn, A., Weickert, J.: High performance parallel optical flow algorithms on the Sony Playstation 3. In: Deussen, O., Keim, D., Saupe, D. (eds.) VMV 2008: Proc. Vision, Modeling, and Visualization, Konstanz, Germany, AKA, October 2008, pp. 253–262 (2008)
20. Gwosdek, P., Bruhn, A., Weickert, J.: Variational optic flow on the Sony Playstation 3 – acurate dense flow fields for real-time applications. Technical Report 233, Department of Mathematics, Saarland University, Germany (April 2009)
21. He, H., Kondi, L.P.: A regularization framework for joint blur estimation and super-resolution of video sequences. In: Proc. 2005 IEEE International Conference on Image Processing, Genova, Italy, September 2005, vol. 3, pp. 329–332 (2005)

22. Keller, S., Lauze, F., Nielsen, M.: A total variation motion adaptive deinterlacing scheme. In: Kimmel, R., Sochen, N.A., Weickert, J. (eds.) Scale-Space 2005. LNCS, vol. 3459, pp. 408–418. Springer, Heidelberg (2005)
23. Keller, S., Lauze, F., Nielsen, M.: Deinterlacing using variational methods. IEEE Transactions on Image Processing 17(11), 2015–2028 (2008)
24. Kokaram, A.: Motion Picture Restoration. Springer, London (1998)
25. Köstler, H., Stürmer, M., Freundl, C., Rüde, U.: PDE based video compression in real time. Technical Report 07-11, Institut für Informatik, Univ. Erlangen-Nürnberg, Germany (2007)
26. Kovačević, J., Safranek, R.J., Yeh, E.M.: Deinterlacing by succesive approximation. IEEE Transactions on Image Processing 6(2), 339–344 (1997)
27. Li, M., Nguyen, T.: A de-intelacing algorithm using Markov random field model. IEEE Transactions on Image Processing 16(11), 2633–2648 (2007)
28. Nguyen, A., Dubois, E.: Spatio-temporal adaptive interlaced-to-progressive conversion. In: Dubois, E., Chiarglione, L. (eds.) Proc. 4th International Workshop on HDTV, Kawasaki, Japan, pp. 749–756 (1992)
29. Patti, A.J., Tekalp, A.M., Sezan, M.I.: Image sequence restoration and deinterlacing by motion compensated kalman filtering. In: Image and Video Processing. Proceedings of SPIE, vol. 1903, pp. 59–70. SPIE Press, Bellingham (1993)
30. Rudin, L.I., Osher, S., Fatemi, E.: Nonlinear total variation based noise removal algorithms. Physica D 60, 259–268 (1992)
31. Tekalp, M.: Digital Video Processing. Prentice Hall, Englewood Cliffs (1995)
32. Wang, Y., Ostermann, J., Zhang, Y.: Video Processing and Communications. Prentice Hall, Upper Saddle River (2002)
33. Wedel, A., Pock, T., Zach, C., Bischof, H., Cremers, D.: An improved algorithm for TV-L^1 optical flow. In: Rosenhahn, B., Cremers, D., Yuille, A. (eds.) Visual Motion Analysis 2008. LNCS, vol. 5604, pp. 23–45. Springer, Heidelberg (2009)
34. Weickert, J.: Theoretical foundations of anisotropic diffusion in image processing. Computing Suppl. 11, 221–236 (1996)
35. Weickert, J.: Anisotropic Diffusion in Image Processing. Teubner, Stuttgart (1998)
36. Woods, N.A., Galatsanos, N.P.: Non-stationary approximate Bayesian super-resolution using a hierarchical prior model. In: Proc. 2005 IEEE International Conference on Image Processing, Genova, Italy, September 2005, vol. 1, pp. 37–40 (2005)
37. Yoo, H., Jeong, J.: Direction-oriented interpolation and its application to deinterlacing. IEEE Transactions on Consumer Electronics 48(4), 954–962 (2002)

Real-Time Synthesis of Body Movements Based on Learned Primitives

Martin A. Giese[1], Albert Mukovskiy[1], Aee-Ni Park[1], Lars Omlor[1], and Jean-Jacques E. Slotine[2]

[1] Section Computational Sensomotorics, Hertie Institute for Clinical Brain Research & Center for Integrative Neuroscience, University of Tübingen, Germany
martin.giese@uni-tuebingen.de
{albert.mukovskiy,aee-ni.park,lars.omlor}@medizin.uni-tuebingen.de
[2] Nonlinear Systems Laboratory, Massachusetts Institute of Technology, USA
jjs@mit.edu

Abstract. The synthesis of realistic complex body movements in real-time is a difficult problem in computer graphics and in robotics. High realism requires the accurate modeling of the details of the trajectories for a large number of degrees of freedom. At the same time, real-time animation necessitates flexible systems that can react in an online fashion, adapting to external constraints. Such online systems are suitable for the self-organization of complex behavior by the dynamic interaction between multiple autonomous characters in the scene. In this paper we present a novel approach for the online synthesis of realistic human body movements. The proposed model is inspired by concepts from motor control. It approximates movements by superposition of movement primitives (synergies) that are learned from motion capture data applying a new blind source separation algorithm. The learned generative model can synthesize periodic and non-periodic movements, achieving high degrees of realism with a very small number of synergies. For obtaining a system that is suitable for real-time synthesis, the primitives are approximated by the solutions of low-dimensional nonlinear dynamical systems (dynamic primitives). The application of a new type of stability analysis (contraction theory) permits the design of complex networks of such dynamic primitives, resulting in a stable overall system architecture. We discuss a number of applications of this framework and demonstrate that it is suitable for the self-organization of complex behaviors, such as navigation, synchronized crowd behavior and dancing.

1 Introduction

The generation of realistic human movements in reactive systems is a difficult task with high relevance for computer graphics and robotics. Many important applications, such as computer games require the online synthesis of such movements, at the same time providing high degrees of realism even for complex human body movements. For the offline synthesis of human movements motion capture has become the standard approach. Appropriate movements are recorded

D. Cremers et al. (Eds.): Visual Motion Analysis, LNCS 5604, pp. 107–127, 2009.

offline and retargeted to the relevant kinematic model. Additional adjustments can be made by editing or blending the recorded movements in order to modify movement style (e.g. [1, 2]), or to make the synthesized movements compatible with constraints of the animation, like kinematic boundary conditions (e.g. [3]). This approach results in highly realistic animations, but requires tedious post-processing of the recorded motion capture data. Recent approaches have tried to simplify this procedure by automatic selection and concatenation of recorded motion segments from large data bases, ensuring that the generated motion sequences fulfill constraints defined by the animator [4, 5]. These methods permit a flexible offline synthesis of quite complex sequences of movements. However, since the retrieval of the relevant trajectory segments from the data basis requires complex search algorithms and the resulting methods are typically not suitable for real-time applications and the online generation of animations.

Other approaches based on physical or dynamical models [6, 7] have focused on the simulation of scenes with many interacting agents or crowds that navigate autonomously or show collective behaviours. The movements of such characters are typically strongly simplified, providing the benefit of a manageable complexity of the dynamics of the underlying simulation system, but at the same time resulting in animations that lack the subtle details of realistic human body movements. Only very recently, work has been presented that simulates highly realistic human behaviour with dynamic models learned from motion capture data [8, 9]. However, the existing approaches are characterized by high-dimensional and complex underlying dynamic control architectures, whose design requires substantial expertise from the animator.

The goal of the presented work was the development of a system for the simulation of highly realistic human movements in real-time by combining motion capture with dynamical systems, but avoiding a detailed simulation of all details of the human body dynamics. To accomplish this goal, we exploited an approach that was inspired by motor control in biological systems. It has been a classical assumption in this field that complex motor behavior is generated by appropriate combination of simpler *movement primitives* or *synergies* [10, 11]. Synergies specify lower-dimensional control units that typically encompass only a subset of the available degrees of freedom. A decomposition in such low-dimensional sub-units has been proposed as way to solve the 'degrees-of-freedom problem', which arises in the synthesis and control of movements in effector systems with many degrees of freedom. Recent studies in motor control have successfully applied unsupervised learning methods to extract low-dimensional spatio-temporal components from trajectories or EMG signals, which have been interpreted as correlates of synergies [12, 13, 14]. We try to extend this approach to computer graphics by learning synergies from motion capture data and mapping them onto structurally stable dynamical systems. By combination of such primitives more complex system architectures can be designed that generate coordinated behavior of multiple characters by self-organization.

Core steps of the developed method are: 1) A novel unsupervised learning method for the approximation of trajectory sets based on anechoic mixing models with

very few source components; 2) a method for establishing mappings between the learned synergies and simple dynamical systems, called *dynamic primitives*, that can be combined into more complex composite systems with well-defined dynamical properties. For the design of such composite systems we exploit *contraction theory*, a novel method for the analysis of the global stability properties of nonlinear dynamical systems.

In the following, after a brief discussion of related approaches (Section 2), we first sketch the novel unsupervised learning algorithm that is suitable for the approximation of trajectories with typically very few hidden source terms (Section 3). We then show how the obtained generative models can be mapped onto dynamical systems that are suitable for a real-time synthesis of movement trajectories (Section 4). Section 5 illustrates how style morphing and navigation can be realized in the proposed framework. Finally, Section 6 shows several applications of the developed framework in computer animation, such as the simulation of reactive behavior or the animation of small crowds.

2 Related Work

Standard dimension reduction methods, such as PCA or ICA, have been quite commonly applied in computer graphics to reduce the dimensionality of motion capture data. Such work shows that the approximation of complex body movements requires typically 8-12 principal components (e.g. [15]). In previous work we have shown for different classes of human movements that by application of mixture models with time delays the number of required source terms for the accurate approximation of human movement data can be significantly reduced (e.g. to 3-4 terms for periodic and some non-periodic movements) [16, 17]. These compact trajectory models provide the advantage that they can be associated with relatively simple dynamical systems that synthesize the trajectories in real-time. In addition, the proposed method, opposed for example to Fourier-based approximations [18], are applicable to periodic as well as non-periodic movements.

To embed the learned synergies into a real-time animation system we define dynamical systems (dynamic primitives) that generate the associated trajectories online. These dynamical systems play a similar role as *Central Pattern Generators* in biological systems. The idea of using dynamical systems for the definition of movement primitives is quite common in robotics (e.g. [19, 20]). In addition, a variety of animation systems have exploited models for central pattern generators. However, in many cases such dynamic primitives have been hand-crafted and thus result in movements that do not approximate natural movements very accurately. In our system we accomplish an excellent approximation of real human movements by learning of the detailed trajectory shapes from motion capture data.

Compared to the existing work, the major contributions of the proposed system are thus as follows: (1) Novel method for the unsupervised learning of highly compact models from trajectory classes of complex human body movements; (2) method for the mapping of the learned trajectory components (synergies) onto

dynamical systems whose stability properties can be designed in a systematic way, making it possible to embed such primitives in more complex systems with well-controlled dynamical properties.

3 Synergy-Based Trajectory Model

3.1 Data Set

For the application presented in this chapter we learned the synergies from a data basis of motion capture data that was recorded using a Vicon 612 Motion Capture System with 8 cameras, using 41 reflecting markers and a sample frequency of 120 Hz. The recorded data basis comprises different types of gaits, partially combined with non-periodic arm movements, periodic gaits, straight walking with neutral and emotional (happy and sad) styles, and walking with a stooped posture. In addition, walking along a circular path (forward and backward) with rotations of 90 deg, and turning on spots (120 deg) left or right per double step were recorded. Movements were retargeted to a skeleton model with 17 joints using an axis-angle representation for the parametrization of the 3D rotations between adjacent segments. In addition, we recorded gaits that were combined with non-periodic arm movements, such as swinging the right or left arm up, or holding the arm in a fixed posture during the whole gait cycle. For these recorded movements the arm moved independently of the legs, forming a separate 'synergy' that was not periodic.

3.2 Learning of Movement Primitives

Joint angle trajectories, after subtraction of the means, were approximated by a weighted mixture of source signals. As shown elsewhere [21], a particularly compact model for the joint angle trajectories x_i can be obtained by approximating the data by an *anechoic mixture model* that is given by the equation:

$$x_i(t) = \sum_j w_{ij} s_j (t - \tau_{ij}) \tag{1}$$

The functions s_j denote hidden source signals and the parameters w_{ij} are the mixing weights. Opposed to common blind source separation techniques, like PCA or ICA, this mixing model allows for time shifts τ_{ij} of the sources in the linear superposition. Time shifts (delays), source signals and mixing weights are determined by a blind source separation algorithm that is based on a time-frequency integral transform.

Opposed to standard anechoic demixing methods in acoustics, our algorithm is suitable for *over-determined problems*, for which the signals outnumber the sources. Our novel algorithm is based on the Wigner-Ville transform. The Wigner-Ville spectrum (WVS) of a random process x is defined by the partial Fourier transform of the symmetric autocorrelation function of x:

$$W_x(t,\omega) := \int E\left\{x(t+\frac{\tau}{2})\,\overline{x(t-\frac{\tau}{2})}\right\} e^{-2\pi i\omega\tau} d\tau \tag{2}$$

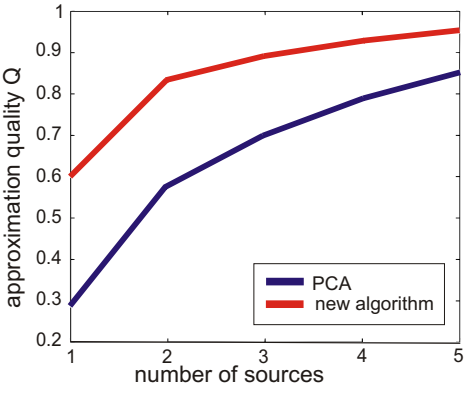


Fig. 1. Comparison of different blind source separation algorithms of our data set that includes periodic and non-periodic movements. The approximation quality Q is shown as a function of the number of sources for traditional blind source separation algorithms (PCA/ICA) and our new algorithm.

Applying this integral transform to equation (1) results in the equation:

$$W_{x_i}(t,\omega) := \sum_j |w_{ij}|^2 \, W_{s_j}(t - \tau_{ij}, \omega) \tag{3}$$

under the assumption that the sources are statistically independent. As two-dimensional representation of one dimensional signals, this equation is redundant and can be solved by computing a set of projections onto lower dimensional spaces that specify the same information as the original problem. It can be shown [22] that this equation can be solved by the iterative solution of the following 2 equations:

$$|\tilde{x}_i(\omega)|^2 = \sum_j |w_{ij}|^2 \, |\tilde{s}_j(\omega)|^2 \tag{4}$$

$$|\tilde{x}_i(\omega)|^2 \frac{\partial}{\partial\omega}\{\tilde{x}_i(\omega)\} = \sum_j |w_{ij}|^2 |\tilde{s}_j(\omega)|^2 [\frac{\partial}{\partial\omega}\{\tilde{s}_i(\omega)\} + \tau_{ij}] \tag{5}$$

where $\tilde{x}$ and $\tilde{s}$ signify the Fourier transformations of the trajectory data and the sources. Detailed comparisons for periodic and non-periodic trajectory data show that this model provides a more compact approximation of human movement trajectories, requiring less source terms than models based on instantaneous mixtures for the same approximation quality. This is illustrated in Fig.1 that shows the approximation quality as a function of the number of sources s_j for the described data set. Instead of the explained variance $E = 1 - \left(\frac{\|D-\hat{D}\|}{\|D\|}\right)^2$ we used the quality measure $Q = 1 - \frac{(\|D-\hat{D}\|)}{\|D\|}$, which is more sensitive for differences in the regime of small approximation errors. In these formulas D signifies

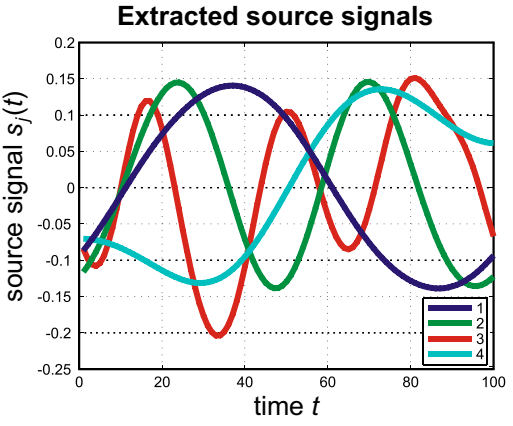


Fig. 2. Four source signals extracted from the motion capture data set including periodic and non-periodic movements. Sources 1-3 are periodic and the 4th source is non-periodic.

the original data matrix and $\hat{D}$ its approximation by the source model, where the norm is the Frobenius norm. The learned sources for a data set including periodic and non-periodic movements are shown in Fig. 2.

The approximation quality of the recorded trajectory set including periodic and non-periodic movements with only four source signals was $Q = 0.92$. This corresponds to an explained variance of approx. $E = 0.99$ and is sufficient for a very accurate approximation of the trajectories. PCA and ICA required more than 7 sources to achieve the same level of accuracy. A detailed analysis shows that mixing weights and delays vary with different motion styles. The generation of intermediate motion styles by interpolation requires thus the interpolation (blending) of all these parameters.

4 Online-Capable Animation Dynamics

The described model for the compact approximation of trajectories based on synergies is not suitable for real-time animation, e.g. for computer games, since the trajectory has to be synthesized by superposition and delaying of source signals whose whole time-course must be known. A real-time capable algorithm can be devised by specifying a dynamical system that produces the same trajectories as solution. We design such a dynamical system, again exploiting the fact that the movement can be approximated by a superposition of a few basic components. For this purpose, we establish a mapping between the solutions of simple dynamical systems and the source signals of the trajectory representation. The complete trajectory is then generated by the superposition of the solutions of a set of such simple dynamical systems (dynamic primitives). By introduction of dynamic couplings between the primitives the temporal coordination between

the different source components can be ensured. In an abstract sense, the resulting system is similar to a set of coupled 'central pattern generators' in a biological system.

4.1 Attractor Dynamics

Our four extracted source-signals are sufficient for a highly accurate approximation of the original trajectories, containing periodic and non-periodic movements. The fourth source contributes exclusively to the reproduction of the non-periodic movements in the data set. The other three sources are periodic and model the periodic parts of the movements.

The model (1) approximates the trajectories of the data set by linear combination of the time-shifted source signals $s_j(t)$. By choosing appropriate mixing weights w_{ij} and time delays τ_{ij} the trajectories of the training data set can be closely approximated. The idea for the online generation of the trajectories is to construct dynamical systems (dynamic primitives) that generate the source signals $s_j(t)$ by iteration of differential equations over time. The relative timing between the different source signals can be stabilized by introduction of dynamic couplings between these differential equations. An additional challenge for the online implementation is an efficient realization of the time delays. The naive implementation by introducing delay lines would lead to a slow system dynamics with rather complex stability properties. This is particularly a problem for more complex systems including multiple interacting characters.

In the following, we first introduce the differential equations for the dynamic primitives that generate the source signals in an online fashion (Section 4.2). Our approach is to choose structurally stable nonlinear dynamical systems for the generation of periodic and non-periodic patterns, and to map their solutions onto the required form of the source signals applying kernel methods (Section 4.3). We then show how these dynamic primitives can be coupled in a way that stabilizes the timing relationships between the different synergies (Section 4.4).

4.2 Dynamic Primitives

For the online generation of the source signals we construct a nonlinear mapping between the solutions of the dynamical systems and the source signals. This gives us a high flexibility for the choice of the dynamical systems, which can be optimized in order to simplify the design of a stable overall system dynamics. As basic building blocks for the dynamics we use nonlinear oscillators for the synthesis of periodic behaviors, and a fixpoint attractor dynamics for the synthesis of non-periodic movements. We decided to use nonlinear dynamical systems whose structural properties do not change in presence of weak couplings. In this way we can design the qualitative properties of the different dynamic primitives – to some degree – independently from their interaction with other system components. The idea to map desired behaviors onto solutions of nonlinear dynamical systems is quite common in robotics and behavioral research [23, 20, 19, 24], but also in computer animation [25].

As basic dynamics for the generation of the periodic signals we used a limit cycle oscillator: The Van der Pol oscillator, for adequate choice of the parameters, has an asymptotically stable limit cycle. Its dynamics is given by the differential equation:

$$\ddot{y}\left(t\right)+\zeta\left(y\left(t\right)^{2}-k\right)\dot{y}\left(t\right)+\omega_{0}^{2}\,y\left(t\right)=0 \tag{6}$$

The parameter ω_0 determines the eigenfrequency of the oscillator, and the parameter k the amplitude of the stable limit cycle. The force that pushes the state back towards the limit cycle is determined by the parameter $\zeta > 0$. For appropriate choice of the oscillator parameters, in absence of external input signals, the form of the stable limit cycle can be made almost ideally circular in the y-$\dot{y}$ plane (phase plane), assuming appropriate scaling of the two axes. This property is critical for the online implementation of the phase delays.

One source is non-periodic and has a ramp-like characteristics (Fig. 2). This source is crucial for the approximation of the non-periodic arm movements, for which the arm moves from a start posture to an end posture. We model this behavior by a fixpoint attractor dynamics. Considering that natural arm movements are characterized by a bell-shaped velocity profile [26, 27], we chose a nonlinear dynamics that generates solutions with this property. In addition this dynamics can generate identical movements in opposite directions by the change of a single parameter. This dynamics is given by the differential equation:

$$\dot{y}(t)=uy(t)(1-y(t)) \tag{7}$$

We restrict the values for y to the interval $I=[0,1]$. For $u<0$ this dynamics has a stable fixpoint in 0, and for $u>0$ a stable fixpoint in 1 that is approached asymptotically from inside the interval I. The value of $|u|$ determines how fast this fixpoint is approached. The solution of this differential equation can be computed analytically and is given by $y(t)=\left(1+\tanh(\frac{u}{2}(t-t_0))\right)/2$, showing that its derivative $\dot{y}(t)=\frac{u}{4}\left(1-\tanh^2(\frac{u}{2}(t-t_0))\right)$ is bell-shaped. By clipping we ensure that in presence of noise y does not leave the permissible interval.

4.3 Mapping between Attractor Solutions and Source Signals

To map the attractor solutions of the differential equations defining the dynamic primitives onto the exact form of the source signals we construct a nonlinear mapping. This mapping is defined by a concatenation of a rotation in phase space, modeling the influence of the time delays τ_{ij}, and a nonlinear function, which is learned from training data points by *Support Vector Regression (SVR)*. The underlying idea is illustrated in Fig. 3.

Treating the oscillatory primitives first, the purpose of the mapping is to associate the points $\mathbf{y}=[y,\dot{y}]'$ along the attractor in the phase plane of the oscillators with the corresponding values of the source function s_j. We try to avoid the introduction of explicit time delays in the implementation since this would lead to a complex system dynamics. As illustrated in Fig. 4 we try to approximate the terms $s_j\left(t-\tau_{ij}\right)$ in (1) in the form:

$$s_j(t-\tau_{ij}(t))\approx f_j\left(\mathbf{M}_{\tau_{ij}}\mathbf{y}(t)\right) \tag{8}$$

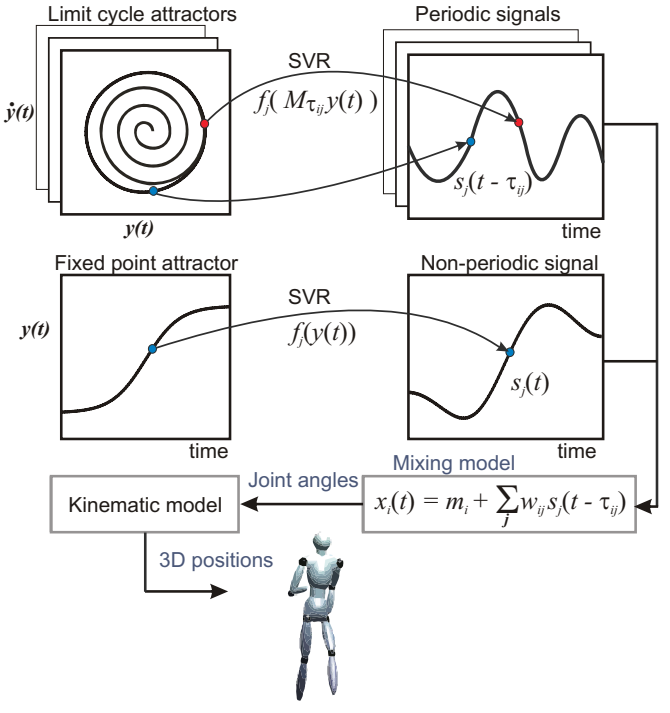


Fig. 3. Illustration of the dynamic architecture for real-time animation. Periodic and non-periodic movements are generated by dynamic primitives. The solutions of these dynamical systems are mapped onto the source signals by a nonlinear mapping that models the time delays and a nonlinear transformation that is learned by SVR. Joint angles trajectories can be synthesized by combining the signals linearly according to the learned mixture model (1). A kinematic model converts the joint angles into 3-D positions for animation.

where $\mathbf{M}_{\tau_{ij}}$ is an orthogonal transformation of the form:

$$\mathbf{M}_{\tau_{ij}} = \begin{bmatrix} \cos(\phi_{ij}) & -\sin(\phi_{ij}) \\ \sin(\phi_{ij}) & \cos(\phi_{ij}) \end{bmatrix} \mathbf{\Sigma} \tag{9}$$

This transformation is a concatenation of scaling and rotation in the two dimensional phase space. The matrix $\mathbf{\Sigma}$ is diagonal and scales the axes of the phase plane in a way that makes the attractor solution approximately circular. The rotation angles are given by $\phi_{ij} = -2\pi\frac{\tau_{ij}}{T}$, where T is the duration of one period of the stable oscillatory solution. The nonlinear function $f_j(\mathbf{y})$ maps the phase plane onto a scalar. It is learned from training data pairs that were obtained by temporally equidistant sampling of the signals $\mathbf{y}(t)$ and $s_j(t - \tau_{ij}(t))$, where we used the solutions of the uncoupled dynamic primitives. The functions were learned by support vector regression [28, 29] using a gaussian kernel.

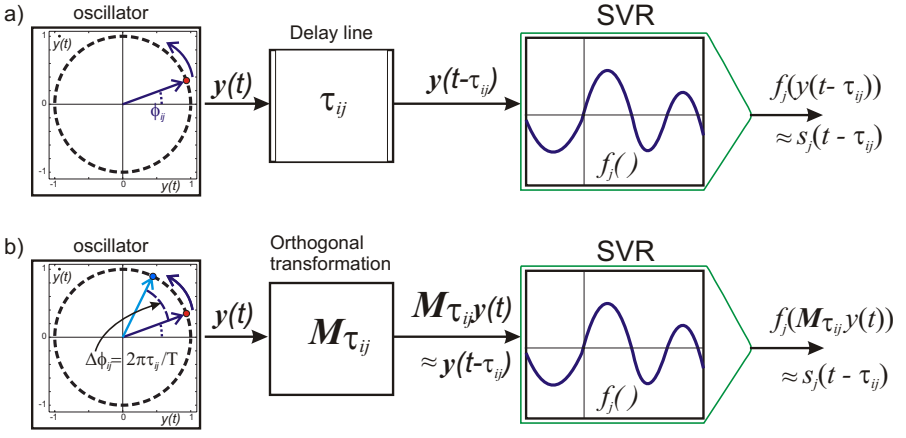


Fig. 4. Online implementation of delays. a) Direct implementation introduces explicit delay lines, resulting in a complex system dynamics that is difficult to control. b) Approximation by a rotation in the phase space, defined by the instantaneous orthogonal transformation $\mathbf{M}_{\tau_{ij}}$ in the phase plane of the oscillator avoids a dynamics with delays.

For the non-periodic source, the solution of the point-attractor equation (7) was mapped in a similar way onto the values of the non-periodic source signal. In principle, here the effect of the time delay can be modeled by an application of a conformal mapping to the solution of this equation. The overall system dynamics was defined by three oscillators and the point attractor dynamics (7). The state variables of the dynamic primitives were mapped onto the source signals by the described nonlinear observers. The synthesized source signals were then linearly combined according to (1), where the complete reconstruction of the joint angle trajectories requires the addition of the average joint angles m_i. An overview of the whole algorithm is given in Fig. 3.

4.4 Dynamic Coupling

The dynamical primitives that generate the signals for different synergies must be synchronized for the generation of coordinated behavior. Synchronization can be easily accomplished by introducing couplings between the dynamic primitives. The oscillatory primitives were modeled by Van der Pol oscillators. Applying concepts from *contraction theory* [30] it can be shown that complex networks of such oscillators can be guaranteed to have a single stable solution if the oscillators are coupled by *velocity couplings*. This type of coupling is defined by the equations (α specifying the coupling force):

$$\begin{aligned}
\ddot{y}_1 + \zeta\left(y_1^2 - k\right)\dot{y}_1 + \omega_0^2\, y_1 &= \alpha\left(\dot{y}_2 - \dot{y}_1\right) + \alpha\left(\dot{y}_3 - \dot{y}_1\right) \\
\ddot{y}_2 + \zeta\left(y_2^2 - k\right)\dot{y}_2 + \omega_0^2\, y_2 &= \alpha\left(\dot{y}_1 - \dot{y}_2\right) + \alpha\left(\dot{y}_3 - \dot{y}_2\right) \\
\ddot{y}_3 + \zeta\left(y_3^2 - k\right)\dot{y}_3 + \omega_0^2\, y_3 &= \alpha\left(\dot{y}_1 - \dot{y}_3\right) + \alpha\left(\dot{y}_2 - \dot{y}_3\right)
\end{aligned} \tag{10}$$

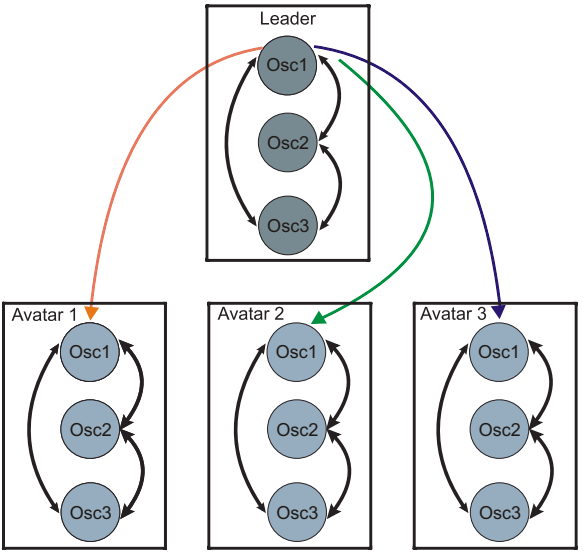


Fig. 5. Coupling of multiple avatars, each of them comprising three coupled oscillators (Osc1..3), permits the simulation of the behavior of coordinated crowds

For values of α below a specific bound, which depends on the coupling graph, the overall system dynamics has only a single stable solution. It is characterized by synchronization of all oscillators. The same type of couplings can be introduced between oscillators that represent dynamic primitives of different characters in the scene. This allows the modeling of synchronized behavior of multiple avatars (e.g. soldiers in lock-step). To implement such couplings we only connected the oscillators assigned to the source with the lowest frequencies (`Osc1` in Fig. 5). By introducing unidirectional couplings it is also possible to make multiple characters following one, who acts as a leader [30].

To synchronize the non-periodic primitives with external events the sign of the parameter u in equation (7) was switched dependent on an external signal, which triggers the raising of the arm. In this way, the previously stable fixed point of this dynamics becomes unstable, while its unstable fixpoint becomes stable. In addition, we added a short pulse input to this equation that displaces the state away from the unstable fixpoint. This ensures a transition to the novel stable point with a well-defined timing.

5 Style Morphing and Navigation

The proposed framework for the real-time synthesis of human movements can be integrated with other functions that are crucial for applications with autonomous characters. We discuss here the integration of style morphing and navigation within the discussed framework.

5.1 Style Morphing

The proposed model for the real-time generation of trajectories permits style morphing by linear interpolation of the average angles m_i, the mixing weights w_{ij} and of the delays τ_{ij}. The interpolation of the mean angles and of the weight matrices is straight forward. However, the interpolation of the delays requires additional approximations in order to avoid artifacts that are caused by ambiguities in the estimation of the delays.

For periodic source signals ambiguities in the estimation of weights and delays can arise that have to be removed prior to the interpolation. Periodic source signals fulfill $s_j(t + rT) = s_j(t)$ with integer r. In addition, source signals can be approximately periodic, fulfilling $s_j(t + qT_\mathrm{n}) \simeq s_j(t)$ with integer q. Such ambiguities are specifically a problem for the source signals that model the higher frequency components, where $T_\mathrm{n} = T/n$ is an integer fraction of the gait cycle time T. This (approximative) periodicity can cause ambiguities in the estimated delays, which might differ by multiples of T_n. If such delays are linearly interpolated they introduce phase differences between the sources that do not interpolate correctly between similar motion styles.

To remove such ambiguities we introduced an additional approximation step where we replaced the delays by the modified source delays $\tilde{\tau}_{ij} = \tau_{ij} - qT/n$, where q was chosen to minimize the values of the delays. This approximation was based on an algorithm that identifies the presence of ambiguities by determining the local extrema of the cross correlation function between the original and time shifted versions of the source signals. This made it possible to restrict and interpolate the delays within the intervals $[-T/2n, T/2n]$, removing the ambiguity.

After the estimation of these modified delays the mixing weights were re-estimated to optimize the accuracy of the obtained model. With the corrected time delays $\tilde{\tau}_{ij}$ and weights the interpolation between two movement styles (a) and (b), e.g. neutral and emotional walking, can be characterized by the equations

$$m_i\,(t) = \lambda\,(t)\,m_i^a + (1 - \lambda\,(t))\,m_i^b \tag{11}$$

$$w_{ij}\,(t) = \lambda\,(t)\,w_{ij}^a + (1 - \lambda\,(t))\,w_{ij}^b \tag{12}$$

$$\tilde{\tau}_{ij}\,(t) = \lambda\,(t)\,\tilde{\tau}_{ij}^a + (1 - \lambda\,(t))\,\tilde{\tau}_{ij}^b \tag{13}$$

The time-dependent morphing parameter $\lambda(t)$ specifies the movement style. Additionally, the gait speed can be adjusted by interpolating the eigenfrequencies of the oscillators:

$$\omega_0(t) = \lambda\,(t)\,\omega_0^a + (1 - \lambda\,(t))\,\omega_0^b \tag{14}$$

The same type of interpolation was also applied to implement direction changes of the avatars for navigation, by morphing between straight walking and turning steps. The change of the morphing parameter can also be made dependent on the behavior of other avatars in the scene. For example, the morphing weight

can be defined as nonlinear function of the distance d from another character: $\lambda(t) = g(d(t))$. Similarly, the eigenfrequency parameter ω_0 can be made dependent on the distance from other agents, resulting in a distance-dependent control of walking speed.

To model transitions between periodic and non-periodic movements, like walking with a rising of the arm vs. walking with arm up or down, we used the non-periodic ramp-like source signal to model change in the mean angles before and after the transition. We learned the superposition weights of periodic and the non-periodic sources during the middle of the transition from training examples.

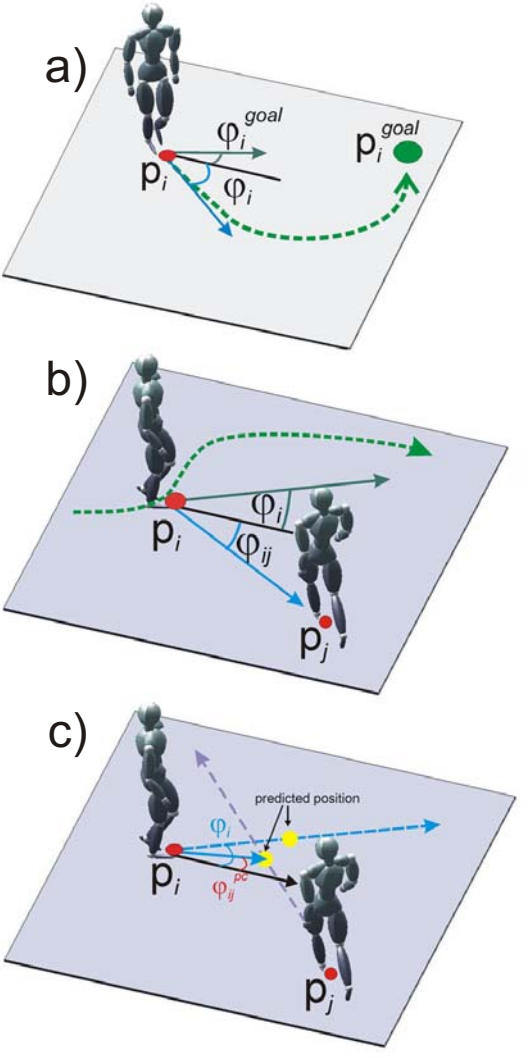


Fig. 6. Navigation dynamics depending on: a) Goal-finding term, b) instantaneous obstacle-avoidance term, and c) predictive obstacle-avoidance term

Transitions between periodic gait steps and the steps containing the non-periodic transitions were generated by linear blending, using the equations above. Using the same method, we were able to generate natural-looking transitions even for movements dependent on periodic and non-periodic primitives.

5.2 Navigation Dynamics

We combined our trajectory generation algorithm with a simplified version of a dynamic navigation model that has been applied successfully in robotics before [23, 31]. We extended this model by inclusion of predicted collisions in the navigation dynamics. The navigation dynamics was given by a differential equation for the heading directions φ_i of the characters. The turning rate of the avatars was controlled by morphing between straight and curved walking steps. The morphing weights were dependent on the temporal change of heading direction $\dot{\varphi}_i$. The navigation dynamics specifies this change by a differential equation that integrates three different components (where $\mathbf{p}_i$ denotes the position of character i):

$$\frac{d\varphi_i}{dt} = \underbrace{h^{\text{goal}}(\varphi_i, \mathbf{p}_i, \mathbf{p}_i^{\text{goal}})}_{\text{goal-finding}} + \underbrace{\sum_j h^{\text{avoid}}(\varphi_i, \mathbf{p}_i, \mathbf{p}_j)}_{\text{instantaneous obstacle avoidance}} + \underbrace{\sum_j h^{\text{pcoll}}(\varphi_i, \varphi_j, \mathbf{p}_i, \mathbf{p}_j)}_{\text{predictive obstacle avoidance}} \tag{15}$$

The first term determines a goal-finding term, where φ_i^{goal} defines the goal direction angle relative to the character i:

$$h^{\text{goal}}\left(\varphi_i, \mathbf{p}_i, \mathbf{p}_i^{\text{goal}}\right) = \sin(\varphi_i^{\text{goal}} - \varphi_i) \tag{16}$$

This term introduces a force that steers the avatars towards the goal e.g. Fig. 6a. The second term implements obstacle avoidance, where obstacles can also be defined by moving objects like other avatars (Fig. 6b). This term is given by the expression:

$$h^{\text{avoid}}\left(\varphi_i, \mathbf{p}_i, \mathbf{p}_j\right) = \sin\left(\varphi_i - \varphi_{ij}\right) \cdot \exp\left(-\frac{(\varphi_{ij} - \varphi_i)^2}{2\sigma_\varphi^2}\right) \cdot \exp\left(-\frac{d_{ij}^2}{2\sigma_d^2}\right) \tag{17}$$

Much more realistic collision avoidance is accomplished by inclusion of a third term in the navigation dynamics that is dependent on the predicted future positions of the avatars (Fig. 6c). This helps to prevent collisions by steering the characters away from each other already at an early stage, when a collision is

likely to occur in the future. The prediction assumes straight trajectories of the avatars and computes the closest point between their predicted trajectories. This third term has the form:

$$h^{\mathrm{pcoll}}\left(\varphi_i, \varphi_j, \mathbf{p}_i, \mathbf{p}_j\right) = \\ \sin(\varphi_i - \varphi_{ij}^{\mathrm{pc}}) \cdot \exp\left(-\frac{(\varphi_{ij}^{\mathrm{pc}} - \varphi_i)^2}{2\sigma_\varphi^2}\right) \cdot \exp\left(-\frac{\left(d_{ij}^{\mathrm{pc}}\right)^2}{2\sigma_d^2}\right) \tag{18}$$

Where $\varphi_{ij}^{\mathrm{pc}}$ signifies the direction of the predicted collision point (Fig. 6c). See [17] for more details.

6 Results

The flexibility of the developed framework for the online synthesis of movement trajectories was tested in different scenarios. In the following, we present a selection of applications including periodic and non-periodic movement primitives.

6.1 Synchronized Behavior of Crowds

The first example demonstrates that, by introduction of dynamic couplings between the dynamical systems that correspond to different characters, we are able to self-organize coordinated behavior of crowds with relatively realistic appearance. This is illustrated schematically in Fig. 7 that shows a few snapshots from an animation where a group of characters starts out with asynchronous step phases. One of the characters acts as a 'leader', and the dynamical systems of the other characters are coupled unidirectionally with its dynamics (cf. Section 4.4). After a short transition period the step cycles of all characters synchronize with the leader. This has the consequence that the 'crowd' leaves the scene with synchronous step pattern ('lock-step').

This shows that the proposed method is suitable for the simulation of coordinated behavior of crowds, like marching soldiers. Another example are dancing scenes that require the coordination of locomotion patterns. A demonstration video[1] can be downloaded.

In the context of this scenario we tried also to compare our method with standard approaches such as PCA (cf. Fig. 8 and demo video[2]). Avatar 3 acts as a 'leader'. It is driven by three coupled oscillators, but without additional external couplings. The other two avatars are coupled to this leader and start with equal initial phases that are different from the phase of the leader. The movement of one character (avatar 1) was generated applying our novel method, using an anechoic mixture model with three sources. The movement of the second character (avatar 2) was generated with a PCA model with 7 components in order to obtain the same approximation quality. This avatar was driven by 7 coupled

[1] http://www.uni-tuebingen.de/uni/knv/arl/avi/synchronization.avi
[2] http://www.uni-tuebingen.de/uni/knv/arl/avi/weakcoupling.avi

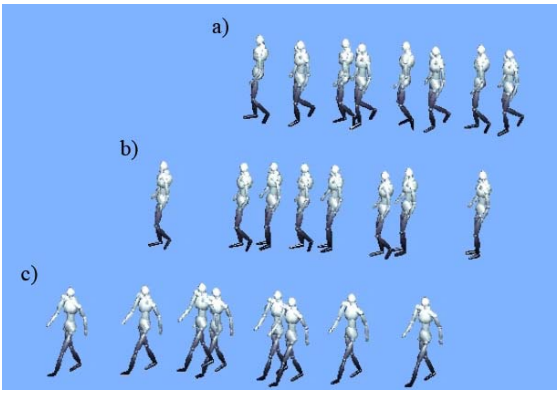


Fig. 7. Autonomous synchronization of gait patterns within a crowd. (a) Avatars start with self-paced walking and are out of phase. After a transitory period (b), the gaits of the characters become completely synchronized (c).

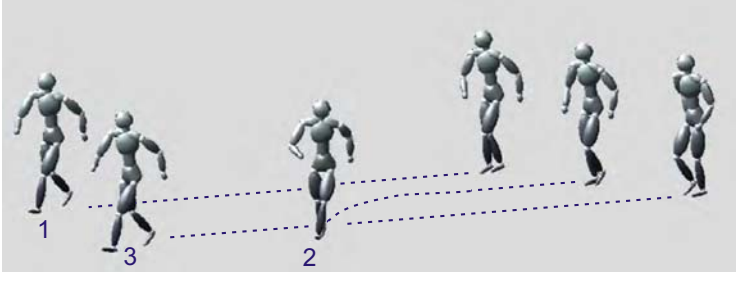


Fig. 8. Example video showing a comparison of the synchronization behavior of a model, which is generated by our method with a standard approach such as PCA. See text for more details.

Van der Pol oscillators, where we tried to optimize the coupling for maximum naturalness of the obtained animation. The detailed comparison shows that the avatar whose motion was generated by the novel architecture (oscillator dynamics with 6 degrees of freedom) shows a quite natural-looking transition from its initial state to the equilibrium state that is synchronized with the leader. The movement of the avatar whose movement was generated using PCA (oscillator dynamics with 14 degrees of freedom) shows artifacts. These artifacts are even increased if translatory body motion is added by enforcing the kinematic foot-contact constraints on the ground, resulting in a turning motion of the avatar (see Fig. 8 and demo movie[3]). If the internal coupling strength within the avatars is increased the synchronization between multiple avatars slows down and an unnatural reduction of step size arises. If the number of components in the PCA

[3] http://www.uni-tuebingen.de/uni/knv/arl/avi/weak_transl.avi

model is increased to 12, similar problems remain also for stronger coupling forces (cf. demo movie[4]). The proposed novel trajectory model thus tends to produce more natural transitions between different coordination states. Present work focuses on a more systematic quantitative comparison between different methods.

6.2 Continuous Style Morphing Online

The algorithm for online style morphing (Section 5.1) was tested by applying it to a complex sequence of locomotion patterns, which is a subset of steps from a folk dance. The movements were generated by interpolation between five prototypical gaits in our data set: straight walking neutral and happy, rotation steps of backwards and forward walks, walking with stooped posture and turning on the spot. Even though these types of locomotion were quite different we were able to approximate them with only three different source terms. Applying the proposed technique for the interpolation between weights, posture vectors, and time delays we were able to create quite realistic transitions between these different patterns, resulting in a complex sequence of steps that could be part of a dancing scenario (demo movie[5]).

6.3 Integration of Periodic and Non-periodic Synergies

We tested that our method correctly identifies the spatial localization of periodic and non-periodic motion components (synergies) in the training data. The mixing weights w_{ij} for the fourth non-periodic source are significantly different from zero only for the angles of the shoulder and elbow joint, reflecting the fact that in our data set the non-periodic movements were mainly executed with the arms. The separation of different spatially localized movement components makes it possible to modify the movement styles of different synergies separately. This is particularly true for periodic and non-periodic primitives, and novel movement patterns can be generated by combining such primitives in ways that were not present in the training data. A simple example can be downloaded as movie[6]. In this case, an arm movement is superposed with different relative timings to the periodic movements of the feet of the two avatars. The sequence is generated using the blending method described in Section 5.1 by recombining the learned synergies with dynamically changing morphing weights and an external trigger signal that initiates the raising and lowering of the arms of the avatars. The whole sequence can be coupled easily to an external rhythm that represents, for example, the beat of the music.

The same method can be applied for more complex scenarios, like dancing of two couples. As illustrated in Fig. 9 and the demo movie[7], one of the two

[4] http://www.uni-tuebingen.de/uni/knv/arl/avi/weak12_transl.avi
[5] http://www.uni-tuebingen.de/uni/knv/arl/avi/stylemorph.avi
[6] http://www.uni-tuebingen.de/uni/knv/arl/avi/armsup.avi
[7] http://www.uni-tuebingen.de/uni/knv/arl/avi/dance2.avi

Fig. 9. Dancing figure from a folk dance. The sequence was generated online by blending and recombination of the learned synergies with dynamically changing morphing weights. An external trigger signal initiates the raising and lowering of the arms of the avatars. (See text for further details.)

couples forms a bridge with the arms while locomoting forwards, while the second couple walks through this bridge one-by-one, in a crouched posture. Then the partners turn around and change roles. The whole scenario was simulated online, modulating the dynamics by few binary control signals that define the action mode of each avatar (forming bridge, crouching, or turning). In this case, periodic and non-periodic movement primitives were coupled in a way that permits an initiation of the the arm movement at any time during the step cycle (e.g. dependent on whether the partner has already completed his turning step). In addition, the dynamics of the characters were coupled to an external rhythm (representing the beat of the music).

6.4 Self-organization of a Folk Dance

To further explore the capabilities of the proposed framework we tried to self-organize a folk dance scenario, where a larger group of avatars has to walk in synchrony with the music in a formation. After reaching the wall of the ball room the characters have to run back to their initial point and to re-synchronize with the music and the other dancers. A video showing this self-organized animation scenario can be downloaded[8].

Figure 10 illustrates the simulated scenario. It is characterized by four spatial sectors:

1) At the entrance of the corridor the characters wait for the corresponding partner and start to move in synchrony. This behavior is implemented by the introduction of couplings between the avatars of one couple and between subsequent couples within the corridor and a coupling to an external periodic signal derived from the music.
2) At the end of the corridor the two avatars of each couple separate, and the coupling between their oscillators is removed. This results in an asynchronous movement that is controlled by the navigation dynamics (Section 5.2). In addition, within this zone the emotional walking style of the characters changes

[8] http://www.uni-tuebingen.de/uni/knv/arl/avi/dance.wmv

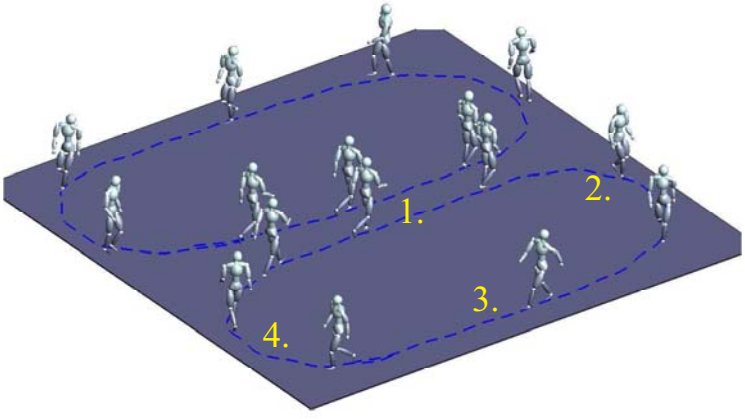


Fig. 10. Simulation of a 'folk dance'. Behavior is self-organized by combining the different elements described in this chapter. Characters act fully autonomously and synchronize with the music in the central corridor. See text for details.

from happy to neutral. The curved walking paths were generated by defining appropriate intermediate goal points.
3) Along the straight paths outside the corridor the avatars accelerate to catch up with their partner at the beginning of the corridor in time, simulated by a temporary increase of the eigenfrequency of the corresponding oscillators.
4) In the last zone the characters decelerate, modeled by a decrease of the eigenfrequency of the oscillators. A difficult problem is the re-synchronization with the correct foot at the entrance of the corridor. This is accomplished by slightly adjusting the oscillator frequencies to ensure re-synchronization with the appropriate leg.

7 Conclusion and Future Work

We have presented a novel framework for real-time character animation that was inspired by the concept of 'synergies' from motor control. Synergies were learned from motion capture data applying a novel algorithm for the solution of anechoic mixture problems. This algorithm learns highly compact models for motion trajectories from motion capture data resulting in models with few source terms and high approximation quality. We demonstrated how this model can be linked to dynamical systems in a way that ensures accurate approximation of the original trajectories and a simple system dynamics. This makes the approach suitable for the design of complex systems with many avatars. Couplings between different dynamic primitives were designed exploiting concepts of contraction theory [30], permitting a design of coupling between different dynamic primitives that ensure a well-controlled overall stable behavior of the system dynamics. The developed framework can be easily combined with key elements of real-time animation systems, such as style morphing, navigation or the coupling to

external rhythms. We demonstrated a variety of application scenarios, like the simulation of coordinated and navigating crowds and the automatic generation of dancing scenes.

Future work will concentrate on the following aspects: 1) Extending the framework for more complex classes of movements; 2) development of a general systematic stability theory for such architectures; and 3) systematic comparison with other methods by comparison with ground truth data for movements in interactive scenarios, including psychophysical experiments.

Acknowledgements

This work has been supported by DFG Forschergruppe 'Perceptual Graphics', the Human Frontier Science Program (HFSP), the EC FP6 project 'COBOL' and the Volkswagenstiftung. Additional support was provided by the Hermann und Lilly-Schilling-Stiftung. We thank W. Strasser and A. Schilling for interesting discussions, and W.Ilg and C. Roether for help with the motion capture.

References

[1] Rose, C., Cohen, M., Bodenheimer, B.: Verbs and adverbs: Multidimensional motion interpolation. IEEE Computer Graphics and Applications 18(5), 32–40 (1998)
[2] Witkin, A., Popović, Z.: Motion warping. In: Proc. ACM SIGGRAPH 1995, vol. 29, pp. 105–108 (1995)
[3] Gleicher, M.: Retargetting motion to new characters. In: Proc. ACM SIGGRAPH 1998, pp. 33–42 (1998)
[4] Gleicher, M., Shin, H.J., Kovar, L., Jepsen, A.: Snap-together motion: Assembling run-time animation. ACM Trans. on Graphics, SIGGRAPH 2003 22(3), 702 (2003)
[5] Arikan, O., Forsyth, D.A., O'Brien, J.F.: Motion synthesis from annotations. ACM Trans. on Graphics, SIGGRAPH 2003 22(3), 402–408 (2003)
[6] Grzeszczuk, R., Terzopoulos, D., Hinton, G.: Neuroanimator: Fast neural network emulation and control of physics based models. In: Int. Conf. on Comp. Graph. and Interactive Techniques, Proc. ACM SIGGRAPH 1998, pp. 9–20 (1998)
[7] Shao, W., Terzopoulos, D.: Artificial intelligence for animation: Autonomous pedestrians. Proc. ACM SIGGRAPH 2005 69(5-6), 19–28 (2005)
[8] Hsu, E., Pulli, K., Popović, J.: Style translation for human motion. ACM Trans. on Graphics, SIGGRAPH 2005 24(3), 1082–1089 (2005)
[9] Chai, J., Hodgins, J.: Performance animation from low-dimensional control signals. ACM Trans. on Graphics, SIGGRAPH 2005 24(3), 686–696 (2005)
[10] Bernstein, N.A.: The coordination and regulation of movements. Pergamon Press, Oxford (1967)
[11] Flash, T., Hochner, B.: Motor primitives in vertebrates and invertebrates. Curr. Opin. Neurobiol. 15(6), 660–666 (2005)
[12] d'Avella, A., Bizzi, E.: Shared and specific muscle synergies in neural motor behaviours. Proc. Natl. Acad. Sci. 102(8), 3076–3081 (2005)
[13] Ivanenko, Y., Poppele, R., Lacquaniti, F.: Five basic muscle activation patterns account for muscle activity during human locomotion. Journal of Physiology 556, 267–282 (2004)

[14] Santello, M., Flanders, M., Soechting, J.: Postural hand synergies for tool use. Journal of Neuroscience 18(23), 10105–10115 (1998)
[15] Safanova, A., Hodgins, J.K., Pollard, N.S.: Synthesizing physically realistic human motion in low-dimensional, behavior-specific spaces. ACM Trans. on Graphics, Proc. SIGGRAPH 2004 23(3), 514–521 (2004)
[16] Park, A., Mukovskiy, A., Omlor, L., Giese, M.A.: Synthesis of character behaviour by dynamic interaction of synergies learned from motion capture data. In: The 16-th Int. Conf. in Central Europe on Comp. Graphics, Visualization and Computer Vision 2008, WSCG 2008, pp. 9–16 (2008)
[17] Park, A., Mukovskiy, A., Omlor, L., Giese, M.A.: Self organized character animation based on learned synergies from full-body motion capture data. In: International Conference on Cognitive Systems, CogSys 2008 (2008)
[18] Unuma, M., Anjyo, K., Takeuchi, R.: Fourier principles for emotion-based human figure animation. Proc. ACM SIGGRAPH 1995 22, 91–96 (1995)
[19] Schaal, S., Ijspeert, A., Billard, A.: Computational approaches to motor learning by imitation. Philos. Trans. R. Soc. of London, Biol. Sci. 358(1431), 537–547 (2003)
[20] Ijspeert, A., Nakanishi, J., Schaal, S.: Learning attractor landscapes for learning motor primitives. Adv. Neural Inf. Process Systems 15, 1547–1554 (2002)
[21] Omlor, L., Giese, M.A.: Extraction of spatio-temporal primitives of emotional body expressions. Neurocomputing 70, 10–12 (2007)
[22] Omlor, L., Giese, M.A.: Blind source separation for over-determined delayed mixtures. Adv. in Neural Inf. Process Systems 19, 1049–1056 (2006)
[23] Schöner, G., Dose, M., Engels, C.: Dynamics of behavior: Theory and applications for autonomous robot architectures. Robotics and Autonomous Systems 16(2-4), 213–245 (1995)
[24] Buchli, J., Righetti, L., Ijspeert, A.J.: Engineering entrainment and adaptation in limit cycle systems - from biological inspiration to applications in robotics. Biological Cybernetics 95(6), 645–664 (2006)
[25] Fod, A., Matarić, M.J., Jenkins, O.C.: Motor primitives in vertebrates and invertebrates. Autonomous Robots 12(1), 39–54 (2002)
[26] Soechting, J., Lacquaniti, F.: Invariant characteristics of a pointing movement in man. Journal of Neuroscience 1, 710–720 (1981)
[27] Morasso, P.: Spatial control of arm movements. Exp. Brain Res. 42(2) (1981)
[28] Vapnik, N.V.: Statistical Learning Theory. Wiley Interscience, Hoboken (1998)
[29] Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines (2001), http://www.csie.ntu.edu.tw/~cjlin/libsvm
[30] Wang, W., Slotine, J.J.E.: On partial contraction analysis for coupled nonlinear oscillators. Biological Cybernetics 92(1), 38–53 (2005)
[31] Warren, W.: The dynamics of perception and action. Psychological Review 113(2), 358–389 (2006)

2D Human Pose Estimation in TV Shows

Vittorio Ferrari[1], Manuel Marín-Jiménez[2], and Andrew Zisserman[3]

[1] ETH Zurich
ferrari@vision.ee.ethz.ch
[2] University of Granada
mjmarin@decsai.ugr.es
[3] University of Oxford
az@robots.ox.ac.uk

Abstract. The goal of this work is fully automatic 2D human pose estimation in unconstrained TV shows and feature films. Direct pose estimation on this uncontrolled material is often too difficult, especially when knowing nothing about the location, scale, pose, and appearance of the person, or even whether there is a person in the frame or not.

We propose an approach that progressively reduces the search space for body parts, to greatly facilitate the task for the pose estimator. Moreover, when video is available, we propose methods for exploiting the temporal continuity of both appearance and pose for improving the estimation based on individual frames.

The method is fully automatic and self-initializing, and explains the spatio-temporal volume covered by a person moving in a shot by soft-labeling every pixel as belonging to a particular body part or to the background. We demonstrate upper-body pose estimation by running our system on four episodes of the TV series *Buffy the vampire slayer* (i.e. three hours of video). Our approach is evaluated quantitatively on several hundred video frames, based on ground-truth annotation of 2D poses[1]. Finally, we present an application to full-body action recognition on the Weizmann dataset.

1 Introduction

Our aim is to detect and estimate 2D human pose in video, i.e. recover a distribution over the spatial configuration of body parts in every frame of a shot. Various pose representations can then be derived, such as a soft-labelling of every pixel as belonging to a particular body part or the background (figure 1b); or the 'stickman' of figure 1c, indicating the location, orientation, and size of body parts. Note, our objective here is not to estimate 3D human pose as in [6, 28, 31].

We wish to obtain pose estimates in highly challenging uncontrolled imaging conditions, typical of movies and TV shows (figures 10, 11). Achieving this is one of the main contributions of the paper. In this setting, images are often very cluttered, and a person might cover only a small proportion of the image area, as they can appear at any scale. Illumination varies over a diverse palette of lighting conditions, and is often quite dark, resulting in poor image contrast. A person's appearance is unconstrained, as

[1] available at www.robots.ox.ac.uk/~vgg/data/stickmen/index.html

D. Cremers et al. (Eds.): Visual Motion Analysis, LNCS 5604, pp. 128–147, 2009.

a b c

Fig. 1. Objective of this work. (a) Input image. (b) Soft-labelling of pixels to body parts or background. Red indicates torso, green upper arms, blue lower arms and head. Brighter pixels are more likely to belong to a part. Color planes are added up, so that purple indicates overlap between lower-arm and torso, yellow between upper-arm and torso, and so on. (c) Stickman representation of pose, obtained by fitting straight line segments to the segmentations in (b). For enhanced visibility, the lower arms are in yellow and the head is in purple.

they can wear any kind of clothing, including body-tight or loose, short or long sleeves, and any colors/textures. The background is unknown and changes over time, preventing the use of background subtraction techniques [5, 9]. Finally, the camera is usually moving, causing motion blur, and multiple people can be present at the same time and can occlude each other during a shot.

Our method covers all poses within the upper-body frontal range. Special attention is given to the arms, as they carry most of the information necessary to distinguish pose. The proposed method supports arms folded over the torso, stretching outwards, pointing forward, etc.

The need for such human centered tracking is evident, with applications ranging from video understanding and search through to surveillance. Indeed 2D human segmentation is often the first step in determining 3D human pose from individual frames [1]. We illustrate the use of the extracted poses with an application to action recognition on the Weizmann dataset.

An earlier version of this work first appeared at [10].

1.1 Approach Overview

We overview the method here for the upper-body case, where there are 6 parts: head, torso, and upper/lower right/left arms (figure 1). Full details are given in section 2. The method is also applicable to full bodies, as demonstrated in section 4.

A recent and successful approach to 2D human tracking in video has been to detect in every frame, so that tracking reduces to associating the detections [24, 30]. We adopt this approach where detection in each frame proceeds in three stages, followed by a final stage of transfer and integration of models across frames.

In our case, the task of pose detection is to estimate the parameters of a 2D articulated body model. These parameters are the (x, y) location of each body part, its orientation θ, and its scale. Assuming a single scale factor for the whole person, shared by all body parts, the search space has $6 \times 3 + 1 = 19$ dimensions. Even after taking into account kinematic constraints (e.g. the head must be connected to the torso), there are still a huge number of possible configurations.

Since at the beginning we know nothing about the person's pose, clothing appearance, location and scale in the image, directly searching the whole space is a time consuming and very fragile operation (there are too many image patches that could be an arm or a torso!). Therefore, in our approach the first two stages use a weak model of a person obtained through an upper-body detector generic over pose and appearance. This weak model only determines the approximate location and scale of the person, and roughly where the torso and head should lie. However, it knows nothing about the arms, and therefore very little about pose. The purpose of the weak model is to *progressively reduce the search space* for body parts.

The next stages then switch to a stronger model, i.e. a pictorial structure [9,23,24] describing the spatial configuration of all body parts and their appearance. In the reduced search space, this stronger model has much better chances of inferring detailed body part positions.

1. Human detection and tracking. We start by detecting human upper-bodies in every frame, using a sliding window detection based on Histograms of Oriented Gradients [7], and associate detections over time. Each resulting track connects the detections of a different person. It carves out of the total spatio-temporal volume the smaller subvolume covered by a person moving through the shot. This reduces the search space by setting bounds on the possible (x, y) locations of the body parts and by fixing their scale, thus removing a dimension of the search space entirely.

2. Foreground highlighting. At this stage the search for body parts is only limited by the maximum extent possible for a human of that scale centered on the detected position. We restrict the search area further by exploiting prior knowledge about the structure of the detection window. Relative to it, some areas are very likely to contain part of the person, whereas other areas are very unlikely. This allows the initialization of a GrabCut segmentation [25], which removes part of the background clutter. This stage further constrains the search space by limiting the (x, y) locations to lie within the foreground area determined by GrabCut.

3. Single-frame parsing. We obtain a first pose estimate based on the *image parsing* technique of Ramanan [23]. The area to be parsed is restricted to the region output of foreground highlighting. Since the person's scale has been fixed by stage 1), no explicit search for body parts over scales is necessary.

In order to reduce the double-counting problems typical of pictorial structures [29], we extend the purely kinematic model of [23] to include *"repulsive" edges* favoring configurations of body parts where the left and right arms are not superimposed.

Both foreground highlighting and parsing stages are run separately for each detection in a track.

4. Spatio-temporal parsing. The appearance of the body parts of a person changes little within a shot. Moreover, the position of body parts changes smoothly over time. We exploit both kinds of temporal continuity in a second pose estimation procedure which (i) uses appearance models integrated from multiple frames where the system is confident about the estimated pose; and (ii) infers over a joint spatio-temporal model of pose, capturing both kinematic/repulsive constraints within a frame, and temporal continuity constraints between frames. As appearance is a powerful cue about the location

of parts, the better appearance models improve results for frames where parsing failed or is inaccurate. At the same time, the spatio-temporal model tightens the posterior distributions over part positions and disambiguates multiple modes hard to resolve based on individual frames.

The spatio-temporal parsing stage runs over an entire track, as a track connects all detections of a person. Multiple persons in the same shot result in separate tracks, for each of which we run spatio-temporal parsing separately.

1.2 Related Works

Our work builds mainly on the *Learning to Parse* approach by Ramanan [23], which provides the pictorial structure inference engine we use in stage 3, and on the *Strike-a-pose* work [24]. The crucial difference to both works is the way the search space of body part configurations is treated. Thanks to the proposed detection and foreground highlighting stages, we avoid the very expensive and fragile search necessary in [23,24]. Moreover, compared to [24], our initial detection stage is *generic over pose*, so we are not limited to cases where the video contains a pre-defined characteristic pose at a specific scale. We also generalize and improve the idea of transferring appearance models of [24]. Rather than using a single frame containing the characteristic pose, we integrate models over multiple frames containing any pose.

Previous use of pictorial structure models have tolerated only limited amounts of background clutter [9,23] and often assume knowledge of the person's scale [23,24] or background subtraction [9]. A few methods operate interactively from regions of interest provided by the user [19].

There are also methods that detect humans using generative models for the entire video sequences, e.g. [14,16]. However, to date these methods have been limited to relatively simple backgrounds and to no occlusion of the person.

Our spatio-temporal model (section 2.4) is most closely related to that of [28,29], but our framework is fully automatic (it does not need any manual initialization or background subtraction).

The work of [20] recovers unusual, challenging body configurations in sports images by combining segmentation and detectors trained to specific body parts, but requires a person centered in the image and occupying most of it.

Finally, very recently two methods [3,12] have been presented for pose estimation of people walking in busy city environments where the camera, multiple people, as well as other objects move simultaneously. Both methods rely heavily on static and dynamic priors specific to walking motion. In contrast, our method makes no assumptions about expected poses, besides the person being upright, and is able to estimate a wide variety of body configurations (figures 10, 11, 12).

2 The Approach in Detail

2.1 Upper-Body Detection and Temporal Association

Upper-body detection. In most shots of movies and TV shows, only the upper-body is visible. To cope with this situation, we have trained an upper-body detector using the

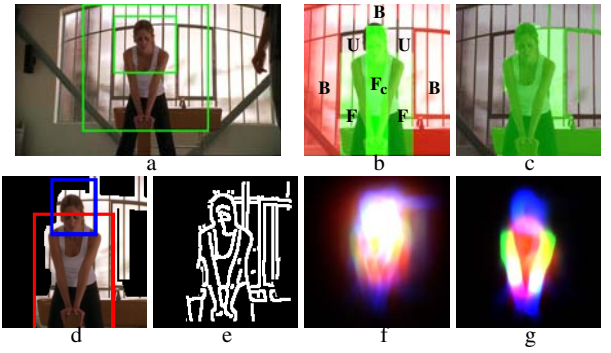


Fig. 2. Overview of the single-frame steps. 1. Upper body detection: The detected person (inner rectangle) and enlarged window where further processing is applied (outer rectangle). **2. Foreground highlighting:** (b) subregions for initializing GrabCut. (c) foreground region output by GrabCut. **3. Parsing:** (d) area to be parsed $\mathcal{F}$ (dilated from (c)) and (e) edges within $\mathcal{F}$. (f) posterior of the part positions $p(l_i|I)$ after the edge-based inference. (g) posterior after the second inference, based on edges and appearance. This visualization is obtained by convolving rectangles representing body parts with their corresponding posterior.

approach of Dalal and Triggs [7], which achieves excellent performance on the related task of full-body pedestrian detection. Image windows are spatially subdivided into tiles and each is described by a Histogram of Oriented Gradients. A sliding-window mechanism then localizes the objects. At each location and scale the window is classified by a linear SVM as containing the object or not. Photometric normalization within multiple overlapping blocks of tiles makes the method particularly robust to lighting variations.

The training data consists of 96 video frames from three movies (*Run Lola run, Pretty woman, Groundhog day*), manually annotated with a bounding-box enclosing an upper-body. The images have been selected to maximize diversity, and include many different actors, with only a few images of each, wearing different clothes and/or in different poses. No images from the test material (shots from *Buffy the Vampire Slayer*) were used for training.

Following Laptev [17], the training set is augmented by perturbing the original examples with small rotations and shears, and by mirroring them horizontally. This improves the generalization ability of the classifier. By presenting it during training with misalignments and variations, it has a better chance of noticing true characteristics of the pattern, as opposed to details specific to individual images. The augmented training set is 12 times larger and contains more than 1000 examples.

We choose an operating point of 90% detection-rate at 0.5 false-positives per image. This per-frame detection-rate translates into an almost perfect per-track detection-rate after temporal association (see below). Although individual detections might be missed, entire tracks are much more robust. Moreover, we remove most false-positives by weeding out tracks shorter than 20 frames.

In practice, this detector works well for viewpoints up to 30 degrees away from straight frontal, and also detects back views (figures 10, 11).

Temporal association. After applying the upper-body detector to every frame in the shot independently, we associate the resulting bounding-boxes over time by maximizing their temporal continuity. This produces *tracks*, each connecting detections of the same person.

Temporal association is cast as a grouping problem [30], where the elements to be grouped are bounding-boxes. As similarity measure we use the area of the intersection divided by the area of the union (IoU), which subsumes both location and scale information, damped over time. We group detections based on these similarities using the Clique Partitioning algorithm of [11], under the constraint that no two detections from the same frame can be grouped. Essentially, this forms groups maximizing the IoU between nearby time frames.

This algorithm is very rapid, taking less than a second per shot, and is robust to missed detections, because a high IoU attracts bounding-boxes even across a gap of several frames. Moreover, the procedure allows persons to overlap partially or to pass in front of each other, because IoU injects a preference for *continuity scale* in the grouping process, in addition to location, which acts as a disambiguation factor.

In general, the 'detect & associate' paradigm is substantially more robust than regular tracking, as recently demonstrated by several authors [22,30].

2.2 Foreground Highlighting

The location and scale information delivered by an upper-body detection greatly constrains the space of possible body parts. They are now confined to the image area surrounding the detection, and their approximate size is known, as proportional to the detection's scale. However, to accommodate for all possible arm poses we must still explore a sizeable area (figure 2a). Stretching out the arms in any direction forms a large circle centered between the shoulders. In challenging images from TV shows, this area can be highly cluttered, confusing the body part estimator.

Fortunately, we have *prior knowledge* about the structure of the search area. The head lies somewhere in the middle upper-half of the detection window, and the torso is directly underneath it (figure 2b). This is known because the detector has been explicitly trained to respond to such structures. In contrast the arms could be anywhere. We propose to exploit this knowledge to initialize GrabCut [25], by learning initial foreground/background color models from regions where the person is likely to be present/absent. The resulting segmentation removes much of the background clutter, substantially simplifying the later search for body parts (figure 2c).

Let $\mathcal{R}$ be a region of interest obtained by enlarging the detection window as in figure 2a. $\mathcal{R}$ is divided into four subregions F, F_c, B, U (see figure 2b). GrabCut is initialized as follows: the foreground model is learnt from F and F_c (F_c is known to belong to the person, while F contains mostly foreground, but some background as well); and the background model from B (it covers mostly background, but it might also include part of the arms, depending on the pose). Furthermore, the region F_c is clamped as foreground, but grabcut is free to set pixel labels in all other subregions (we have extended the original GrabCut algorithm to enable these operations). The U region is neutral and no color model is learnt from it. The setup accurately expresses our prior knowledge and results in a controlled, upper-body-specific segmentation, assisted by as

Fig. 3. Examples of foreground highlighting

much information as we can derive from the previous object detection process. Near the head, B and F_c compete for the U region, with the foreground growing outwards until it meets a background-colored area, resulting in a good head segmentation. Along the sides, the background floods into the initial F to segment the shoulders, while at the same time the arms get labeled as foreground because they are colored more similarly to the initial F than to the initial B (figure 3).

The above procedure is rather conservative, and it often retains parts of the background. The goal is not to achieve a perfect segmentation, but to reduce the amount of background clutter (figure 3). It is more important not to lose body parts, as they cannot be recovered later. To validate this, we have inspected 1584 frames of a *Buffy* episode (i.e. every 10th frame) and only in 71 a body part was lost (4.5%). In contrast to traditional background subtraction, used in many previous works to extract silhouettes [5, 9, 13], our method does not need to know the background *a priori*, and allows the background to change over time (in video).

2.3 Single-Frame Parsing

Our main goal is to explain the spatio-temporal volume covered by a person moving in a shot. In particular, we want to estimate the 2D pose of the person, as the location, orientation and size of each body part. Ideally, the exact image regions covered by the parts should also be found. For estimating 2D pose in individual video frames, we build on the *image parsing* technique of Ramanan [23]. In the following we first briefly summarize it, and then describe our extensions.

Image parsing [23]. A person is represented as a pictorial structure composed of body parts tied together in a tree-structured conditional random field (figure 5a). Parts, l_i, are oriented patches of fixed size, and their position is parametrized by location and orientation. The posterior of a configuration of parts $L = \{l_i\}$ given an image I can be written as a log-linear model

$$P(L|I) \propto \exp\left(\sum_{(i,j)\in E} \Psi(l_i, l_j) + \sum_i \Phi(l_i)\right) \tag{1}$$

The pairwise potential $\Psi(l_i, l_j)$ corresponds to a spatial prior on the relative position of parts and embeds the kinematic constraints (e.g. the upper arms must be attached to the torso). The unary potential $\Phi(l_i)$ corresponds to the local image evidence for a part in a particular position (likelihood). Since the model structure E is a tree, inference is performed exactly and efficiently by sum-product Belief Propagation [4].

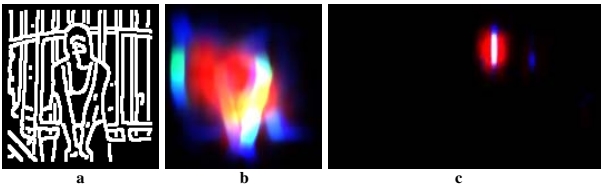


Fig. 4. The *image parsing* pose estimation algorithm of [23] applied to the image in figure 1. (a) All edges inside region $\mathcal{R}$, without filtering them through foreground highlighting. (b) Parsing applied to the whole region $\mathcal{R}$. It achieves a worse estimate than when helped by foreground highlighting, figure 2g, because it is attracted by the bars in the background. (c) Parsing applied directly to the whole image, without reducing the search space to $\mathcal{R}$ based on the initial person detection. It fails entirely.

The key idea of [23] lies in the special treatment of Φ. Since the appearance of neither the parts nor the background is known at the start, only edge features are used. A first inference based on edges delivers soft estimates of body part positions, which are used to build appearance models of the parts (e.g. in figure 2f the torso is in red). Inference in then repeated using an updated Φ incorporating both edges and appearance. The process can be iterated further, but in this paper we stop at this point. The technique is applicable to quite complex images because (i) the appearance of body parts is a powerful cue, *and* (ii) appearance models can be learnt from the image itself through the above two-step process.

The appearance models used in [23] are color histograms over the RGB cube discretized into $16 \times 16 \times 16$ bins. We refer to each bin as a *color* c. Each part l_i has foreground and background likelihoods $p(c|fg)$ and $p(c|bg)$. These are learnt from a part-specific soft-assignment of pixels to foreground/background derived from the posterior of the part position $p(l_i|I)$ returned by parsing. The posterior for a pixel to be foreground given its color $p(fg|c)$ is computed using Bayes' rule and used during the next parse.

As in [23], in our implementation we explicitly maintain a 3D binned volume to represent the possible (x, y, θ) positions of each part (discretization: every pixel for (x, y) and 24 steps for θ). This dense representation avoids the sampling needed by particle representations (e.g. [28,29]). The kinematic potential Ψ has a relative location (x, y) and a relative orientation (θ) components. The former gives zero probability if $(l_j - l_i)$ is out of a box-shaped tolerance region around the expected relative location (i.e. the parts *must* be connected [23]). The relative orientation component is a discrete distribution over $\theta_i - \theta_j$. The parameters of Ψ are learned from training data in [23]. The relative orientation prior is nearly uniform, allowing our approach to estimate a variety of poses.

When [23] is run unaided on a highly cluttered image such as figure 1a, without any idea of where the person might be or how large it is, parsing fails entirely (figure 4c). There are simply too many local image structures which could be a limb, a head, or a torso. This is assessed quantitatively in section 3.

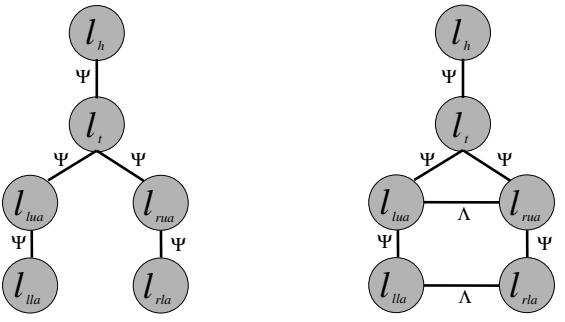


Fig. 5. Single-frame models. Each node represents a body part (h: head, t: torso, left/right upper/lower arms lua, rua, lla, rla). (a) The kinematic tree includes edges between every two body parts which are physically connected in the human body. (b) The repulsive model extends the kinematic tree with edges between opposite-sided arm parts.

We reduce the space explored by parsing based on three sources of information:

- (i) the *location and scale information* supplied by the upper-body detector, is used to define the enlarged search region $\mathcal{R}$. Parsing is run only within $\mathcal{R}$, rescaled to a fixed size, tuned to roughly yield the part sizes expected by the parser. Thanks to the proper use of scale information from detection, we effectively obtain scale-invariant pose estimation, without having to explicitly search for body parts at multiple scales. This significantly reduces ambiguity and false positive detections.
- (ii) *Foreground highlighting*. We further simplify pose estimation by restricting the area to be parsed to the region $\mathcal{F} \subset \mathcal{R}$ output of foreground highlighting (figure 2d). This is realized by removing all edges outside $\mathcal{F}$ and setting all pixels $\mathcal{R} \setminus \mathcal{F}$ to black. This causes the image evidence $\Phi(l_i)$ to go to $-\infty$ for $l_i \notin \mathcal{F}$, and hence it is equivalent to constraining the space of possible poses.
- (iii) *Head and torso constraints*. A final assistance is given by mildly constraining the (x, y) location of the head and torso based on our prior knowledge about the spatial structure of $\mathcal{R}$ (see section 2.2). The constraints come in the form of broad subregions $\mathcal{H}, \mathcal{T} \in \mathcal{R}$ where the head and torso must lie, and are realized by setting $\Phi(l_{head}), \Phi(l_{torso})$ to $-\infty$ for $l_i \notin \mathcal{H}, \mathcal{T}$ (figure 2d). These constraints directly reflect our prior knowledge from the detection process and therefore do not limit the range of poses covered by the parser (e.g. for the arms).

All the above aids to pose estimation are made possible from the initial generic upper-body detection. Foreground highlighting and location constraints can only be automated when building on a detection window. The combined effect of these improvements is a vastly more powerful parser, capable of estimating 2D pose in a highly cluttered image, even when the person occupies only a small portion of it. Moreover, parsing is now faster, as it searches only an image region, supports persons at multiple scales, and multiple persons at the same time, as each detection is parsed separately.

Repulsive model. A well-known problem with pictorial structure models evaluated as trees such as the one above, is that different body parts can take on similar (x, y, θ)

states, and therefore cover the same image pixels. Typically this happens for the left and right lower (or upper) arms, when the image likelihood for one is substantially better than the likelihood for the other. It is a consequence of the assumed independence of the left and right arms in the tree. This is referred to as the *double-counting problem* and was also noticed by other authors [29]. One solution, adopted in previous work, is to explicitly model limb occlusion by introducing layers into the model [2,15,29], though the graphical model is then no longer a tree.

Here, in order to alleviate the double-counting problem we add to the kinematic tree model two *repulsive edges* (figure 5b). The first edge connects the left upper arm (lua) to the right upper arm (rua), while the second edge connects the left lower arm (lla) to the right lower arm (rla). The posterior of a configuration of parts in the extended model becomes

$$P(L|I) \propto \exp\left(\sum_{(i,j)\in E} \Psi(l_i, l_j) + \sum_i \Phi(l_i) + \Lambda(l_{lua}, l_{rua}) + \Lambda(l_{lla}, l_{rla})\right) \tag{2}$$

The repulsive prior $\Lambda(l_i, l_j)$ gives a penalty when parts l_i and l_j overlap, and no penalty when they don't

$$\Lambda(l_i, l_j) = \begin{cases} w_\Lambda & \text{if } |l_i - l_j| \leq t_\Lambda \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

Therefore, the extended model prefers configurations of body parts where the left and right arms are not superimposed. It is important to notice that this new model does not forbid configurations with overlapping left/right arms, but is *biased against them*. If the image evidence in their favor is strong enough, inference will return configurations with overlapping arms (figure 10-c2). This properly reflects our prior knowledge that, in the majority of images, the arms don't occlude each other.

Since the extended graphical model has loops, we perform approximate inference with sum-product Loopy Belief Propagation, which in practice delivers a good estimate of the posterior marginals and is computationally efficient. The weight w_Λ and the threshold t_Λ are manually set and kept fixed for all experiments of this paper.

Figure 6 shows a typical case where the purely kinematic model delivers a posterior whose mode puts both lower arms on the left side, while the extended model yields the correct pose, thanks to the repulsive edges.

2.4 Spatio-Temporal Parsing

Parsing treats each frame independently, ignoring the temporal dimension of video. However, all detections in a track cover the same person, and people wear the same clothes throughout a shot. As a consequence, the appearance of body parts is quite stable over a track. In addition to this continuity of appearance, video offers also continuity of geometry: the position of body parts changes smoothly between subsequent frames.

In this section, we exploit the continuity of appearance for improving pose estimations in particularly difficult frames, and the continuity of geometry for disambiguating multiple modes in the positions of body parts, which are hard to resolve based on individual frames.

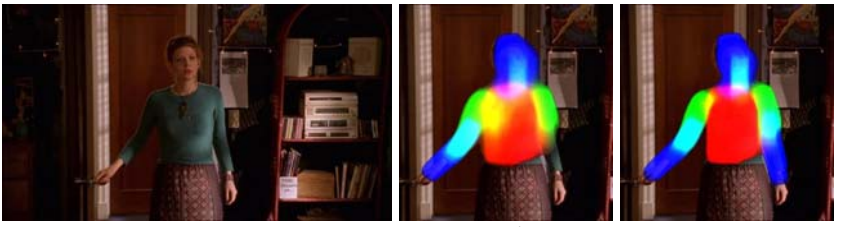


Fig. 6. Impact of repulsive model. (a) Original image. (b) Pose estimated by the kinematic tree model. As in previous figures, the visualization is obtained by convolving rectangles representing body parts with their corresponding posterior probability over (x, y, θ). The right (in the image) upper arm (green) has two equally probable modes, one at the correct position, and one on the left upper arm. For the right lower arm (blue) instead, nearly all of the probability mass is on the left side, while only very little is on the correct position. This double-counting phenomenon visibly affects the estimation. (c) Pose estimated after extending the model with repulsive edges. The position of the right lower arm is now correctly estimated.

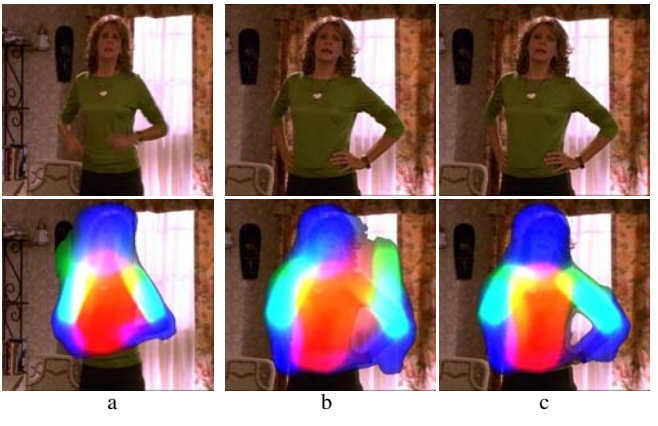


Fig. 7. Impact of transferring appearance models. (a) one of several frames with low TPE after single-frame parsing, from which integrated appearance models are learnt (top). The pose estimate is quite clear (bottom). (b) a frame with high TPE. The system is uncertain whether the right arm lies on the window or at its actual position (bottom). (c) Parsing the frame in (b) while using the learned integrated appearance models. The right arm ambiguity is now resolved (bottom), as the system acquires from other frames the knowledge that white is a color occurring on the background only.

Learning integrated appearance models. The idea is to find the subset of frames where the system is confident of having found the correct pose, integrate their appearance models, and use them to parse the whole track again (figure 7). This improves pose estimation in frames where parsing has either failed or is inaccurate, because appearance is a strong cue about the location of parts.

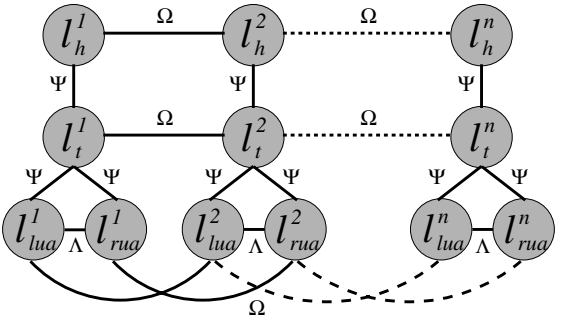


Fig. 8. Spatio-temporal model. For clarity, only head (l_h), torso (l_t), and left/right upper arms (l_{lua}, l_{rua}) are drawn.

Frames where parsing infers a highly confident configuration of body parts provide good reference appearance models (figure 7a). The measure of confidence used here is the entropy of the posterior of the part positions $p(l_i|I)$, accumulated over all parts $L = \{l_i\}$ to give the *total pose entropy* TPE:

$$TPE(L|I) = -\sum_i \left(\sum_{x,y,\theta} p(l_i = \{x, y, \theta\}|I) \cdot \log p(l_i = \{x, y, \theta\}|I) \right) \quad (4)$$

Rather than simply selecting the single frame with the lowest TPE, we learn models by *integrating* over all frames with a similar low TPE. It can be shown [26] that the distribution minimizing the total KL-divergence to a set of distributions is their average. Hence, we integrate the foreground and background likelihoods $\{p_r(c|fg)\}, \{p_r(c|bg)\}$ from the reference frames r by averaging them. The integrated posteriors $p_i(fg|c)$ are then obtained by applying Bayes' rule.

The integrated models are richer, in that $p_i(fg|c)$ is nonzero for a broader range of colors, so they generalize to a larger number of frames. Moreover, they are more accurate, because estimated over a wider support. Examples of the benefits brought by using the learned integrated appearance models to re-parse frames are shown in figure 7, and by the difference between figure 2g (purely single-frame) and figure 1b (re-parsing).

Spatio-temporal inference. We extend the single-frame person model to include dependencies between body parts over time (figure 8). The extended model has a node for every body part in every frame of a continuous temporal window (11 frames in our experiments). The posterior of all configurations of parts $\{L^t\} = \{l_i^t\}$ given all frames $\{I^t\}$ can be written as

$$P(\{L^t\}|\{I^t\}) \propto$$

$$\exp\left(\sum_{t,i} \left(\sum_{j|(i,j)\in E} \Psi(l_i^t, l_j^t) + \Phi(l_i^t) + \Omega(l_i^t, l_i^{t+1}) + \Lambda(l_{lua}^t, l_{rua}^t) + \Lambda(l_{lla}^t, l_{rla}^t) \right) \right) \quad (5)$$

In addition to the *kinematic/repulsive* dependencies Ψ, Λ between different parts in a single frame, there are *temporal* dependencies Ω between nodes representing the same

part in subsequent frames. As a temporal prior $\Omega(l_i^t, l_i^{t+1})$ we use a simple box-shaped distribution limiting the difference in the $l_i^t = (x, y, \theta)$ position of a body part between frames. We use the integrated appearance models to obtain a better image likelihood Φ. Approximate inference in the spatio-temporal model with loops is carried out with sum-product Loopy Belief Propagation.

The spatio-temporal inference is a batch process treating all frames in the temporal window simultaneously, as opposed to traditional tracking, where only past frames can influence estimation in the current frame. The inference procedure outputs the full marginal posteriors $p(l_i^t|\{I^t\})$, defining the probability of every possible (x, y, θ) body part position in every frame. This is better than a single MAP solution [24], as any remaining ambiguity and uncertainty is visible in the full posteriors (e.g. due to blurry images, or tubular background structures colored like the person's arms). Finally, our joint spatio-temporal inference is better than simply smoothing the single-frame posteriors over time, as the kinematic dependencies within a frame and temporal dependencies between frames *simultaneously help each other*.

Thanks to the proposed joint spatio-temporal inference, the final pose estimates are tighter and more accurate than single-frame ones. As a typical effect, multiple modes in the positions of body parts are disambiguated, because modes not consistently recurring over time are attenuated by the temporal prior Ω (figure 9). Moreover, the estimated poses are now more temporally continuous, which is useful for estimating the motion of body parts for action recognition.

3 Upper-Body Pose Estimation Results

We have applied our pose estimation technique to episodes 2,4,5 and 6 of season five of *Buffy the vampire slayer*, for a total of more than 70000 video frames over about 1000 shots.

The examples in figures 10 and 11 show that the proposed method meets the challenges set in the introduction. It successfully recovers the configuration of body parts in spite of extensive clutter, persons of different size, dark lighting and low contrast (c3, f2, f3). Moreover, the persons wear all kinds of clothing, e.g. ranging from long sleeves to sleeveless (b3, a3, h3), and this is achieved despite the fact that their appearance is unknown *a priori* and was reconstructed by the algorithm. The system correctly estimated a wide variety of poses, including arms folded over the body (c1, c2, g1), stretched out (e3, f3), and at rest (e1, f1). The viewpoint doesn't have to be exactly frontal, as the system tolerates up to about 30 degrees of out-of-plane rotation (c1). Persons seen from the back are also covered, as the upper-body detector finds them and we don't rely on skin-color segmentation (e1, f1). Finally, the method deals with multiple persons in the same image and delivers a separate pose estimate for each (h2, note how the pose of each person is estimated independently).

We quantitatively assess these results on 69 shots divided equally among three of the episodes. We have annotated the ground-truth pose for four frames spread roughly evenly throughout each shot, by marking each body part by one line segment [20] (figure 10a). Frames were picked where the person is visible at least to the waist and the arms fit inside the image. This was the sole selection criterion. In terms of imaging

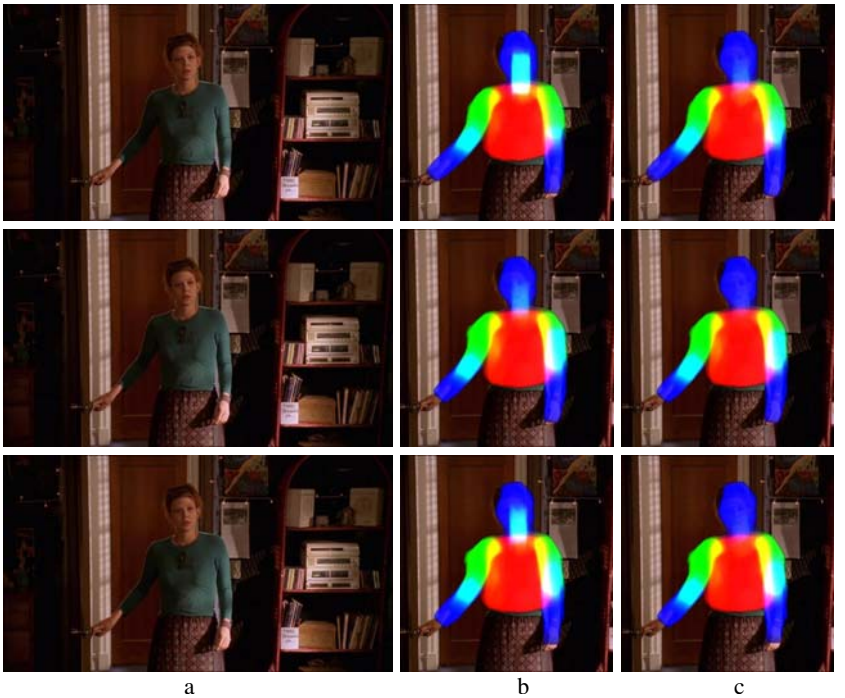


Fig. 9. Impact of spatio-temporal parsing. (a) Three subsequent video frames. (b) Pose estimated by single-frame parsing after transferring appearance models, but without dependencies between body parts over time. In the first and third frames, the right upper arm has a strong second mode on the face, but not in the second frame. (c) Pose estimated by the complete spatio-temporal model. The spurious mode has been largely eliminated.

conditions, shots of all degrees of difficulty have been included. A body part returned by the algorithm is considered correct if its segment endpoints lie within 50% of the length of the ground-truth segment from their annotated location.

The initial detector found an upper-body in 88% of the $69 \times 4 = 276$ annotated frames, and this places an upper bound on performance. Table 1 shows the percentage of the $243 \times 6 = 1458$ body parts in these frames which have been correctly estimated by several versions of our system. Our best result is 62.6%. The image parser of [23] using software supplied by the author, and run unaided directly on the image, achieves only 9.6%, thus highlighting the great challenge posed by this data, and the substantial improvements brought by our techniques. Helping [23] by constraining it by the location and scale delivered by the initial human detection causes performance to jump to 41.2% (section 2.1). Adding foreground highlighting further raises it to 57.9% (section 2.2). These results confirm that both search space reduction stages we proposed (starting from a detection and foreground highlighting) contribute considerably to the quality of the results. Transferring appearance models increases performance moderately to 59.4% (section 2.4). The improvement appears relatively small because in many cases the localization refinements are too fine to be captured by our coarse

Fig. 10. Pose estimation results I. (a1) example ground-truth 'stickman' annotation. All other subfigures are the output of the proposed method, with body part segmentations overlaid. For illustration, in (a2) we also overlay the stickman derived by our method. The color coding is as follows: head = purple, torso = red, upper arms = green, lower arms = yellow. In (c2) a pose with crossed arms is correctly estimated: the repulsive model does not prevent our system from dealing with these cases.

evaluation measure. On the other hand, this suggests that the proposed approach performs well also on static images. Extending the purely kinematic model of [23] with repulsive priors brings a last visible improvement to 62.6%, thanks to alleviating the double-counting problem (section 2.3).

Somewhat surprisingly, including temporal priors does not improve our evaluation score (section 2.4). This is due to two reasons. The first is that many cases where the estimated poses become more temporally continuous do not result in a better score. Our measure does not prize temporal smoothness, it only looks at the position of body parts in individual frames. The second reason is that temporal integration occasionally worsens the estimated poses. Due to the temporal prior, the posterior probability of a (x, y, θ) state in a frame depends also on nearby states in neighboring frames. Therefore, if a body part is 'missing' at its correct position in a frame, i.e. the unary

Fig. 11. Pose estimation results II. More example pose estimations. A fair sample of failures are also included, e.g. (f1) is missed as a detection, and the wrong pose is obtained in (e3) (rear person). Notice how the leftmost person in (h2) is largely occluded.

potential gives it near-zero probability, the posterior probability of the correct position in neighboring frames is decreased by the temporal prior (i.e. it propagates the miss over time; the same behavior also eliminates incorrect modes). A potential solution is to model occluded/missing body parts by extending the state space with a replica for each state, labeled as occluded/missing. This replica would have a low, but non-zero probability. In this fashion, the joint probability of a configuration of body parts including such a state would also be non-zero. This is a topic of our current research.

4 Application: Action Recognition on the Weizmann Dataset

Determining human pose is often a first step to action recognition. For example [18] classifies frames by their pose as a pre-filter to spatio-temporal action detection, and [27] specifies actions by pose queries.

In this section we apply the extracted pose representation to the task of action recognition on the Weizmann dataset [5], which includes nine actions performed by nine different subjects who are fully visible in all frames (including their legs). Following

Table 1. Percentage of correctly estimated body parts by various versions of our method

Method	Performance
Ramanan NIPS 2006 [23]	9.6%
+ detection	41.2%
+ foreground highlighting	57.9%
+ appearance transfer	59.4%
+ repulsive model	**62.6%**
+ complete spatio-temporal model	61.7%

Fig. 12. Pose estimation on the Weizmann dataset [5]. Our methods performs well also on full bodies, and handles a variety of poses (left: a jumping-jack; right: walking).

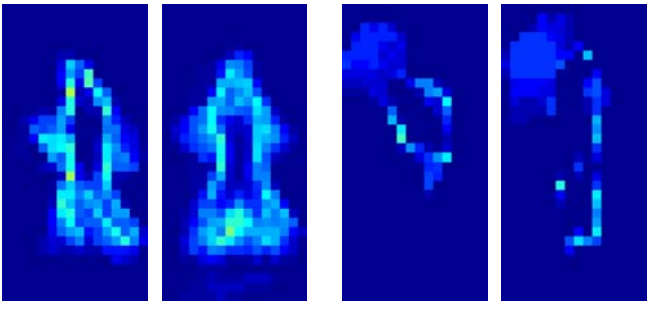


Fig. 13. Action descriptor. Accumulated motion differences for two subjects walking (left) and waving with one hand (right). For illustration, the difference images in this figure are computed from all body parts. Our descriptor instead, is a concatenation of difference images for each body part groups, which provides more discriminative power.

the standard leave-one-out evaluation protocol [5,13,21], we train on eight subjects and test on the remaining one. The experiment is then repeated by changing the test subject and recognition rates are averaged.

Here, we replace the upper-body detector by the standard HOG based pedestrian detector of Dalal and Triggs [7], and employ a full-body pictorial structure including also upper and lower legs (figure 12). Our action descriptor is inspired by motion history images [8] and is obtained as follows. First, for each frame we derive a soft-segmentation from the posteriors of the part positions $p(l_i|I)$ delivered by our pose estimator. Next,

we subtract these soft-segmentations between pairs of subsequent frames, and accumulate the differences over the whole sequence. The resulting accumulated difference image is then subsampled to a 16x32 grid (figure 13). The final descriptor is obtained by computing a separate difference image for each of the four body part groups (torso, arms, legs, and head) and concatenating them. The descriptor is informative because it encodes how much motion each body part group performs, and at which position relative to the coordinate frame of the detection window. It is robust because differences are accumulated over many frames, limiting the impact of incorrect pose estimates in a few frames. For each action, we train a one-vs-all linear SVM on this descriptor, and use them to classify the sequences of the test subjects.

Although previous works using background subtraction achieve perfect results on this dataset [5,13], the only work we are aware of tackling the task without *any* static background assumption[2] only obtains 73% recognition rate [21]. While operating in the same conditions, our method achieves the significantly higher rate of 88%. These results demonstrate the suitability of our technique to full body pose estimation.

5 Appraisal and Future Work

We have demonstrated automated upper body pose estimation on extremely challenging video material – the principal objective of this work.

The numerous works defining action descriptors based on body outlines [5,13] could benefit from our technique, as it provides outlines without resorting to traditional background segmentation, requiring a known and static background.

Of course, further improvements are possible. For example the body part segmentations could be improved by a further application of GrabCut initialized from the current segmentations. Another possible extention is to explicitly model dependencies between multiple people, e.g. to prevent the same body part from being assigned to different people.

The upper body detector and tracking software is available at `www.robots.ox.ac.uk/~vgg/research/pose_estimation/index.html`

Acknowledgements. We thank Pawan Kumar for helpful discussions on probabilistic models, Varun Gulshan for the extended GrabCut, and Deva Ramanan for the image parsing code [23]. We are grateful for support from EU Project CLASS, the Swiss National Science Foundation, the Royal Academy of Engineering, Microsoft, and the Spanish Ministry of Education and Science (grant FPU AP2003-2405, project TIN2005-01665).

References

1. Agarwal, A., Triggs, B.: 3d human pose from silhouettes by relevance vector regression. In: CVPR (2004)
2. Agarwal, A., Triggs, B.: Tracking articulated motion using a mixture of autoregressive models. In: Pajdla, T., Matas, J(G.) (eds.) ECCV 2004. LNCS, vol. 3023, pp. 54–65. Springer, Heidelberg (2004)

[2] The recent work of [32] proceeds without background subtraction at test time, but uses it for training.

3. Andriluka, M., Roth, S., Schiele, B.: People-tracking-by-detection and people-detection-by-tracking. In: CVPR (2008)
4. Bishop, C.: Pattern recognition and machine learning. Springer, Heidelberg (2006)
5. Blank, M., Gorelick, L., Shechtman, E., Irani, M., Basri, R.: Actions as space-time shapes. In: ICCV (2005)
6. Bray, M., Kohli, P., Torr, P.: Posecut: Simultaneous segmentation and 3d pose estimation of humans using dynamic graph-cuts. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006, Part II. LNCS, vol. 3952, pp. 642–655. Springer, Heidelberg (2006)
7. Dalal, N., Triggs, B.: Histogram of Oriented Gradients for Human Detection. In: CVPR, vol. 2, pp. 886–893 (2005)
8. Davis, J., Bobick, A.: The representation and recognition of action using temporal templates. In: CVPR (1997)
9. Felzenszwalb, P., Huttenlocher, D.: Pictorial structures for object recognition. IJCV 61(1) (2005)
10. Ferrari, V., Marin-Jimenez, M., Zisserman, A.: Progressive search space reduction for human pose estimation. In: CVPR (June 2008)
11. Ferrari, V., Tuytelaars, T., Van Gool, L.: Real-time affine region tracking and coplanar grouping. In: CVPR (2001)
12. Gammeter, S., Ess, A., Jaeggli, T., Schindler, K., Van Gool, L.: Articulated multi-body tracking under egomotion. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part II. LNCS, vol. 5303, pp. 816–830. Springer, Heidelberg (2008)
13. Ikizler, N., Duygulu, P.: Human action recognition using distribution of oriented rectangular patches. In: ICCV workshop on Human Motion Understanding (2007)
14. Jojic, N., Winn, J., Zitnick, L.: Escaping local minima through hierarchical model selection: Automatic object discovery, segmentation, and tracking in video. In: CVPR (2006)
15. Kumar, M.P., Torr, P.H.S., Zisserman, A.: Learning layered pictorial structures from video. In: ICVGIP, pp. 148–153 (2004)
16. Kumar, M.P., Torr, P.H.S., Zisserman, A.: Learning layered motion segmentations of video. In: ICCV (2005)
17. Laptev, I.: Improvements of object detection using boosted histograms. In: BMVC (2006)
18. Laptev, I., Perez, P.: Retrieving actions in movies. In: ICCV (2007)
19. Lin, Z., Davis, L., Doermann, D., DeMenthon, D.: An interactive approach to pose-assisted and appearance-based segmentation of humans. In: ICCV workshop on Interactive Computer Vision (2007)
20. Mori, G., Ren, X., Efros, A., Malik, J.: Recovering human body configurations: Combining segmentation and recognition. In: CVPR (2004)
21. Niebles, J., Fei-Fei, L.: A hierarchical model model of shape and appearance for human action classification. In: CVPR (2007)
22. Ozuysal, M., Lepetit, V., Fleuret, F., Fua, P.: Feature harvesting for tracking-by-detection. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006. LNCS, vol. 3953, pp. 592–605. Springer, Heidelberg (2006)
23. Ramanan, D.: Learning to parse images of articulated bodies. In: NIPS (2006)
24. Ramanan, D., Forsyth, D.A., Zisserman, A.: Strike a pose: Tracking people by finding stylized poses. In: CVPR, vol. 1, pp. 271–278 (2005)
25. Rother, C., Kolmogorov, V., Blake, A.: Grabcut: interactive foreground extraction using iterated graph cuts 23(3), 309–314 (2004)
26. Schroff, F., Criminisi, A., Zisserman, A.: Single-histogram class models for image segmentation. In: Kalra, P.K., Peleg, S. (eds.) ICVGIP 2006. LNCS, vol. 4338, pp. 82–93. Springer, Heidelberg (2006)
27. Shechtman, E., Irani, M.: Matching local self-similarities across images and videos. In: CVPR (2007)

28. Sigal, L., Bhatia, S., Roth., S., Black, M., Isard, M.: Tracking loose-limbed people. In: CVPR (2004)
29. Sigal, L., Black, M.J.: Measure locally, reason globally: Occlusion-sensitive articulated pose estimation. In: CVPR, vol. 2, pp. 2041–2048 (2006)
30. Sivic, J., Everingham, M., Zisserman, A.: Person spotting: video shot retrieval for face sets. In: Leow, W.-K., Lew, M., Chua, T.-S., Ma, W.-Y., Chaisorn, L., Bakker, E.M. (eds.) CIVR 2005. LNCS, vol. 3568, pp. 226–236. Springer, Heidelberg (2005)
31. Sminchisescu, C., Triggs, B.: Estimating articulated human motion with covariance scaled sampling. In: IJRR (2003)
32. Thurau, C., Hlavac, V.: Pose primitive based human action recognition in videos or still images. In: CVPR (2008)

Recognition and Synthesis of Human Movements by Parametric HMMs*

Dennis Herzog and Volker Krüger

Computer Vision and Machine Intelligence Lab
Aalborg University Copenhagen
Lautrupvang 15, 2750 Ballerup, Denmark
{deh,vok}@cvmi.aau.dk

Abstract. The representation of human movements for recognition and synthesis is important in many application fields such as: surveillance, human-computer interaction, motion capture, and humanoid robots. Hidden Markov models (HMMs) are a common statistical framework in this context, since they are generative and are able to deal with the intrinsic dynamic variation of movements performed by humans. In this work we argue that many human movements are parametric, i.e., a parametric variation of the movements in dependence of, e.g., a position a person is pointing at. The parameter is part of the semantic of a movement. And while classic HMMs treat them as noise, we will use parametric HMMs (PHMMs) [6,19] to model the parametric variability of human movements explicitly. In this work, we discuss both types of PHMMs, as introduced in [6] and [19] , and we will focus our considerations on the recognition and synthesis of human arm movements. Furthermore, we will show in various experiments the use of PHMMs for the control of a humanoid robot by synthesizing movements for relocating objects at arbitrary positions. In vision-based interaction experiments, PHMM are used for the recognition of pointing movements, where the recognized parameterization conveys to a robot the important information which object to relocate and where to put it. Finally, we evaluate the accuracy of recognition and synthesis for pointing and grasping arm movements and discuss that the precision of the synthesis is within the natural uncertainty of human movements.

1 Introduction

One of the major problems in action and movement recognition is to recognize actions that are of the same type but can have very different appearances depending on the situation they appear in. In addition, for some actions these differences are of major importance in order to convey their meaning. Consider for example the movement of a human pointing at an object, "This object there...", with the finger pointing at a particular object (as in Figure 1). Clearly, for such an action, the action itself needs to be recognized but also the point

* This work was partially funded by PACO-PLUS (IST-FP6-IP-027657).

D. Cremers et al. (Eds.): Visual Motion Analysis, LNCS 5604, pp. 148–168, 2009.

in 3D space at which the human is pointing. Only together do these two pieces of information convey the full semantics of the movement. In this work, we are concerned with two major problems: (1) the recognition of parametric actions, i.e., which action it is and which parameterization it has, and (2) the synthesis of parametric actions. The first problem is mainly a vision problem. The second problem is mainly a robotics problem where a robot needs to perform an action that is scene dependent, i.e., the parameters of the action are given by the scene state. Both problems need to be considered in imitation learning, where one is interested in teaching robots through simple demonstrations [1,2,17].

Fig. 1. *Capturing Session for Our Dataset.* The person wears a black suit with model markers (tiny balls) that are easy to detect for the cameras. For capturing, a model is used, see Figure 2.

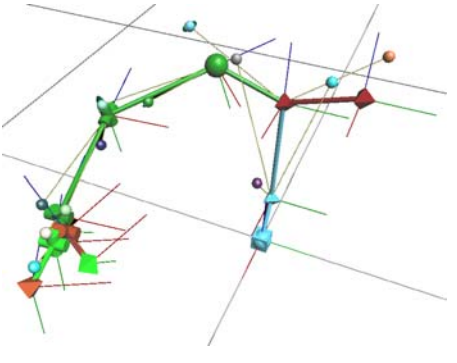


Fig. 2. *Capture Model.* For motion capture, the markers of the model (tiny balls) are aligned to captured markers, see Figure 1.

To approach the recognition, synthesis and training of movements in a single framework, Hidden Markov Models (HMMs) are useful: An HMM is a generative statistical model, which means that it is possible to train the model based on a small set of demonstrated training movements and then to generate (synthesize)

a novel movement using the trained HMM. It captures the variance of the movements, and it is applicable for recognition, as one can efficiently calculate conditional probabilities of observed movements, given an HMM. However, classical HMMs are limited if we want to model a *class* of movements, e.g., pointing movements, where the movements are the same except for some parameterization: a) the classical HMM will learn the average of the demonstrated movements with large covariances in order to model the variability in the data and b) it will consider the semantically important parameterization as noise (see Figure 8, left). To model the systematic and parametric variability of the data rather than treating it as noise, parametric hidden Markov models were introduced [19,6]. In [19], Wilson and Bobick extend the framework of HMMs by extending the Baum-Welch learning algorithm. In [6], an exemplar-based PHMM was introduced that solves the parameterization through interpolation of classic HMMs.

In this chapter, we aim at a systematic evaluation of the two parametric HMMs for the synthesis and recognition of full arm movements. We perform experiments for online recognition and robot control. The basis for a systematic evaluation are movements such as pointing at objects and grasping at objects. The movements are recorded in a table-top scenario, where a person sits in front of a table (see Figure 1). They are performed to 7×5 object positions, which define the two dimensional parameterization of the movement and the PHMMs. For each of the positions, we evaluate the performance of synthesis (error of the synthesized 3D point trajectories of shoulder, elbow, wrist, etc.) and recognition (error of estimated positions at table-top) for both types of PHMMs. The performance concerning the recognition of the movement type is evaluated for arbitrary object locations. Noise experiments and the online recognition based on noisily estimated arm-poses (of a person advising a robot) through a 3D tracker show the robustness of the PHMM framework. The applicability of the PHMMs for synthesis is demonstrated in a robot demo, where a humanoid robot needs to grasp objects at arbitrary locations. Our main contribution are:

- evaluating both PHMMs also for synthesis as such
- evaluating both PHMMs
  - on full arm movements (21 dimensions, seven 3D points)
  - for recognition and synthesis
  - systematically for the movement parameterization
- real-life applications enabled through PHMMs
  - robot control
  - vision-based recognition for interaction

Overview

In the following section, we give a short overview of the related work. In Section 2 we provide an introduction to hidden Markov models, and the parametric HMMs (Section 2.2), and a detailed description of exemplar-based PHMM (Section 2.3). In Section 3, the application of PHMMs for recognition and synthesis is described (Section 3.1) and evaluated for full arm movements (Section 3.2). In addition,

two online applications of PHMMs (online recognition and control of a humanoid robot) are given (Section 3.3). Conclusions in Section 4 complete the chapter.

Related Work

Most approaches for movement representation that are of interest in our problem context are trajectory based: trajectories for the training, e.g., sequences of human body poses, are encoded in a suitable manner. Newly incoming trajectories are then compared with the previously trained ones. A recent review can be found in [13]. – Some of the most common approaches to represent movement trajectories use hidden Markov models (HMMs) [9,15], since HMMs are an advantageous statistical framework, as mentioned in Section 1. One approach to overcome that HMM model intra-class variability rather as noise is to use set of hidden Markov models (HMMs) in a mixture-of-experts approach, as first proposed in [10]. In order to deal with a large parameter space one ends up with a lot of experts, and training becomes un-sustainable. The parametric extension [19] of the classic HMM overcomes this problem. A more recent approach was presented by [1]. In this work, the parametric variation is carried out in spline space (key-points) for trajectories of a robot's end-effector. But this approach does not seem suitable for recognizing arm movements. In addition to HMMs, there are also other movement representations that are interesting in our context, e.g., [12,18]. However, these approaches share the same problems as the HMM based approaches.

2 Parametric Hidden Markov Models

A parametric HMM (PHMM) is an extension of the hidden Markov model (HMM) [9,15] by additional latent style parameters $(\phi_i) = \boldsymbol{\phi}$ which model the systematic variation within a class of modeled sequences. That is, in our context, the variation within arm movements of specific type. Consider, for example, the movement *pointing at an object* (see Figure 1), where an actor sits in front of a table and points in a similar way at objects that are placed at different table-top positions. The movements vary then according to the object position, using a classical HMM, we would model the differences between the "pointing at an object" movements as noise, i.e., the classical HMM would average out the localization a human is pointing at. This is a problem because the pointed-at localization is an important piece of information. A parametric HMMs (parameterized by the pointed at position $\boldsymbol{\phi}$) is able to preserve this information. The parametric extension of HMMs for the recognition of movements has been introduced in [19]. After a short introduction to HMMs, we will describe two approaches to the parametric extension of the HMMs: the approach [19], and an exemplar-based approach [6] that uses model interpolation.

2.1 Hidden Markov Models

A hidden Markov model is a finite state machine extended in a probabilistic way. Generally, it is denoted by a triple $\boldsymbol{\lambda} = (\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{\pi})$. The outputs of each state i are

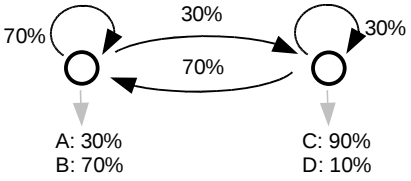


Fig. 3. *Two-State HMM.* The figure shows an HMM with discrete output symbols. Both, the transition arcs and the output symbols are labeled with probabilities. Possible output sequences are for example: ACBBC, or ACBBAD. The sequence DDDDD is a rather unlikely output sequence due to the probabilities of the self transition and the output symbol D of the right state.

described in case of a discrete HMMs by probability mass functions $b_i(\cdot)$, where $\boldsymbol{B} = (b_i)$. For continuous HMMs, the b_i are, e.g., Gaussian density functions $b_i(\boldsymbol{x})$. Figure 3 shows an HMM with discrete output symbols. In the rest of this work we consider only continuous HMMs (see Figure 4), where each observation density function b_i is a multivariate Gaussian density $b_i(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$. Let us consider now a continues sequence $\boldsymbol{X} = \boldsymbol{x}_1 \cdots \boldsymbol{x}_n$ observed at discrete time steps t. If we denote the hidden states for each time step by q_t then we have, e.g., for $q_t = i$, that $P(\boldsymbol{x}_t|q_t) = b_i(\boldsymbol{x}_t)$. However, the hidden states of the HMM are generally unknown (hidden). The model $\boldsymbol{\lambda}$ further defines the prior state distribution $P(q_{t=1} = i) =: \pi_i$ and the state transition probabilities $P(q_{t+1} = j|q_t = i) =: a_{ij}$ which are compactly denoted by the probability mass distribution vector $\boldsymbol{\pi} = (\pi_i)$ and the state transition matrix $\boldsymbol{A} = (a_{ij})$, respectively. The HMM model describes processes which follow the first order Markov chain property: $P(q_t|q_{t-1}, q_{t-2}, \ldots) = P(q_t|q_{t-1})$.

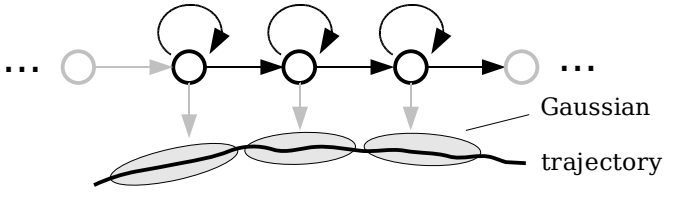


Fig. 4. *Left-Right HMM.* The figure shows a left-right HMM with continuous outputs. A possible output of this model could be a sampled version of the depicted trajectory. Since a hidden sequence has to pass the states in the order from left to right, each state of a model (trained for the depicted trajectory) is basically responsible for a certain part of the trajectory.

In the following we will only deal with those aspects about HMMs which are important in our context. An extensive tutorial to HMMs is given by [15]; a deeper introduction can be found in [9]. There are several important aspects about HMMs that are to be considered. On the one hand there are different types of HMMs concerning the outputs and the state transition structure, on

the other hand efficient algorithms are required, e.g., in order to estimate the parameters of an HMM based on a training set.

One type of HMM is the left-right model [15] which restricts the transition structure of an HMM. In this model, back transitions from a state to a previous state have the likelihood 0 ($a_{ij} = 0, j < i$). We use in the following a specific type of left-right model, where transitions of each state are leading only from the state to the state itself or to the direct successor [4], see Figure 4. If such a model, e.g., with single Gaussian outputs, describes a trajectory as shown in Figure 4 then it becomes strait forward to generate or synthesize a meaningful sequence by simply taking the means of the Gaussians in the order of the states. Such a left-right model can handle temporal variations of the movements as the number of repetitive outputs of a state can differ to allow different progression rates of the time sequences.

Under certain circumstances it is better if one has more control over the time duration or the number of sequential time steps for which a state is responsible for generating outputs. Such an extension is given by semi-observable hidden Markov models (SHMMs) [14], where the number of outputs of a state in sequence is modeled through a probability distribution. However, the algorithms of the SHMM have unsolved issues concerning the implementation [14]. To achieve similar control over the time durations of a state, we replace each state of the HMM by a certain number of states (see Figure 5) which share the same output distribution. This way, one can control the minimal and maximal duration for which a single distribution generates outputs through the state transition structure, see Figure 5.

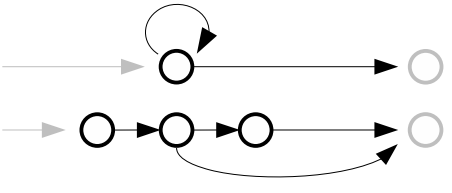


Fig. 5. *Explicit Time Durations.* Each state is replaced by some states (3 states) which share the same output distribution. The depicted transition structure allows each state triple to generate outputs from the shared distribution for at least 2 and maximal 3 states.

For the application of HMMs in general, and also in our case of classification, one is interested in calculating the likelihood $P(\boldsymbol{X}|\boldsymbol{\lambda})$ of an observation $\boldsymbol{X}$ given a model $\boldsymbol{\lambda}$ as this gives a hint whether the sequence belongs to the class represented by the model $\boldsymbol{\lambda}$. On the other hand one likes to estimate the model parameters of an HMM given a training set $\mathcal{X}$ of observations, e.g., of a class of similar movements.

The first problem—known as evaluation problem—is efficiently solved by the forward-backward algorithm which overcomes the enumeration

$$P(\boldsymbol{X}|\boldsymbol{\lambda}) = \sum_{\boldsymbol{Q}} P(\boldsymbol{X}|\boldsymbol{\lambda}, \boldsymbol{Q}) P(\boldsymbol{Q}|\boldsymbol{\lambda})$$

of each possible state sequence $\boldsymbol{Q}$, where the exponential number $\#\boldsymbol{Q}$ of state sequences makes the calculation generally too expensive. The forward-backward algorithm is a dynamic programming approach. The dynamic programming step of the forward procedure uses the recursive formula of the forward variable

$$\alpha_t(i) := P(\boldsymbol{x}_1 \cdots \boldsymbol{x}_t, q_t = i|\boldsymbol{\lambda}) = \left(\sum_j \alpha_{t-1}(j) a_{ij} \right) b_j(\boldsymbol{x}_t),$$

where each state i has to be considered in each iteration $t = 1, \ldots, T$ over the length T of the sequence $\boldsymbol{X}$. Finally, we can evaluate

$$P(\boldsymbol{X}|\boldsymbol{\lambda}) = \sum_{i=1}^{N} \alpha_T(i).$$

Similarly, the value $P(\boldsymbol{X}|\boldsymbol{\lambda})$ can also be evaluated by defining a backward variable $\beta_t(i) := P(\boldsymbol{x}_{t+1} \cdots \boldsymbol{x}_T|q_t = i, \boldsymbol{\lambda})$ instead. The forward/backward procedures are very efficient: $\mathcal{O}(N^2 T)$, where N is the number of states. The forward/backward procedure and its time consumption are described in detail in [15].

The second problem is the estimation of the model parameters $\boldsymbol{\lambda} = (\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{\pi})$. This is generally treated as a maximum likelihood problem. There exist no analytical solution to this problem. The Baum-Welch expectation-maximization algorithm (EM) [15] adapts therefore the model parameters iteratively. Given a training set $\mathcal{X} = \{\boldsymbol{X}^k = \boldsymbol{x}_1^k \cdots \boldsymbol{x}_{T_k}^k\}$, the algorithms is proven to modify the parameters in each EM iteration towards more likely ones. Each EM iteration consists of two steps: the E (expectation) step, and a subsequent M (maximization) step. In the *E step* the joint posterior probabilities $\xi_t^k(i,j) = P(q_t = i, q_{t+1} = j|\boldsymbol{X}^k, \bar{\boldsymbol{\lambda}})$ of being in state i and j at time t and $t+1$, and the marginal distribution $\gamma_t^k(i) = P(q_t = i|\boldsymbol{X}^k, \bar{\boldsymbol{\lambda}})$ are calculated. Here, $\bar{\boldsymbol{\lambda}}$ denotes the model parameters of the previous iteration step. The posterior probability $\gamma_t(i)$ can be interpreted as the responsibility of state i for generating the output $\boldsymbol{x}_t$. Both, $\gamma_t^k(i)$, and $\xi_t^k(i,j)$ can be calculated based on the forward-backward variables that are computed for each sequence $\boldsymbol{X}^k$. See [9] for more details. In the *M step* the complete-data likelihood function $Q(\boldsymbol{\lambda}, \bar{\boldsymbol{\lambda}}) = \sum_{\boldsymbol{Q}} P(\boldsymbol{Q}|\mathcal{X}, \boldsymbol{\lambda}) \log P(\mathcal{X}, \boldsymbol{Q}|\bar{\boldsymbol{\lambda}})$ is maximized. This maximization is achieved, e.g., by using standard constrained optimization techniques, i.e., Lagrange multipliers. In this way one yields the following update formulas for the mean $\boldsymbol{\mu}_i$ and the covariance $\boldsymbol{\Sigma}_i$ of each Gaussian $b_i(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$:

$$\boldsymbol{\mu}_i = \frac{\sum_{t,k} \gamma_t^k(i) \boldsymbol{x}_t^k}{\sum_{t,k} \gamma_t^k(i)}$$

$$\boldsymbol{\Sigma}_i = \frac{\sum_{t,k} \gamma_t^k(i) (\boldsymbol{x}_t^k - \boldsymbol{\mu}_i)(\boldsymbol{x}_t^k - \boldsymbol{\mu}_i)^T}{\sum_{t,k} \gamma_t^k(i)}.$$

Let us consider now a common scenario of recognition or classification within an HMM-based approach. Suppose that for each sequence class k a representative data set $\mathcal{X}^k$ is given. Then, we train for each set an HMM $\boldsymbol{\lambda}^k$ by maximizing the likelihood function $\mathrm{P}(\mathcal{X}^k|\boldsymbol{\lambda}^k)$ with the EM algorithm mentioned above. A specific output sequence $\boldsymbol{X} = \boldsymbol{x}_1 \dots \boldsymbol{x}_T$ can then be classified by identifying the model $\boldsymbol{\lambda}^k$ for class k which yields the highest likelihood $\mathrm{P}(\boldsymbol{X}|\boldsymbol{\lambda}^k)$. Therefore, one efficiently calculates the likelihoods with the forward-backward algorithm mentioned above. — One obvious approach for handling whole classes of parameterized actions for the purpose of parameter recognition is now a mixture-of-experts approach [10] with sampling of the parameter space. The parameter vector would be, e.g., in the case of a pointing movement, the position $\boldsymbol{\phi} = (x, y, z)$ at which the index-finger is pointing at. Clearly, this approach suffers from the great number of HMMs needed to be trained and stored for all possible pointing positions. Furthermore, we are interested in the valuable information about $\boldsymbol{\phi}$. Therefore, we introduce in the following the parameterization of the movements as additional model parameters.

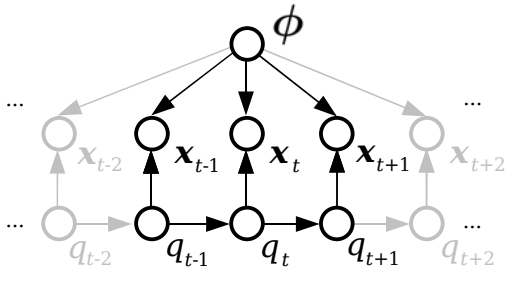


Fig. 6. *Graphical Model for the PHMM in* [19]. The figure shows the dependency on the variable $\boldsymbol{\phi}$ which models the systematic variation of the outputs $\boldsymbol{x}_t$ of an output sequence $\boldsymbol{x}_1\boldsymbol{x}_2 \dots \boldsymbol{x}_t \dots \boldsymbol{x}_T$. The variable q_t denotes the hidden state at time step t.

2.2 Parametric HMMs

The underlying idea of a parametric extention of HMMs (PHMM) is to introduce a new latent variable $\boldsymbol{\phi} = (\phi_i)$ which adapts the basis HMM parameters. If the hidden variable is known and assumed to be constant then the PHMM becomes the classic HMM; thus we tend to denote a PHMM simply by $\boldsymbol{\lambda}^{\boldsymbol{\phi}}$. The latent variable is supposed to describe some systematic variation (see also Section 2, above) within the parametric movement class represented by the PHMM. As a consequence, the number of latent variables ϕ_i has to be chosen with care for a specific problem.

Two approaches to PHMMs have been considered yet. The approach of Wilson and Bobick in [19] considers a variation of the Gaussian means $\boldsymbol{\mu}_i = \boldsymbol{f}_i(\boldsymbol{\phi})$ of the observation densities in dependence of the latent parameter $\boldsymbol{\phi}$ of a movement. Figure 6 shows this dependency in a graphical model.

The exemplar-based approach in [6] uses some local exemplar HMMs $\boldsymbol{\lambda}^{\boldsymbol{\phi}_i}$ for specific movement parameterizations $\boldsymbol{\phi}_i$. The generation of an HMM $\boldsymbol{\lambda}^{\boldsymbol{\phi}}$ for specific parameters is carried out by component-wise linear interpolation of the local HMMs. Both approaches handle the training of a PHMM in a supervised manner. For each sequence of the training set the parameter $\boldsymbol{\phi}$ (e.g., the pointed at position of a pointing movement) is required.

Here, we want to give the complete explanation of the exemplar-based framework of PHMMs [6] which mainly consists of a synchronization procedure for the local HMMs. This procedure ensures meaningful interpolation what is indispensable at least for the synthesis of movements.

Regarding the approach of Wilson and Bobick we refer to [19]. However, we like to summarize some interesting points of that work: they considered two cases for the dependency of the Gaussian mean $\boldsymbol{\mu}_i = f_i(\boldsymbol{\phi})$. In the linear case, each function $\boldsymbol{f}_i(\boldsymbol{\phi})$ is of the form $\boldsymbol{\mu}_i = \bar{\boldsymbol{\mu}}_i + \boldsymbol{W}_i\boldsymbol{\phi}$, where the matrices $\boldsymbol{W}_i$ describe the linear variation of the means of each state i. In a more general approach, they use for each state i a neural network that is trained to approximate even a nonlinear dependency $\boldsymbol{f}_i(\boldsymbol{\phi})$. What is remarkable about their approach is the concise way of the training in the linear case. Here, they apply an EM method. Like in the original Baum-Welch EM algorithm, the maximization is given as an analytical solution, the update formula for $\bar{\boldsymbol{\mu}}_i$ is replaced by an update formula for $\bar{\boldsymbol{\mu}}_i$ and $\boldsymbol{W}_i$:

$$[\boldsymbol{W}_i \mid \bar{\boldsymbol{\mu}}_i] = \left[\sum_{k,t} \gamma_t^k(i)\boldsymbol{x}_t^k\boldsymbol{\Omega}_k^T\right]\left[\sum_{k,t} \gamma_t^k(i)\boldsymbol{x}_t^k\boldsymbol{\Omega}_k\boldsymbol{\Omega}_k^T\right]^{-1}, \tag{1.1}$$

where $\boldsymbol{\Omega}_k = [(\boldsymbol{\phi}_k)^T \mid 1]^T$, and $\boldsymbol{\phi}_k$ is the parameterization of a training sequence $\boldsymbol{X}^k$. In the nonlinear case the maximization step of the EM algorithm itself contains an iteration for the estimation of the neural network parameters, called a generalized EM method (GEM) [19]. Irrespectively of the approach, the authors are aiming at recognition of movements, and consider no synthesis. However, their PHMM seems to be suited for the synthesis of movements as well.

The exemplar-based approach to PHMMs in [6] has the advantage that all model parameters are adapted depending on the parameterization $\boldsymbol{\phi}$ and this somehow *for free* through the interpolation of the exemplar HMMs. However, the approach is nontrivial. The HMMs need to be setup in a state-wise aligned sense thus that the interpolation becomes meaningful. The setup is described in the following Section 2.3, and can be regarded as a method for training of PHMMs. One advantage of this setup is that it is rather simple to handle and should perform better in the case that the variation of the observation covariances matters. A drawback is that one has to chose specific parameterizations for which the exemplar HMMs are to be trained.

2.3 Exemplar-Based PHMM Framework

As mentioned above, the basic idea of our PHMM approach to the handling of whole classes of parameterized sequences is to train some local exemplar HMMs

$\boldsymbol{\lambda}^{\phi_i}$. Thus each HMM represents sequences of certain parameterization $\boldsymbol{\phi}_i$. A new HMM $\boldsymbol{\lambda}^{\boldsymbol{\phi}}$ for a novel parameter $\boldsymbol{\phi} = (\phi_i)$ is deduced by component-wise linear interpolation of the nearby exemplar models. For brevity, we consider now the case of a single parameter $\phi = u$ and two exemplar models $\boldsymbol{\lambda}^{u=0}$ and $\boldsymbol{\lambda}^{u=1}$, see Figure 7. The new transition probabilities a_{ij}^u and each Gaussian $\mathcal{N}_i^u(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_i^u, \boldsymbol{\Sigma}_i^u)$ of a model $\boldsymbol{\lambda}^u$ for $u \in [0,1]$ is then generated by interpolation

$$\begin{aligned} \boldsymbol{\mu}_i^u &= (1-u)\boldsymbol{\mu}_i^0 + u\boldsymbol{\mu}_i^1, \\ \boldsymbol{\Sigma}_i^u &= (1-u)\boldsymbol{\Sigma}_i^0 + u\boldsymbol{\Sigma}_i^1, \\ a_{ij}^u &= (1-u)a_{ij}^0 + ua_{ij}^1, \end{aligned} \tag{1.2}$$

where the normalization of the probabilities $\sum_j a_{ij} = 1$ is preserved for each state i. In order to yield a meaningful model through the state-wise interpolation, one has to ensure that corresponding states of the exemplar HMMs model the same semantic part of the trajectory (see Figure 7). The following consideration leads to this requirement: consider the n-th state of the two HMMs $\boldsymbol{\lambda}^{u=0}$ and $\boldsymbol{\lambda}^{u=1}$, where one of the two states models a part of a forward motion of a hand while the other models a part of a backward motion. Clearly, interpolation of such two states does not make sense. Hence, the state-wise alignment where states match semantically is vital. This is described in the Section 2.3. The expansion to the multi-variate case of the parameterization $\boldsymbol{\phi}$ is straightforward, e. g., by using bilinear ($\boldsymbol{\phi} = (u, v)$) or trilinear interpolation.

Synchronization of HMM States. Here we discuss, how to assure that the corresponding states of the exemplar HMMs model the same semantic parts of movements. This is necessary to achieve meaningful state-wise interpolation, as mentioned above. The required alignment of the states to the sequences is similar to the alignment of two sequences by dynamic time warping (DTW) [11]. However, this technique can not be applied here, since we need to align the states of the exemplar HMMs and this in the presence of many training sequences.

The underlying idea of the synchronization technique in [6] is to set up local exemplar HMMs $\boldsymbol{\lambda}^{\boldsymbol{\phi}}$ by using the invariance of HMMs to temporal variations. We proceed in two steps. — In *step one*, a global HMM $\boldsymbol{\lambda}$ is trained based on the whole training set $\mathcal{X}$ which contains all available sequences for a single movement type but this for arbitrary parameterizations $\boldsymbol{\phi}$. For the training of the global HMM, the EM algorithm (see Section 2.1) is used. A trained global HMM is sketched in Figure 7 through the light gray Gaussians. The situation that movements of different parameterizations are covered in such a symmetrical way by the Gaussians of the global HMM as in Figure 7 can be ensured by:

- *The use of a left-right HMMs,* as shown in Figure 4. In this left-right model, each hidden state sequence has to pass the states in a given order. Thus, each state is responsible for some part of any sequence. This way, a state can not be used just for one movement of a specific parameterization and skipped for the remaining movements.

– *Constraining the "time-warp capabilities" of the HMM.* This is done in order to prevent an asymmetric alignment of the states of the exemplar HMMs. This is for example the case, when some states generate very few output for movements of one parameterization and many outputs for movements of other parameterizations. To assure that all states have approximately the same output duration, explicit time durations are added to the states of the HMMs in the way as discussed in Section 2.1. For the model training during the experiments, we ensure that each Gaussian can generate outputs for maximal six time steps and at least for four. The idea of using a constrained time-warping, has already been used for dynamic time warping (DTW) [11]. A proper and an improper state alignment is shown for DTW in [11] (Figure 2 B and C).

Based on these settings one yields a symmetric global HMM as sketched in Figure 7, and shown in Figure 8 (left) for real data.

In *step two*, one considers the partial training set $\mathcal{X}^{\boldsymbol{\phi}}$ of a specific parameterization $\boldsymbol{\phi}$ for which we finally want to setup a local exemplar HMM $\boldsymbol{\lambda}^{\boldsymbol{\phi}}$. Again, we use the EM algorithm. But now, we use the parameters of the global HMM $\boldsymbol{\lambda}$ as initial values for the EM. In addition, to preserve the state alignment for the local HMM as it is given by the global HMM, we fix the means after the first EM step. (In the context of Figure 7 this results in a shrinking of the dark gray ellipsoids to the light gray ones.) In the following, we will exemplify that this adapted EM procedure gives a proper state alignment of the local HMMs: In the first E step of the EM algorithm, the posterior probabilities $\gamma_t^k(i) = P(q_t = i|\boldsymbol{X}^k, \boldsymbol{\lambda})$ of being in state i at time step t are computed for all sequences $\boldsymbol{X}^k = \boldsymbol{x}_1^k \dots \boldsymbol{x}_T^k$ in $\mathcal{X}^{\boldsymbol{\phi}}$. But this is done based on the current parameter values, which are in this iteration the values of the global HMM. Thus, $\gamma_t^k(i)$ is the responsibility of state i for generating $\boldsymbol{x}_t^k$, as given by the global HMM $\boldsymbol{\lambda}$. In the following M step of the EM algorithm, each $\boldsymbol{\mu}_i$ of a Gaussian of state i is re-estimated as a $\gamma_t^k(i)$-weighted mean:

$$\boldsymbol{\mu}_i = \frac{\sum_{t,k} \gamma_t^k(i) \boldsymbol{x}_t^k}{\sum_{t,k} \gamma_t^k(i)} \tag{1.3}$$

Now, consider Figure 7. The responsibilities $\gamma_t^0(i)$ of the upper sequence $\boldsymbol{x}_1^0 \boldsymbol{x}_2^0 \cdots$ are large for the time steps $t = 1, 2$ and the state $i = 1$ but are small for all other states $i > 1$. This is caused by the position of the Gaussian $\mathcal{N}_1$ of the global HMM $\boldsymbol{\lambda}$. As consequence, the mean $\boldsymbol{\mu}_1^0$ of $\mathcal{N}_1^0$ of the local exemplar HMM $\boldsymbol{\lambda}^0$, as calculated by Equation (1.3), is (as requested) between $\boldsymbol{x}_1^0$ and $\boldsymbol{x}_2^0$. Similarly, $\boldsymbol{\mu}_1^1$ of the exemplar HMM $\boldsymbol{\lambda}^1$ computed based on the lower sequence $\boldsymbol{x}_1^1 \boldsymbol{x}_2^1 \dots$ would be between $\boldsymbol{x}_1^1$ and $\boldsymbol{x}_2^1$. Hence, a symmetrical alignment of $\boldsymbol{\mu}_1^0$ and $\boldsymbol{\mu}_1^1$ of the local exemplar HMMs $\boldsymbol{\lambda}^0$ and $\boldsymbol{\lambda}^1$ is inherited from the global model $\boldsymbol{\lambda}$.

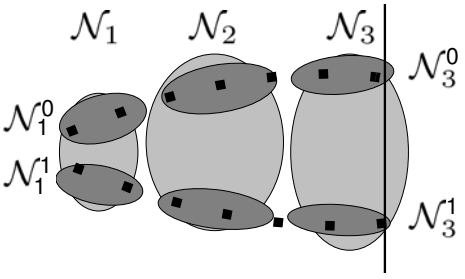


Fig. 7. *State-Alignment for Local Exemplar HMMs.* The upper three dark ellipsoids are depicting Gaussians $\mathcal{N}_1^0, \ldots, \mathcal{N}_3^0$ for the states $i = 1, 2, 3$ of a local exemplar HMM $\boldsymbol{\lambda}^0$, whereas the lower three dark ellipsoids belong to an exemplar model $\boldsymbol{\lambda}^1$. The dots within the dark ellipsoids are sketching training sequences with parameterization $u = 0$, and $u = 1$ depending on their target points on the vertical line. In addition, the Gaussians $\mathcal{N}_i$ of a global model $\boldsymbol{\lambda}$ are indicated in light gray. This global $\boldsymbol{\lambda}$ is a model for all training sequences with $u \in [0, 1]$, and it is used to align the local HMMs. Figure 8 shows such an alignment for some recorded movements.

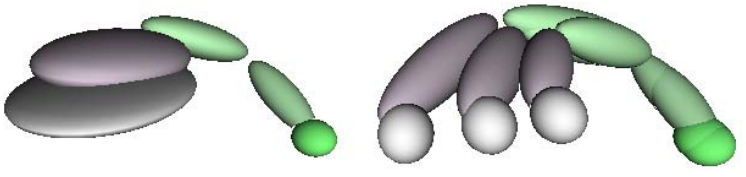


Fig. 8. *State-Alignment of Exemplar HMMs for Recored Data.* Similar to Figure 7, the state outputs of a global HMM (left) and three local exemplar HMMs (right) of an exemplar-based PHMM are shown. The exemplar HMMs (right) are trained on recorded 3D trajectories of a finger tip of a person performing pointing movements for three positions at table-top (see also Figure 1). The index finger starts always at the same point which is modeled by the Gaussian sketched by the green ball. The specific pointed at positions are modeled by the light gray balls of the three local exemplar HMMs. The global HMM (left) (used to setup the local exemplar HMMs) has a disc like Gaussian that models all pointed at positions which cover the plane of the table-top. — *Note, the HMM (left) models the systematic variation of the movements simply as noise.*

3 Recognition and Synthesis

In this section we consider PHMM-based recognition and synthesis of movements. We focus our considerations on pointing (see Figure 1) and grasping movements, which are probably two of the most important movements in human-robot interaction scenarios. After a short explanation in Section 3.1 about the application of the PHMMs for recognition and synthesis, we evaluate in Section 3.2 the recognition and synthesis performance of exemplar-based PHMM [6], and compare it to the linear PHMM [19] which models the linear variation of

the means of the Gaussian outputs. In addition to the off-line experiments, we present in Section 3.3 an application for online recognition and a robot motor control demo.

3.1 PHMM-Based Approach to Synthesis and Recognition

In the following we describe how the recognition and synthesis of parametric movements is performed within an exemplar-based approach to PHMM. For the linear PHMM [19], the procedure is basically the same, an advantage is then that one does not need consider the exemplar HMMs. This simplifies the following considerations for the linear PHMM.

Synthesis. The synthesis of a movement of a specific parameterization $\boldsymbol{\phi}$ from an exemplar-based PHMM $\boldsymbol{\lambda}^{\boldsymbol{\phi}}$ is performed as follows: At first, a set of local HMMs $\boldsymbol{\lambda}^{\boldsymbol{\phi}_i}$ with parameterizations nearest to $\boldsymbol{\phi}$ is selected under the constraint that $\boldsymbol{\phi}$ and thus the HMM $\boldsymbol{\lambda}^{\boldsymbol{\phi}_i}$ can be accurately interpolated. To be more specific, in the case of our pointing movement (see Figure 1) and a pointed at position $\boldsymbol{\phi} = (u, v)$ at table-top, we select at first four HMMs $\boldsymbol{\lambda}^{(u,v)_i}$ for pointed at positions $(u, v)^i$ such that (u, v) is in the rectangle defined by the corners $(u, v)^i$. Then, the HMM $\boldsymbol{\lambda}^{(u,v)_i}$ are interpolated, as described in Section 2.3. This gives a classic HMM $\boldsymbol{\lambda}^{(u,v)=\text{const}}$ for movements pointing at the position (u, v). The synthesis itself is then achieved by simply taking the sequence of the Gaussians' means, i.e., $\boldsymbol{\mu}_1^{uv} \dots \boldsymbol{\mu}_N^{uv}$. This sequence is meaningful because of the choice of the left-right Markov model that we use for the PHMM. We use then linear spline interpolation to expand the sequence to a continuous function $\boldsymbol{f}(t)$. If necessary, the expansion can be performed under consideration of the timing which is encoded in the transition matrix $\boldsymbol{A}^{(u,v)}$ of $\boldsymbol{\lambda}^{(u,v)}$. We neglect this as it is not important for our experiments.

Recognition. The recognition of an observed movement within an exemplar-based approach to PHMM is very similar compared to the nonparametric case of classification based on general HMMs. Here, we classify the type of the movement, but in addition, we identify also its parameterization. Suppose we have for each parametric movement class k a parametric HMM $\boldsymbol{\lambda}_k^{\boldsymbol{\phi}}$ that represents the class. Now, let us assume that we need to classify a sequence $\boldsymbol{X}$. We start by estimating the most likely parameter $\boldsymbol{\phi}_k$ for each possible action class k. This involves maximizing the likelihood functions:

$$\boldsymbol{\phi}_k = \underset{\boldsymbol{\phi} \in [0-\varepsilon, 1+\varepsilon]^d}{\arg\max} P(\boldsymbol{X} | \boldsymbol{\lambda}_k^{\boldsymbol{\phi}}), \tag{1.4}$$

where each parameter is assumed to be in a normalized interval $[0, 1]$ and ε defines the range for which one allows the extrapolation of model parameters, e.g., $\varepsilon = 0.1$. The class identity for the movement $\boldsymbol{X}$ is now given by that class k for which the likelihood $P(\boldsymbol{X}|\boldsymbol{\lambda}_k^{\boldsymbol{\phi}})$ is the highest. In addition, the associated parameter $\boldsymbol{\phi}_k$ gives the most likely parameterization for $\boldsymbol{X}$. In the experiments we maximize the log likelihood functions $l_k(u, v) = \log P(\boldsymbol{X}|\boldsymbol{\lambda}_k^{(u,v)})$ w.r.t. the

position (u, v) at table-top, see Figure 9. This is done using a gradient descent method and numerical derivatives of $l_k(u, v)$. In our evaluation of the movement representation, the parameterization is given by the 2-D coordinates of a plane at table-top. In the experiment we have up to 3×3 local exemplar HMMs for each movement, where the associated positions at table-top form a regular raster. In this case the first guess of the most likely parameter $(u, v)_k$ is always estimated based on the local exemplar HMMs for the outermost raster positions. Then, the estimate is refined based on the four closest exemplar HMMs to the first guess.

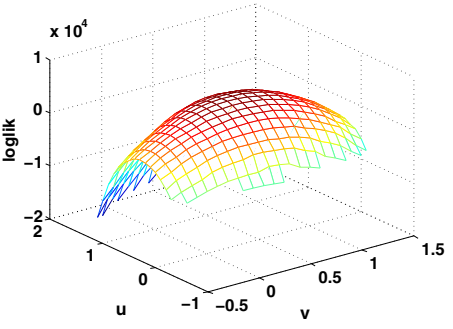


Fig. 9. *Plot of Log Likelihood Function* $l(u, v) = \log P(\boldsymbol{X}|\boldsymbol{\lambda}^{(u,v)})$. The movement $\boldsymbol{X}$ is a pointing movement which leads to the middle of the table top, where the parameterization is $(u, v) = (0.5, 0.5)$.

3.2 Off-Line Evaluation of the PHMM-Based Approach

For evaluation, we use motion data acquired with an eight camera visual marker motion capture system of *Vicon*. The setup of the capture session for data acquisition takes place, as follows: The person or actor sits in front of a table (see Figure 1). The pointing and grasping movements are performed in such a way that they are starting and ending in the same base position (arm hanging down). The recognition and synthesis experiments are based on the trajectories of seven 3D points located at different segments of the body. The seven data points are: sternum; shoulder, and elbow of the right arm; the index finger, its knuckle, and the thumb of the right hand. The motion capturing is here performed by aligning a model of the right arm (Figure 2) to the captured marker locations (see Figure 1). The pointing movements are movements such as pointing at a specific object in order to communicate a specific object, e.g., "This object here is...". The grasping movements reach towards a particular object in order to grasp it. For both movements, the parameterization is given through the associated position (u, v) at table-top.

We evaluate the model performance systematically for different movement parameterizations. Therefore, we introduced a regular 7×5 raster that covers the table-top region of 80cm $\times$ 30cm. For evaluation, we recored 4 test repetitions for each raster position and movement type (pointing, grasping), all in all 240

repetitions for testing. For the training of the PHMMs, we recorded a different set of movements for a 3×3 raster with 10 repetitions for each position and action type.

The training and setup of the exemplar-based PHMMs [6] for grasping and pointing is done as described in Section 2.3. For both PHMM (the exemplar-based [6], and the linear PHMM [19]) we trained PHMMs with 20 states (or outputs), where the hidden state sequences are allowed to stay between 4 and 6 steps in each state. The training sequences are rescaled to 100 samples. We train the PHMMs based either on the movement data of the full 3×3 raster (9 exemplar HMMs) or on a smaller 2×2 raster, where we use only the outermost positions or movements of the 3×3 raster. In the following, we will refer to the trained PHMMs simply by terms as, e.g., "the exemplar-based 3×3 PHMM for grasping", or "the linear 2×2 PHMM" etc, in order to indicate the different train sets and types of PHMMs.

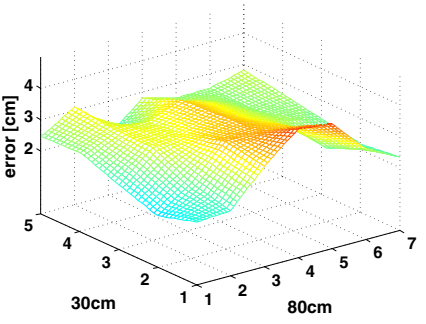
Evaluation of Synthesis. The performance of synthesis is evaluated by plotting the errors of the synthesized movements for each position of the 5×7 test raster, see Figure 10. The error for each synthesized movement is calculated as a distance measure between the synthesis $\boldsymbol{f}(t)$ and a statistical ground truth average movement $\overline{\boldsymbol{f}}(t)$ of the test movements. We perform the averaging by training an HMM of heigh accuracy (80 states) on the four movements. Then we synthesize the averaged movement $\overline{\boldsymbol{f}}(t)$ from the HMM. Both movements $\boldsymbol{f}(t)$ and $\overline{\boldsymbol{f}}(t)$ are stacked 3D trajectories $\boldsymbol{f}(t) = (\boldsymbol{f}_i(i))_{i=1}^{7}$, where the $\boldsymbol{f}_i(t),\ i = 1, 2, \ldots$ are elbow, wrist, etc. The error ε of the synthesized movement $\boldsymbol{f}(t)$ is calculated as the route-mean-square error between the synthesis and the ground truth $\overline{\boldsymbol{f}}(t)$:

$$\varepsilon = \sqrt{\frac{1}{7}\sum_{i=1}^{7} \int (\boldsymbol{f}_i(\alpha(t)) - \overline{\boldsymbol{f}}_i(\overline{\alpha}(t)))^2 \mathrm{d}t \Big/ \int \alpha(t)\mathrm{d}t}, \tag{1.5}$$

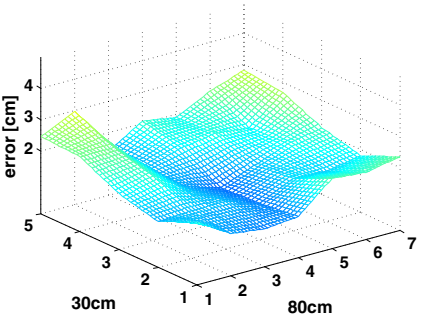
where $\alpha(t)$ and $\overline{\alpha}(t)$ are warping functions.

The Figure 10 (a)–(d) compare the error (as calculated above) of pointing movements that are synthesized based on linear and exemplar-based PHMMs. Clearly, the performance in the middle of the region increases for the exemplar-based 3×3 PHMM. For the linear PHMM, the performance does not increase for a training raster of higher resolution. This is not surprising, since the linear PHMM cannot adopt to the non-linear details of the movement variation. The results for the grasping movements are similar to those for the pointing movements, and this for both types of PHMMs (for the exemplar-based PHMM, see Figures 10, (e), and (f)).

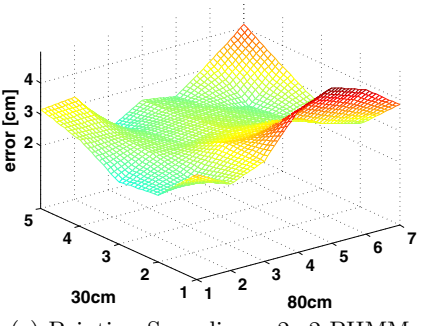
Summarizing, the synthesis errors are approximately 1.9 cm for the exemplar-based PHMM for grasping and pointing, if the outer regions are neglected, where the pose of the person is extremely stretched. For the linear PHMMs, the errors are slightly higher (≈2.8 cm). As a reference for the errors, we compare with the accuracy of the human performer of the recorded movements. For both movement types the average error or deviation ε between pairs of performed movements of



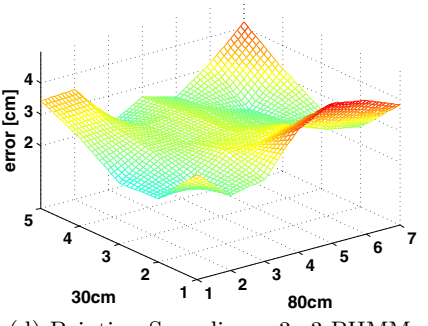
(a) Pointing Syn., 2×2 eb-PHMM



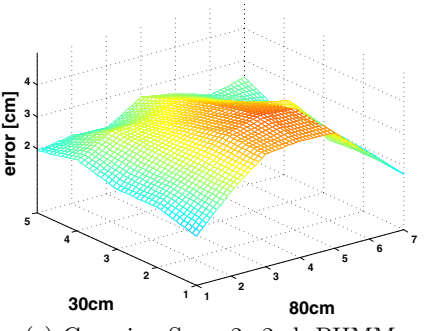
(b) Pointing Syn., 3×3 eb-PHMM



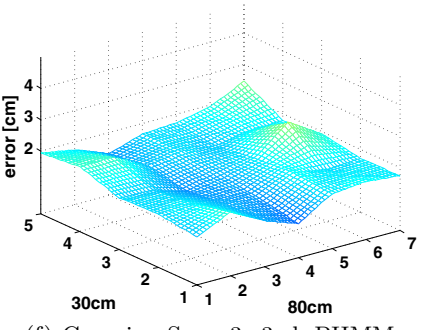
(c) Pointing Syn., linear 2×2 PHMM



(d) Pointing Syn., linear 3×3 PHMM



(e) Grasping Syn., 2×2 eb-PHMM



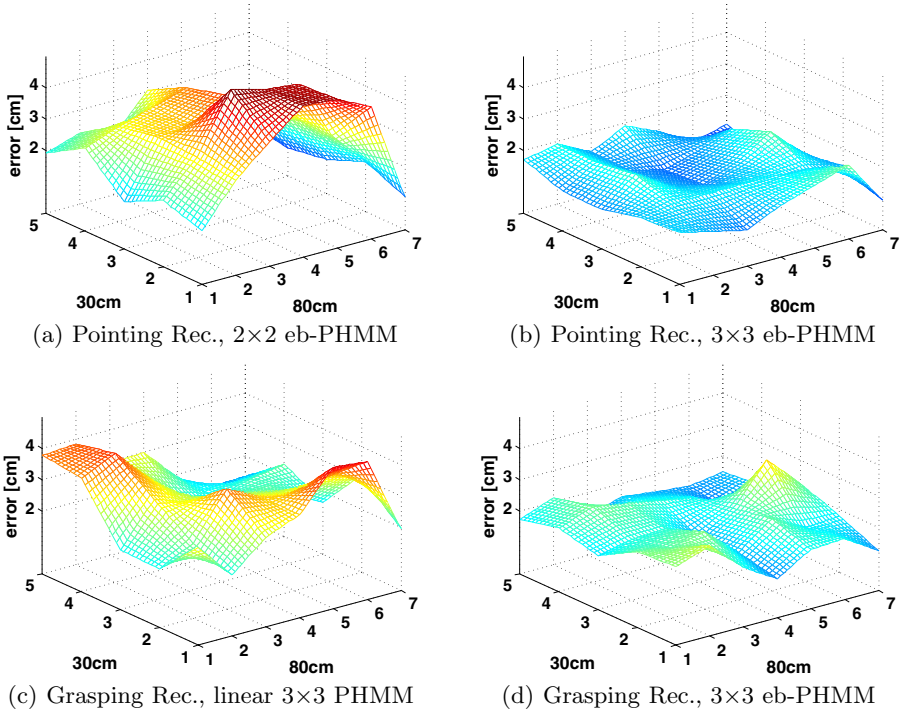
(f) Grasping Syn., 3×3 eb-PHMM

Fig. 10. *Synthesis Error.* The figures show the overall root-mean-square error for pointing and grasping sequences synthesized for 7×5 raster positions. The error is compared for the synthesis based on linear and exemplar-based PHMMs, where the training is based on 3×3 or 2×2 training rasters.

same type is ≈2.6 cm for the position in the middle of the raster. We assume now that a movement is a performance of an intended optimal movement. The error of a performance by the human is then at least about ≈1.3 cm. The accuracy of the exemplar-based PHMM (≈ 1.9 cm) is then comparable with the accuracy of the human performer.

Evaluation of Recognition. Here, we considered two things: the recognition performance in terms of the recognized associated position of an action, and the rate of correct classification of the actual action itself. In addition, we tested the robustness to noise. It is worth to take a look at Figure 9, which shows that the maximization of the log likelihood $l(u, v) = \log P(\boldsymbol{X}|\boldsymbol{\lambda}^{(u,v)})$ w.r.t. (u, v) for a given movement is solvable by standard optimization techniques (smoothness, strict concavity).

Recognition. For each position of the 5×7 raster, the recognition error is the deviation of the estimated position (u, v) for an input movement to the true raster position. For each raster position, we average the error based on four movements. The recognition results are very similar to the synthesis results. We depicted only some examples for the exemplar-based PHMMs (Figures 11 (a), (b), and (d)) and the linear PHMM (Figure 11 (c)). Like for the synthesis based on the exemplar-based PHMM, the performance increases here again in the inner table-top region for the 3×3 PHMM if we compare to the 2×2 PHMM.



(a) Pointing Rec., 2×2 eb-PHMM (b) Pointing Rec., 3×3 eb-PHMM

(c) Grasping Rec., linear 3×3 PHMM (d) Grasping Rec., 3×3 eb-PHMM

Fig. 11. *Recognition Error.* The figures show the error of the recognized (pointed at, and reached for) positions of pointing and grasping movements performed for 7×5 raster positions. The error is the averaged error for four movement performances per raster position. The figures compare the recognition performance of the linear and exemplar-based PHMMs for different training rasters.

In contrast to synthesis, the accuracy of the recognized position by the linear PHMM is in the very inner region equally good compared to ours. For both PHMM types we have errors of ≈2 cm (at least in the inner region of the raster). The errors of the synthesis results can be explained, to some degree, through the performance of the movements through the human. The performer missed the true raster positions slightly. The average deviation from the true position is in the middle of the raster is about ≈0.8 cm. In addition, one has to take into account that we estimate the parameterization of a movement based on the whole performance. Since the movement starts end ends with the arm hanging down besides the person, it seems to be unavoidable that the person has to correct the movement during the performance in order to attain the right position at table top. In such a case, the first part of the movement is misleading for the estimation of the attained position.

Classification. The rate of correct-classified types of the 280 grasping and pointing test movements is 94% if we use our 2×2 PHMMs. The rate decreases insignificantly to 93% for the 3×3 PHMMs. The rate for linear PHMM is also insignificantly different (95%), and this, independent from the training raster.

Robustness of the Recognition. We tested the robustness of the recognition of the parameterization of movements by adding Gaussian noise to each component of the samples of the movements. Here, we detected no significant influence for independent distributed noise with $\sigma < 15$cm. Obviously, that is caused by the large number of samples of a sequence.

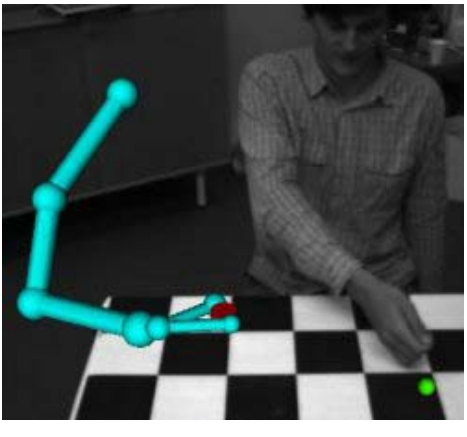


Fig. 12. *Online Recognition Demo.* A person is advising a virtual robot arm to relocate objects (currently, a red one is grasped by the robot). The ball nearby the person's hand indicates the recognized position, and a high likelihood (green) of pointing.

Fig. 13. *Noise of the 3D Body Tracker.* The two consecutive frames show the estimated arm by a 3D body tracker [3]. The estimation is in this specific experiment not very accurate (noisy).

3.3 Online Implementations

In addition to our off-line experiments, we have also implemented and tested our approach for online applications. We have done two implementations, both available via the web links [5,7].

The first implementation, a *recognition demo* (link: [5]), consists of an augmented reality setup, see Figure 12. The table-top scenario contains an animated robot arm, a stereo camera rig, and two objects on a table. The idea of the demo is that the person advises the robot arm which object to grasp next in order to relocate it. The important part is here the recognition of pointing actions and pointed at positions. The recognition is performed by a PHMM which represents shoulder, elbow, and wrist trajectories. To achieve online recognition, we calculate the likelihood of pointing for each frame over the recent history. The trajectories are generated by vision-based 3D body tracking [3]. This experiment

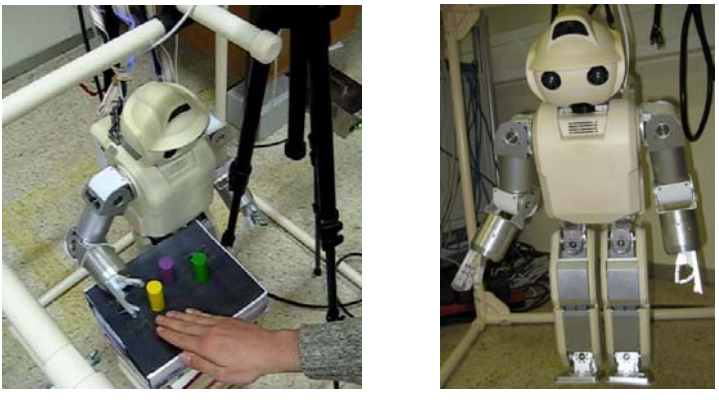


Fig. 14. *Robot Synthesis Demo.* A person advises the robot HOAP-3 (right) which object to clean up next.

shows the robustness of the PHMM-based recognition. Figure 13 gives an impression about the noise generated by the body tracking.

The second implementation is a *robot demo* (link: [7]) where we have replaced the animated robot arm with the humanoid robot HOAP-3 by Fujitsu (Figure 14). Starting point of the demo was a children's toy box with specific holes for different object shapes. The aim was here to tell the robot though pointing and grasping gestures which object belongs into which hole such that the robot would be able to learn and perform the appropriate actions later in a different setup. For training and testing the objects can be anywhere on the table. The main contribution of this demo is the transfer of the movement to the robot. Again, the synthesis of movements is done based on PHMMs that are trained as described in Section 3.2, above. In this experiment, the robot's embodiment differs from that of the human demonstrator, but the appearance of the human like way of moving is still preserved. For a complete description we refer to [8].

4 Summary

In this paper we have applied parametric HMMs in a variety of different experiments to test their ability (a) to recognize human movements and (b) to synthesize human-like movements on a humanoid robot. One of the most important outcomes in the context of recognition is that PHMMs can be very effectively used not only for the recognition of the action itself but also for the recognition of its parameterization. A further impressive outcome is the robustness of the PHMMs to noise: Apart from the theoretic findings by adding white noise with increasing variance to the off-line test data, we were also able to identify the action and its parameterization based on on-line data coming from a vision-based 3D human body tracker. It is apparently one of the great strengths of the HMM framework to average out the noise and to identify the consistency within the input data. We achieved fairly accurate results for action recognition ($\approx 95\%$). For parametric arm movements (similar to those we considered so far), classical HMMs achieve such a rate only on the basis of a tuned feature set [16].

Another important outcome is the possibility to use the PHMMs as a motion model to drive the arm movement of a humanoid robot with sufficient precision and dependent on context and object locations. The synthesis error (averaged error of the trajectories of shoulder, elbow, wrist, etc.) and the error of the recognized parameterization of movements are quite small ($\approx 2\,\text{cm}$), especially, if one takes the natural variance of human performances for comparison.

References

1. Asfour, T., Welke, K., Ude, A., Azad, P., Hoeft, J., Dillmann, R.: Perceiving objects and movements to generate actions on a humanoid robot. In: Proc. Workshop: From features to actions – Unifying perspectives in computational and robot vision, ICRA, Rome, Italy (April 2007)

2. Dariush, B.: Human Motion Analysis for Biomechanics and Biomedicine. Machine Vision and Applications 14, 202–205 (2003)
3. Grest, D., Woetzel, J., Koch, R.: Nonlinear Body Pose Estimation from Depth Images. In: Kropatsch, W.G., Sablatnig, R., Hanbury, A. (eds.) DAGM 2005. LNCS, vol. 3663, pp. 285–292. Springer, Heidelberg (2005)
4. Gunter, S., Bunke, H.: Optimizing the number of states, training iterations and Gaussians in an HMM-based handwritten word recognizer. In: Proc. Seventh International Conference on Document Analysis and Recognition, vol. 01, pp. 472–476 (2003)
5. Herzog, D., Grest, D., Krueger, V.: An Online Recognition Demo, http://www.cvmi.aau.dk/~deh/demo/recognition.avi
6. Herzog, D., Krüger, V., Grest, D.: Parametric Hidden Markov Models for Recognition and Synthesis of Movements. In: Proceedings of the British Machine Vision Conference (BMVC), Leeds, UK, September 2008, vol. 1, pp. 163–172 (2008)
7. Herzog, D., Ude, A., Krueger, V.: HOAP-3 Online Demo, http://www.cvmi.aau.dk/~deh/demo/robot.avi
8. Herzog, D., Ude, A., Krueger, V.: Motion Imitation and Recognition using Parametric Hidden Markov Models. In: Proc. 8th IEEE-RAS International Conference on Humanoid Robots Humanoids 2008, December 2008, pp. 339–346 (2008)
9. Huang, X., Ariki, Y., Jack, M.: Hidden Markov Models for Speech Recognition. Edinburgh University Press (1990)
10. Jacobs, R., Jordan, M., Nowlan, S., Hinton, G.: Adaptive mixtures of local experts. Neural Computation 3, 79–87 (1991)
11. Keogh, E., Pazzani, M.: Derivative dynamic time warping. In: Proceedings of the 1st SIAM International Conference on Data Mining (SDM 2001), Chicago, USA (2001)
12. Lu, C., Ferrier, N.: Repetitive Motion Analysis: Segmentation and Event Classification. IEEE Transactions on Pattern Analysis and Machine Intelligence 26(2), 258–263 (2004)
13. Moeslund, T., Hilton, A., Krueger, V.: A survey of advances in vision-based human motion capture and analysis. Computer Vision and Image Understanding 104(2-3), 90–127 (2006)
14. Murphy, K.P.: Hidden semi-Markov models (HSMMs). In: unpublished notes (2002)
15. Rabiner, L.R., Juang, B.H.: An introduction to hidden Markov models. In: IEEE ASSP Magazine, January 1986, pp. 4–15 (1986)
16. Ramana, P.K.R., Grest, D., Krüger, V.: Human Action Recognition in Table-top Scenarios: An HMM-based Analysis to Optimize the Performance. In: Proceedings of Computer Analysis of Images and Patterns, Vienna, pp. 101–108 (2007)
17. Schaal, S.: Is Imitation Learning the Route to Humanoid Robots? Trends in Cognitive Sciences 3(6), 233–242 (1999)
18. Vecchio, D., Murray, R., Perona, P.: Decomposition of Human Motion into Dynamics-based Primitives with Application to Drawing Tasks. Automatica 39(12), 2085–2098 (2003)
19. Wilson, A.D., Bobick, A.F.: Parametric hidden Markov models for gesture recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence 21(9), 884–900 (1999)

Recognizing Human Actions by Their Pose

Christian Thurau[1] and Václav Hlaváč[2]

[1] Fraunhofer IAIS
Schloss Birlinghoven, 53754 Sankt Augustin, Germany
christian.thurau@iais.fraunhofer.de
[2] Czech Technical University in Prague
Faculty of Electrical Engineering
Center for Machine Perception
121 35 Prague 2, Karlovo namesti 13
hlavac@fel.cvut.cz

Abstract. The topic of human action recognition from image sequences gained increasing interest throughout the last years. Interestingly, the majority of approaches are restricted to dynamic motion features and therefore not universally applicable. In this paper, we propose to recognize human actions by evaluating a distribution over a set of predefined static poses which we refer to as pose primitives. We aim at a generally applicable approach that also works in still images, or for images taken from a moving camera. Experimental validation takes varying video sequence lengths into account and emphasizes the possibility for action recognition from single images, which we believe is an often overlooked but nevertheless important aspect of action recognition.

The proposed approach uses a set of training video sequences to estimate pose and action class representations. To incorporate the local temporal context of poses, atomic subsequences of poses using n-gram expressions are explored. Action classes can be represented by histograms of poses primitive n-grams which allows for action recognition by means of histogram comparison. Although the suggested action recognition method is independent of the underlying low-level representation of poses, representations remain important for targeting practical problems. Thus, to deal with common problems in video based action recognition, e.g. articulated poses and cluttered background, a recently introduced Histogram of Oriented Gradient based descriptor is extended using a non-negative matrix factorization reconstruction.

1 Motivation

The last years gave rise to many different ways of representing human activities in computer vision applications [4,11,15,16,17,20,23,24,38]. While most presented representations lead to convenient recognition results on standard data sets, we could also observe an (possibly unnecessary) increase in complexity of representations. We advocate that a common life approach in representing the activity as a sequence of pose primitives is of advantage. The intuitively understandable *pictographs*, graphic record in hieroglyphic symbols, are used in our everyday

D. Cremers et al. (Eds.): Visual Motion Analysis, LNCS 5604, pp. 169–192, 2009.

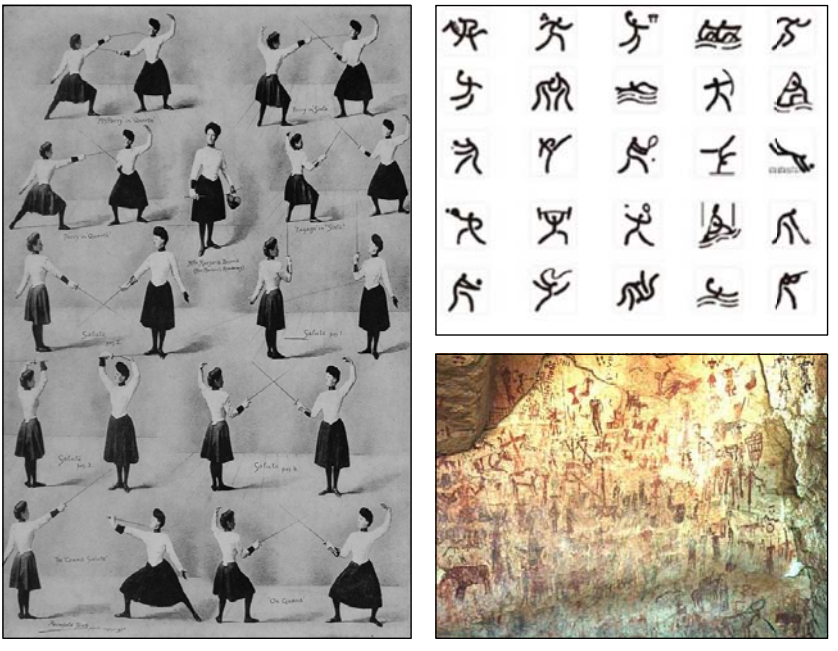


Fig. 1. Examples of prototypical poses

life, for instance, for information signs, icons on our computer screens, or sports events. Pictographs represent the meaning of an activity very concisely. Interestingly, despite their simplicity, they were not yet explored for the use as an action representation in recognition tasks, to our knowledge.

Let us present a few examples in Fig. 1 in a chronological order. Ancient cave paintings already show similarities to modern pictographs as they try to explain complex actions with only a very few stylized poses, see Fig. 1, bottom right. Egyptian hieroglyphs developed from such paintings. The one hundred years old example shows pose primitives depicted in the women fencing instruction book from 1908, see Fig. 1, left side. The newest example are the pictographs from the Beijing Olympic games 2008, see Fig. 1, top right. Notice that all these pictographs are intuitively understood by us humans even if the underlying action is rather complex.

The terminology in the scope of this contribution has not settled yet. Let us consider that only a single person is observed for simplicity. This detailed look does not undermine the generality because there can be several independently represented persons in the observed scene. Let us introduce needed concepts informally in a top-down manner on the illustrative example, say a person who first walks, then runs, stumbles over an obstacle, falls and stays down. The top level concept is the *activity* which captures the whole scenario of the example. The activity can be decomposed into *actions* (jumping, running, stumbling, falling, staying down). The action can be broken down into (body) *poses*. If we

wished to provide a low level quantitative description of a single person's action then we could sample it densely in time to very many time snaps. This low-level representations are called *poses*.

If the activity representation is leveraged to a qualitative level then similar poses are grouped to *pose primitives* (sometimes called prototypical poses in the literature) capturing characteristic properties of the consequent poses having the same meaning. Such pose primitives are typically shown in pictographs discussed above. We chose the word 'primitive' because it is a common term for describing basic building blocks of action/motion in robotics or cognitive psychology (see also Section 2 for further explanations). Further, the idea of constructing more complex action patterns by sequencing pose primitives relates to common ideas in structural pattern recognition and text mining. We believe that methods of these disciplines will be used soon also for activity description/recognition. Similarly on the action level, we need a concept which encompasses many instances of semantically same actions which look similarly. We will call such grouped actions as *action class*.

Poses are time ordered. We like to depart from the 'time series'-like representation. This makes sense because a difficult problem of low level time warping of the observed time series and stored pattern time series can be naturally bypassed by working on the higher level representation. The action is a sequence of pose primitives. Sometimes, periodicity or other more complicated pattern can be observed in this sequence. For example, walking consists of several motion primitives that are periodically repeated.

In this contribution, in contrast to other recent approaches which use motion-dependent features, we propose a strictly pose based action classifier. We assume that arbitrary human action can be constituted by sequencing a set of static prototypical poses. However, we do not use a single pose for representing a single activity class but a distribution over a predefined set of poses. In the following, we will put a special emphasize on action class representations using poses, low-level image features for describing a single poses. The actual classification process does not model or assume a sequential ordering of poses and is thereby applicable to variable sequence lengths or still images. However, we will show that considering the local temporal ordering of (two or three subsequent) poses can improve action recognition.

We can proceed to the 'informal story' of the contribution. The intention is to learn pose primitives and action classes from a few training videos. During training we try to remove as much background clutter as possible. This is achieved by detecting persons in the video and applying background subtraction. Note that background subtraction is only necessary during training. From these cleaned videos poses and poses primitives can be learned. Pose primitives are clustered to action classes. All these steps could be conducted in unsupervised manner, however, for clustering action classes we usually rely on a labeled set of training sequences. The human comes into play if she/he intends to label naturally found pose primitives and action classes by semantically meaningful labels. In run mode, a video or a still image with possibly cluttered background serves as

input. After a first step of human detection, the pose primitives are recognized in it. This opens the possibility to recognize also action classes by analyzing sequences of pose primitives and not the underlying raw image data.

The remainder of this contribution is organized as follows: in the next Section, we will briefly summarize related work. In Section 3, we introduce a general approach for pose based activity recognition. The approach does not make assumptions about low-level features. For targeting practical problems, we have to consider pose representations that possibly work for single images. Consequently, in Section 4 we will present a Histogram of Oriented Gradient (HOG) based pose representation which is also applicable for still images with unknown background. In Section 5, we will present detailed experimental evaluation of the proposed approach. Finally, the conclusion and outlook on future work will be presented in Section 6.

2 Related Work

Representations for action and motion are not only of importance for video-based recognition, but also for cognitive psychology (how do humans internally represent motion?), or robotics (how to classify, imitate and represent human action?). A basic common idea of recent approaches is to derive a set of basic building blocks that are sufficient for constructing more complex motions or actions. Note that, although different disciplines use different terms (action-, movement-, or motion-primitives) that serve as a basic building block, certain common ideas can be noticed. Since we focus on distributions over a set of predefined poses in this contribution, we use the term pose-primitive. In the following, we briefly summarize related approaches and try to show similarities among terminologies in different fields. Moreover, we give an overview of recent approaches in video-based action recognition, with a special emphasize on action representation and low-level image features.

Results in behavioral biology indicate that instead of directly controlling motor activation commands, movements are the outcome of so called combination of movement primitives or motor primitives [8,10]. Recently, experiments in cognitive psychology explain the representation of motor skills in human long-term memory by sequences of *basic-action concepts* [26]. Basic-action concepts are in this context similar to movement primitives but assume an internal tree-like structuring of primitive units (called basic action units). Moreover, pose based action recognition has been proposed as a model for action recognition in the cortex. Neurons in the superior temporal sulcus, so called "snapshot" neurons, were found that respond to static presentations and action [35]. A more detailed review of the problems of motor control can be found in [10,9,37].

The idea of movement primitives found its way into various robotic and motion control applications. A movement primitive is in this context often referred to as a computational element in a sensorimotor map that transforms desired limb-trajectories into actual motor commands [30]. The work of Fod et al. [9] is related to our approach. There, the concept of movement primitives is applied in

a computational approach by linear superposition of clustered primitive motor commands that finally result in a complex motion in simulated agents. Our previous work used a probabilistic sequencing of action primitives for imitating human activities in simulated game worlds [32], deriving action primitives in a similar way as in [9]. In this paper, we follow that line of research and adapt it to the domain of image-based action recognition.

For video based recognition, the idea of representing actions by a set of basic building blocks is not new, see also [22] for a brief overview. Often, the term key-frame is used in a similar way. In [5], an action recognition approach is proposed that matches shapes from video sequences against stored key-frames, thereby recognizing specific tennis strokes in sports videos. Moeslund et al. [21] have recognized actions by comparing strings of manually found motion primitives, where the motion primitives are extracted from four features in a motion image. In [25], silhouette key-frame images are extracted using optical flow data. The key-frames are grouped in pairs and used to construct a grammar for action recognition. In [12], the idea of recognizing human behavior using an activity language is further extended. Recently, Weinland et el. [36] have proposed a time-invariant representation of actions by making direct use of distances to key-poses. Further papers related to our approach are the works of Hamid et al. [13], and Goldenberg et al. [11]. The work [13] applied a sequence comparison using bags of event n-grams to recognize higher-level behaviors, where n-grams are build using a set of basic hand crafted events. Goldenberg et al. [11] extracted *eigenshapes* from periodic motions for behavior classification.

Having a closer look at the underlying features representing activities, we can spot two main classes, *dynamic features* [4,16,17,23,24] and *static pose based features* [7,11,15,20,31,38]. Generally, it depends on the application whether to choose dynamic or static image features (obviously, we can not extract dynamic features from still images).

Regarding dynamic features, Blank et al. [4] use three dimensional space-time shapes extracted from silhouette images of humans for classifying activities. In [17], shape and motion cues are used for action recognition of two different actions in the movie 'Coffee and Cigarettes'. Jhuang et al. [16] introduce a biologically inspired action recognition approach which uses hierarchically ordered spatio-temporal feature detectors. Niebles et al. [23,24] extended the *bag-of-features* concept to account for activity recognition. In [24], human action-categories are represented and learned using space-time interest points. In [23], a hierarchical bag of features approach is applied to image sequences. Note that [23] compared static and dynamic image features and concluded that dynamic features should be preferred. This might appear contrary to the results of this paper and others (e.g. [15]) where solely static features are used. Schindler et al. [27] combine form and motion features and thereby outperform each feature taken alone, which might be an expected outcome. However, although the mentioned papers were evaluated on the same data-set, it might not be useful to interpret the results solely based on the used features.

Besides motion features, we can find static pose based representations where poses are often described by silhouettes [11,15]. A *bag-of-rectangles* method is used for action recognition in [15], effectively modeling human poses for individual frames, and thereby recognizing various action classes. Goldenberg et al. [11] use *Principal Component Analysis* (PCA) to extract eigenshapes from silhouette images for behavior classification. Since silhouettes usually require background subtraction and are therefore rather restrictive, other static pose descriptors were suggested [20,38,31]. Lu et al. [20] represent actions of hockey players as PCA-HOG descriptors and action recognition is based on a set of predefined poses. Zhang et al. [38] recognize different actions by finding articulated poses in infrared images. They cluster a set of primitive poses based on HOG descriptors. Ferrari et al. [7] estimate the body pose by means of pictorial structures and afterwards classify poses into action classes using a linear SVM [7]. In [31], we introduced a first approach towards action recognition using histograms of clustered HOG descriptors of human poses.

The methods estimating the pose which are the most related to our approach are [38,3,1]. In [38], individual object detectors are trained for a number of poses. To deal with varying backgrounds, a weighting of foreground (pose) pixels and background pixels is applied during training. Bissacco et al. [3] use a Latent Dirichlet Allocation (LDA) based segmentation of human poses represented by HOG descriptors. Agarwal et al. [1] estimate 3D human poses using a local basis representation extracted by means of *non-negative matrix factorization* (NMF). Similar to this paper, they apply NMF to a set of clean (no background clutter) human poses, and use the NMF weights to reconstruct novel poses. In contrast to [1], we include a set of background bases to further alleviate the influence of background clutter to the pose estimation.

3 Pose Based Action Recognition

In pose based action recognition, the task is to infer an action class based on a single recognized pose or on a sequence of recognized poses. Generally, the sequential ordering of poses is assumed to be of great importance for action recognition. Consequently, a lot of approaches try to model the underlying sequence of motion/poses directly, often using Markovian models. For handling classical computer vision problems, e.g. object recognition, a trend towards unordered 'bag of words' representations has appeared lately, see, e.g. [29]. In this contribution, we follow this line of research by interpreting human poses as unordered events. We ignore the global sequential ordering of poses and implement a simple classification scheme using learned histograms of pose primitives. The idea of classifying action classes by means of histogram comparison was first introduced in [13] where classes of higher level action categories were recognized using a set of predefined events. In contrast to [13], we focus on more primitive actions and use a simple (clustered) pose based representation which makes the overall approach more similar to the popular 'bag of words' paradigm.

Fig. 2. Visualization of silhouette-based pose primitives

Note that the actual low level feature representation of poses is not of interest at this point. The later sections will show detailed results on three varying low-level representations. In the following, we explain in detail how to classify a sequence of poses using histograms, and we also highlight the importance of incorporating the local temporal context by means of n-gram expressions.

3.1 Histogram Based Action Recognition

The underlying assumption of our approach is that we can construct arbitrarily complex actions by sequencing pose primitives which mean a given set of clustered poses here. Figure 2 shows an example of pose primitives that were clustered from action sequences. As an input to clustering we used a reduced set of silhouette vectors. Interestingly, the visual appearance of pose primitives is surprisingly similar to idealized pictographs often used for describing sport activities. While the idea of activity representation by identification of sequences of prototypical pose primitives seems to be closely related to highly optimized activity representations found in pictographs, for convenient recognition results we do not limit the representation to a single pictograph.

For clustering, we usually use a conventional k-means clustering. In the following, we denote an individual pose primitive by $\mathbf{a_i}$. Each frame in a sequence showing a human can be mapped on one particular pose primitive. A video sequence can be expressed as a sequence of pose primitives by $\mathbf{A}=[\mathbf{a_1}, \ldots, \mathbf{a_n}]$. The assignment of observations to pose primitives is done using a simple similarity ranking. This results in a soft distribution over all pose primitives for each frame.

As already mentioned, instead of directly analyzing the sequential ordering of pose primitives, we concentrate on the number of occurrences of pose primitives. Thus, a histogram of a video sequence of pose primitives serves as an action representation. During training we assume a set of well separated training videos showing the execution of one particular action class. Histograms of pose primitives of these videos are our class representations and are used for recognizing novel pose sequences or single poses.

Fig. 3 shows visualization of pose primitive histograms for various action classes. We show three examples showing three pairs of action classes, left hand waving vs. right hand waving in Fig. 3(a); walk vs. run in Fig. 3(b) and jumping jack vs. run Fig. 3(c) (see also Figure 8 for a visualization of these action classes). It can be seen that histograms over pose primitives are more similar for related action classes (e.g., wave with one hand and wave with two hands), and more distinct for intuitively more different actions (e.g., jumping jack and run).

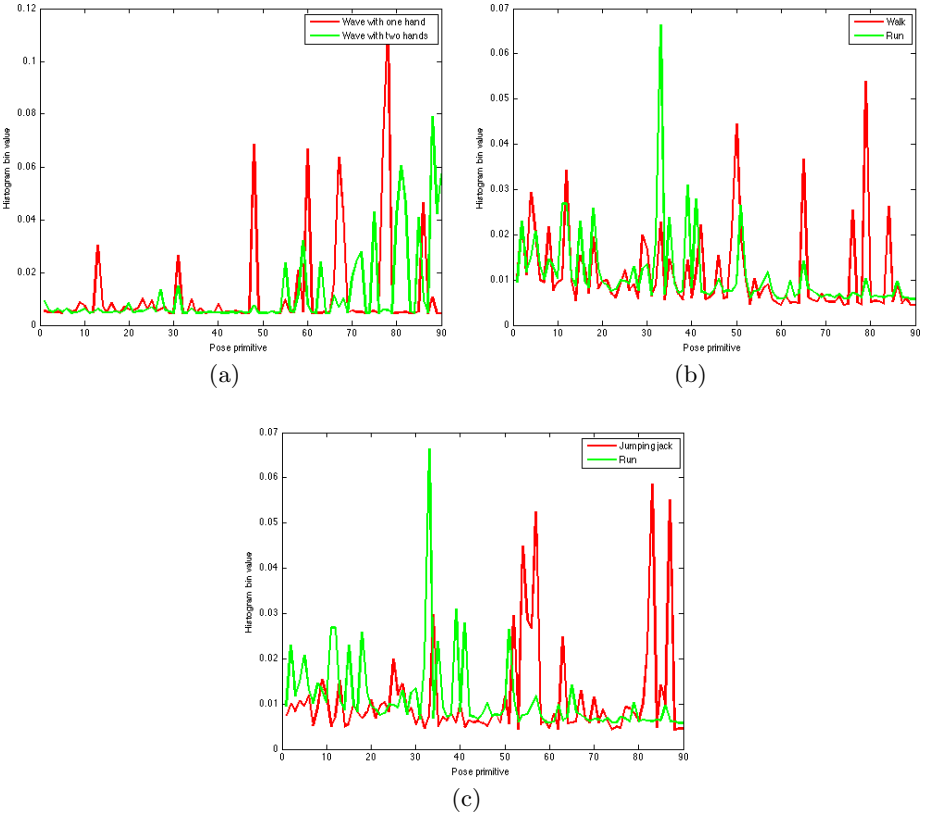


Fig. 3. Three pairs of histograms plotting allowing to compare three pairs of action classes

Given a sequence $\mathbf{A}$ of pose primitives, a histogram $\mathrm{H}(\mathbf{A})$ for all pose primitives $[\mathbf{a}_1, \ldots, \mathbf{a}_n]$ that were clustered during the training phase is defined as a mapping

$$\mathrm{H}(\mathbf{A})\colon \mathbf{A} \rightarrow \mathbb{N}^+ \cup 0, \mathrm{H}(\mathbf{A})_{a_i} := \mathrm{occ}(a_i, \mathbf{A})\ , \tag{1}$$

where $i = 1, \ldots, n$, and n denotes the total number of pose primitives, and $\mathrm{occ}(a_i, \mathbf{A})$ denotes the number of occurrences of $\mathbf{a}_i$ in $\mathbf{A}$. A normalized pose primitive histogram $\phi(\mathbf{A})$ is then simply defined as $\phi(\mathbf{A}) = \mathrm{H}(\mathbf{A}) / \mid \mathrm{H}(\mathbf{A}) \mid$.

Given the normalized histogram $\phi(\mathbf{A})$ for a sequence of pose primitives $\mathbf{A}$, the classification is performed according to the minimum distance D to a set of normalized training histograms $\phi(\mathbf{T}_i)$ of pose primitive sequences $\mathbf{T}_i$.

$$j = \underset{i}{\operatorname{argmin}}\, D(\phi(\mathbf{A}), \phi(\mathbf{T}_i)), \tag{2}$$

Effectively, we compare the distribution of pose primitives for each observed sequence, i.e., estimating the correct action class from a given sequence of poses is now a problem of inferring the most similar known distribution of pose prototypes.

In previous work [33,31,34], we classified based on the minimum distance to a single training histogram using a simple Nearest Neighbor (NN) classifier. While this often gave convenient recognition rates, it also showed to be more sensitive to parameter settings, especially the outcome of pose primitive clustering and the number of pose primitives used for clustering. A bad initialization of clusters could lead to mediocre classification results. In this contribution, for a more robust representation, we follow the *single histogram class model* proposed in [28], which, in our case, is a simple mean of the histograms belonging to one action class ($\phi(\mathbf{T}_c) = \sum_{c_i}^{n} \phi(\mathbf{T}_{c_i})/n$). Use of additional weightings of certain histogram bins, as proposed in [28], did not improve overall classification rates and is therefore not applied. Using single class histogram models, we could increase the stability of recognition results and we reduce the computational complexity since we now represent each activity class by only one histogram. However, as we will see in Section 5, peak classification results can be slightly lower compared to conventional NN classification.

3.2 Histogram Distance Measures

The classification based on the minimum distance to a set of action class histograms requires an appropriate distance measure. Various distance, divergence, or similarity measures are regularly applied for comparing histograms. Since the histogram based approach in pose based action recognition is applied for the first time, to our knowledge, Section 5 will present detailed experimental evaluation of different common histogram distance measures. We list them already here,

$$
\begin{aligned}
&L_1 - \text{distance:} && D_{\text{L1}} = \mid \phi(\mathbf{A}) - \phi(\mathbf{T}) \mid ,\\
&L_2 - \text{distance:} && D_{\text{L2}} = \mid \phi(\mathbf{A}) - \phi(\mathbf{T}) \mid^2 ,\\
&\text{Kullback-Leibler divergence:} && D_{\text{KL}} = \phi(\mathbf{A}) \log \tfrac{\phi(\mathbf{A})}{\phi(\mathbf{T})} ,\\
&\chi^2 \text{ significance :} && D_{\chi^2} = \tfrac{1}{2} \textstyle\sum_i \frac{(\mathbf{a}_i - \mathbf{a}_i^{\mathbf{T}})^2}{\mathbf{a}_i + \mathbf{a}_i^{\mathbf{T}}} ,\\
&\text{Bhattacharyya distance:} && D_{\text{BD}} = \text{-ln} \textstyle\sum_i \sqrt{\mathbf{a}_i \mathbf{a}_i^{\mathbf{T}}} ,
\end{aligned}
$$

where $\mathbf{a}_i$, $\mathbf{a}^{T_i}$ denote histogram bin entry i of $\phi(\mathbf{A})$ and $\phi(\mathbf{T})$, respectively.

At this point, it is important to have a look at the influence of different sequence lengths of observed pose sequences. Especially the case in which it has to be decided about the action class by observing a single frame only. In such a case, the selection of histogram distance measure might be even more important since the corresponding query histogram is usually only very sparsely covered (for a discrete assignment to pose primitives it would only contain one entry). It can be seen that there are differences among similarity measures on how they take sparsely covered query histogram bins into account. Following observations in [28], especially the KL-divergence, but also Bhattacharyya distance and χ^2 significance might be of interest for this type of histogram comparisons, since they do not penalize zero query histogram bins that much.

Consider, for instance, the KL-divergence as a histogram distance measure. We get for $\text{argmin}_i\, D(\phi(\mathbf{A}), \phi(\mathbf{T}_i))$

$$j = \underset{i}{\text{argmin}}\, D_{\text{KL}}(\phi(\mathbf{A}) \mid\mid \phi(\mathbf{T}_i)), \tag{3}$$

where $i = 1, \ldots, n$ and

$$D_{\text{KL}}(\phi(\mathbf{A}) \mid\mid \phi(\mathbf{T}_i)) = \sum_k \mathbf{a}_k \log\left(\frac{\mathbf{a}_k}{\mathbf{a}_k^{T_i}}\right). \tag{4}$$

For still images, the sequence $\mathbf{A}$ reduces to $\mathbf{A} = [\mathbf{a}_l]$, where l denotes the recognized pose primitive index, thus

$$j = \underset{i}{\text{argmin}}\, D_{\text{KL}}(\phi(\mathbf{A}) \mid\mid \phi(\mathbf{T}_i)) = \underset{i}{\text{argmin}} \log\frac{1}{\mathbf{a}_l^{\mathbf{T}_i}}, \tag{5}$$

where $\mathbf{a}_l^{\mathbf{T}_i}$ corresponds to the histogram entry for pose primitive $\mathbf{a}_l$ in the training histogram $\mathbf{T}_i$. By selecting the action class with the highest probability, it can be seen that the KL-divergence extends rather intuitively to single frame recognition of action classes. More observed poses yield less sparsely populated query histograms and sequences are usually classified more precisely. In contrast, e.g. L_2 distance punishes empty bins of the query histogram more severely and as such might be more likely to produce false classifications.

3.3 Local Temporal Context

As noted in [13], activities are not defined by their content or pose alone. Classification of unordered sequences of poses can only work if a sufficient number of pose primitives are exclusive to a specific action class. For example, an action class of 'hand waving' does contain poses that are most likely exclusive to that class, whereas more general classes, e.g., walking or running, are more likely to consist of poses that are non-exclusive to a particular class.

If we want to stay within the paradigm of unordered pose classification we can not completely alleviate this effect. However, we can make basic pose instances more expressive by incorporating the local temporal context. The local ordering of events or specific poses allows to introduce the context without the need for considering a complete sequence and its sequential structure. Instead of having a look at each frame exclusively, we perform a short subsequencing of poses by means of n-gram expressions. This idea was stimulated by the success of local statistics expressed as n-grams in speech recognition, mathematical linguistics and text mining.

Converting pose primitive sequences to sequences of n-gram instances does not require any modifications to the presented approach. Instead of computing histograms of pose primitives, histograms of n-gram instances are computed. The underlying pose primitives as a descriptor for each frame stay the same. The only effect is that we now have much larger *vocabulary* of pose primitives

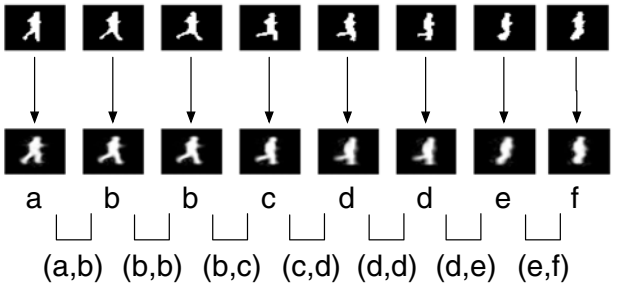


Fig. 4. Sample sequence showing the original silhouettes (upper row), their reconstruction using pose primitives (middle), and their bi-gram representation (lower row)

(The maximum number of instances corresponding to all possible combinations of pose primitives is k^n, where k denotes the number of individual pose primitives and n denotes the n-gram degree). This larger vocabulary is on the one hand more precise and on the other hand does not sacrifice local matching accuracy between a novel observed pose and a pose primitive (which would be the outcome if we would simply try to cluster k^n pose primitives). See also [13] for a more comprehensive introduction to n-grams in the context of activity recognition, and Figure 4 for illustration.

The subsequences are overlapped in our case. For example, the *bi*-gram ($n = 2$) expression of a sequence of pose primitives $\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_n]$ could be simply written as $\mathbf{A}_{bi-\text{gram}} = [\{\mathbf{a}_1\mathbf{a}_2\}, \{\mathbf{a}_2\mathbf{a}_3\}, \ldots, \{\mathbf{a}_{n-1}\mathbf{a}_n\}]$. For using discrete assignments to pose primitives, the actual number of possible subsequences tends to be much lower than k^n and represents only a fraction of all possible n-gram instances. Generally, we can formulate n-gram expressions as a simple stochastic Markov process $\mathrm{P}(\mathbf{a}_i \mid \mathbf{a}_{i-1}, \ldots, \mathbf{a}_{i-n})$, where n corresponds to the degree of the n-gram.

Concerning the subsequences length, with an increase in n, the possible number of n-grams is increasing and as such leads to an increase in computational complexity, and also very low probabilities for individual observations. Moreover, it showed that pose primitive sequences are not rapidly changing, i.e. even in a standard 25 FPS video sequence a person's pose would be constantly assigned to one specific pose primitive over usually 4-6 frames. Thus, the underlying data seems more suited for *bi*-grams since the introduction of *tri* or higher order n-grams would not capture additional contextual information. Consequently, the later presented experiments concentrate on the use of *bi*-grams.

Obviously, it is impossible to extract n-grams from still images. Thus, in order to compare n-grams of pose primitives against single frame recognition, we had to consider two frames in the case of n-grams. Interestingly, for the evaluation data used, the performance of single frame recognition without n-gram expressions (truly single frame recognition) only decreases slightly. This indicates that the extracted pose primitives already provide a decent separability between action classes, at least for the benchmark data used.

4 HOG Based Pose Representations

The presented approach for action class recognition using primitive poses does not impose a specific low-level image representation of poses. Nevertheless, for targeting practical problems, a convenient representation is crucial. In the following, we motivate a novel representation for human poses which is based on NMF basis reconstruction of conventional gradient histograms.

Often silhouettes are used as a low-level pose representation [4,11,15]. Silhouettes bring certain simplifications and benefits to action class recognition: they are independent of a specific persons appearance and clothing texture (silhouettes remain the same), they ignore possibly irritating background clutter, and they are often straightforward to extract from a video using background subtraction. However, the arguably biggest disadvantage is that silhouettes are dependent on well working background subtraction and that they are sacrificing possibly useful information within the body appearance of persons (e.g., hand postures in front of the body or clothing texture).

There are cases of practical interest in which background subtraction cannot be used or it is difficult, e.g., for still images or video sequences from a moving camera. To guarantee applicability in still images, we use the raw image data extracted from a single frame as an input. As mentioned before, most difficulties in pose matching arise from cluttered background and pose articulations. Often, background objects are falsely recognized as limbs or parts of a pose.

4.1 Histograms of Oriented Gradients

The task we are facing here is similar to the pedestrian detection task in computer vision, although we have to consider pose articulations and possibly cluttered background in the vicinity of a person. Instead of recognizing pedestrians standing in an upright position, we have to accurately match a detected person/pose to the best matching pose primitive. In [6], a *Histogram of Oriented Gradients* (HOG) descriptor was introduced for pedestrian detection in raw images. That approach marks up to now one of the most successful representations

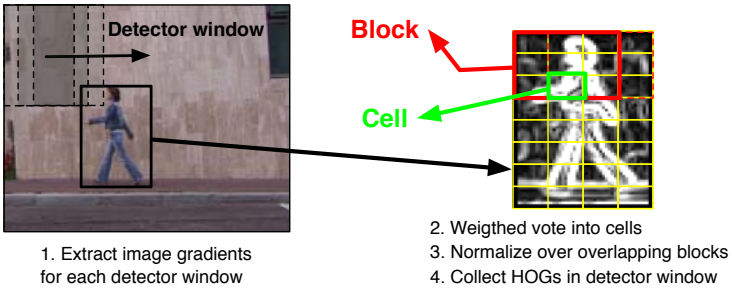


Fig. 5. Computation of a Histogram of Oriented Gradients descriptor

in that domain. Given the success of gradient histograms for pedestrian detection, we build on this work and extend it to cope with the task at hand.

Basically, HOG descriptors are locally normalized gradient histograms, see also Fig. 5 for visualization. Using a sliding window approach, for each detector window it is decided whether it contains a human or not. In this paper, we stay close to the parameters suggested in [6]. The detector window is divided into cells of size 5×5. For the detector window the image gradient is computed using a simple filter mask ($[-1, 0, 1]$). Every pixel calculates a vote for an edge orientation histogram for its cell, weighted by the magnitude (as suggested in [6], we used 9 orientation bins). Groups of 3×3 cells form a so called block. Blocks overlap with each other. For each block, the orientation histograms of the included cells are concatenated and normalized using L2-normalization. The final descriptor is the concatenated vector of all normalized blocks for the detector window.

4.2 NMF Basis Pose Representation

Recent contributions showed applicability of gradient histograms for recognition of more articulated poses [3,1]. Both works model the statistics of gradient histograms for different poses. [3] use Latent Dirichlet Allocation applied to HOG descriptors, resulting in a model-based pose descriptor. In [1], *non-negative matrix factorization* (NMF) is applied to a local image feature representation of human poses. The human poses are presented with a clean background, thus, the NMF weights selectively encode pose features. Our work builds on [1], therefore we first briefly introduce NMF.

Applying non-negative matrix factorization to a non-negative data matrix $\mathbf{V}$ leads to a factorization $\mathbf{V} \approx \mathbf{WH}$ where both $\mathbf{W}$ and $\mathbf{H}$ are constrained to be non-negative. In the following, we interpret $\mathbf{W}$ as a set of basis vectors and $\mathbf{H}$ as coefficients. Consequently, the original data matrix $\mathbf{V}$ can be reconstructed using $\mathbf{W}$ and $\mathbf{H}$.

To find a factorization $\mathbf{V} \approx \mathbf{WH}$, we use the standard multiplicative update rule

$$\mathbf{H}_{i,j}^{t+1} \leftarrow \mathbf{H}^{t}{}_{i,j} \frac{\sum_k \mathbf{W}_{k,i} \mathbf{V}_{i,j} / (\mathbf{WH})_{k,j}}{\sum_m \mathbf{W}_{m,i}} , \tag{6}$$

$$\mathbf{W}_{j,k}^{t+1} \leftarrow \mathbf{H}^{t}{}_{j,k} \frac{\sum_j \mathbf{H}_{i,j} \mathbf{V}_{k,j} / (\mathbf{WH})_{k,j}}{\sum_l \mathbf{H}_{k,l}} . \tag{7}$$

Under these update rules the divergence $D(\mathbf{V}||\mathbf{WH})$ is non increasing, see also [19] for further details.

For image processing, unlike PCA or similar techniques, NMF can result in part based representations of objects. For example, in [18] face images were decomposed into a set of meaningful parts, e.g., ears, eyes. Note that NMF will not always result in part-based representations. While this was not an issue in the presented work, future work might as well include the recently introduced NMF with sparseness constraint that is able to enforce part-based representations [14].

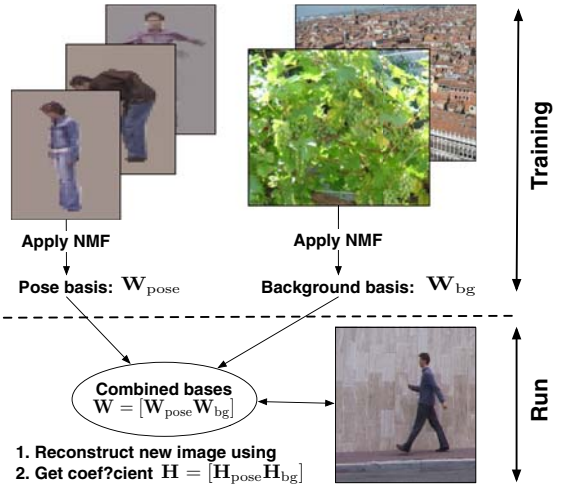


Fig. 6. We extract a set of NMF bases W from clean pose images and background images. Novel images are reconstructed by these bases.

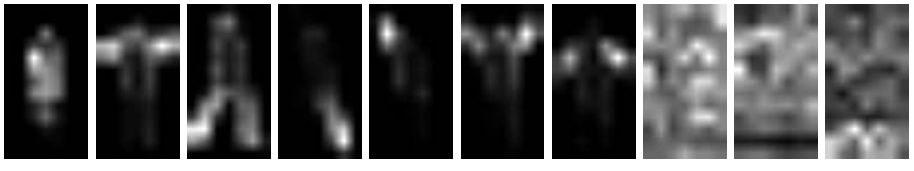


Fig. 7. Visualization of basis vectors as the outcome of non-negative matrix factorization (NMF). The first seven images correspond to basis vectors learned from human poses, the remaining 3 were extracted from background images.

In [1], a set of basis vectors $\mathbf{W}$ are learned from human pose images with clean background by application of non-negative matrix factorization. The idea is that the basis vectors are able to reconstruct a novel descriptor, and thereby also ignore possible background clutter. In contrast to [1], we aim at finding a larger variety of articulated poses. If reconstructing a gradient image solely from pose bases $\mathbf{W_{pose}}$, we found that also background clutter would often be reconstructed which often leads to false recognitions. Therefore, we apply two modifications: we compute the bases $\mathbf{W}$ for the whole image containing the pose, and we add a second set of bases $\mathbf{W_{bg}}$ that is computed from background images as well (see also Fig. 6).

This leads to the following reconstruction $\mathbf{V} = [\mathbf{W_{pose}}\mathbf{W_{bg}}][\mathbf{H_{pose}}\mathbf{H_{bg}}]$.

During training, pose $(\mathbf{W_{pose}}, \mathbf{H_{pose}})$ and background $(\mathbf{W_{bg}}, \mathbf{H_{bg}})$ parameters are learned independently of each other.

Fig. 7 visualizes some resulting bases. Application of NMF results in some interesting decomposition of human poses in which we can identify individual

body parts. Complete poses are the outcome of an appropriate combination of basis vectors. It can be seen that the $\mathbf{W_{pose}}$ represents meaningful parts of a pose. The later carried out experiments contain detailed results on parameter selection.

For estimating the pose from a novel image $\mathbf{V^{new}}$, we first compute the coefficient $\mathbf{H^{new}}$ corresponding to $\mathbf{W} = [\mathbf{W_{bg}}\mathbf{W_{pose}}]$. For this, we use the standard iterative algorithm while holding $\mathbf{W}$ fixed. Thereby, the weights are combined so that they give the best possible explanation for $\mathbf{V^{new}}$ under the usage of $\mathbf{W_{bg}}$ and $\mathbf{W_{pose}}$.

Due to the strictly additive combination of weights in NMF, we usually have either a usage of background weights $\mathbf{W_{bg}}$ or pose weights $\mathbf{W_{pose}}$ for certain descriptor parts. The outcome are the coefficients $\mathbf{H^{new}}$ composed of separate coefficients for pose and background weights $\mathbf{H^{new}} = [\mathbf{H_{bg}^{new}}\mathbf{H_{pose}^{new}}]$. Effectively, we decouple the background appearance from the foreground by means of NMF basis reconstruction.

Note that the proposed extraction of background bases $\mathbf{W_{bg}}$ can not result in accurate modeling of arbitrary background appearances, due to a lack of training samples and insufficient number of bases. However, including $\mathbf{W_{bg}}$ considerably reduces the effect of falsely combining pose bases $\mathbf{W_{pose}}$ for modeling the background. While the background bases $\mathbf{W_{bg}}$ can only result in an approximate modeling of background appearances, the pose bases $\mathbf{W_{pose}}$ are sufficient for providing an accurate modeling of poses. That said, we did not find a significant increase in pose estimation when using background images for training that correspond to the same class of images. While using the same class of background images certainly leads to a more accurate reconstrunction of the background, it seems to be rather unimportant for pose estimation given that $\mathbf{W_{bg}}$ are extracted from a broad range of images.

The resulting coefficients $\mathbf{H_{pose}}$ can now be compared against example poses or pose primitives. Pose primitives are clustered from a set of training sequences and their corresponding coefficients $\mathbf{H_{pose}}$. For clustering, we use the Euclidean distance and k-means clustering.

5 Experimental Results

In a series of experiments, we used the Weizmann Institute action-data set [4] for benchmarking the proposed recognition approach. The data set contains 10 different actions performed by 9 subjects, see also Figure 8. For the training phase, we also acquired each sequence without the background. We measured the average precision of leave one subject out cross validation series. Figure 9 shows the actual input data we used for evaluation. Figure 9(a) shows one original image frame, Figure 9(b) corresponding background subtracted image. Figure 9(c)-9(e) show centered images that were used as an input to the proposed approach. For these, we extracted a 54×90 pixel wide region around the center location of each subject. Figure 9(d) shows a *clean* human pose that was used for estimating pose basis Vectors $\mathbf{W}_{\text{pose}}$, and Figure 9(e) a human pose including background that served as an input for testing only.

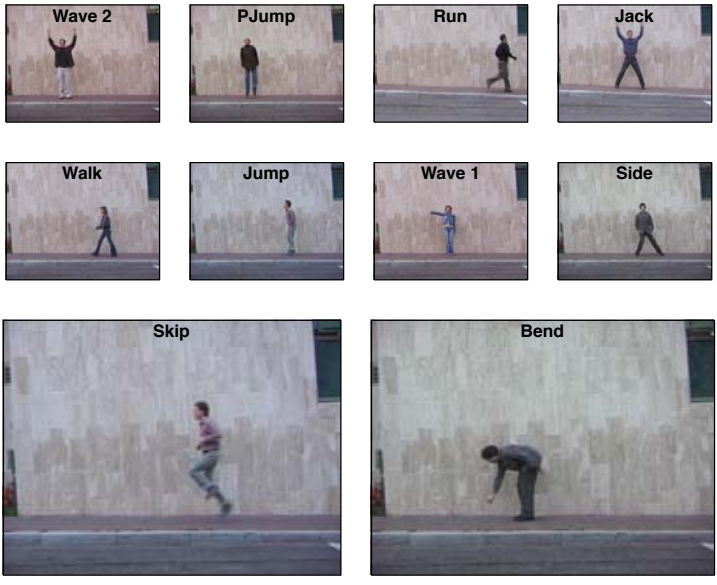


Fig. 8. The Weizmann Institute data set consists of 10 actions performed by 9 subjects. The videos are filmed from the same view. Most difficulties in analyzing this data set are from different appearances, clothing and size of people, and inner-class variations in action execution. Moreover, the movement appearances of certain actions, e.g. walking, running, and jumping on one leg, are quite similar.

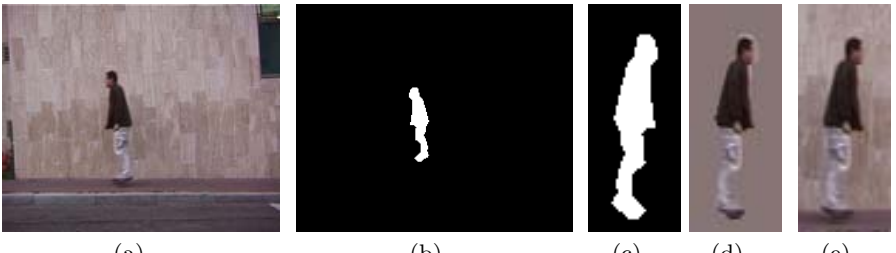


Fig. 9. Examples of training and test data used

Note that for evaluation of action recognition in still images, we also used the average precision among all test sequences. While we do not achieve real-time computation yet, the approach is reasonably fast. Dependent on the resolution, and including a HOG based approach for person detection, we achieve 2-4 frames per second in a basic MATLAB implementation.

As mentioned before, the proposed approach is not dependent on the lower level representation of poses. Still, it is interesting to explore which representations are suitable. We used three different descriptors for representing human

Fig. 10. For estimating background basis vector W_{bg} we used a common image database consisting of approx. 500 random images downloaded from the web. The Figures shows a few exemplary images.

poses. These are **(i)** conventional silhouette images (we also reduced silhouettes vector dimension by application of NMF), **(ii)** the proposed HOG based descriptor including NMF mapping using background and pose basis vectors, and **(iii)** the same NMF/HOG representation without the background basis vectors (a representation more similar to [1]). For estimating background NMF basis vector $\mathbf{W}_{bg}$ we used a standard set of about 500 random images, see also Figure 10. The data set does not show any humans and was used in other recent contributions [6].

Overall, the introduced approach achieves high precision. We were able of classifying up to approx. 95% of complete sequences correctly using a leave one out classification procedure. For recognizing action classes from only a single frame, we reached approx. 85%. However, the recognition results are influenced by the low-level pose representation, the used histogram distance measure, and the the number of pose primitives. Also the n-gram degree shows to influence the overall results.

Simple silhouettes derived by background subtraction usually gave the best results, Figure 12(b). As mentioned before, silhouettes ignore variations in subjects clothing and are independent of background clutter. While HOG descriptors also offer a convenient representation of poses, they are still influenced by clothing and background clutter. The proposed NMF basis representation partly alleviates the effects of background clutter and leads to a clear improvement over purely pose based NMF reconstruction.

Regarding histogram distance measures for classifying pose primitive histograms, we compared L_1-distance, L_2-distance, KL-divergence, χ^2 similarity, and the Bhattacharyya distance. While we already explained the suitability of KL-divergence in Section 3, it showed that especially for classification based on only a single observed frame, the χ^2 distance shows slightly superior performance. Also see Figure 12 for detailed results on complete sequences, and Figure 13 for results classifying single frames. As expected, the L_1 and L_2 distance are not suitable for classifying still images as they tend to punish empty histogram bins more severely.

The experiments showed that the number of pose primitives plays an important role for recognition. Following the interpretation as pictographs, we require at least one pose primitive per action class. Since the pose primitives are

Table 1. Comparison of different approaches evaluated on the Weizmann Institute action recognition benchmark set [4]. Note that the cited papers all use slightly different evaluation schemes with variations in image sequence lengths and separation of test and training set. The most common evaluation method is leave one out cross validation and recognition of complete sequences.

Methods	(%) sequences	(%) still images
No background subtraction and applicable to single frames		
This paper	93	70
Niebles et al. [23]	72.8	55.0
Weinland et al. [36]	93.6	-
Ferrari at al. [7]	88.0	-
Thurau et al. [34]	94.40	70.4
Background subtracted images or use of motion information		
This paper	95	84
Blank et al. [4]	99.61	-
Ikizler et al. [15]	100.00	-
Weinland et al. [36]	100.00	-
Jhuang et al. [16]	98.8	-
Schindler et al. [27]	100	-
Ali et al. [2]	89.70	-
Thurau et al. [33]	99.10	

clustered in an unsupervised manner, the actual required amount tends to be higher. Selecting suitable pose primitives for each action class by a human expert might lead to a more optimized representation. However, above a certain value (about 40-60 pose primitives), we usually could not spot a further increase in classification precision, where substantially higher numbers lead to overfitting. Interestingly, the additional information gained by classifying complete sequences requires a smaller amount of pose primitives. For single frame recognition, increasing the number of pose primitives generally shows to have a greater impact on recognition rates. The primitives tend to get more exclusive to a particular action class. For example, if 10 action classes are expressed by a distribution over 10 pose primitives (a rather low number), we have an average of one pose primitive exclusively assigned to a particular action class. It showed that the it requires at least 40-50 pose primitives for decent precision, which corresponds to 4-5 pose primitives (on average) which are exclusive to one particular action class. Note that it is at least questionable if a human observer would be able of classifying all single-frames correctly, since there are a lot of ambiguities. For instance, a person standing in an upright position is a very common pose which can be found among most action classes.

Comparison of uni-gram and bi-gram subsequences generally gave slightly better results for bi-grams. However, the used test data might not fully reflect the advantages gained from taking the local temporal context into account. There are simply not many ambiguities that arise from local subsequences. Consider,

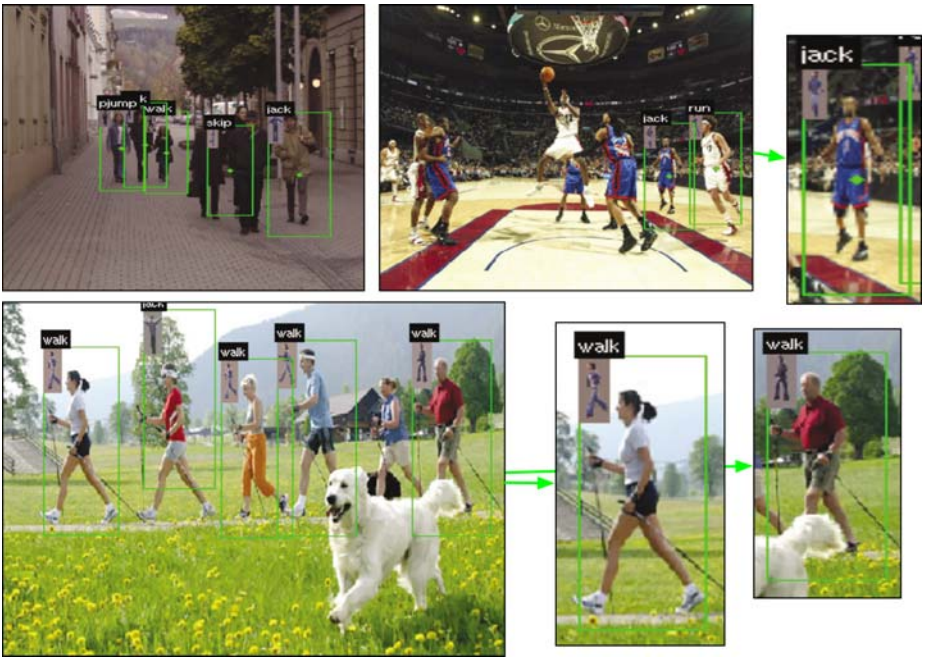
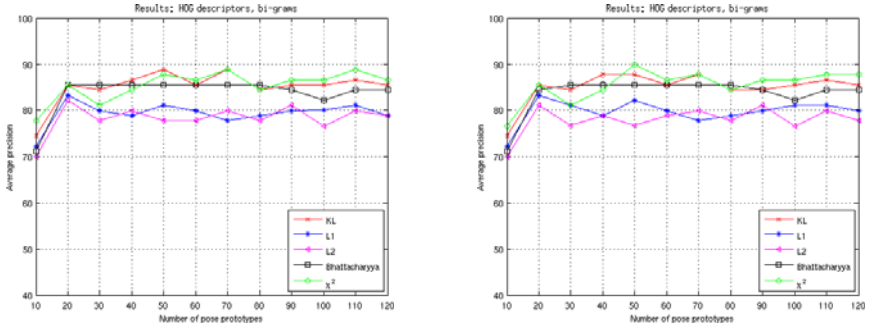


Fig. 11. Preliminary experimental results on images downloaded from the Internet

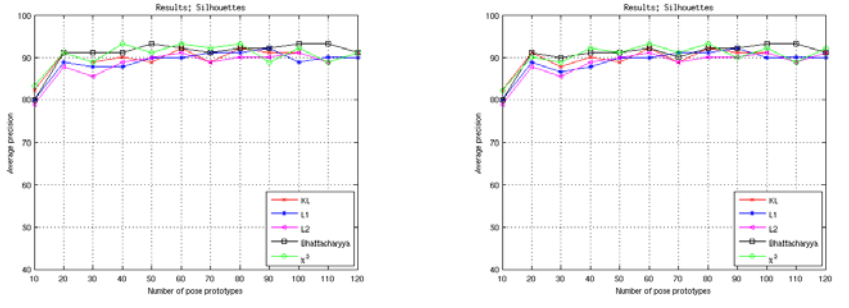
for instance, a person's forward and backward arm movement. In order to discriminate these two classes, we definitely have to take the local temporal context of pose sequences into account. Otherwise, on a strictly per frame analysis, the underlying poses are identical.

In Table 1, we compare various recent results on the used benchmark data. Besides [23,7], the approach presented is arguably the most flexible, since it can be applied to still images and, using the HOG based descriptors, does not require background subtraction or similar techniques (note that background subtraction is still used during training). Also note that the low-level features used for representing poses are of great importance not only for the presented approach. Most approaches achieving above 90% recognition rates are using silhouettes and thus ignoring variations in peoples' clothing. In [33,34] we already used a similar approach to the one presented here, however, we did not model single class histograms and the evaluation scheme varied which explains the slightly higher precision.

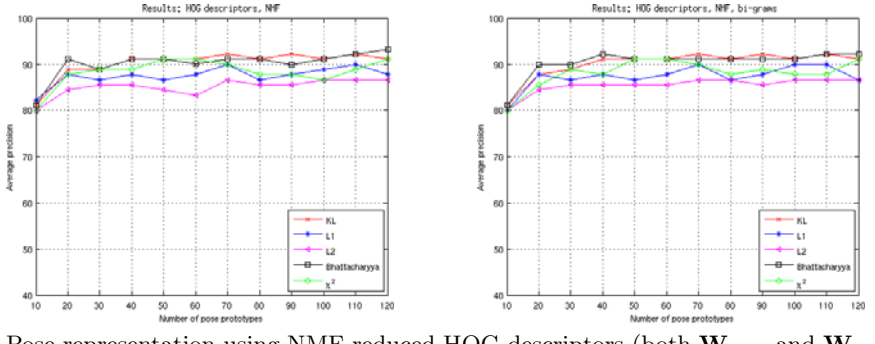
Additionally, we present early results on images downloaded from the Internet in Figure 11. Here, we used a HOG/NMF based representation of poses. Especially for heavily articulated/occluded poses where no similar pose can be found in the training set, the detection also showed shortcomings (e.g., the jumping basketball player in the upper-middle image). Other representations for poses, e.g., pictorial structures, might overcome problems resulting from changing viewpoints. For detecting people, we used a simple likelihood ratio based on HOG



(a) Pose representation using NMF reduced HOG descriptors (only $\mathbf{W}_{pose}$ is used, $\mathbf{W}_{bg}$ is ignored)

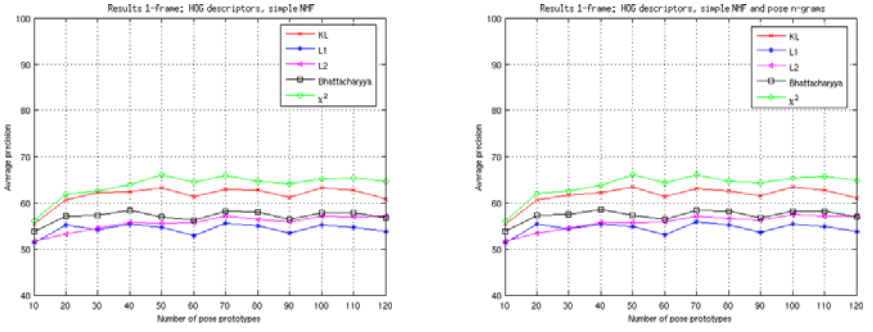


(b) Pose representation using NMF reduced silhouettes

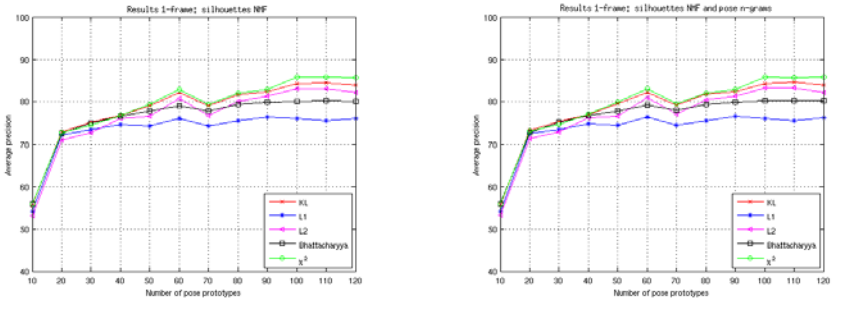


(c) Pose representation using NMF reduced HOG descriptors (both $\mathbf{W}_{pose}$ and $\mathbf{W}_{bg}$ are used)

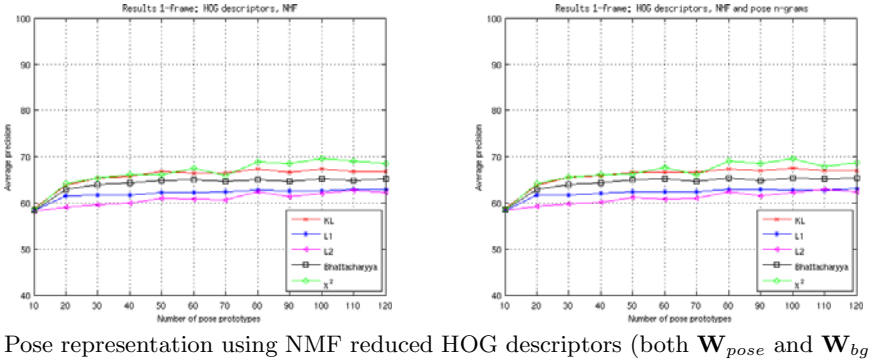
Fig. 12. Average precision for classifying the test data set using full length test sequence using a leave-one out procedure. Here, we varied the number of pose primitives, histogram distance measures, low-level pose representations, and the use of n-grams. Figure 12(a) shows results for using HOG-descriptors and a simplified NMF mapping using only W_{pose}. Figure 12(c) shows results for using HOG-descriptors and the proposed NMF mapping including W_{pose}, W_{bg}. Figure 12(b) shows results for using conventional silhouettes representations. For each Figure, the left diagram corresponds to a single pose representation, the right diagram shows results for using a *bi*-gram representation.



(a) Pose representation using NMF reduced HOG descriptors (only $\mathbf{W}_{pose}$ is used, $\mathbf{W}_{bg}$ is ignored)



(b) Pose representation using NMF reduced silhouettes



(c) Pose representation using NMF reduced HOG descriptors (both $\mathbf{W}_{pose}$ and $\mathbf{W}_{bg}$ are used)

Fig. 13. Average precision for classifying the test data on a *per-frame basis*. We measured the average correctly classified frames per sequence, this alleviates the influence of longer lasting sequences. We evaluated the data for a changing number of pose primitives and for various distance measures. It can be seen that overall the χ^2 distance gives the best results, closely followed by KL-divergence. Interestingly, for silhouettes the recognition rates can go up to 86%. Thus, on average 8 out of 10 frames contain enough information for successfully recognizing action classes. However, taking background clutter and variations in clothing into account 13(c), the classification rates decrease to approx. 70%.

descriptor basis encodings (see also [34]). Although the first results seem promising, it is yet to explore how well the proposed approach transfers to larger image sets. For further experiments, the training data should also cover a broader range of action classes. Here, we used the Weizmann Institute data set for training, thus only covering the in Figure 8 introduced action classes.

6 Conclusion

We presented a strictly pose based approach for action recognition. The approach builds on the idea that distinctive poses already contain sufficient information for recognizing action classes. We interpret an action class as a distribution over a set of learned pose primitives. This representation does not model a sequential ordering of poses and is, therefore, independent of the actual length of a novel query pose sequences. Rather than taking still images as a special case of action recognition, which is often done in literature, it emerges as a natural use case of our approach. Additionally, we manage certain ambiguities that can result from single frame analysis by making use of the local context of poses. The local context is included using n-grams in a similar fashion as they are used in statistical language models.

For handling practical problems, where we can not access silhouette images using background subtraction, we elaborated a novel pose descriptor that extends a conventional Histogram of Oriented Gradient descriptor. We base the descriptor on a non-negative matrix factorization based reconstruction of gradient histograms. The main idea is that different kinds of learned building blocks, which can correspond to limbs or body parts or background, can be efficiently used to estimate the pose of a human in front of a cluttered image.

The success of the conducted experiments indicate that the pose of a human already contains sufficient information about the underlying action. While we believe that additional information gained by dynamic features (or by taking the global sequential ordering of poses into account) could result in a better precision for activity recognition, it is interesting to see that we can neglect motion features to a certain extent. Effectively, we could show that recognizing actions in single images and videos can be handled using the same methodology.

The presented action representation is inspired by pictographs which are met in our everyday life. However, we can not yet express a complex action class using only one particular (very expressive or unique) pose, up to now we are limited to distributions over a set of a few poses. Future research should continue on exploring these most efficient and simple action class descriptors and possibly make them directly usable for visual action recognition. Not only do they provide a means for describing action classes, they could also allow efficient video synopsis, i.e. describing long lasting activities using one expressive pictograph, or partially observed image labeling, i.e. labeling of extracted pictographs by a human expert.

Acknowledgments

Christian Thurau was supported by a grant from the European Community under the EST Marie Curie Project VISIONTRAIN MRTN-CT-2004-005439. Václav Hlaváč was supported by the Czech Ministry of Education project MSM6840770038 and by the European Commission project ICT-215078 DIPLECS.

References

1. Agarwal, A., Triggs, B.: A Local Basis Representation for Estimating Human Pose from Cluttered Images. In: Narayanan, P.J., Nayar, S.K., Shum, H.-Y. (eds.) ACCV 2006. LNCS, vol. 3851, pp. 50–59. Springer, Heidelberg (2006)
2. Ali, S., Basharat, A., Shah, M.: Chaotic Invariants for Human Action Recognition. In: ICCV 2007 (2007)
3. Bissacco, A., Yang, M.H., Soatto, S.: Detecting Humans via Their Pose. In: NIPS 2006 (2006)
4. Blank, M., Gorelick, L., Shechtman, E., Irani, M., Basri, R.: Actions as Space-Time Shapes. In: ICCV 2005 (2005)
5. Carlsson, S., Sullivan, J.: Action recognition by shape matching to key frames. In: Workshop on Models versus Exemplars in Computer Vision (2001)
6. Dalal, N., Triggs, B.: Histograms of Oriented Gradients for Human Detection. In: CVPR 2005 (2005)
7. Ferrari, V., Marin, M., Zisserman, A.: Progressive Search Space Reduction for Human Pose Estimation. In: CVPR 2008 (2008)
8. Flash, T., Hochner, B.: Motor primitives in vertebrates and invertebrates. Current Opinion in Neurobiology 15(6), 660–666 (2005)
9. Fod, A., Matarić, M., Jenkins, O.: Automated Derivation of Primitives for Movement Classification. Autonomous Robots 12(1), 39–54 (2002)
10. Ghahramani, Z.: Building blocks of movement. Nature 407, 682–683 (2000)
11. Goldenberg, R., Kimmel, R., Rivlin, E., Rudzsky, M.: Behavior classification by eigendecomposition of periodic motions. Pattern Recognition 38, 1033–1043 (2005)
12. Guerra-Filho, G., Aloimonos, Y.: A Sensory-Motor Language for Human Activity Understanding. In: 6th IEEE-RAS International Conference on Humanoid Robots (HUMANOIDS 2006), pp. 69–75 (2006)
13. Hamid, R., Johnson, A., Batta, S., Bobick, A., Isbell, C., Coleman, G.: Detection and Explanation of Anomalous Activities: Representing Activities as Bags of Event n-Grams. In: CVPR 2005 (2005)
14. Hoyer, P.O.: Non-negative Matrix Factorization with sparseness constraints. Journal of Machine Learning Research 5, 1457–1469 (2004)
15. Ikizler, N., Duygulu, P.: Human Action Recognition Using Distribution of Oriented Rectangular Patches. In: Human Motion ICCV 2007 (2007)
16. Jhuang, H., Serre, T., Wolf, L., Poggio, T.: A Biologically Inspired System for Action Recognition. In: ICCV 2007 (2007)
17. Laptev, I., Perez, P.: Retrieving actions in movies. In: ICCV 2007 (2007)
18. Lee, D.D., Seung, H.S.: Learning the parts of objects by non-negative matrix factorization. Nature 401(6755), 788–799 (1999)
19. Lee, D.D., Seung, H.S.: Algorithms for non-negative matrix factorizationi. In: NIPS 2001 (2001)

20. Lu, W.L., Little, J.J.: Simultaneous Tracking and Action Recognition using the PCA-HOG Descriptor. In: CRV 2006 (2006)
21. Moeslund, T., Fihl, P., Holte, M.: Action Recognition using Motion Primitives. In: Danish Conference on Pattern Recognition and Image Analysis (2006)
22. Moeslund, T., Reng, L., Granum, E.: Finding Motion Primitives in Human Body Gestures. In: Wolfmann, J., Cohen, G. (eds.) Coding Theory 1988. LNCS (LNAI), vol. 388, pp. 133–144. Springer, Heidelberg (1989)
23. Niebles, J.C., Fei-Fei, L.: A Hierarchical Model of Shape and Appearance for Human Action Classification. In: CVPR 2007 (2007)
24. Niebles, J.C., Wang, H., Fei-Fei, L.: Unsupervised Learning of Human Action Categories Using Spatial-Temporal Words. In: BMVC 2006 (2006)
25. Ogale, A.S., Karapurkar, A., Aloimonos, Y.: View-invariant modeling and recognition of human actions using grammars. In: ICCV Workshop on Dynamical Vision (2005)
26. Schack, T., Mechsner, F.: Representation of motor skills in human long-term memory. Neuroscience Letters 391, 77–81 (2006)
27. Schindler, K., van Gool, L.: Action Snippets: How many frames does human action recognition require? In: CVPR 2008 (2008)
28. Schroff, F., Criminisi, A., Zisserman, A.: Single-Histogram Class Models for Image Segmentation. In: Kalra, P.K., Peleg, S. (eds.) ICVGIP 2006. LNCS, vol. 4338, pp. 82–93. Springer, Heidelberg (2006)
29. Sivic, J., Russell, B.C., Efros, A.A., Zisserman, A., Freeman, W.T.: Discovering object categories in image collections. In: Proceedings of the International Conference on Computer Vision (2005)
30. Thoroughman, K., Shadmehr, R.: Learning of action through adaptive combination of motor primitives. Nature 407, 742–747 (2000)
31. Thurau, C.: Behavior Histograms for Action Recognition and Human Detection. In: Human Motion ICCV 2007 (2007)
32. Thurau, C., Bauckhage, C., Sagerer, G.: Synthesizing Movements for Computer Game Characters. In: Rasmussen, C.E., Bülthoff, H.H., Schölkopf, B., Giese, M.A. (eds.) DAGM 2004. LNCS, vol. 3175, pp. 179–186. Springer, Heidelberg (2004)
33. Thurau, C., Hlaváč, V.: n-grams of action primitives for recognizing human behavior. In: Kropatsch, W.G., Kampel, M., Hanbury, A. (eds.) CAIP 2007. LNCS, vol. 4673. Springer, Heidelberg (2007)
34. Thurau, C., Hlaváč, V.: Pose primitive based human action recognition in videos or still images. In: International Conference on Computer Vision and Pattern Recognition (CVPR 2008). IEEE, Los Alamitos (2008)
35. Vangeneugden, J., Pollick, F., Vogels, R.: Functional differentiation of macaque visual temporal cortical neurons using a parameterized action space. J. Vis. 8(6), 232–232 (2008), http://journalofvision.org/8/6/232/
36. Weiland, D., Boyer, E.: Action Recognition using Exemplar-based Embedding. In: CVPR 2008 (2008)
37. Wolpert, D.M., Ghahramani, Z., Flanagan, J.R.: Perspectives and problems in motor learning. TRENDS in Cognitive Sciences 5(11), 487–494 (2001)
38. Zhang, L., Wu, B., Nevatia, R.: Detection and Tracking of Multiple Humans with Extensive Pose Articulation. In: ICCV 2007 (2007)

View-Based Approaches to Spatial Representation in Human Vision

Andrew Glennerster[1], Miles E. Hansard[2], and Andrew W. Fitzgibbon[3]

[1] University of Reading, Reading, UK
a.glennerster@reading.ac.uk
http://www.personal.rdg.ac.uk/~sxs05ag/

[2] INRIA Rhône-Alpes, Montbonnot, France
[3] Microsoft Research, Cambridge, UK

Abstract. In an immersive virtual environment, observers fail to notice the expansion of a room around them and consequently make gross errors when comparing the size of objects. This result is difficult to explain if the visual system continuously generates a 3-D model of the scene based on known baseline information from interocular separation or proprioception as the observer walks. An alternative is that observers use view-based methods to guide their actions and to represent the spatial layout of the scene. In this case, they may have an expectation of the images they will receive but be insensitive to the rate at which images arrive as they walk. We describe the way in which the eye movement strategy of animals simplifies motion processing if their goal is to move towards a desired image and discuss dorsal and ventral stream processing of moving images in that context. Although many questions about view-based approaches to scene representation remain unanswered, the solutions are likely to be highly relevant to understanding biological 3-D vision.

1 Is Optic Flow Used for 3-D Reconstruction in Human Vision?

Optic flow, or motion parallax, provides animals with information about their own movement and the 3-D structure of the scene around them. Throughout evolution, motion is likely to have been more important for recovering information about scene structure than binocular stereopsis, which is predominantly used by hunting animals who are required to remain still. The discrimination of 3-D structure using motion parallax signals is known to be highly sensitive, almost as sensitive as for binocular stereopsis [1,2]. Motion parallax has also been shown to provide an estimate of the viewing distance and metric shape of objects, when combined with proprioceptive information about the distance the observer has travelled [3,4]. An important and unsolved challenge, however, is to understand how this information is combined into a consistent representation as the observer moves through a scene.

D. Cremers et al. (Eds.): Visual Motion Analysis, LNCS 5604, pp. 193–208, 2009.

In computer vision, the problem is essentially solved. Photogrammetry is the process of using optic flow to recover information about the path of the camera, its internal parameters such as focal length and the 3-D structure of the scene [5,6]. For most static scenes the process is robust, reliable and geometrically accurate. It does not suffer from the systematic spatial distortions that are known to afflict human observers in judgements of metric shape and distance [7,8,9,10,11]. Most approaches to visual navigation in robotics are based on a partial reconstruction of the 3D environment, again using photogrammetric principles [12,13,14].

It is questionable whether human vision is designed to achieve anything like photogrammetric reconstruction. Gibson [15], for example, argued strongly against the idea that vision required explicit internal representation, but did not provide a clear description of an alternative [16]. Similar arguments are still popular [17] but are no more computationally precise. In fact, although there have been some attempts at generating non-metric representations for navigation in robots (see Section 3), there is as yet no well-developed rival to 3-D reconstruction as a model for representing the spatial layout of a scene, either in computer vision or models of human vision.

1.1 An Expanding Room

In our Virtual Reality Laboratory, we have developed a paradigm in which information about the distance of objects from vergence, motion parallax and proprioception conflict with the assumption of scene stability. Observers wear a head mounted display that has a wide-field of view and high resolution. In an immersive virtual environment, they make judgements about the relative size or distance of objects, allowing us to probe the representation of space generated by the visual system and to assess the way in which different cues are combined. Figure 1 illustrates one experiment [18]. As the observer moves from one side of the room to the other, the virtual room expands around him/her. Remarkably, almost all observers fail to notice any change in size of the room even though the room expands by a factor of 4. Even with feedback about the true size of objects in the room, they fail to learn to use veridical stereo and motion parallax cues appropriately [18,19].

Our data suggest a process of scene representation that is very different from photogrammetry. For example, Figure 1b shows data for 5 human observers who were asked to judge the size of a cube seen on the right of the room (when the room is large) compared to a reference cube shown on the left of the room (when the room is small). The reference cube disappears when the observer walks across the room, so they must remember its size. Participants make a binary, forced choice judgement on each trial about the relative size of the reference and comparison cubes. Over many trials, it is possible to deduce the size of the comparison cube that they would match with the reference cube. As Figure 1b shows, this size varies with the viewing distance of the comparison cube: at 6m participants choose a match that is almost four times as large as the reference cube, i.e. almost equal to the expansion of the room, whereas at 1.5m they choose a more veridical match.

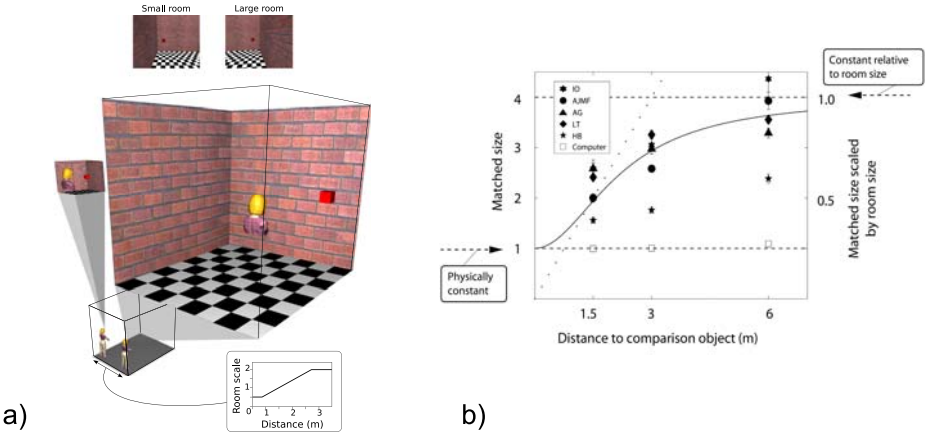


Fig. 1. An expanding virtual room. a) As the observer walks from one side of the room to the other the virtual scene around them expands. The centre of expansion is mid-way between the eyes so the view from that point is unaffected. This stimulus separates purely visual cues to the 3-D structure of the scene from others, such as proprioception. b) Data from an experiment [18] in which participants matched the size of a cube visible when the room was large with one visible only when the room was small. The correct matched size ratio is 1 while a size ratio of 4 is predicted if participants judged the cube size relative to the room or texture elements. Data for 5 participants is shown (solid symbols) and for a simulation using a 3-D reconstruction package (open symbols, see text). The curve shows the predictions of a cue combination model. The dotted line shows the predicted responses if participants matched the retinal size of the cubes. Figure reproduced, with permission, from [18] © Elsevier.

Figure 1b also shows the predicted responses from a 3-D reconstruction package (*Boujou* [20]) supplied with the same images that a participant would see as they carried out the task. In this case, we provided one veridical baseline measure, which sets the scale of the reconstruction, so the height of the simulation data on the y-axis is not informative. The important point, though, is that there is no variation of predicted matching size with viewing distance for the 3-D reconstruction, unlike the pattern of the human data.

In some ways, it is not difficult to explain the human observers' behaviour. The fitted curve in Figure 1b illustrates the predictions of a cue combination model. Matches for a distant comparison object are close to that predicted by a texture-based strategy (judging the cube size relative to the bricks, for example), since here stereo and motion parallax cues to distance are less reliable and hence are given a lower weight [21]. At close distances, stereo and motion parallax are more reliable (they support lower depth discrimination thresholds when compared to texture cues) and hence have a greater effect on matched sizes [22,18,21]. However, combining cues in this way is very specific to the task. It does not imply that there should be related distortions in other tasks, for example those requiring the comparison of two simultaneously visible objects.

The difficulty with a task-specific explanation, such as the cue combination model in Figure 1b, is that it avoids the question of scene representation in the human visual system. It is all very well to say that the world can act as 'an outside memory', and be accessed by looking at the relevant parts of the scene when necessary [23,17]. There must be a representation of some form to allow the observer to turn their head and eyes or walk to the right place to find the information. The representation may not be a 3-D reconstruction, but it must nevertheless be structured. There is as yet no satisfactory hypothesis about what that representation might be but the proposal in this paper is that it may have similarities to 'view-' or 'aspect-graphs', which are introduced in the next section.

1.2 Moving between Views

An alternative to 3-D reconstruction has been described in relation to navigation, in which a robot or animal stores views or 'snapshots' of a scene and records something about the motor output required to move between one view and the next, without integrating this information into a Cartesian map. The snapshots are the nodes in a 'view graph' and the movements are the edges [24,25] (see Section 3). One possible explanation of people's perceptions in the expanding room is that, as they move across a room, observers generally have an expectation of the images that they will receive: the fact that they do not notice that they are in an expanding room implies that they are relatively insensitive to the rate at which those images arrive as they walk.

This is illustrated in Figure 2a, where the grey plane represents a manifold of all the images that could be obtained by a person walking around the room. Each point on the plane represents one image and neighbouring points represent images visible from neighbouring vantage points. The dotted line shows, schematically, the set of images that a monocular observer might receive as they walk from the left to the right of the room. Potentially, that eye could receive exactly the same set of images whether the room was static or expanding: there is no way to tell from the monocular images alone. However, as Figure 2a shows, the rate at which new images arrive for each step the observer takes is different in the expanding room. On the left, where the room is small, new images arrive rapidly, whereas on the right they do so more slowly. Proprioceptive signals from the muscles provide information about the distance walked and these should be sufficient to tell observers about the size of the room. But in the expanding room, observers seem to disregard this information when it conflicts with the assumption that the room is stable, at least in determining their subjective experience of the room.

One attractive aspect of the view graph idea is its close similarity to known biological mechanisms. Moving from image to image or, more generally, from sensory state to sensory state, with an expectation of the sensory input that will be received as a consequence of each movement, is a familiar biological operation associated with well established circuitry, notably in the cerebellum [26,27,28]. The same cannot be said of operations required for general 3-D coordinate

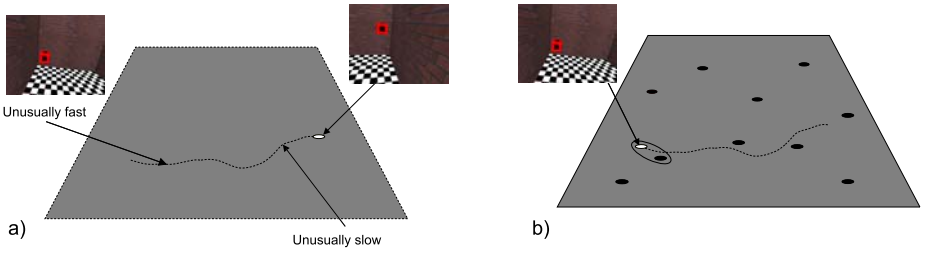


Fig. 2. Possible images in an expanding room. a) The grey plane symbolises a manifold of views of the room shown in figure 1. Each point corresponds to a single view and neighbouring points correspond to views from neighbouring vantage points. The current view is indicated by the white dot. The dashed line indicates the set of views that a hypothetical observer receives as they walk across the room. The manifold of views for the expanding room is identical to that for a static room: the only difference is that when the observer is on the right hand side of the room, where it is large, views change at an unusually slow rate as the observer walks while on the left side of the room, where the room is small, the views change unusually fast. b) The grey plane now symbolises a manifold of potential sensory (or sensory+motivational) states with the current state indicated by the white dot. The state includes visual and proprioceptive input. The black dots indicate stored states. Both the current and the stored states can be described by points in the same high dimensional space.

transformations, particularly translations of the origin. There are no detailed suggestions about how such operations might be achieved in the brain.

On the other hand, Figure 2b illustrates operations that are known to take place in the brain and it looks very similar to Figure 2a. Now the grey plane illustrates a high dimensional space of potential sensory (and motivational) states. Each black dot represents a potential sensory+motivational context that will lead to a particular motor output. The white dot represents the current sensory+motivational context, which is recognised as most similar to one of the stored contexts. The match leads to an action, whose consequence is a new sensory+motivational context, and so the movement progresses.

From this perspective, one can suggest a potential explanation for the fact that observers see the expanding room as static. Even though the proprioceptive signals are different in the two cases, the powerful visual feedback is exactly what is expected in a static room and hence, overall, the sensory context is sufficiently similar for it to be recognised as the expected context rather than an alternative one. In terms of Figure 2b, the path through sensory+motivational space as observers walk across the room is very similar, despite the different proprioceptive inputs, and hence their subjective experience is too.

It is not always true that the proprioceptive input is ignored. When observers move from a small room to a clearly separate large room their size judgements are much better than in the expanding room [18]. In that case, when confronted with a new room of unknown size, there is no strong expectation about the sensory feedback they will receive and so the proprioceptive information becomes

decisive. Even in the expanding room, proprioceptive (and stereo) input contributes to some judgements while not affecting observers' subjective impression of the room size [18,19,21]. Overall, the results in the expanding room suggest a representation that is looser, less explicit and more task-dependent than a 3-D reconstruction.

2 Biological Input and Output of Optic Flow Processing

For photogrammetry, the goal is clear: compute the 3-D structure of the scene and the path of the camera through it. Corresponding features are identified across frames, then the most likely 3-D scene, camera movement and camera intrinsic parameters (such as focal length) are computed given the image motion of the features. The camera motion is described in the same 3-D coordinate frame as the location of the points. The process can be written as:

$$\mathbf{x}_j^i = P^i \mathbf{X}_j \tag{1}$$

where $\mathbf{x}$ gives the image coordinates of points in the input frames, $\mathbf{X}$ describes the 3-D locations of the points in the scene and P is the projection matrix of the camera. P includes the intrinsic parameters of the camera, which remain constant, and the extrinsic parameters (camera pose) which are the only things that change from frame to frame. Equation 1 applies to the j^{th} point in the i^{th} frame.

In computer vision applications, the camera is generally free to translate and rotate in any direction. This is not true in biology. The 6 degrees of freedom of camera motion are essentially reduced to 3. Animals maintain fixation on an object as they move and are obliged to do so for at least 150 ms (this the minimum saccadic latency) [29]. Although the eye can rotate around 3 axes and the optic centre can translate freely in 3 dimensions, the rotation and translation of the eye are tightly coupled so that for each translation of the optic centre there is a compensatory rotation of the eye. Two dimensions of rotation maintain fixation while the third restricts the torsion of the eye with respect to the scene. This means that as a monocular observer moves through a static environment, maintaining fixation on an object, there is only one image that the eye can receive for each location of the optic centre.

A similar pattern of eye movements predominates throughout the animal kingdom and must clearly carry an evolutionary benefit. Land [30] has documented eye movement strategies in many species. Animals fixate while moving, then they make a 'saccade' or rapid rotation of the eyes and fixate a new target as they continue to move. They do this even when their eyes are fixed within the head, as is the case in many insects. In a rare condition in which the eye muscles become paralysed, patients' eyes are fixed with respect to their head [31]. In these cases, the head makes a rapid rotation between periods of fixation, so their visual diet is similar to that of a person with freely moving eyes. And when the eye and head are fixed with respect to the body, for example in a hover fly, the whole body makes saccades.

Why should animals do this? The constraint that eye rotation and translation are tightly linked does reduce the number of free parameters and so can help in the estimation of camera motion and scene reconstruction [32,33,34,35]. However, the rationale we describe here is different. If the goal is to navigate across a manifold of images, as we discussed in relation to Figure 2, then the restrictive pattern of eye movements that animals adopt makes sense. We illustrate this claim in the next sections by considering the role of the dorsal and ventral streams of visual processing during a simple set of movements.

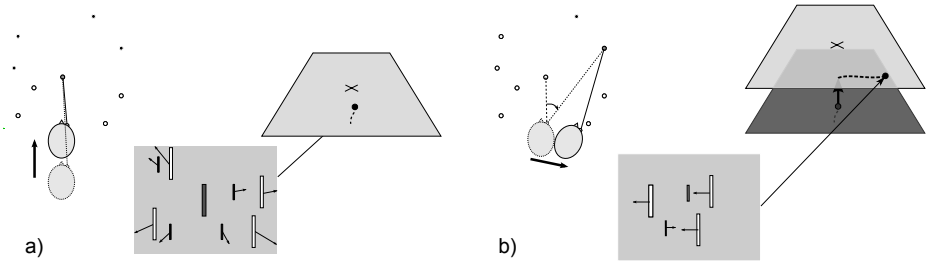


Fig. 3. Moving across a surface of images. a) An observer is shown in plan view moving towards a fixated object. The pattern of flow on the retina is illustrated and, on the right, the movement of the current image across a surface of images, similar to that shown in Figure 2a. b) The observer makes a saccade to a new fixation point which is illustrated by the current image jumping to a new surface of images (upper surface). The observer then moves laterally while maintaining fixation on the same object. Objects with a crossed disparity (open symbols) move in one direction in the image while those with an uncrossed disparity (filled symbols) move in the opposite direction. The cross marks the view that would be obtained if the viewer were at the location of the fixated object.

2.1 Dorsal Stream

Figure 3 shows a simple sequence of head and eye movements to illustrate a possible role of the dorsal stream in controlling head movements. In Figure 3a, the observer moves towards a fixated target. A fast, tight loop from retina to eye muscles keeps the eye fixating as the observer moves. This circuit can even bypass cortical motion processing [36]. The obligation to maintain fixation imposes a tight constraint on the type of image changes that can arise. The plane shown on the right in Figure 3a represents the set or manifold of images that can be obtained when the eye fixates a particular object. Each point on the manifold represents one image. As the observer approaches the fixated object, the current image (black dot) is shown moving towards the view that would be obtained if the observer were at the location of the object (shown by the cross). The retinal flow created by this movement is approximately radial expansion outwards from the fovea. There are many neurons in MSTd (the dorsal part of the medial superior temporal cortex, in the dorsal stream) that are sensitive to flow of this type [37,38].

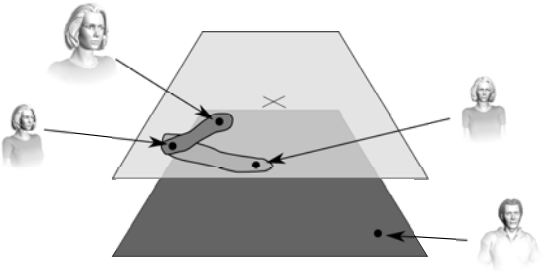


Fig. 4. Ventral stream neuron receptive fields. Neurons in the ventral stream respond selectively to certain objects or complex features while showing considerable invariance to size or viewing direction. The dark grey region illustrates a 'receptive field' of a size-invariant neuron, i.e. the set of images to which it might respond. The lighter grey 'receptive field' shows the set of images to which a view-invariant neuron might respond. As in Figure 3, each surface of images comprises the images that can be obtained by the observer while they fixate on a particular object – in this case a woman and a man.

Figure 3b illustrates in the same way a saccade followed by a lateral movement of the observer. Fixating a new object moves the current image to a new surface of images. The observer then moves laterally, staying the same distance from the fixation point. This produces a quite different pattern of retinal flow in which objects that are closer than the fixation point (shown as open symbols) move one way on the retina while more distant objects move in the opposite direction (closed symbols). The distinction between objects in front of and behind the fixation point can be signalled by disparity (crossed versus uncrossed) and indeed this seems to be a trick the visual system uses to disambiguate the flow. The same area that contains neurons responsive to expansion, MSTd, also contains neurons responsive to one direction of motion when the stimulus disparity is crossed and the *opposite* direction of motion when the stimulus disparity is uncrossed [39]. These neurons are ideally suited to signalling the type of flow that occurs when the observer moves his or her head laterally while fixating an object.

The two components of observer motion shown in Figure 3a and b can be detected independently. The neurons sensitive to forward motion can be used as a signal of progress towards the goal in Figure 3a with the neurons sensitive to lateral motion signalling error; the role of these types of neurons can then be reversed to control the lateral motion shown in Figure 3b. Greater calibration would be required to move on a path between these extremes, but the principle of using these two components remains [40] and the simplicity of the control strategy relies on the restriction that the observer fixates on a point as he or she moves. This is a quite different hypothesis about the role of dorsal stream neurons from the idea that they contribute to a computation of scene structure and observer heading in the same coordinate frame [38].

2.2 Ventral Stream

The ventral stream of visual processing has a complementary role. Neurons in this part of the visual pathway provide, as far as possible, a constant signal despite the observer's head movements, indicating *which* object the observer is looking at (i.e. which surface the current image is on) in contrast to the dorsal stream which signals how the observer is moving in relation to the fixated object, independent of the identity of that object.

As illustrated in Figure 4, it is possible to show on the surface of images the 'receptive fields' of different types of ventral stream neurons. Rather than a receptive field on the retina, here we mean the set of images to which this neuron would respond. The dark grey patch on the left side of Figure 4 shows a hypothetical 'receptive field' for a size-independent cell [41]: it would respond to an object from a range of viewing distances. The overlapping lighter grey patch shows the receptive field of a view-independent cell [42], which responds to the same object seen from a range of different angles.

Combining both of these types of invariance, one could generate a receptive field that covered the entire surface of images. In other words the neuron would respond to the view of a particular object from a wide range of angles and distances. Neurons with this behaviour have been reported in the hippocampus of primates [43]. The hippocampus, which lies at the end of the ventral stream, contains a large auto-association network [44] which has the effect of providing a constant output despite moderate changes in the input. Thus, there are mechanisms throughout the ventral stream that encourage a stable output to be maintained for the period of fixation. In terms of the surface of images, the argument here is that the ventral stream is designed to determine which surface of images contains the current image and the dorsal stream is designed to determine how the current image is moving across it.

So far, we have only considered the example of a short sequence of head and eye movements. In section 3, we discuss view graphs and how these can provide an extended spatial representation of observer location.

3 Representing Observer Location

A view graph (or aspect graph) representation consists of discrete reference views which form the nodes of the graph. In some simple examples, these have been the views at junctions in a maze [45], while in more realistic cases they have been the views at various locations in an open environment [46]. The edges of the graph are the movements that take the observer from one view to the next. A view graph is not a continuous representation of space (in contrast to the coordinate frame of $\mathbf{X}$ in Equation 1) but, when the observer is not at a location from which a reference view was taken, it is still possible to determine which of the reference views is most similar to the current view. This allows space to be divided up into regions, as the example below illustrates.

Figure 5 illustrates this principle for a 2-dimensional world. Suppose the room contained 3 objects as shown and that a 'view' consisted of a list of the angles

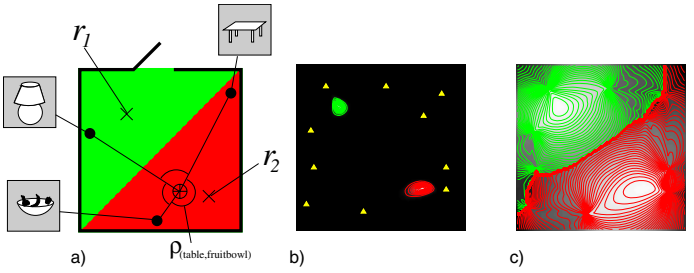


Fig. 5. Reference locations. a) Two reference locations r_1 and r_2 are shown and three objects, with the angle subtended between pairs of objects shown, e.g. $\rho_{(\text{table,fruitbowl})}$. The green and red areas indicate a possible division of the room into locations where the best answer to the question 'Where am I?' would be r_1 and r_2 respectively. b) Plan view of a scene in which the location of objects are marked by yellow triangles. Contour plots show the likelihoods that the observer is at the two candidate locations, colour coded according to which of the two likelihoods is greater. c) Same as b) but showing log likelihoods.

subtended at the optic centre by each pair of objects. Figure 5a shows two reference locations, r_1 and r_2, and the angles subtended between each of the three pairings of objects at a third, arbitrary location. The colours show a hypothetical division of the room into regions where the view is most like that at r_1 (green) and most like the view at r_2 (red). Figure 5b and c illustrate a specific example of this principle. There are now more than three objects, whose locations are shown by yellow triangles, but still two reference locations. The plots show the likelihoods that the view at each location is a noisy version of the view r_1 or r_2, computed as follows.

Scene features at positions (x_p, y_p), $p = 1, \ldots, N$, are imaged from viewpoint (x, y), and represented by the visual angles:

$$\boldsymbol{\rho}_{p,q}(x,y) = \big(\rho_{1,q}(x,y), \ldots, \rho_{p,q}(x,y), \ldots, \rho_{N,q}(x,y)\big), \quad q = 1, \ldots, N, \tag{2}$$

where $\rho_{p,q}$ is the angle between points (x_p, y_p) and (x_q, y_q). In order to avoid the use of a distinguished reference point, we compute all N^2 possible angles from each viewpoint. As well as general views $\boldsymbol{\rho}(x, y)$, we have R different *reference views*,

$$\big\{\boldsymbol{\rho}_{p,q}(x_1, y_1), \ldots, \boldsymbol{\rho}_{p,q}(x_r, y_r), \ldots, \boldsymbol{\rho}_{p,q}(x_R, y_R)\big\}, \quad q = 1, \ldots, N, \tag{3}$$

taken from distinct locations (x_r, y_r). Each reference view $\boldsymbol{\rho}(x_r, y_r)$ is accompanied by a corresponding list of variances, $\boldsymbol{\sigma}^2(x_r, y_r)$:

$$\boldsymbol{\rho}_{p,q}(x_r, y_r) = \big(\rho_{1,q}(x_r, y_r), \ldots, \rho_{p,q}(x_r, y_r), \ldots, \rho_{N,q}(x_r, y_r)\big), \quad q = 1, \ldots, N, \tag{4}$$

$$\boldsymbol{\sigma}^2_{p,q}(x_r, y_r) = \big(\sigma^2_{1,q}(x_r, y_r), \ldots, \sigma^2_{p,q}(x_r, y_r), \ldots, \sigma^2_{N,q}(x_r, y_r)\big), \quad q = 1, \ldots, N. \tag{5}$$

In our simulations, we take $\sigma_{p,q}^2 = 1$ in all cases. The fit of the r^{th} reference view to the visual angles obtained at observer position (x, y) is defined as the squared difference between $\boldsymbol{\rho}(x, y)$ and $\boldsymbol{\rho}(x_r, y_r)$, summed over all angles,

$$E_r(x, y) = \sum_{q=1}^{N} \sum_{p=1}^{N} \frac{1}{\sigma_{p,q}^2(x_r, y_r)} \big(\rho_{p,q}(x, y) - \rho_{p,q}(x_r, y_r)\big)^2. \tag{6}$$

We use E_r to compute the likelihood of the current view $\boldsymbol{\rho}(x, y)$, under the hypothesis that the viewpoint coincides with that of model view $\boldsymbol{\rho}(x_r, y_r)$. Specifically, we represent the likelihood as

$$\mathsf{L}\big(\boldsymbol{\rho}(x, y) \,|\, \boldsymbol{\rho}(x_r, y_r)\big) = e^{-E_r(x,y)}, \tag{7}$$

which is proportional to the probability of the r^{th} location-hypothesis:

$$\mathsf{P}\big(x = x_r, y = y_r \,|\, \boldsymbol{\rho}(x, y)\big) \propto e^{-E_r(x,y)}. \tag{8}$$

The normalizing constant is obtained by integrating P over all viewpoints, (x, y). The figures plot the maximum likelihood at each point (x, y), colour-coded by the reference location r for which $\mathsf{L}\big(\boldsymbol{\rho}(x, y) \,|\, \boldsymbol{\rho}(x_r, y_r)\big)$ is maximum.

If the visual system represents places using angles subtended between objects in a similar way, then fixating on an object as the observer moves is a sensible strategy. It allows changes in angles (at least, those with respect to the fixated object) to be monitored easily as the observer moves. Figure 6 illustrates the idea. If the optic centre, O, of the eye/camera is at the location marked by the blue cross and the fixation point is F, then the eccentricity of a point, P, imaged on peripheral retinal is the same as the angle (ρ) between that point and the fixation point subtended at the optic centre. If the observer maintains fixation on the same point, then change in eccentricity of the peripheral object signals change in the angle ρ, Figure 6c. This retinal flow, $\Delta\rho$, is directly relevant to computing changes in likelihood that the current location is r_1 or r_2, as Figures 6b and d illustrate.

The example in Figures 5 and 6 is limited to control of movement between two reference locations. Figure 7 shows a larger set of reference locations chosen by a robot, equipped with an omnidirectional camera, at which it took 'snapshots' or reference views as it explored a real scene [47]. New snapshots were constrained to differ (above a criterion level) from those already stored. In general, views are connected as the nodes in a view graph, where the edges connect 'neighbouring' views. New edges were formed by the robot as it explored: as it left one reference location it compared its current view to all other 'snapshots' it had stored and then used a homing strategy [48,49] to approach the reference location with the most similar view. The edges in the view graph contained no information about the direction or distance between two vertices, only that they were neighbours. This experiment illustrates how the idea of reference views can be extended over a large (potentially limitless) spatial range. Equally, of course, the resolution of the representation can be increased within a particular region of space by including more reference locations there. One can imagine many situations in

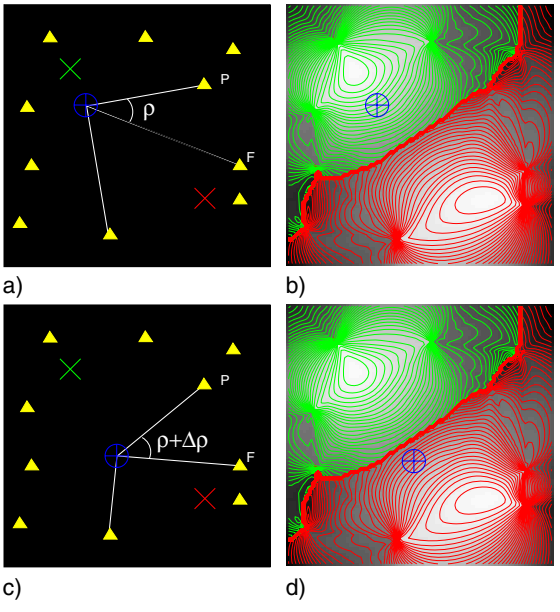


Fig. 6. Retinal flow and fixation. a) The observer's location is marked by the blue target and the fixated object is shown by a yellow triangle, F. Two other objects imaged on peripheral retina are shown, one of which (P) is at an eccentricity of ρ. Movement of the observer causes a change in the angles between pairs of objects subtended at the optic centre (c and d). Because the observer maintains fixation as he or she moves, retinal flow provides a straight-forward record of the changes in these angles with respect to the fixation point. For example, as shown in c), the change in the angle ρ to $\rho + \Delta\rho$ results in flow, $\Delta\rho$, at the point on the retina where the object P is imaged. The change in ρ (and other angles) with respect to the fixation point can be used to determine whether the current view is becoming less like that from reference location r_1 and more like that from r_2 (b and d).

which fine distinctions between different observer locations are important in one part of a scene but less important in others.

View graphs, then, provide a way of extending the ideas about control of observer location to a wider region of space. Section 2 dealt only with observer movement relative to the fixation point, and saccades. Now these can be seen as examples of moving over a small part of an infinitely extendible view graph. However, view graphs do not directly answer the question raised in section 1 about scene representation. For example, in Figure 7, how might the location of the objects (rather than the robot) be represented? The grey object appears in many views, including the snapshots from all the reference locations shown by filled symbols. Is there a sensible coordinate frame in which to unite all the information about an object's location, other than a 3-D world-based frame? Current implementations do not do so and it is not immediately obvious how it should be done.

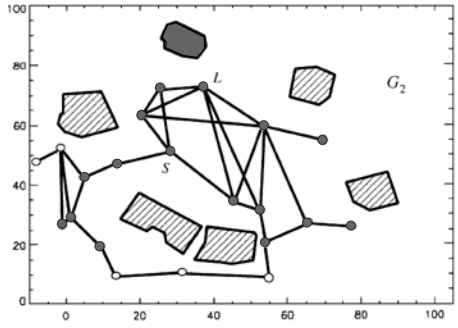


Fig. 7. View graph of a scene. Circles show the locations of reference views (or 'snapshots'), which are the nodes in the view graph (adapted from a figure in Franz et al [47], with permission). The filled circles indicate the reference views that include an image of the grey object. It is not clear how best to define the location of objects in the view graph.

One might argue that in simple animals the distinction does not exist: for an ant to know where an object is located is much the same as knowing how to arrive at it. The same may be true for humans if an object is a long way away (like the Eiffel tower). In that case, the represented location of an object can be a node in the view graph. However, this only avoids the question of uniting estimates of object location across views, which is relevant at closer distances for actions such as reaching or pointing. Humans can point to an object that is not currently visible and can do so in a way that accounts for their rotations *and* translations since they last saw it. They can also direct their hand to an object that they have seen in peripheral vision but never fixated, even when they make a saccade before pointing at it (although they make systematic error in this case [50]). These capacities, however poorly executed, are evidence of a representation of object location that is maintained across head and eye movements. View graphs, or some similar view-based approach, must clearly be able to support similar behaviour if they are to become candidate models of human spatial representation.

4 Conclusion

We have argued that the human perception of space in an expanding virtual room may be understood in terms of a representation like a view graph, or a manifold of images, where the observer has an expectation of the images they will receive as they move across the room but are insensitive to the discrepancy between visual and proprioceptive feedback. It is clear that view graph models are currently inadequate in many ways but we have argued that an explanation along these lines is more likely to explain human perception in the expanding room than one based on 3-D reconstruction. It is clear from the expanding room

and many other perceptual phenomena that the development and successful implementation of non-Cartesian, non-metric representations will be of great relevance to the challenge of understanding human 3-D vision.

Acknowledgements

Funded by the Wellcome Trust. We are grateful to Bruce Cumming, Andrew Parker and Hanspeter Mallot for helpful discussions.

References

1. Rogers, B.J., Graham, M.: Similarities between motion parallax and stereopsis in human depth perception. Vision Research 22, 261–270 (1982)
2. Bradshaw, M.F., Rogers, B.J.: The interaction of binocular disparity and motion parallax in the computation of depth. Vision Research 36, 3457–3768 (1996)
3. Bradshaw, M.F., Parton, A.D., Eagle, R.A.: The interaction of binocular disparity and motion parallax in deptermining perceived depth and perceived size. Perception 27, 1317–1331 (1998)
4. Bradshaw, M.F., Parton, A.D., Glennerster, A.: The task-dependent use of binocular disparity and motion parallax information. Vision Research 40, 3725–3734 (2000)
5. Fitzgibbon, A.W., Zisserman, A.: Automatic camera recovery for closed or open image sequences. In: Burkhardt, H.-J., Neumann, B. (eds.) ECCV 1998. LNCS, vol. 1406, pp. 311–326. Springer, Heidelberg (1998)
6. Hartley, R., Zisserman, A.: Multiple view geometry in computer vision. Cambridge University Press, Cambridge (2000)
7. Foley, J.M.: Binocular distance perception. Psychological Review 87, 411–433 (1980)
8. Gogel, W.C.: A theory of phenomenal geometry and its applications. Perception and Psychophysics 48, 105–123 (1990)
9. Johnston, E.B.: Systematic distortions of shape from stereopsis. Vision Research 31, 1351–1360 (1991)
10. Tittle, J.S., Todd, J.T., Perotti, V.J., Norman, J.F.: A hierarchical analysis of alternative representations in the perception of 3-D structure from motion and stereopsis. J. Exp. Psych.: Human Perception and Performance 21, 663–678 (1995)
11. Glennerster, A., Rogers, B.J., Bradshaw, M.F.: Stereoscopic depth constancy depends on the subject's task. Vision Research 36, 3441–3456 (1996)
12. Basri, R., Rivlin, E., Shimshoni, I.: Visual homing: Surfing on the epipoles. International Journal of Computer Vision 33, 117–137 (1999)
13. Davison, A.J.: Real-time simultaneous localisation and mapping with a single camera. In: Proceedings. Ninth IEEE International Conference on computer vision, pp. 1403–1410 (2003)
14. Newman, P., Ho, K.L.: SLAM-loop closing with visually salient features. In: Proceedings IEEE International Conference on Robotics and Automation, pp. 635–642 (2005)
15. Gibson, J.J.: The ecological approach to visual perception. Houghton Mifflin, Boston (1979)

16. Ullman, S.: Against direct perception. Behavioural and Brain Sciences 3, 373–415 (1980)
17. O'Regan, J.K., Noë, A.: A sensori-motor account of vision and visual consciousness. Behavioural and Brain Sciences 24, 939–1031 (2001)
18. Glennerster, A., Tcheang, L., Gilson, S.J., Fitzgibbon, A.W., Parker, A.J.: Humans ignore motion and stereo cues in favour of a fictional stable world. Current Biology 16, 428–443 (2006)
19. Rauschecker, A.M., Solomon, S.G., Glennerster, A.: Stereo and motion parallax cues in human 3d vision: Can they vanish without trace? Journal of Vision 6, 1471–1485 (2006)
20. 2d3 Ltd. Boujou 2 (2003), http://www.2d3.com
21. Svarverud, E., Gilson, S.J., Glennerster, A.: Absolute and relative cues for location investigated using immersive virtual reality. In: Vision Sciences Society, Naples, Fl (2008)
22. Ernst, M.O., Banks, M.S.: Humans integrate visual and haptic information in a statistically optimal fashion. Nature 415, 429–433 (2002)
23. O'Regan, J.K.: Solving the real mysteries of visual perception: The world as an outside memory. Canadian Journal of Psychology 46, 461–468 (1992)
24. Schölkopf, B., Mallot, H.A.: View–based cognitive mapping and path planning. Adaptive Behavior 3, 311–348 (1995)
25. Koenderink, J.J., van Doorn, A.J.: The internal representation of solid shape with respect to vision. Biological Cybernetics 32, 211–216 (1979)
26. Marr, D.: A theory of cerebellar cortex. J. Physiol (Lond.) 202, 437–470 (1969)
27. Albus, J.: A theory of cerebellar function. Mathematical Biosciences 10, 25–61 (1971)
28. Miall, R.C., Weir, D.J., Wolpert, D.M., Stein, J.F.: Is the cerebellum a Smith predictor? Journal of Motor Behaviour 25, 203–216 (1993)
29. Carpenter, R.H.S.: Movements of the eyes. Pion, London (1988)
30. Land, M.F.: Why animals move their eyes. Journal of Comparative Physiology A: Neuroethology, Sensory, Neural, and Behavioral Physiology 185, 1432–1351 (1999)
31. Gilchrist, I.D., Brown, V., Findlay, J.M.: Saccades without eye movements. Nature 390, 130–131 (1997)
32. Aloimonos, Y., Weiss, I., Bandopadhay, A.: Active vision. In: Proceedings of the International Conference on Computer Vision, London, UK, June 8–11, pp. 35–54 (1987)
33. Bandopadhay, A., Ballard, D.: Egomotion perception using visual tracking. Computational Intelligence 7, 39–47 (1990)
34. Sandini, G., Tistarelli, M.: Active tracking strategy for monocular depth inference over multiple frames. IEEE Transactions on Pattern Analysis and Machine Intelligence 12, 13–27 (1990)
35. Daniilidis, K.: Fixation simplifies 3D motion estimation. Computer Vision and Image Understanding 68, 158–169 (1997)
36. Cohen, B., Reisine, H., Yokota, J.-I., Raphan, T.: The nucleus of the optic tract: Its function in gaze stabilization and control of visual-vestibular interaction. Annals of the New York Academy of Sciences 656, 277–296 (1992)
37. Saito, H., Yukie, M., Tanaka, K., Hikosaka, K., Fukada, Y., Iwai, E.: Integration of direction signals of image motion in the superior temporal sulcus of the macaque monkey. J. Neuroscience 6, 145–157 (1986)
38. Perrone, J.A., Stone, L.S.: A model of self-motion estimation within primate extrastriate visual cortex. Vision Research 34, 2917–2938 (1994)

39. Roy, J.P., Wurtz, R.H.: The role of disparity-sensitive cortical neurons in signalling the direction of self-motion. Nature 348, 160–162 (1990)
40. Glennerster, A., Hansard, M.E., Fitzgibbon, A.W.: Fixation could simplify, not complicate, the interpretation of retinal flow. Vision Research 41, 815–834 (2001)
41. Rolls, E.T., Bayliss, G.C.: Size and contrast have only small effects on the responses to faces of neurons in the cortex of the superior temporal sulcus of the monkey. Experimental Brain Research 65, 38–48 (1986)
42. Booth, M.C.A., Rolls, E.T.: View-invariant representations of familiar objects by neurons in the inferior temporal cortex. Cerebral Cortex 8, 510–525 (1998)
43. Georges-Francois, P., Rolls, E.T., Robertson, R.G.: Spatial view cells in the primate hippocampus: allocentric view not head direction or eye position or place. Cerebral Cortex 9, 197–212 (1999)
44. Treves, A., Rolls, E.T.: Computational analysis of the role of the hippocampus in memory. Hippocampus 4, 374–391 (2004)
45. Gillner, S., Mallot, H.A.: Navigation and acquisition of spatial knowledge in a virtual maze. Journal of Cognitive Neuroscience 10, 445–463 (1998)
46. Franz, M.O., Mallot, H.A.: Biomimetic robot navigation. Robotics and Autonomous Systems 30, 133–153 (2000)
47. Franz, M.O., Schölkopf, B., Mallot, H.A., Bülthoff, H.H.: Learning view graphs for robot navigation. Autonomous Robots 5, 111–125 (1998)
48. Cartwright, B.A., Collett, T.S.: Landmark learning in bees: experiments and models. Journal of Comparative Physiology 151, 521–543 (1983)
49. Hong, J., Tan, X., Pinette, B., Weiss, R., Riseman, E.: Image-based homing. IEEE Control Systems Magazine 12(1), 38–45 (1992)
50. Henriques, D.Y.P., Klier, E.M., Smith, M.A., Lowy, D., Crawford, J.D.: Gaze-centered remapping of remembered visual space in an open-loop pointing task. Journal of Neuroscience 18, 1583–1594 (1998)

Combination of Geometrical and Statistical Methods for Visual Navigation of Autonomous Robots

Naoya Ohnishi[1,⋆] and Atsushi Imiya[2,⋆⋆]

[1] School of Science and Technology, Chiba University, Japan
[2] Institute of Media and Information Technology, Chiba University, Japan
Yayoicho 1-33, Inage-ku, Chiba, 263-8522, Japan
imiya@faculty.chiba-u.jp

Abstract. For visual navigation of an autonomous robot, detection of collision-free direction from an image/ image sequence captured by imaging systems mounted on the robot is a fundamental task. This collision free direction provides the next view to direct attention for computing the next collision free direction. Therefore, the robot requires a cyclic mechanism directing attention to the view and computing the collision free direction from that view. We combine a geometric method for free space detection and a statistical method for visual navigation of the mobile robot. Firstly, we deal with a random-sampling-based method for the detection of free space. Secondly, we deal with a statistical method for the computation of the collision avoiding direction. The robot finds free space using the visual potential defined from a series of views captured by a monocular camera system mounted on the robot to observe the view in front of the robot, We examine the statistical property of the gradient field of the visual potential. We show that the principal component of the gradient of the visual potential field yields the attention direction of the mobile robot for collision avoidance. Some experimental results of navigating the mobile robot in synthetic and real environments are presented.

1 Introduction

For visual control of an autonomous robot, detecting the direction that allows it to avoid collision with obstacles is a fundamental task. The robot mounted with a monocular and/or multiple vision system computes control features to decide navigation direction from the view in front of the robot. For the visually controlled robot, this collision-avoidance direction provides the direction to the next attention field from which the robot computes the next control feature. In this paper, we introduce a method for the computation of the collision-avoidance

⋆ Present Address: Power and Industry Systems Research and Development Centers, Power Systems Company, TOSHIBA, Fuchu, Tokyo, Japan.

⋆⋆ All academic correspondences are addressed to A.Imiya.

D. Cremers et al. (Eds.): Visual Motion Analysis, LNCS 5604, pp. 209–234, 2009.

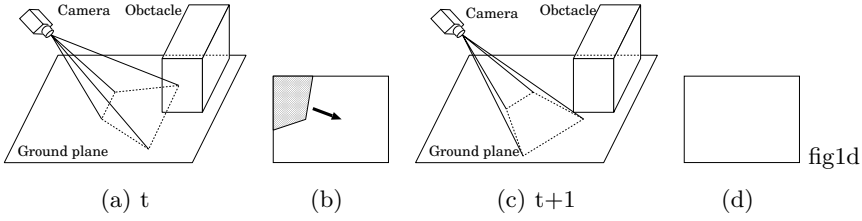


Fig. 1. Next attention view. (a) Distribution of the obstacle and camera at time t. (b) Captured image at the time t. The black region expresses the obstacle. The arrow is the direction to the next view. (c) Distribution of the obstacle and camera at time $t+1$. (d) Captured image at the time $t+1$. The distribution of obstacle areas on the image at t defines the direction to the next view at the time $t+1$ So that, at the time $t+1$, no obstacle is captured by the camera system mounted on the robot.

direction from a series of views captured by a monocular camera system mounted on the robot to observe the view in front of the robot. Figure 1 illustrates the relation between the collision-free direction and the next view for the visual navigation.

Ohnishi and Imiya introduced the visual potential for the computation of the collision-avoidance force [16,17] using the dominant plane and the potential field yielded from the dominant plane. The dominant plane is the planar area in the robot workspace corresponding to the largest part of an image or at least the half of an image. Therefore, the dominant plane derives the collision-free region. The robot computes the navigation direction from the configuration of the collision-free region in front of the robot. We examine a statistical property of the gradient of the potential field, that is, the principal component of the gradient of the visual potential defines the collision-avoidance direction. In the accompanying papers [18,19], they have presented the statistical properties of the dominant plane, obtained using independent components. This property explains the motion detection property and the attention direction in the visual field in the human brain.

In this paper, we assume the following constraints for robot navigation.

1. The ground plane, on which the robot moves, is the planar area.
2. The camera mounted on the mobile robot is downward-looking.
3. The robot observes the world using the camera mounted on itself for navigation.
4. The camera on the robot captures a sequence of images since the robot is moving.

These assumptions are illustrated in Fig. 2 (a). Therefore, if there are no obstacles around the robot, and since the robot does not touch the obstacles, the ground plane corresponds to the dominant plane in the image observed through the camera mounted on the mobile robot. Furthermore, using images captured by the camera mounted on the robot, decides the direction to move. Since the robot moves, an imaging system mounted on the robot captures an

image sequence. Using the optical flow field computed from this image sequence, we can detect the dominant plane of each image in an image sequence [15]. This dominant plane is accepted as a free space for robot navigation [16,17]. This free space decides the navigation direction and the next view direction.

The overview of our algorithm is shown in Fig. 2 (b). The mobile robot dynamically computes the local path with the camera capturing the images. Therefore, the dominant plane determines free-space in the workspace for visual navigation of the robot. Furthermore, using the dominant plane, the robot selects a possible region for the corridor path in a robot work space.

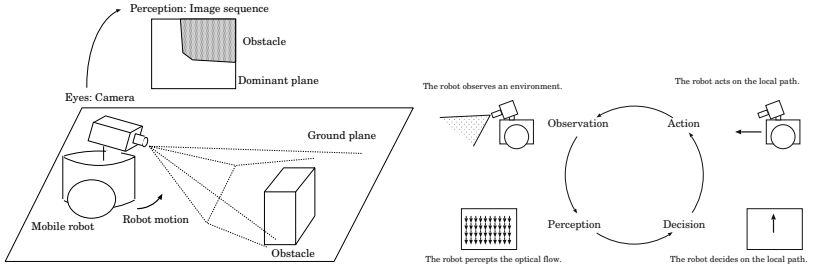


Fig. 2. Perception and cognition of motion and obstacles in a workspace by an autonomous mobile robot and Observation-Perception-Decision-Action cycle for vision based robot navigation. (a)The mobile robot has a camera, which corresponds to eyes. The robot perceives an optical flow field from ego-motion. (b)At first, the mobile robot which equipped with a camera observes an environment. Next, an optical flow field relative to the robot motion is computed from images obtained by the camera. The optical flow field is used to decide the local path. The robot moves to the direction of the computed local path.

The potential field method [9] is an established method of path planning for mobile robots. Mobile robot navigation by the potential field method using a monocular or stereo camera has also been proposed [1,12,25]. Adorini *et al* [1] used calibrated omnidirectional cameras for obstacle detection. Murray and Little [12] proposed an algorithm for autonomous exploration by a mobile robot using calibrated stereo vision. Wong and Spetsakis [25] used optical flow observed through a calibrated monocular camera. Path planning is the problem of deriving the optimal or suboptimal path from a starting point to a destination using the map of the robot workspace, which is called the configuration space, and landmarks [6,23]. Navigation is the problem of deriving the optimal direction in which to move from a sequence of snapshots of the obstacle configuration in the robot workspace [4,22]. Our method solves the navigation problem by using the visual potential field and optical flow field of the image. The optical flow field [2,8,10], is an apparent motion of the scene and a fundamental feature for understanding environment information. Since both the visual potential field and

optical flow field are apparent information on the image captured by the camera mounted on the mobile robot, our method does not require environmental maps or camera calibration.

The visual potential is the sum of the gradient field in the obstacle area, which defines the direction to the next view requiring attention for visual navigation, and the optical flow field [16]. In this paper, we use the principal component of the gradient field as the control force. For visual navigation, the robot computes the attractive force from the obstacle area to the feasible region to avoid collision with obstacles in the workspace. As this attractive force to avoid collision with the obstacle, our robot adopts the principal direction of the gradient field computed using the obstacle region in the image. Therefore, our robot uses the principal direction of the gradient field and the optical flow as the attractive force from the obstacle and the guiding force to the destination, respectively.

2 Optical Flow and Dominant Plane

In this section, we summarise the optical-flow-based dominant plane detection algorithms proposed in our previous papers [15,16,17,18]. In refs. [15] and [21], a temporal-model-based and model-based methods are proposed, and a statistical method is proposed in ref. [19].

2.1 Computation of Optical Flow

Setting $I(x, y, t)$ and $\dot{\boldsymbol{x}} = (\dot{x}, \dot{y})^\top = (u, v)^\top$ to be the time-varying gray-scale-valued image at time t and the optical flow, respectively, optical flow $\dot{\boldsymbol{x}}$ at each point $\boldsymbol{x}$ satisfies

$$I_x\dot{x} + I_y\dot{y} + I_t = 0. \tag{1}$$

The computation of $\dot{\boldsymbol{x}}$ from $I(x, y, t)$ is an ill-posed problem [8,13,10], Therefore, additional constraints are required to compute $\dot{\boldsymbol{x}}$. We use the Lucas-Kanade method with pyramids [3].

Equation (1) can be solved by assuming that the optical flow vector of pixels is constant in the neighbourhood of each pixel. We set the window size to be 5×5. Then, equation (1) is expressed as a system of linear equations,

$$I_{x\alpha\beta}u + I_{y\alpha\beta}v + I_{t\alpha\beta} = 0, \quad |\alpha| \leq 2, |\beta| \leq 2 \tag{2}$$

where $I_{x\alpha\beta}(x, y, t)$, $I_{y\alpha\beta}(x, y, t)$, and $I_{t\alpha\beta}(x, y, t)$ are the values of $I_x(x, y, t)$, $I_y(x, y, t)$, and $I_t(x, y, t)$, respectively, in the pixel $(x + \alpha, y + \beta)^\top$.

In Lucus-Kanade with pyramid transform, optical flow $\dot{\boldsymbol{x}}^i = (\dot{x}^l, \dot{y}^l)^\top$ on each layer is solved by the Lucas-Kanade method [10]. Then, these optical flow vectors at coarse resolution are propagated to the fine grid using the warp operation $w(I(x, y, t), \dot{\boldsymbol{x}}) = I(\boldsymbol{x}+\dot{\boldsymbol{x}}, t)$ and linear interpolation. In this algorithm, I_t^l stands for the pyramidal representation at the level l of image $I(x, y, t)$ at time t, that is, the pyramid representation is expressed as

$$I^{l+1}(x,y,t) = \sum_{\alpha,\beta=-1}^{1} a_{\alpha\beta} I^l(2x-\alpha, 2y-\beta, t), \tag{3}$$

where

$$a_{\alpha\beta} = \frac{1}{2^{1+|\alpha|}} \frac{1}{2^{1+|\beta|}}, \ |\alpha| \leq 1, |\beta| \leq 1, \tag{4}$$

setting $I^0(x,y,t) = I(x,y,t)$ for the original image. For the computation of $\dot{\boldsymbol{x}} = \boldsymbol{u}^L$, see appendix. Furthermore, in the algorithm 1, $\dot{\boldsymbol{x}}_t^l = (\dot{x}^l, \dot{y}^l)^\top$ is computed from $\dot{\boldsymbol{x}}_t^{l+1} = (\dot{x}^{l+1}, \dot{y}^{l+1})^\top$ as $\dot{x}^l = E(\dot{x}^{l+1})$ and $\dot{y}^l = E(\dot{y}^{l+1})$, where the operation E achieves the upsampling and linear interpolation to yield the optical flow field in the l-th layer from that in the$(l+1)$-th layer. The operation E is expressed as

$$E(\dot{x}_{mn}) = 4 \sum_{\alpha,\beta=-1}^{1} a_{\alpha\beta} \dot{x}_{\frac{m-i}{2}, \frac{n-j}{2}}, \ E(\dot{y}_{mn}) = 4 \sum_{\alpha,\beta=-1}^{1} a_{\alpha\beta} \dot{y}_{\frac{m-i}{2}, \frac{n-j}{2}}, \tag{5}$$

for both $\frac{m-i}{2}$ and $\frac{n-j}{2}$ are integers.

Algorithm 1. Optical Flow Computation by the Lucas-Kanade Method Using Pyramids

Data: I_t^l, I_{t+1}^l, $\quad 0 \leq l \leq$ the maximum of the layers;
Result: $\dot{\boldsymbol{x}}^l$;
$l :=$ *the* maximum of the layers ;
while $l \neq 0$, **do**
  $\dot{\boldsymbol{x}}^l := u(I_t^l, I_{t+1}^l)$;
  $I_{t+1}^{l-1} := w(I_{t+1}^{l-1}, \dot{\boldsymbol{x}}_t^l)$;
  $l := l - 1$;
end

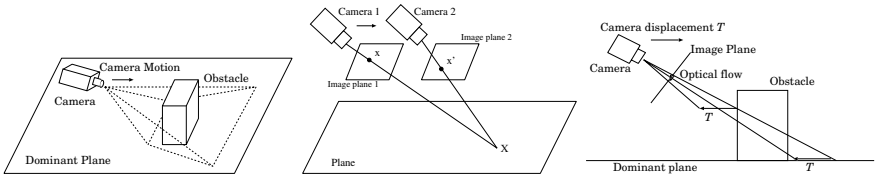
2.2 Optical Flow on Dominant Plane

Setting $\boldsymbol{H}$ to be a 3×3 matrix [7], the homography between the two images of a planar surface can be expressed as

$$\boldsymbol{\xi}' = \boldsymbol{H}\boldsymbol{\xi}, \tag{6}$$

where $\boldsymbol{\xi} = (x, y, 1)^\top$ and $\boldsymbol{\xi}' = (x', y', 1)^\top$ are homogeneous coordinates of corresponding points in two successive images. Assuming that the camera displacement is small, matrix $\boldsymbol{H}$ is approximated by affine transformations. These geometrical and mathematical assumptions are valid when the camera is mounted on a mobile robot moving on the dominant plane. Therefore, the corresponding points $\boldsymbol{x} = (x, y)^\top$ and $\boldsymbol{x}' = (x', y')^\top$ on the dominant plane are expressed as

$$\boldsymbol{x}' = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}, \tag{7}$$



(a) (b) (c)

Fig. 3. Homography transform of images caused by the translation of the camera and the difference in optical flow between the dominant plane and an obstacle. (a) shows the camera motion and (b) shows homography transform on the image plane caused by the robot motion. (c) If the camera moves distance T approximately parallel to the dominant plane, the camera observes the difference in the optical flow vectors between the dominant plane and an obstacle.

where $\boldsymbol{A}$ and $\boldsymbol{b}$ are a 2×2 affine-coefficient matrix and a 2-dimensional vector, which are approximations of $\boldsymbol{H}$.

Figure 3 shows the homography transform of images caused by the translation of the camera. (a) shows the camera motion and (b) shows the homography transform on the image plane caused by the robot motion.

2.3 Three Methods for the Dominant Plane Detection

Temporal-Model-Based Method. We can estimate the affine coefficients using the RANSAC-based algorithm [5,7,15]. Using estimated affine coefficients, we can estimate optical flow on the dominant plane $\hat{\boldsymbol{x}} = (\hat{x}, \hat{y})^\top$,

$$\hat{\boldsymbol{x}} = (\boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}) - \boldsymbol{x}, \tag{8}$$

for all points $\boldsymbol{x}$ in the image. We call $\hat{\boldsymbol{x}}$ the *planar flow*, and $\hat{\boldsymbol{x}}(x, y, t)$ the *planar flow field* at time t, which is a set of planar flow $\hat{\boldsymbol{x}}$ computed for all pixels in an image.

Algorithm 2. Affine Coefficient Estimation

repeat
  Randomly select three points from $\{\boldsymbol{x}\}$;
  Estimate $\boldsymbol{A}$ and $\boldsymbol{b}$ in Eq. (7) from $\hat{\boldsymbol{x}} = \boldsymbol{x}' - \boldsymbol{x}$;
  Compute planar flow field $\hat{\boldsymbol{x}}$ using Eq. (8);
until $\dot{\boldsymbol{x}} \neq \hat{\boldsymbol{x}}$ *for more than half of the image*;

If an obstacle exists in front of the robot, the planar flow on the image plane differs from the optical flow on the image plane, as shown in Fig. 3 (c). Since the planar flow vector $\hat{\boldsymbol{x}}$ is equal to the optical flow vector $\dot{\boldsymbol{x}}$ on the dominant plane, we use the difference between these two flows to detect the dominant plane.

We set ε to be the tolerance of the difference between the optical flow vector and the planar flow vector. Therefore, if the inequality

$$|\dot{\boldsymbol{x}} - \hat{\boldsymbol{x}}| < \varepsilon \tag{9}$$

is satisfied, we accept point $\boldsymbol{x}$ as a point on the dominant plane. Our algorithm is summarised below.

Algorithm 3. Dominant Plane Detection

Compute optical flow field $\dot{\boldsymbol{x}}(x, y, t)$ from two successive images;
repeat
  Compute optical flow field $\dot{\boldsymbol{x}}(x, y, t)$ from two successive images;
  Compute affine coefficients in Eq. (7) from three randomly selected points;
  Estimate planar flow field $\hat{\boldsymbol{x}}(x, y, t)$ from affine coefficients;
  Match the computed optical flow field $\dot{\boldsymbol{x}}(x, y, t)$ and estimated planar flow field $\hat{\boldsymbol{x}}(x, y, t)$ using Eq. (9);
  if $|\dot{\boldsymbol{x}} - \hat{\boldsymbol{x}}| < \varepsilon$ **then** assign these points as the dominant plane;
  if *the dominant plane occupies more than half of the image* **then** output the dominant plane $d(x, y, t)$ as a binary image;
until *predetermined number of times*;

Then, the image is represented as a binary image by the dominant plane region and the obstacle region. Therefore, we set $d(x, y, t)$ to be the dominant plane as

$$d(x, y, t) = \begin{cases} 255, & \text{if } \boldsymbol{x} \text{ is on the dominant plane} \\ 0, & \text{if } \boldsymbol{x} \text{ is on the obstacle area.} \end{cases} \tag{10}$$

We call $d(x, y, t)$ the dominant plane map. Then, we have the following definition.

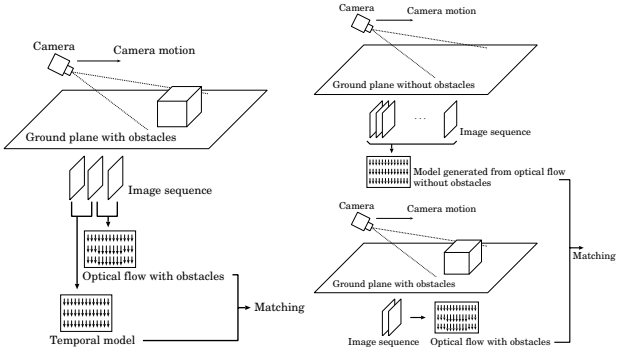
Definition 1. *(Obstacle Map) We call the compliment of the dominant plane with respect to a scene the obstacle region.*

Model-Based Method. For the comparison of algorithms in refs. [14] and [21], we briefly summarise the model-based method for the detection of the ground plane [11,21]. The model-based method requires learning and detection phases.

In the learning phase, the robot equipped with the camera system moves in a workspace without the presence of obstacles and captures an image sequence. Setting $I_m(x, y, t)$ to be the image without obstacles at time index t to create a model vector of the camera motion, the optical flow $\boldsymbol{u}_m = (\dot{x}_m, \dot{y}_m)^\top$ can be expressed as

$$\frac{\partial I_m}{\partial x}\dot{x}_m + \frac{\partial I_m}{\partial y}\dot{y}_m + \frac{\partial I_m}{\partial t} = 0. \tag{11}$$

Setting $\boldsymbol{u}_m(1)$ to be the estimated optical flow between the successive images $I_m(x, y, 0)$ and $I_m(x, y, 1)$, $\boldsymbol{u}_m(1)$ is the model vector of the camera motion in



(a)Temporal-model-based method (b) Model-based method

Fig. 4. Two methods for the visual navigation. (a) Temporal-model-based navigation using optical flow. This method enables the detection of obstacles without a model. The first two images from three successive images are used for the construction of temporal models. (b) Model-based navigation using optical flow. The upper part is the learning phase for the construction of the model of a robot motion and an environment. During navigation, obstacles are detected by matching the optical flow of this model and the observed ones.

the workspace without obstacles. The method yields the identical model vector

$$\boldsymbol{u}_m(1) = \boldsymbol{u}_m(2) = \ldots = \boldsymbol{u}_m(t), \tag{12}$$

where $\boldsymbol{u}_m = (\dot{x}_m(t), \dot{y}_m(t))^\top$

In the detection phase, the robot moves in a workspace with obstacles at the same speed as that in the learning phase. Set $I(x, y, t)$ and $\boldsymbol{u}(t)$ to be the image with obstacles at time t captured by the mobile robot and the optical flow between the successive image $I(x, y, t-1)$ and $I(x, y, t)$. The ground plane with $\boldsymbol{u}_m(t)$ is distinguished from the other regions according to the optical flow $\boldsymbol{u}(t)$. Therefore, the ground plane is detected as the region that satisfies

$$|\boldsymbol{u}_m - \boldsymbol{u}(t)| < \varepsilon, \tag{13}$$

where ε is the tolerance of the difference between the optical flow vector and the model vector, the model flow vector field $\hat{\boldsymbol{u}}$ is computed as the average,

$$\boldsymbol{u}_m = \frac{1}{n} \sum_{\tau=0}^{n} \boldsymbol{u}_m(\tau). \tag{14}$$

Fig. 4 compares temporal-model-based and model-based navigation.

Statistical Method. As previously introduced [20,26], we accept the assumption that the optical flow fields observed by the moving camera are linear combinations of the optical flow fields of the dominant plane and the obstacles [19].

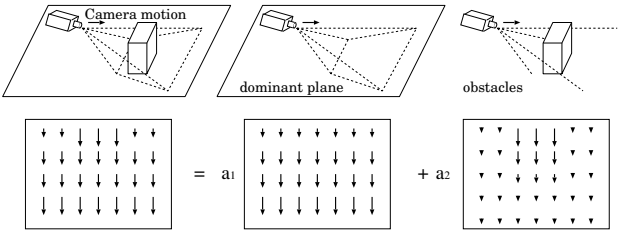


Fig. 5. Linear combination of optical flow field in the scene. The optical flow field (bottom right) is expressed as a linear combination of those shown at bottom middle and bottom right. a_1 and a_2 are mixture coefficients.

This assumption is numerically and geometrically acceptable if motion displacement is small compared with the size of obstacles, as shown in the numerical experiment. Therefore, the independent component component (ICA in abbreviation) is suitable for the separation of the optical flow field into independent flow components.

ICA requires at least two input signals for separation of an input signal into two independent components. We use the optical flow field $\dot{\boldsymbol{x}}$ and planar flow field $\hat{\boldsymbol{x}}$ for the inputs to ICA. Setting $\boldsymbol{v}_\alpha$ and $\boldsymbol{v}_\beta$ to be the output optical flow fields of ICA, we have the equations

$$\begin{cases} \boldsymbol{u} = a_{11}\boldsymbol{v}_\alpha + a_{12}\boldsymbol{v}_\beta, \\ \hat{\boldsymbol{x}} = a_{21}\boldsymbol{v}_\alpha + a_{22}\boldsymbol{v}_\beta, \end{cases} \tag{15}$$

where a_{ij} is the mixture coefficient. ICA estimates the independent optical flow fields $\boldsymbol{v}_\alpha$ and $\boldsymbol{v}_\beta$ from optical flow field $\boldsymbol{u}$ and planar flow field $\hat{\boldsymbol{u}}$. The motions of the dominant plane and obstacles in the images are different, and the dominant-plane motion is smooth on the images compared with the obstacle motion, as shown in Fig. 6. Consequently, the output signal of the obstacle motion has larger variance than the output signal of the dominant-plane motion. We are required to determine whether components have optical flow of the dominant plane or of obstacle areas. The difference in the variances of the norms of vectors $\boldsymbol{v}_\alpha$ and $\boldsymbol{v}_\beta$ enables us to order independent components [19]. Therefore, if the variances σ_α^2 and σ_β^2, of $|\boldsymbol{v}_\alpha|$ and $|\boldsymbol{v}_\beta|$, respectively satisfy the inequality $\sigma_\beta^2 > \sigma_\alpha^2$, we accept the output flow field $\boldsymbol{v}_\alpha$ as the the optical flow field on the dominant plane. Figure 7 is the diagram of the ICA-based statistic method for the free space detection.

Since the planar flow field is subtracted from the optical flow field including the obstacle motion, the speed $|\boldsymbol{v}(\boldsymbol{x})|$ of the point $\boldsymbol{x}$ is constant on the dominant plane. However, the length of the flow vector $|\boldsymbol{v}(\boldsymbol{x})|$ is not constant, because of the form of Eq. (15). Then, we use the median value of $|\boldsymbol{v}(\boldsymbol{x})|$ for the detection of the dominant plane. Since the dominant plane occupies the largest domain in the image, we compute the distance between each $|\boldsymbol{v}(\boldsymbol{x})|$ and the median of the speed, as shown in Fig. 6.

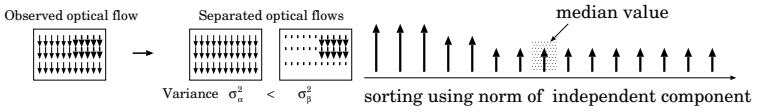


Fig. 6. Difference in the motions of the dominant plane and obstacles. (a) The dominant-plane motion is smooth on the images compared with obstacle motion. Therefore, the order of the components can be determined by using variances σ_α^2 and σ_β^2. (b) Sorting using the norm $\boldsymbol{l}$ for determination of output order. The area which has the median value of the component is detected as the dominant plane, since the dominant plane occupies the largest domain in the image.

The area which has the median value of the component is the dominant plane. Setting m to be the median value of the speed, we define the measure as $\varepsilon(x, y, t) = ||\boldsymbol{v}(\boldsymbol{x})| - m|$. Therefore, the area in which $\varepsilon(x, y, t) \approx 0$ is the dominant plane.

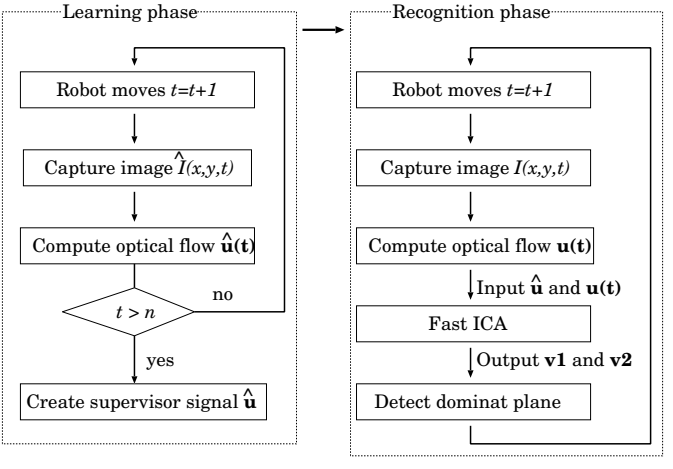


Fig. 7. Procedure for the free space detection using optical flow and ICA

3 Determination of the Navigation Direction

The robot should move on the dominant plane without collision with obstacles. We generate an artificial repulsive force from the obstacle area in the dominant plane map $d(x, y, t)$ using the gradient vector field. The potential field on the image is an approximation of the projection of the potential field in the workspace to the image plane. Therefore, we use the gradient vector of the dominant plane as the repulsive force field from obstacles.

3.1 Gradient Vector of Dominant Plane as Repulsive Force Field

Since the dominant plane map $d(x, y, t)$ is a binary image sequence, the computation of the gradient is numerically unstable and unrobust. Therefore, as a preprocess to the computation of the gradient, we smooth the dominant plane map $d(x, y, t)$ by convolution with Gaussian $G*$, that is, we adopt $\boldsymbol{g}$,

$$\boldsymbol{g}(x, y, t) = \nabla(G * d(x, y, t)) = \begin{pmatrix} \frac{\partial}{\partial x}(G * d(x, y, t)) \\ \frac{\partial}{\partial y}(G * d(x, y, t)) \end{pmatrix}, \tag{16}$$

as the potential generated by obstacles. Here, for a 2D Gaussian,

$$G_\sigma(x, y) = \frac{1}{2\pi\sigma} e^{-\frac{x^2+y^2}{2\sigma^2}}. \tag{17}$$

$G * d(x, y, t)$ is given as

$$G * d(x, y, t) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} G(u - x, v - y) d(x, y, t) du dv. \tag{18}$$

Definition 2. *(Obstacle Potential) For the obstacle region O, we define the potential*

$$p_o(x, y, t) = G_\sigma * d(x, y, t) \quad d(x, y, t) = \begin{cases} 255, & \textit{if } \boldsymbol{x} \in O \\ 0, & \textit{otherwise,} \end{cases} \tag{19}$$

we define the gradient of the potential as

$$\boldsymbol{g}(x, y, t) = \nabla p_o(x, y, t). \tag{20}$$

We select parameter σ in the Gaussian kernel of Eq. (18) to be half the image size. An example of the gradient vector field is shown in Fig. 8(c).

3.2 Repulsive Force by Principal Component Analysis

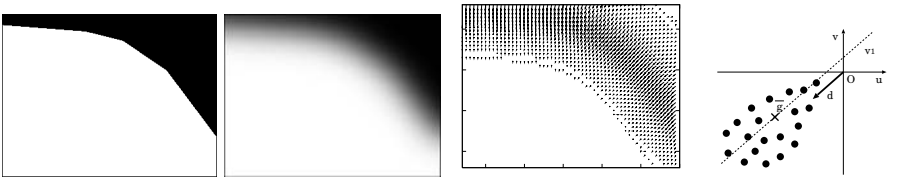
Setting $\mathbf{G} = \{\boldsymbol{g}_i\}_{i=1}^N$ to be the collection of the gradient vectors in the x-y coordinate system, we compute the principal direction of vectors in the vector set $\mathbf{G}$. An example of the collection of gradient vectors $\mathbf{G}$ is plotted in Fig. 9(a). Using the centroid $\overline{\boldsymbol{g}} = \frac{1}{N} \sum_{i=1}^N \boldsymbol{g}_i$ of $\mathbf{G}$, we define matrix $\boldsymbol{G}$ as

$$\boldsymbol{G} = \begin{pmatrix} \boldsymbol{h}_1 \, \boldsymbol{h}_2 \cdots \boldsymbol{h}_N \end{pmatrix} \tag{21}$$

for $\boldsymbol{h}_i = \boldsymbol{g}_i - \overline{\boldsymbol{g}}$. Then, the 2×2 matrix

$$\boldsymbol{M} = \frac{1}{N} \boldsymbol{G}\boldsymbol{G}^\top \tag{22}$$

is the correlation matrix of vectors in $\mathbf{G}$. For eigenvalues λ_i for $i = 1, 2$ such that $\lambda_1 \geq \lambda_2$ of matrix $\boldsymbol{M}$, we set the corresponding principal components as $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$, respectively. From eigenvectors $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ of matrix $\boldsymbol{M}$, we compute the direction of the attractive force to avoid collision with the obstacles, since



(a) $d(x, y, t)$ (b) $G * d(x, y, t)$ (c) $\boldsymbol{g}(x, y, t)$ (d) $\boldsymbol{d}$

Fig. 8. (a) Dominant plane map $d(x, y, t)$. The black and white regions indicate obstacle and dominant plane regions, respectively. (b) Dominant plane after Gaussian convolution operation $G * d(x, y, t)$. (c) Gradient vector field $\boldsymbol{g}(x, y, t)$ as repulsive force field from obstacles. Since the obstacle is at the upper right of the image, the directions of the gradient vectors are to the lower left. (d) Geometrical relationship between the principal component and the repulsive force. Black dots and x indicate collection of gradient vectors $\mathbf{G} = \{\boldsymbol{g}_i\}_{i=1}^{N}$ and its mean value $\overline{\boldsymbol{g}}$, respectively. The dashed line and bold line indicate principal component $\boldsymbol{v}_1$ and the repulsive force $\boldsymbol{r}$, respectively.

potential is computed from the binary function whose value is one in the obstacle area. Figure 9(b) shows an example of the first principal component of $\mathbf{G}$.

Since the robot is required to avoid collision with obstacles, the direction of the repulsive force is in the direction from the obstacle region to the dominant plane. Therefore, we define the direction of the repulsive force from obstacles as

$$\boldsymbol{r} = arg\{\min \angle[\overline{\boldsymbol{g}}, \boldsymbol{v}_1]\}, \tag{23}$$

where $\angle[\boldsymbol{a}, \boldsymbol{b}]$ is the smaller angle between vectors $\boldsymbol{a}$ and $\boldsymbol{b}$. That is, we select the direction of the mean value $\overline{\boldsymbol{g}}$ in principal component $\boldsymbol{v}_1$ as the direction of the repulsive force from obstacles. The geometrical relationship between principal component $\boldsymbol{v}_1$ and the repulsive force $\boldsymbol{r}$ is shown in Fig. 8 (d).

3.3 Mobile Robot Navigation by Direction of Attention

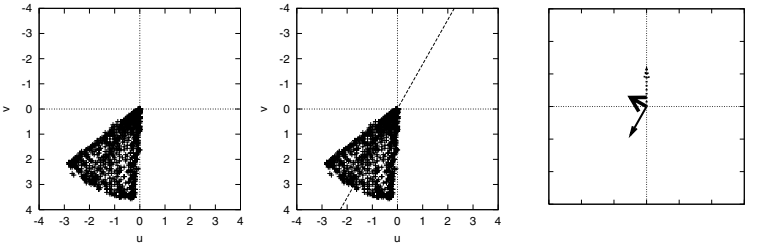
Since vector $\boldsymbol{r}$ defines the direction of attention for avoiding collision with obstacles, we set the control force as

$$\boldsymbol{p} = \boldsymbol{r} + \boldsymbol{w} \tag{24}$$

for guiding force $\boldsymbol{w}$ computed from the translational optical flow field. For example, if the robot moves forward, guiding force $\boldsymbol{w} = (0, 1)^\top$. This direction $\boldsymbol{p}$ defines the direction of attention for the next view.

Setting $\boldsymbol{p}(t)$ to be the control force $\boldsymbol{p}$ at time t, we define the translational and rotational velocities for a nonholonomic mobile robot. The paper [16] defined the control method of the mobile robot from $\boldsymbol{p}(t)$ as

$$\theta(t) = \arccos \frac{\boldsymbol{p}(t)^\top \boldsymbol{y}}{|\boldsymbol{p}||\boldsymbol{y}|}, \tag{25}$$



(a) $\mathbf{G}$ (b) $\boldsymbol{v}_1$ (c) $\boldsymbol{p}$

Fig. 9. Gradient field and its principal component. (a) $\mathbf{G} = \{\boldsymbol{g}_i\}_{i=1}^{N}$ of Fig. 8 is plotted in the u-v coordinate. (b) Plotted $\mathbf{G}$ and its first principal component $\boldsymbol{v}_1$. The dashed line is the first principal component. (c) Direction of attention. The solid arrow is the repulsive force $\boldsymbol{r}$. The dashed arrow is the forward direction of the robot $\boldsymbol{w} = (0, 1)^\top$. The bold arrow is the control force $\boldsymbol{p}$. Since the obstacle is at the upper right of the image, the repulsive force $\boldsymbol{v}_1$ is towards the lower left.

where $\boldsymbol{y} = (0, 1)^\top$ is the y axis of the image, which is the forward direction of the mobile robot. Figures 10(a) and (b) shows the relationships among $\boldsymbol{p}(t)$, $\theta(t)$, and the direction of robot motion [16].

We define the robot translational velocity $T(t)$ and the rotational velocity $R(t)$ at time t as $T(t) = T_m \cos\theta(t)$ and $R(t) = R_m \sin\theta(t)$, where T_m and R_m are the maximum translational and rotational velocities of the mobile robot between time t and $t+1$. Setting $\boldsymbol{X}(t) = (X(t), Y(t))^\top$ to be the position of the robot at time t in the world coordinate system, we have the relations

$$T(t) = \sqrt{\dot{X}(t)^2 + \dot{Y}(t)^2}, \;\; R(t) = \tan^{-1}\frac{\dot{Y}(t)}{\dot{X}(t)}. \tag{26}$$

Therefore, we have the control law

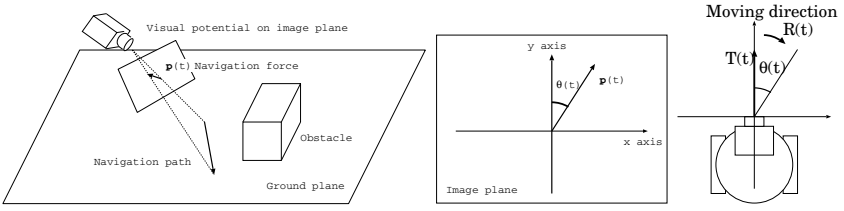
$$\dot{X}(t) = T(t)\cos R(t), \;\; \dot{Y}(t) = T(t)\sin R(t) \tag{27}$$

as shown in Fig. 10 (c). The control strategy is illustrated in Figure 10.

4 Experiments for Dominant Plane Detection

We experimentally evaluate the robustness of our temporal-model-based method with comparison to the model-based method.

We use the Lucas-Kanade method with pyramids [3] for the computation of the optical flow. The tolerance for the matching of flow vectors in Eq. (9) is set to be $\varepsilon = 0.2$, which was determined experimentally.



(a) (b) (c)

Fig. 10. Navigation using the potential field. (a) Navigation force $\boldsymbol{p}(t)$ is the local navigation path. (b) The angle between the control force $\boldsymbol{p}(t)$ and y axis is $\theta(t)$. (c) Robot displacement $T(t)$ and rotation angle $R(t)$ at time t are determined using $\theta(t)$.

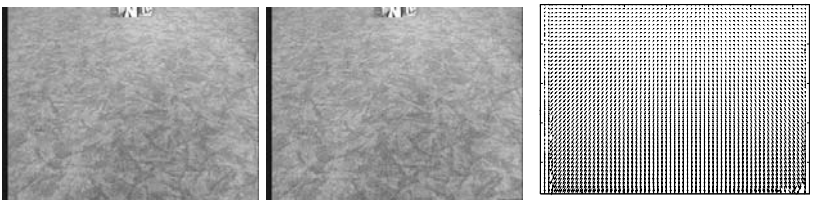


Fig. 11. Model Image Generation Left and Middle: image sequence without obstacles. The pair of images is successive image $I_m(t)$ and $I_m(t+1)$ for the computation of the optical-flow model. Right: optical-flow model $\boldsymbol{u}_m$.

Fig. 12. Same as Fig. 11 but for the image sequence with obstacles. The images $I(t)$ and $I(t+1)$ and the optical flow field $\boldsymbol{u}$ are used for the detection of the ground plane.

4.1 Comparison with Model-Based Method

In this experiment, we detect the dominant plane of the successive images obtained by a camera mounted on a robot.

First, we show the result of the ground plane detection employing the model-based method. Two images in left and middle of Fig. 11 are the successive images captured in the workspace without obstacles, and right, the optical flow $\boldsymbol{u}_m$. Figure 12 shows the image sequence and the optical flow $\boldsymbol{u}$ in the detection phase.

Fig. 13. Dominant plane detected by model-based method. Left: observed optical flow. Middle and Right. dominant plane The white and black regions represent the ground plane without obstacles and that with obstacles, respectively.

Fig. 14. Dominant plane detection by temporal model-based method Left: planar flow. Middle and Right: dominant plane. The dominant plane is detected by the temporal-model-based method. The white and black regions represent the ground plane without obstacles and that with obstacles, respectively.

Matching the optical flow $\boldsymbol{u}$ and the model vector $\boldsymbol{u}_m$, the same region and a different region are painted with white and black, respectively. Therefore, the white and black regions represent the ground plane without obstacles and that with obstacles, respectively. The result is shown in Fig. 13.

Next, we show the result of the dominant plane detection employing our temporal-model-based method. This method uses the images sequence shown in Fig. 12 for the detection of the dominant plane. The top of Fig. 14 shows the planar flow $(\hat{x}, \hat{y})^\top$ estimated from the optical flow $(\dot{x}, \dot{y})^\top$ shown in the middle of Fig. 12.

4.2 Dominant Plane Detection for Long Sequence

We present the dominant plane detection in a long image sequence using our temporal-model-based method and the model-based method, and compare the obtained results.

To evaluate the accuracy of the dominant plane detection, we define the error ratio $E(t)$.

$$E(t) = 100 \frac{\sum_{x,y} |G(x,y,t) - D(x,y,t)|}{\sum_{x,y} 1}.[\%] \tag{28}$$

Here, $D(x, y, t)$ and $G(x, y, t)$ are the dominant planes on the image coordinate system (x, y) at frame t detected from the optical flow and the ground truth which is extracted manually, respectively.

$$D(x, y, t) = \begin{cases} 0 \text{ for detected dominant plane,} \\ 1 \text{ for detected other area,} \end{cases}, \tag{29}$$

$$G(x, y, t) = \begin{cases} 0 \text{ for dominant plane,} \\ 1 \text{ for other area,} \end{cases}, \tag{30}$$

$E(t)$ roughly describes the percentage of errors between the actual and estimated dominant planes.

Figures 15 and 16 show the results of the dominant plane detection and the error ratio, respectively.

Figure 15 shows the results of the detection using the two methods. In Fig. 15, the first, second, third, fourth, and fifth rows show the image sequence, computed optical flow, estimated planar flow, results of the detection using the temporal-model-based method, and results of the detection using the model-based method, respectively.

The results in Fig. 16 show that the error ratio decreases in a subsequent frame even if the error ratio increases in a sequence. Therefore, our method enables the detection of the dominant plane without increasing the error ratio.

Our method runs on a 800MHz, AMD-K6 processor for 320×240 images and takes 0.25 seconds per frame for the detection of the dominant plane. The specifications of the mobile robot are described in Table 1.

Table 1. Specifications of our mobile robot

Name	Magellan Pro, AAI Systems, Inc.
Size	Circular - 16-inch diameter
Weight	50 pounds
Drive	2-wheel
CPU	800MHz, AMD-K6 processor
Main memory	256MB
OS	Red Hat Linux
Compiler	GNU C++ Compiler
Camera	SONY EVI-D30

In Fig. 16, the left and right figures show the error ratios $E(t)$ of the temporal-model-based and model-based methods, respectively. These plots of error ratios show that the temporal-model-based method is more robust than the model-based method.

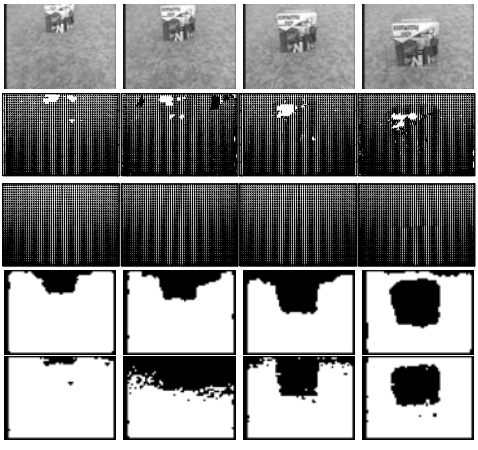


Fig. 15. Dominant plane detection in long sequence. Detected dominant planes in 150, 200, 250, and 300 frames. The first, second, third, fourth, and fifth rows show the image sequence, computed optical flow, estimated planar flow, results of the detection using the temporal-model-based method, and results of the detection using the model-based method, respectively.

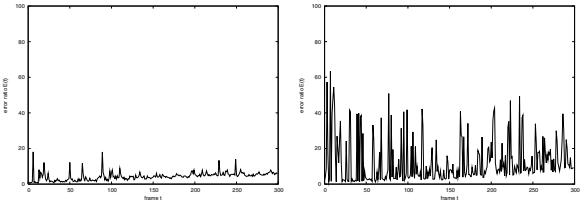


Fig. 16. Error ratios $E(t)$ of temporal-model-based method and model-based method. Left and Right are temporal-model-based method and model-based method, respectively. The vertical axis indicates the error ratio $E(t)$ in Eq. (28), while the horizontal axis, the frame number of the image sequence.

4.3 Dominant Plane Detection Using Uncalibrated Image Sequences

Our method detects the dominant plane under the condition that the camera displacement of the image sequence is unknown. On the other hand, the model-based method requires the camera displacement of the image sequence. Therefore, the detection of the obstacles from uncalibrated image sequences is not possible using the model-based method.

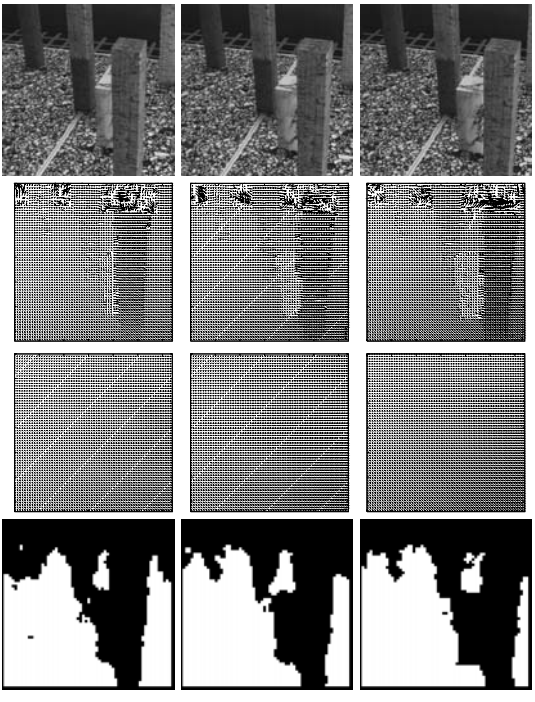


Fig. 17. Detected dominant planes in 0, 10 and 20 frames. The first row shows the original image sequence, the second row, the optical flow sequence, the third row, the planar flow sequence, and the fourth row, the detected dominant plane. In the images of the dominant plane, the white areas are the dominant planes, and the black areas are the obstacles. The top black areas in the images have an infinite depth from a camera. Therefore, these areas are not dominant planes.

For the experiment on the dominant plane detection using image sequences, we use the marbled-block[1] and rotating block[2] image sequences. These image sequences are shown in the first rows of Figs. 17 and 18. The image sequence in Fig. 17 is captured by a translating camera. The image sequence in Fig. 18 is captured by a rotating camera. In Figs. 17 and 18, the four rows are the original image sequence, optical flow sequence, planar flow sequence, and detected dominant plane, from top to bottom, respectively. In the images of the dominant plane, the white areas are the dominant planes, and the black areas are the obstacles. These results show that our method enables the accurate detection of the dominant plane from the uncalibrated image sequences.

[1] Marbled-block sequence: recorded and first evaluated by Otte and Nagel, http://i21www.ira.uka.de/image_sequences/

[2] Rotating blocks: Otago optical flow evaluation sequences, http://www.cs.otago.ac.nz/research/vision/Research/

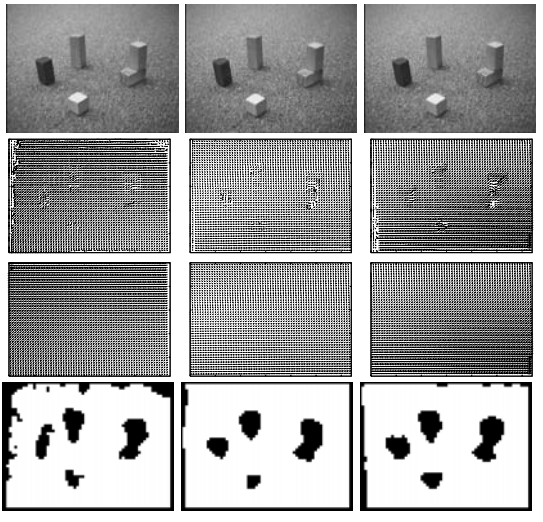


Fig. 18. Detected dominant planes in 0, 1 and 2 frames. The first row shows the original image sequence, the second row, the optical flow sequence, the third row, the planar flow sequence, and the fourth row, the detected dominant plane. In the images of the dominant plane, the white areas are the dominant planes, and the black areas are the obstacles.

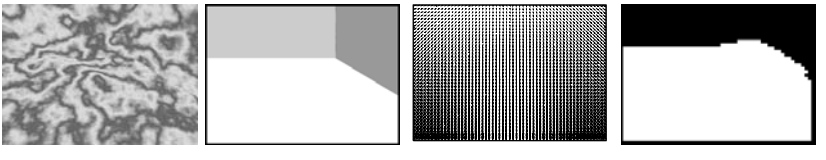


Fig. 19. Captured image and computed optical flow. From left to right, a captured image, the ground truth of the planes, the computed optical planar flow, and the estimated dominant plane (the white region), respectively. There are three orthogonal planes in front of the camera, the floor is the dominant plane. The camera moves forward.

4.4 Statistical Method for Dominant Plane Detection

We clarify the relation between the RANSAC-based method an ICA-based method for the dominant plane detection using an image sequence captured from a synthetic environment. We use the *Fast ICA package for MATLAB*[3]. The first frame of the image sequence is shown in Fig. 19(left). There are three orthogonal planes in front of the camera, and the camera moves forward. Therefore, the computed optical flow is Fig. 19(right). In the experiments, the width of

[3] http://www.cis.hut.fi/projects/ica/fastica/

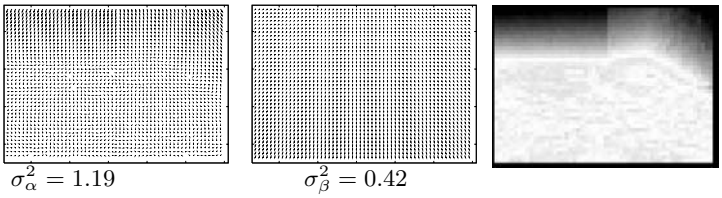


Fig. 20. Output optical flow fields and those variances

images is 320 pixel. Figures in the second row in Fig. 19 is the estimated planar flow and the detected dominant plane.

Figure 20 shows output optical flow fields and those variances. The variances σ_α and σ_β are used for determining the dominant plane.

These numerical examples show that the RANSAC-based algorithm achieves to segment planar areas whose variances of the lengths of flow vectors are difference.

4.5 Discussion

The model-based method is a simple method for computation of the detection of obstacles, since this method only requires the matching of flow vectors. Therefore, this method can be performed faster than our temporal-model-based method. On the other hand, our method requires the estimation of the planar flow. This estimation process in our method takes considerable time, compared with the model-based method, since our method generates the planar flow temporary. Thus, to speed up this estimation process, the planar flow is estimated by an affine transformation.

The advantage of the temporal-model-based method is that it is independent of the workspace configuration, since our method generates the planar flow at every frame for the matching with the optical flow. Therefore, our method enables the accurate detection of the dominant plane. In contrast, the model-based method uses the model created at the learning phase in advance. The method is valid for the same workspace in which the model is learned.

From these observation, we adopt the temporal-model-based method for the dominant-plane detection as the preprocessing for the navigation direction computation.

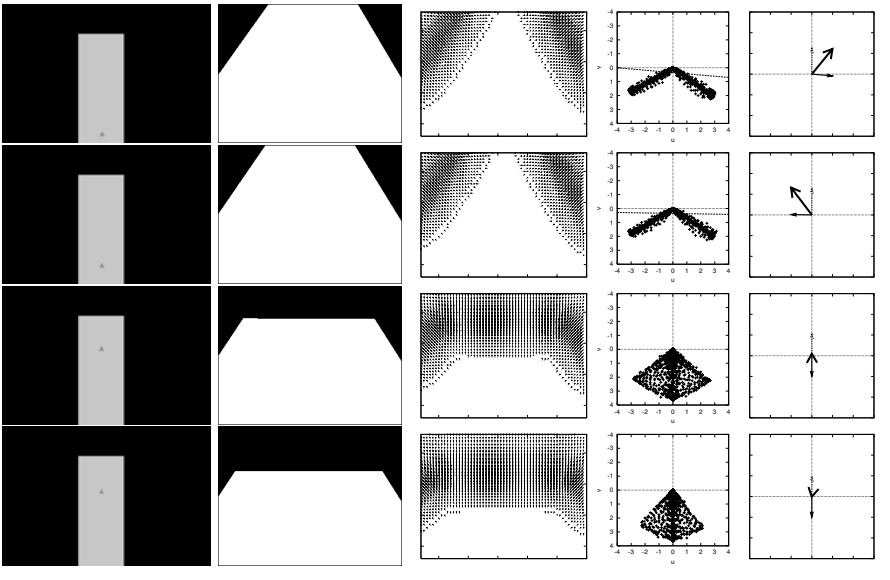
5 Experiments for the Navigation Direction Computation

5.1 Experiments in Synthetic Environments

We carried out experiments on navigating the mobile robot using the proposed algorithm in synthetic environments. The mobile robot moves forward without collision with obstacles and walls. Therefore, we carry out experiments for the following basic situations.

(a) (b)

Fig. 21. Experimental results in synthetic environments. (a) Corridor with a dead end. (b) Curved pathway. The black regions are obstacles. The white regions are the corridor. The triangles in the map indicate the positions of the mobile robot.

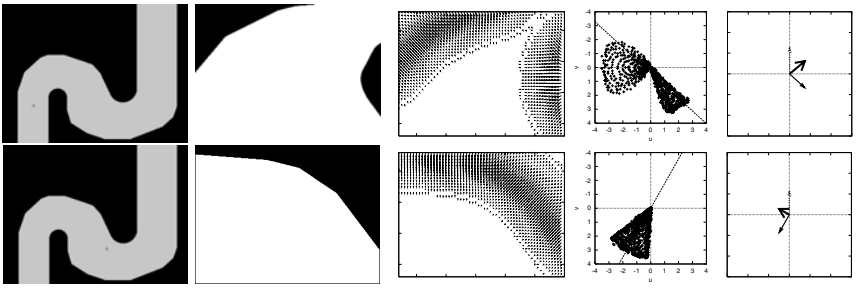


(a) (b) $I(x, y, t)$ (c) $\boldsymbol{g}(x, y, t)$ (d) $\boldsymbol{v}_1$ (e) $\boldsymbol{p}$

Fig. 22. Control forces at several frames in Fig. 21 (a). (a) Configuration of the robot and obstacles. The triangle is the mobile robot. (b) Captured image $I(x, y, t)$. (c) Gradient vector field $\boldsymbol{g}(x, y, t) = \nabla(G * d(x, y, t))$. (d) Plotted gradient vectors $\mathbf{G}$ and its first principal component $\boldsymbol{v}_1$. (e) Attention direction. The solid arrow is the repulsive force $\boldsymbol{r}$. The dashed arrow is the forward direction of the robot $\boldsymbol{w} = (0, 1)^\top$. The bold arrow is the control force $\boldsymbol{p}$.

1. The mobile robot moves along the centre curve of a corridor.
2. The mobile robot stops at a dead end.
3. The robot turns at the corner.

Figures 21 (a) and (b) show the experimental results of motion control in corridors with a dead end and a curved pathway, respectively. These results



(a) (b) $I(x, y, t)$ (c) $\boldsymbol{g}(x, y, t)$ (d) $\boldsymbol{v}_1$ (e) $\boldsymbol{p}$

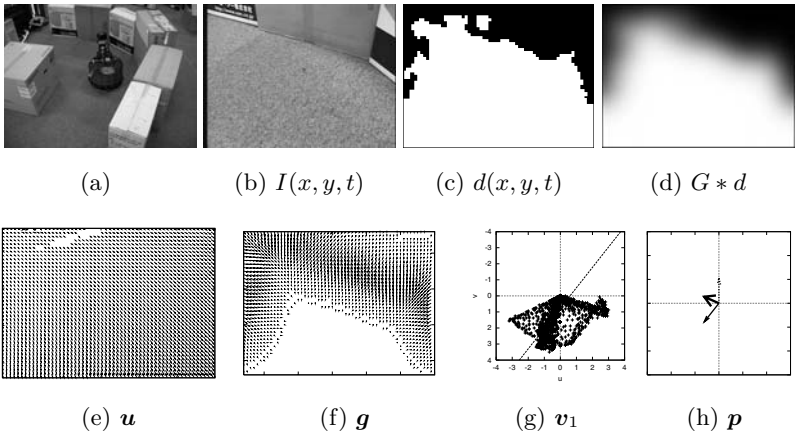
Fig. 23. Control forces at several frames in Fig. 21 (b). Captions of (a)-(e) are the same as those of Fig. 22.

(a) (b) (c)

Fig. 24. Experimental results in synthetic environments. (a) Room with one obstacle. (b) Room with three obstacles. (c) Curved pathway.

show that the mobile robot moved along the corridor without collision with obstacles. Figures 22 and 23 show the captured images and the control force at several frames in the experiments shown in Figs. 21 (a) and (b), respectively. The first and second columns in Fig. 22 show that the robot moved along the centre of the corridor. The third and fourth columns in Fig. 22 show that the robot stopped at the dead end. It known that the repulsive force from obstacle $\boldsymbol{r}$ is almost equal to the forward direction of the robot $\boldsymbol{w}$ at the dead end. Therefore, the control force $\boldsymbol{p} \simeq 0$ and the robot stops at the dead end.

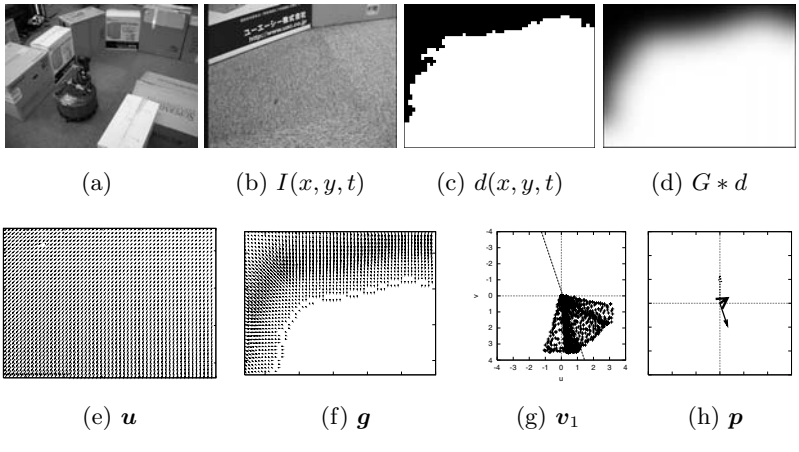
Figure 23 shows that the robot turned right and left at the corners. If the obstacle is at the upper left in the image, the repulsive force is towards the lower right. If the obstacle is at the upper right in the image, the repulsive force is towards the lower left. Therefore, the mobile robot moved along the curved pathway without collision with obstacles. Other experimental results are shown Fig. 24. These results show that PCA to the gradient filed of the obstacle region extracts the control force for the visual navigation.



(a) (b) $I(x, y, t)$ (c) $d(x, y, t)$ (d) $G * d$

(e) $\boldsymbol{u}$ (f) $\boldsymbol{g}$ (g) $\boldsymbol{v}_1$ (h) $\boldsymbol{p}$

Fig. 25. Experimental results for the counterclockwise curved pathway. (a) Snapshot of the experiment. (b) Captured image $I(x, y, t)$. (c) Detected dominant plane $d(x, y, t)$. (d) Gaussian image $G*d(x, y, t)$. (c) Optical flow $\boldsymbol{u}$. (f) Gradient vector field $\boldsymbol{g}(x, y, t) = \nabla(G * d(x, y, t))$. (g) Plotted gradient vectors $\mathbf{G}$ and its first principal component $\boldsymbol{v}_1$. (h) Direction of attention $\boldsymbol{p}$. Since the obstacle is at the upper right in the image, the direction of attention is towards the upper left.



(a) (b) $I(x, y, t)$ (c) $d(x, y, t)$ (d) $G * d$

(e) $\boldsymbol{u}$ (f) $\boldsymbol{g}$ (g) $\boldsymbol{v}_1$ (h) $\boldsymbol{p}$

Fig. 26. Experimental results for the clockwise curved pathway. Captions of (a)-(h) are the same as those of Fig. 25. Since the obstacle is at the upper left in the image, the direction of attention is towards the upper right.

5.2 Experiments in Real Environments

We carried out experiments on navigating the mobile robot using the proposed algorithm in real environments. The environments for the experiments are counterclockwise- and clockwise-curved pathways. In these environments, the robot moved without collision with obstacles in the corridor.

The experimental results in the counterclockwise- and clockwise-curved pathways are shown in Figs. 25 and 26, respectively. In these figures, starting from the upper left, the snapshot of the experiment, captured image $I(x, y, t)$, detected dominant plane $d(x, y, t)$, Gaussian image $G * d(x, y, t)$, gradient vector field $\boldsymbol{g}(x, y, t) = \nabla(G * d(x, y, t))$, plotted gradient vectors $\mathbf{G}$ and its first principal component $\boldsymbol{v}_1$, and direction of attention $\boldsymbol{p}$ at a frame are shown.

Figure 25 shows that the robot moved along the counterclockwise-curved pathway without collision with obstacles. Since the obstacle is at the upper right in the image, the direction of attention is towards the upper left. Figure 25 shows that the robot moved along the clockwise-curved pathway without collision with obstacles. Since the obstacle is at the upper left in the image, the direction of attention is towards the upper right. These results show the performance of the our algorithm for the real environments.

6 Conclusions

We introduced the visual potential as a geometric feature for robot navigation. The visual potential is generated from the free space map, which is obtained as dominant plane in the view. The first part of the paper compares three optical-flow based methods for the dominant plane detection. We showed that the independent component analysis of the optical flow filed detects the dominant plane in the view. Then, we developed an algorithm for navigating a mobile robot using the principal component of the gradient of the obstacle potential field captured by a camera mounted on the robot. We defined the principal component as the repulsive force from obstacles for avoiding collision with obstacles. Therefore, the principal component indicates the direction of attention for the next view. We set the direction of the principal component vector using the mean value of the obstacle potential field. Our algorithm enabled a mobile robot to avoid obstacles without referring to an environmental map. Some experimental results for a mobile robot navigating in synthetic and real environments were presented.

References

1. Adorini, G., Cagnoni, S., Mordonini, M., Sgorbissa, A.: Omnidirectional stereo systems for robot navigation. In: OMNIVIS (2003)
2. Barron, J.L., Fleet, D.J., Beauchemin, S.S.: Performance of optical flow techniques. International J. of Computer Vision 12, 43–77 (1994)
3. Bouguet, J.-Y.: Pyramidal implementation of the Lucas Kanade feature tracker description of the algorithm. Intel Corporation, Microprocessor Research Labs, OpenCV Documents (1999)

4. Conner, D.C., Rizzi, A.A., Choset, H.: Composition of local potential functions for global robot control and navigation. In: International Conference on Intelligent Robots and Systems, vol. 4, pp. 3546–3551 (2003)
5. Fischler, M.A., Bolles, R.C.: Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Comm. of the ACM 24, 381–395 (1981)
6. Guilherme, N.D., Avinash, C.K.: Vision for mobile robot navigation: A survey. IEEE Trans. on PAMI 24, 237–267 (2002)
7. Hartley, R., Zisserman, A.: Multiple View Geometry in Computer Vision. Cambridge University Press, Cambridge (2000)
8. Horn, B.K.P., Schunck, B.G.: Determining optical flow. Artificial Intelligence 17, 185–203 (1981)
9. Khatib, O.: Real-time obstacle avoidance for manipulators and mobile robots. International J. of Robotics Research 5, 90–98 (1986)
10. Lucas, B., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: International Joint Conference on Artificial Intelligence, pp. 674–679 (1981)
11. Mallot, H.A., Bulthoff, H.H., Little, J.J., Bohrer, S.: Inverse perspective mapping simplifies optical flow computation and obstacle detection. Biological Cybernetics 64, 177–185 (1991)
12. Murray, D., Little, J.: Using real-time stereo vision for mobile robot navigation. Autonomous Robots 8, 161–171 (2000)
13. Nagel, H.-H., Enkelmann, W.: An investigation of smoothness constraint for the estimation of displacement vector fields from image sequences. IEEE Trans. on PAMI 8, 565–593 (1986)
14. Ohnishi, N., Imiya, A.: Featureless robot navigation using optical flow. Connection Science 17, 23–46 (2005)
15. Ohnishi, N., Imiya, A.: Dominant plane detection from optical flow for robot navigation. Pattern Recognition Letters 27, 1009–1021 (2006)
16. Ohnishi, N., Imiya, A.: Navigation of nonholonomic mobile robot using visual potential field. In: International Conference on Computer Vision Systems (2007)
17. Ohnishi, N., Imiya, A.: Corridor navigation and obstacle avoidance using visual potential for mobile robot. In: 4th Canadian Conference on Computer and Robot Vision, pp. 131–138 (2007)
18. Ohnishi, N., Imiya, A.: Independent component analysis of layer optical flow and its application. In: 2nd International Symposium on Brain, Vision and Artificial Intelligence, pp. 171–180 (2007)
19. Ohnishi, N., Imiya, A.: Independent component analysis of optical flow for robot navigation. Neurocomputing 71, 2140–2163 (2008) (accepted for publication)
20. Park, K.-Y., Jabri, M., Lee, S.-Y., Sejnowski, T.J.: Independent components of optical flows have MSTd-like receptive fields. In: Proc. of the 2nd International Workshop on ICA and Blind Signal Separation, pp. 597–601 (2000)
21. Santos-Victor, J., Sandini, G.: Uncalibrated obstacle detection using normal flow. Machine Vision and Applications 9, 130–137 (1996)
22. Tews, A.D., Sukhatme, G.S., Matarić, M.J.: A multi-robot approach to stealthy navigation in the presence of an observer. In: ICRA, pp. 2379–2385 (2004)
23. Trihatmo, S., Jarvis, R.A.: Short-safe compromise path for mobile robot navigation in a dynamic unknown environment. In: Australian Conference on Robotics and Automation (2003)
24. Vaina, L.M., Beardsley, S.A., Rushton, S.K.: Optic flow and beyond. Kluwer Academic Publishers, Dordrecht (2004)

25. Wong, B., Spetsakis, M.: Scene reconstruction and robot navigation using dynamic fields. Autonomous Robots 8, 71–86 (2000)
26. Zemel, R.S., Sejnowski, T.J.: A model for encoding multiple object motions and self-motion in area mst of primate visual cortex. Neuroscience 18, 531–547 (1998)

Appendix

We have a system of linear equations,

$$I^l_{x(\alpha\beta)}u^l + I^l_{y(\alpha\beta)}v^l + I^l_{t(\alpha\beta)} = 0, \quad |\alpha| \leq 2, |\beta| \leq 2$$

where $I^l_{x(\alpha\beta)} = I_x(x+\alpha, y+\beta, t)^l$, $I^l_{y(\alpha\beta)} = I_y(x+\alpha, y+\beta, t)^l$, and $I^l_{t(\alpha\beta)} = I_t(x+\alpha, y+\beta, t)^l$. Therefore, if the vectors

$$\boldsymbol{a}^l = (I^l_{x(-2\,-2)}, I^l_{x(-2\,-1)}, \cdots, I^l_{x(22)})^\top$$
$$\boldsymbol{b}^l = (I^l_{y(-2\,-2)}, I^l_{y(-2\,-1)}, \cdots, I^l_{y(22)})^\top$$

are independent, we obtain the unique solution $\boldsymbol{u}^l(x,y) = (u^l(x,y), v^l(x,y))^\top$

$$\boldsymbol{u}^l(x,y) = -(\boldsymbol{A}_l^\top \boldsymbol{A}_l)^{-1}\boldsymbol{A}^\top \boldsymbol{c}^l$$

where

$$\boldsymbol{A}_l = (\boldsymbol{a}^l, \boldsymbol{b}^l), \quad \boldsymbol{c}^l = (I^l_{t(-2\,-2)}, I^l_{t(-2\,-1)}, \cdots, I^l_{t(22)})^\top,$$

for the centre point $(x,y)^\top$ of the 5×5 window on each layer.

Motion Integration Using Competitive Priors

Shuang Wu[1], Hongjing Lu[2], Alan Lee[2], and Alan Yuille[1]

[1] Department of Statistics, UCLA
[2] Department of Psychology, UCLA

Abstract. Psychophysical experiments show that humans are better at perceiving rotation and expansion than translation [5,9]. These findings are inconsistent with standard models of motion integration which predict best performance for translation. To explain this discrepancy, our theory formulates motion perception at two levels of inference: we first perform model selection between the competing models (e.g. translation, rotation, and expansion) and then estimate the velocity using the selected model. We define novel prior models for smooth rotation and expansion using techniques similar to those in the slow-and-smooth model [23] (e.g. Green functions of differential operators). The theory gives good agreement with the trends observed in four human experiments.

1 Introduction

As an observer moves through the environment, the retinal image changes over time to create multiple complex motion flows, including translational, circular and radial motion. Imagine that you are in a moving car and seeing a walker through a set of punch holes, as illustrated in figure (1). In each hole, the component of the line motion orthogonal to the lines orientation can be clearly perceived; however, the component of motion parallel to the lines orientation is not observable. This is often referred to as the aperture problem. The inherent ambiguity of motion signals requires the visual system to employ an efficient integration strategy to combine many local measurements in order to perceive global motion. In the example (figure (1)), the visual system needs to effectively integrate local signals to perceive multiple complex motion flows, including the movement of the walker and the moving background respectively.

Human observers are able to process different motion patterns and infer ego motion and global structure of the world. Psychophysical experiments have identified a variety of phenomena, such as motion capture and motion cooperativity [16], which appear to be consequences of such integration. A number of computational Bayesian models have been proposed to explain these effects based on prior assumptions about motion. In particular, it has been shown that a slow-and-smooth prior, and related models, can qualitatively account for a range of experimental results [21,22,23] and can quantitatively account for others [10,17].

However, the integration strategy modeled by the slow-and-smooth prior may not generalize to more complex motion types, such as circular and radial motion, which are critically important for estimating ego motion. In this paper we

D. Cremers et al. (Eds.): Visual Motion Analysis, LNCS 5604, pp. 235–258, 2009.

are concerned with two questions. (1) What integration priors should be used for a particular motion input? (2) How can local motion measurements be combined with the proper priors to estimate motion flow? Within the framework of Bayesian inference, the answers to these two questions are respectively based on model selection and parameter estimation. In the field of motion perception, most work has focused on the second question, using parameter estimation to estimate motion flow. However, Stocker and Simoncelli [18] recently proposed a conditioned Bayesian model in which strong biases in precise motion direction estimates arise as a consequence of a preceding decision about a particular hypothesis (left vs. right motion).

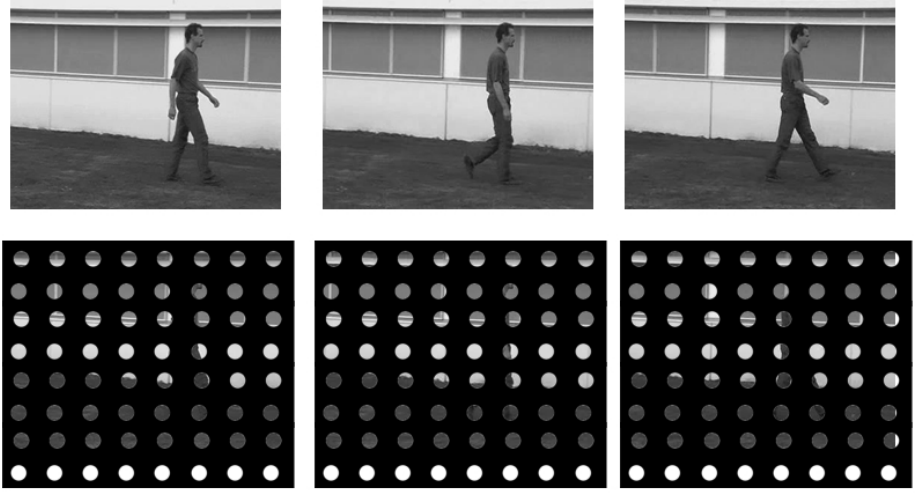


Fig. 1. An illustration of observing a walker with a moving camera. Top panel, three example frames. Bottom panel, observing the scene through a set of punch holes.

The goal of this paper is to provide a computational explanation for both of the above questions using Bayesian inference. To address the first question, we develop new prior models for smooth rotation and expansion motion. To address the second, we propose that the human visual system has available multiple models of motion integration appropriate for different motion patterns. The visual system decides the best integration strategy based upon the perceived motion information, and this choice in turn affects the estimation of motion flow.

In this paper, we first present a computational theory in section (4) that includes three different integration strategies, all derived within the same framework. We test this theory in sections (5,6) by comparing its predictions with human performance in psychophysical experiments, in which subjects were asked to detect global motion patterns or to discriminate motion direction in translation (left vs. right), rotation (clockwise vs. counter-clockwise), and radial motion

(inward vs. outward). We employ three commonly used stimuli: random dot patterns, moving gratings and plaids (two superimposed gratings each with different orientations) to show that the model can be applied to a variety of inputs. We compare the predictions of the model predictions with human performance across four psychophysical experiment.

2 Background

There is an enormous literature on visual motion phenomena and there is only room to summarize the work most relevant to this paper. Our computational model relates most closely to work [10,21,23] that formulates motion perception as Bayesian inference with a prior probability biasing towards slow-and-smooth motion. But psychophysical [1,5,9,11], physiological [4,19] and fMRI data [12] suggests that humans are sensitive to a variety of motion patterns including translation, rotation, and expansion. In particular, Freeman *et al.* [5] and Lee *et al.* [9] demonstrated that human performance on discrimination tasks for translation, rotation, and expansion motion was inconsistent with the predictions of the slow-and-smooth theory (our simulations independently verify this result). Instead, we propose that human motion perception is performed at two levels of inference: (i) model selection, and (ii) estimating the velocity with the selected model. The concept of model selection has been described in the literature, see [8], but has only recently been applied to model motion phenomena [18]. Our new motion models for rotation and expansion are formulated very similarly to the original slow-and-smooth model [23], and similar mathematical analysis [3] is used to obtain the forms of the solutions in terms of Greens functions of the differential operators used in the priors.

Smoothness priors for motion have a long history in computer vision beginning with the influential work of Horn and Schunk [7] and related work by Hildreth [6]. The slow-and-smooth prior on the velocity [23] was developed to explain psychophysical phenomena such as motion capture [14]. The nature of the psychophysics stimuli and the experimental phenomena meant that the first derivative regularizers used in [6,7] needed to be supplemented by zeroth order regularizers (e.g. slowness of the velocity) and higher order regularizers. Specifically, higher order regularizers were required to ensure that the theory was well-posed for the sparse stimuli (e.g. moving dots) used in typical psychophysical experiments. Moreover, zeroth order regularizers were needed to ensure that the interactions (e.g. motion capture) between different stimuli decreased with distance [24]. Justifications for the slowness and smoothness assumptions came from analysis of the probability distribution of velocity flow in the image plane assuming isotropic distributions of velocity in space [20,25], namely for most distributions of velocity in space the projection onto the image plane would induce velocity distributions which were peaked at zero velocity and zero velocity gradient. More recently, empirical studies by Roth and Black [15] on motion statistics yield similar results. We note that a limitation of the slow-and-smooth prior, for computer vision purposes, is that it is a quadratic regularizer and

hence tends to smooth the velocity over motion discontinuities (although the limited range of interaction of the slow-and-smooth model means that this effect is more localized than for standard quadratic regularizers). Hence non-quadratic regularizers/priors are preferred for computer vision applications [2]. The nature of the psychophysical stimuli in this paper, however, means that discontinuities can be ignored and so we can use quadratic regularizers and take advantage of their good computational properties.

In order to relate theories like slow-and-smooth to psychophysics we have to understand the experimental design and model how human subjects address the specific experimental tasks. The experiments we describe in this paper use a standard experimental design which we describe for dot stimuli (a similar design is used for gratings and plaid stimuli). For these stimuli, some of the dots (the signal) move coherently, whereas other dots (the noise) move randomly. The experimental task is to measure some property of the coherent stimulus (e.g. does it move to the left or the right? does it rotate or contract?). The difficulty of performing this task depends on the proportion of signal dots to noise dots. The *coherence threshold* is defined to be the value of the ratio at which human subjects achieve 75 % accuracy for the task. Hence stimulus parameters (i.e. proportion of signal to noise) can be adjusted until this degree of accuracy is obtained. To evaluate the performance of a theory, we compute the analogous threshold for the theory and compare it to the threshold found in the experiments. An alternative criterion is to measure the minimum speed or contrast (e.g. brightness of the dots compared to the background) required to achieve a certain performance level (e.g. 75 % accuracy) at a visual task (e.g. is the motion to the right or the left?).

3 Overview of the Theory

We formulate motion perception as a problem of Bayesian inference with two stages followed by post-processing. We hypothesize that the human visual system has a set of models M for estimating the velocity. Firstly, we perform model selection to find the model that best explains the observed motion pattern. Secondly, we estimate motion properties using the selected model. The post-processing models how human subjects use the estimated motion to perform psychophysical tasks.

Formally a model M estimates the velocity field $\{\boldsymbol{v}\}$ defined at all positions $\{\boldsymbol{r}\}$ in the image from local velocity measurements $\{\boldsymbol{u}\}$ at discrete positions $\{\boldsymbol{r}_i,\ i = 1, \ldots N\}$. The model is specified by a prior $p(\{\boldsymbol{v}\}|M)$ on the velocity field and a likelihood function $p(\{\boldsymbol{u}\}|\{\boldsymbol{v}\})$ for the measurements. This specifies a full distribution $p(\{\boldsymbol{v}\}, \{\boldsymbol{u}\}, M) = p(\{\boldsymbol{u}\}|\{\boldsymbol{v}\})p(\{\boldsymbol{v}\}|M)P(M)$, where $P(M)$ is the prior over the models (which is a uniform distribution).

The prior and likelihoods are expressed as Gibbs distributions:

$$p(\{\boldsymbol{v}\}|M) = \exp(-E(\{\boldsymbol{v}\}|M)/T), \tag{1}$$

$$p(\{\boldsymbol{u}\}|\{\boldsymbol{v}\}) = \exp(-E(\{\boldsymbol{u}\}|\{\boldsymbol{v}\})/T), \tag{2}$$

and are specified in sections (4.1,4.2) respectively.

The first stage of the theory selects the model M^* that best explains the measurements $\{\boldsymbol{u}\}$. It is chosen to maximize the posterior of the model evidence:

$$M^* = \arg\max_M P(M|\{\boldsymbol{u}\}) = \arg\max_M \frac{P(\{\boldsymbol{u}\}|M)P(M)}{P(\{\boldsymbol{u}\})} \tag{3}$$

$$\text{where} \quad p(\{\boldsymbol{u}\}|M) = \int p(\{\boldsymbol{u}\}|\{\boldsymbol{v}\})p(\{\boldsymbol{v}\}|M)d\{\boldsymbol{v}\}. \tag{4}$$

The second stage of the theory estimates the velocity flow by using the selected model M^*. This yields:

$$\boldsymbol{v}^* = \arg\max_{\boldsymbol{v}} p(\{\boldsymbol{v}\}|\{\boldsymbol{u}\}, M^*) \tag{5}$$

$$\text{where} \quad p(\{\boldsymbol{v}\}|\{\boldsymbol{u}\}, M^*) = \frac{p(\{\boldsymbol{u}\}|\{\boldsymbol{v}\})p(\{\boldsymbol{v}\}|M^*)}{p(\{\boldsymbol{u}\}|M^*)}, \tag{6}$$

The post-processing estimates the properties of the motion flow required to perform the psychophysics tasks.In the psychophysics experiments in this paper, the human subjects must determine whether the motion is translation, expansion/contraction, or rotation and then estimate the appropriate motion properties: (i) direction of motion (if translation), (ii) direction of expansion/contraction (if expansion/contraction), or (iii) direction of rotation (if rotation).

To address these tasks, we use the following post-processing strategy which we illustrate for the rotation stimuli. First we compute the two stages of the theory as described above. If the rotation model is selected, then we estimate the center of rotation (by least squares fit) and compute the direction of rotation (i.e. clockwise or counterclockwise) for the velocity field evaluated at each data point (i.e. dots), see Appendix B for details. We alter the parameters of the stimuli (i.e. the proportion of noise dots) until 75 % of the stimulus data points are estimated to have the correct direction of rotation (for each stimulus). Alternatively, we consider a set of stimuli, compute the velocity flow for each stimulus and estimate the centers of rotation, compute the histograms of the estimated rotated directions for the data points for all images, and determine the stimulus parameters so that 75 % of the histogram values are in the correct direction.

4 Model Formulation

This section specifies the priors for the different models, the likelihood functions, closed form solutions for the velocity flows, and analytic formulae for performing model selection. Some details of the derivations are described in appendix (9).

4.1 The Priors

We define three priors corresponding to the three different types of motion – translation, rotation, and expansion. For each motion type, we encourage slowness and spatial smoothness. The prior for translation is very similar to the

slow-and-smooth prior [23] except we drop the higher-order derivative terms and introduce an extra parameter (to ensure that all three models have similar degrees of freedom).

We define the priors by their energy functions $E(\{\boldsymbol{v}\}|M)$, see equation (11). We label the models by $M \in \{t, r, e\}$, where t, r, e denote translation, rotation, and expansion respectively. (We note that the prior for expansion will also account for contraction). We use the convention that $\boldsymbol{v} = (v_x, v_y)$, and $\nabla = (\frac{\partial}{\partial x}, \frac{\partial}{\partial y})$, $\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$.

1. Slow-and-smooth-translation:

$$E(\{\boldsymbol{v}\}|M = t) = \int \lambda(|\boldsymbol{v}|^2 + \mu|\nabla \boldsymbol{v}|^2 + \eta|\nabla^2 \boldsymbol{v}|^2)d\boldsymbol{r}, \tag{7}$$

2. Slow-and-smooth-rotation:

$$E(\{\boldsymbol{v}\}|M = r) = \int \lambda\{|\boldsymbol{v}|^2 + \mu[(\frac{\partial v_x}{\partial x})^2 + (\frac{\partial v_y}{\partial y})^2 + (\frac{\partial v_x}{\partial y} + \frac{\partial v_y}{\partial x})^2] + \eta|\nabla^2 \boldsymbol{v}|^2\}d\boldsymbol{r}, \tag{8}$$

3. Slow-and-smooth-expansion:

$$E(\{\boldsymbol{v}\}|M = e) = \int \lambda\{|\boldsymbol{v}|^2 + \mu[(\frac{\partial v_x}{\partial y})^2 + (\frac{\partial v_y}{\partial x})^2 + (\frac{\partial v_x}{\partial x} - \frac{\partial v_y}{\partial y})^2] + \eta|\nabla^2 \boldsymbol{v}|^2\}d\boldsymbol{r}, \tag{9}$$

These priors are motivated as follows. The $|\boldsymbol{v}|^2$ and $|\nabla^2 \boldsymbol{v}|^2$ terms bias towards slowness and smoothness and are used in all three models. However, the priors differ in the use of first derivative terms. The translation prior prefers constant translation motion with $\boldsymbol{v}$ constant, since $\nabla \boldsymbol{v} = 0$ for this type of motion. The translation prior is similar to the first three terms of the slow-and-smooth energy function [23] but with a restriction on the set of parameters. Formally $\lambda(|\boldsymbol{v}|^2 + \frac{\sigma^2}{2}|\nabla \boldsymbol{v}|^2 + \frac{\sigma^4}{8}|\nabla^2 \boldsymbol{v}|^2)d\boldsymbol{r} \approx \lambda \sum_{m=0}^{\infty} \frac{\sigma^{2m}}{m! 2^m}|D^m \boldsymbol{v}|^2 d\boldsymbol{r}$ (for appropriate parameter values). Our computer simulations showed that the translation prior performs similar to the original slow-and-smooth prior.

The first derivative terms in the slow-and-smooth-rotation prior is motivated by the ideal form of rigid rotation:

$$v_x = -\omega(y - y_0), \quad v_y = \omega(x - x_0) \tag{10}$$

where (x_0, y_0) are the (unknown) centers, ω is the (unknown) angular speed. This rigid rotation is preferred by the slow-and-smooth-rotation prior (at least, by its first derivative terms), since we have $\frac{\partial v_x}{\partial y} + \frac{\partial v_y}{\partial x} = 0$ and $\frac{\partial v_x}{\partial x} = \frac{\partial v_y}{\partial y} = 0$ (independent of (x_0, y_0) and ω).

The first derivative terms in the slow-and-smooth-expansion prior is motivated by the ideal form of rigid expansion:

$$v_x = e(x - x_0), \quad v_y = e(y - y_0) \tag{11}$$

where (x_0, y_0) are the (unknown) centers, and e is the (unknown) expansion rate. Such a rigid expansion is preferred by the slow-and-smooth-expansion prior (first order terms), since we have $\frac{\partial v_x}{\partial x} - \frac{\partial v_y}{\partial y} = 0$, and $\frac{\partial v_x}{\partial y} = \frac{\partial v_y}{\partial x} = 0$ (again independent of (x_0, y_0) and e).

Note that we will use equations (10,11) in the post-processing stage where we estimate the centers (x_0, y_0) and the rotation/expansion ω, e, see Appendix B.

4.2 The Likelihood Functions

The likelihood function differs for the three classes of stimuli used in the psychophysics experiments that we considered: (i) For moving dots, as used in [5], the motion measurement is the local velocity $\boldsymbol{u}$ of each dot (assumed known since the motion is small and so correspondence between dots at different time frames is unambiguous); (ii) For a moving grating [13] [9], the local motion measurement is the velocity component in the direction orthogonal to the grating orientation; (iii) For a moving plaid (with two superimposed grating each with different orientation) [13] [9], the local motion measurement uses the velocity components of each grating (effectively this reduces to the moving dot case).

For the dot stimuli, the energy term in the likelihood function, see equation (2), is set to be

$$E(\{\boldsymbol{u}|\boldsymbol{v}\}) = \sum_{i=1}^{N} |\boldsymbol{v}(\boldsymbol{r}_i) - \boldsymbol{u}(\boldsymbol{r}_i)|^2 \tag{12}$$

For the grating stimuli, see figure (5), the likelihood function uses the energy function

$$E_n(\{\boldsymbol{u}\}|\{\boldsymbol{v}\}) = \sum_{i=1}^{N} |\boldsymbol{v}(\boldsymbol{r}_i) \cdot \hat{\boldsymbol{u}}(\boldsymbol{r}_i) - |\boldsymbol{u}(\boldsymbol{r}_i)||^2 \tag{13}$$

where $\hat{\boldsymbol{u}}(\boldsymbol{r}_i)$ is the unit vector in the direction of $\boldsymbol{u}(\boldsymbol{r}_i)$ (i.e. in the direction of local image gradient and hence perpendicular to the orientation of the bar).

For the plaid stimuli, see figure (6), the likelihood function uses the energy function

$$E_p(\{\boldsymbol{u}\}|\{\boldsymbol{v}\}) = \sum_{i=1}^{N} (|\boldsymbol{v}(\boldsymbol{r}_i) \cdot \hat{\boldsymbol{u}}_1(\boldsymbol{r}_i) - |\boldsymbol{u}_1(\boldsymbol{r}_i)||^2 + |\boldsymbol{v}(\boldsymbol{r}_i) \cdot \hat{\boldsymbol{u}}_2(\boldsymbol{r}_i) - |\boldsymbol{u}_2(\boldsymbol{r}_i)||^2) \tag{14}$$

where there are two sets of measurements $\{u_1\}$ and $\{u_2\}$. This specifies the velocity in two different directions and hence can be reformulated in terms of model for the dots.

4.3 MAP Estimator of Velocities

The MAP estimate of the velocities for each model is obtained by solving

$$\boldsymbol{v}^* = \arg\max_{\boldsymbol{v}} p(\{\boldsymbol{v}\}|\{\boldsymbol{u}\}, M) = \arg\min_{\boldsymbol{v}} \{E(\{\boldsymbol{u}\}|\{\boldsymbol{v}\}) + E(\{\boldsymbol{v}\}|M)\} \tag{15}$$

For the slow-and-smooth model [23], it was shown using regularization analysis [3] that this solution $\boldsymbol{v}^*$ can be expressed in terms of a linear combination of the Green function G of the differential operator which imposes the slow-and-smoothness constraint (the precise form of this constraint was chosen so that G was a Gaussian [24]).

In this paper, we perform a similar analysis for the three types of models $M \in \{t, r, e\}$ introduced in this paper and show that the solution $\boldsymbol{v}^*$ can also be expressed as a linear combination of Green functions G^M where M indicates the model. The precise forms of these Green functions are determined by the differential operators associated with the slow-and-smoothness terms of the models and are derived in appendix (9 A1). The main difference with the Green functions described by Yuille and Grzywacz [23],[25] is that the models for expansion/contraction and rotation require two vector valued Green functions $\boldsymbol{G}_1^M = (G_{1x}^M, G_{1y}^M)$ and $\boldsymbol{G}_2^M = (G_{2x}^M, G_{2y}^M)$ (with the constraint that $G_{1x}^M = G_{2y}^M$ and $G_{2x}^M = G_{1y}^M$). These vector-valued Green functions are required to perform the coupling between the different velocity component required for rotation and expansion, see figure (2). For the translation model there is no coupling required and so $G_{2x}^M = G_{1y}^M = 0$.

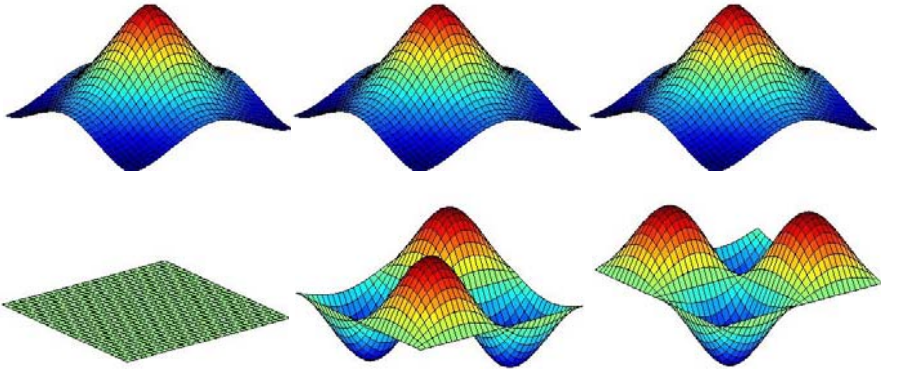


Fig. 2. The vector-valued Green function $\boldsymbol{G} = (G_1, G_2)$. Top panel, left-to-right: $G_{1x}^{M=t}, G_{1x}^{M=r}, G_{1x}^{M=e}$ for the translation, rotation, and expansion models. Bottom panel: left-to right: $G_{2x}^{M=t}, G_{2x}^{M=r}, G_{2x}^{M=e}$ for translation, rotation, and expansion models. Observe that the G_{1x}^M are similar for all models, $G_{2x}^{M=t}$ vanishes for the translation model (i.e. no coupling between velocity components), and $G_{2x}^{M=r}$ and $G_{2x}^{M=e}$ both have two peaks which correspond to the two directions of rotation and expansion. Recall that $G_{1x}^M = G_{2y}^M$ and $G_{2x}^M = G_{1y}^M$.

Our analysis, see Appendix A, shows that the estimated velocities for model M can be expressed in terms of the Greens functions $\boldsymbol{G}_1^M, \boldsymbol{G}_2^M$ of the model and coefficients $\{\alpha_i\}, \{\beta_i\}$:

$$\boldsymbol{v}(\boldsymbol{r}) = \sum_{i=1}^{N} [\alpha_i \boldsymbol{G}_1^M(\boldsymbol{r} - \boldsymbol{r}_i) + \beta_i \boldsymbol{G}_2^M(\boldsymbol{r} - \boldsymbol{r}_i)], \tag{16}$$

where the coefficients $\{\alpha_i\}, \{\beta_i\}$ are obtained by solving linear equations, derived in appendix (9), which depend on the likelihood function and hence on the form of the stimuli (e.g. dot, grating, or plaid).

For *the dot stimuli*, the $\{\alpha\}, \{\beta\}$ are obtained by solving the linear equations:

$$\sum_{i=1}^{N} [\alpha_j \boldsymbol{G}_1^M(\boldsymbol{r}_i - \boldsymbol{r}_j) + \beta_j \boldsymbol{G}_2^M(\boldsymbol{r}_i - \boldsymbol{r}_j)] + \alpha_i \boldsymbol{e}_1 + \beta_i \boldsymbol{e}_2 = \boldsymbol{u}(r_i),\ i = 1, \ldots N, \tag{17}$$

where $\boldsymbol{e}_1, \boldsymbol{e}_2$ denote the (orthogonal) coordinate axes (i.e. the x and y axes). If we express the $\{\alpha\}, \{\beta\}$ as two N-dimensional vectors A and B (recall N is the number of dots), the $\{u_x\}$ and $\{u_y\}$ as vectors $U = (U_x, U_y)$, and define $N \times N$ matrices $g_{1x}^M, g_{2x}^M, g_{1y}^M, g_{2y}^M$ to have components $G_{1x}^M(\boldsymbol{r}_i - \boldsymbol{r}_j), G_{2x}^M(\boldsymbol{r}_i - \boldsymbol{r}_j), G_{1y}^M(\boldsymbol{r}_i - \boldsymbol{r}_j), G_{2y}^M(\boldsymbol{r}_i - \boldsymbol{r}_j)$ respectively, then we can express these linear equations as:

$$[\begin{pmatrix} g_{1x}^M & g_{2x}^M \\ g_{1y}^M & g_{2y}^M \end{pmatrix} + I] \begin{pmatrix} A \\ B \end{pmatrix} = \begin{pmatrix} U_x \\ U_y \end{pmatrix} \tag{18}$$

For *the grating stimuli*, the $\{\alpha\}, \{\beta\}$ satisfy similar linear equations:

$$[\begin{pmatrix} \tilde{g}_{1x}^M & \tilde{g}_{2x}^M \\ \tilde{g}_{1y}^M & \tilde{g}_{2y}^M \end{pmatrix} + I] \begin{pmatrix} A \\ B \end{pmatrix} = \begin{pmatrix} U_x \\ U_y \end{pmatrix} \tag{19}$$

where $\tilde{g}_{1x}^M$ is the matrix with components $\tilde{G}_{1x}^M(\boldsymbol{r}_i - \boldsymbol{r}_j) = [\boldsymbol{G}_1^M(\boldsymbol{r}_i - \boldsymbol{r}_j) \cdot \hat{\boldsymbol{u}}(r_i)] \hat{\boldsymbol{u}}_x(r_i)$, and similarly for $\tilde{g}_{1y}^M$, $\tilde{g}_{2x}^M$ and $\tilde{g}_{2y}^M$.

For *the plaid stimuli*, the $\{\alpha\}, \{\beta\}$ satisfy:

$$(\begin{pmatrix} \tilde{g}_{1x}^M & \tilde{g}_{2x}^M \\ \tilde{g}_{1y}^M & \tilde{g}_{2y}^M \end{pmatrix}_{u_1} + \begin{pmatrix} \tilde{g}_{1x}^M & \tilde{g}_{2x}^M \\ \tilde{g}_{1y}^M & \tilde{g}_{2y}^M \end{pmatrix}_{u_2} + I) \begin{pmatrix} A \\ B \end{pmatrix} = \begin{pmatrix} U_{1x} \\ U_{1y} \end{pmatrix} + \begin{pmatrix} U_{2x} \\ U_{2y} \end{pmatrix}, \tag{20}$$

where U_1, U_2 are the velocity components of the two gratings respectively.

These equations (18,19,20) can all be solved by matrix inversion to determine the coefficients $\{\alpha\}, \{\beta\}$. These solutions can be substituted into equation (16) to obtain the velocity field.

4.4 Model Selection

We re-express model evidence $p(\{\boldsymbol{u}\}|M)$ in terms of (A, B):

$$p(\{\boldsymbol{u}\}|M) = \int p(\{\boldsymbol{u}\}|A, B, M) p(A, B) dA dB, \tag{21}$$

where $p(A, B)$ is the uniform distribution.

The model evidence can be computed analytically (exploiting properties of multi-dimensional Gaussians), see Appendix B. The result can be expressed compactly by introducing new notation in the form of $2N \times 2N$ matrices:

$$g^M = \begin{pmatrix} g_{1x}^M & g_{2x}^M \\ g_{1y}^M & g_{2y}^M \end{pmatrix} \quad \tilde{g}^M = \begin{pmatrix} \tilde{g}_{1x}^M & \tilde{g}_{2x}^M \\ \tilde{g}_{1y}^M & \tilde{g}_{2y}^M \end{pmatrix}.$$

For *the dot stimuli* this gives:

$$p(\{\boldsymbol{u}\}|M) = \frac{1}{(\pi T)^N \sqrt{\det(g^M + I)}} \exp[-\frac{1}{T}(U^T U - U^T \frac{g^M}{g^M + I} U)] \quad (22)$$

Similarly, for *the gratings stimuli* we obtain:

$$p(\{\boldsymbol{u}\}|M) = \frac{1}{(\pi T)^N} \frac{\sqrt{\det(g^M)}}{\sqrt{\det(\tilde{\Sigma})}} \exp[-\frac{1}{T}(U^T U - U^T \tilde{g}^M \tilde{\Sigma}^{-1} \tilde{g}^M U)] \quad (23)$$

where $\tilde{\Sigma} = \tilde{g}^M \tilde{g}^M + g^M$.

For the *plaids stimuli* we obtain: (we omit the M in g^M to simplify notation.)

$$p(\{\boldsymbol{u}_1\}, \{\boldsymbol{u}_2\}|M) = \frac{1}{(\pi T)^N} \frac{\sqrt{\det(g)}}{\sqrt{\det(\tilde{\Sigma})}} \exp -\frac{1}{T}[U_1^T U_1 + U_2^T U_2 - \Phi^T \tilde{\Sigma}^{-1} \Phi] \quad (24)$$

where $\tilde{\Sigma} = \tilde{g}_1^M \tilde{g}_1^M + \tilde{g}_2^M \tilde{g}_2^M + g^M$ and $\Phi = \tilde{g}_1^T U_1 + \tilde{g}_2^T U_2$.

5 Experiment 1: Random Dot Motion

We first investigate motion perception with the moving dots stimuli (figure 3) used by Freeman and Harris [5]. The stimuli consisted of 128 moving dots in a display window. All the dots in the stimulus had the same speed in all the three motion patterns, including translation, rotation and expansion.

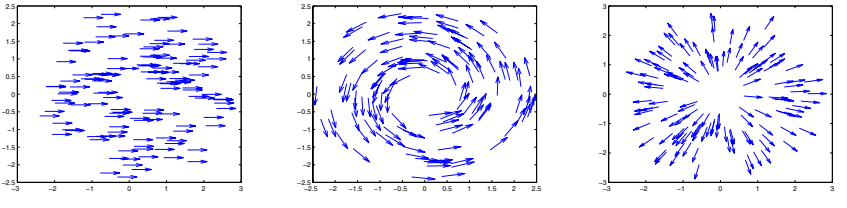


Fig. 3. Moving random dot stimuli – translation, rotation and expansion

The reported simulations are based upon 1000 trials for each speed condition. The parameter values used in the simulations are $\lambda = 0.001$, $\mu = 12.5$, $\eta = 78.125$ and $T = 0.0054$.

Model selection results are shown in the panels of figure (4). It is shown that the correct model is always selected over the entire range of speed, and for all 3 type of motion stimuli.

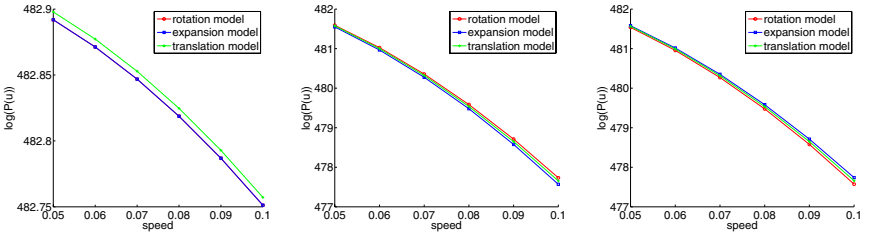


Fig. 4. Model selection results in Experiment 1 with random dot motion. Plots the log probability of the model as a function of speed for each type of stimuli. left: translation; middle: rotation; right: expansion. Green curves with cross are from translation model. Red curves with circles are from rotation model. Blue curves with squares are from expansion model.

6 Experiment 2: Randomly Oriented Gratings

6.1 Stimulus

When randomly oriented grating elements drift behind apertures, the perceived direction of motion is heavily biased by the orientation of the gratings, as well as by the shape and contrast of the apertures. Recently, Nishida and his colleagues developed a novel global motion stimulus consisting of a number of gratings elements, each with randomly assigned orientation [13]. Lee and his colleagues [9] extended this stimulus to rotational and radial motion. A coherent motion in this type of stimuli is perceived when the drifting velocities, also termed the component of velocity, of elements are consistent with a given global motion pattern. Examples of the stimuli used in these psychophysical experiments are shown in left side of figure (5). The stimuli consisted of 728 gratings (drifting sine-wave gratings windowed by stationary Gaussians). The orientations of the gratings were randomly assigned, and their drifting velocities were determined by a specified global motion. The motions of signal grating elements were consistent with global motion, but the motions of noise grating elements were randomized. The task was to identify the global motion direction as one of two alternatives: left/right for translation, clockwise/counterclockwise for rotation, and inward/outward for expansion. Motion sensitivity was measured by the coherence threshold, defined as the proportion of signal elements that yielded a performance level of 75% correct.

Similar stimuli with 328 gratings were generated to test our computational models. The smaller grating number is used to shorten simulation time, and experiments show that model behavior remains the same when grating number increases. The input for the models is the component velocity perpendicular to the assigned orientation for each grating. Two levels of inference are involved. Our simulations first select the correct model for each stimulus and then estimate the velocity flow using the selected model.

All reported simulations in Experiment 2-4 are based up 50 trials for each level of coherence ratio. The parameter values used in the simulations are $\mu = 12.5$,

Table 1. Model selection result for the grating rotation stimuli. The values are logarithms of model evidence. The correct model, rotation model, always wins. As the coherence ratio increases, the rotation model's advantage also increases.

coherence ratio	10%	20%	30%	40%	50%
rotation model	1009.0	1060.1	1146.3	1256.9	1418.8
expansion model	1008.9	1059.5	1144.9	1254.4	1414.7
translation model	1008.9	1059.7	1145.5	1255.5	1416.5

Table 2. Model selection result for the grating expansion stimuli. The correct model, expansion, always wins.

coherence ratio	10%	20%	30%	40%	50%
rotation model	1011.5	1057.4	1151.8	1275.9	1430.0
expansion model	1011.7	1058.0	1153.4	1278.5	1434.2
translation model	1011.6	1057.6	1152.5	1277.1	1431.9

Table 3. Model selection result for the grating translation stimuli. All three models (rotation/expansion/translation) have virtually the same model evidence. This is due to the fact that rotation/expansion models also favor translation as translation model does.

coherence ratio	10%	20%	30%	40%	50%
rotation model	990.2	1076.3	1159.7	1265.6	1456.1
expansion model	990.2	1076.2	1159.7	1265.7	1456.3
translation model	990.2	1076.2	1159.7	1265.6	1456.2

$\eta = 78.125$, $v = 0.0244$ and $T = 4e-5$. For rotation/expansion model, $\lambda = 0.005$; for translation model, $\lambda = 0.0025$.

6.2 Model Selection Results

Model selection results are shown in tables (1,2,3). From the table we see that the rotation model has the highest model evidence for the rotation stimulus over the entire range of the coherence ratio. As coherence ratio increases, the advantage of rotation model over the other two models also increases. Analogous results are found for the expansion stimuli. However, for translation stimulus, all models have very similar model evidence values. This is caused by translation being favored by all three models, and indeed they all produce very similar motion field estimates.

6.3 Comparison between Human Performance and Model Prediction

Once the appropriate model has been selected, the velocity flow is estimated using the selected model (more specifically, this is computed from equations (16,17) with the appropriate M).

The post-processing in Experiments 2 4 includes a simple decision rule the input of which is the velocity flow estimated by the selected model.First the selected model estimates velocity flow in each trial. At each level of coherence ratio in the range of 0 to 50%, the post-processing enables the model to predict accuracy in discriminating motion direction for each stimulus (e.g., left vs. right for translation, clockwise vs. counter-clockwise for rotation, and inward vs. outward for expansion). To address this task, we pool estimated motion directions of 328 elements to generate a motion direction histogram for each trial. Note we use rotation direction and expansion direction to generate histograms. We then compute the average direction histogram over 50 trials for stimuli with one moving direction (e.g., $P(\boldsymbol{v}|L)$ for leftward translation), and the average direction histogram over 50 trials for stimuli with the opposite moving direction, $P(\boldsymbol{v}|R)$ for rightward translation. Based on the two computed histograms, a decision boundary is searched to achieve maximum accuracy. We apply the same post-processing on each of the three motion patterns to estimate accuracy at each level of coherence ratio, and then estimate the coherence threshold needed to achieve 75% correct in the discrimination task.

The results of psychophysical experiments (middle panel of figure 5) showed worse performance for perceiving translation than rotation/expansion motion [9]. Clearly, as shown in the third panel of the same figure, the model performs best for rotation and expansion, and is worst for translation. This finding agrees with human performance in psychophysical experiments.

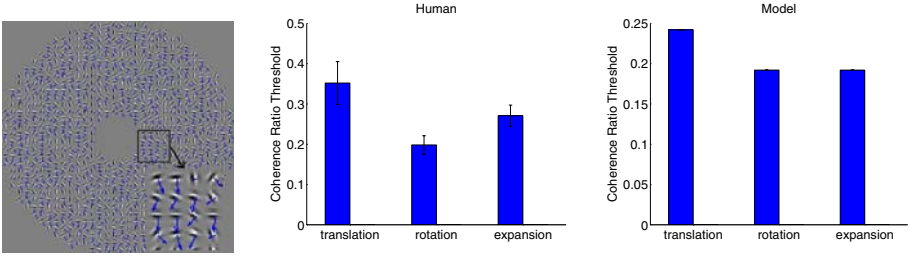


Fig. 5. Stimulus and results in Experiment 2 with randomly-oriented grating stimuli. Left panel: illustration of grating stimulus. Blue arrows indicate the drifting velocity of each grating. Middle panel: human coherence thresholds for different motion stimuli. Right panel: Model prediction of coherence thresholds which are consistent with human trends.

7 Experiment 3: Plaid Stimuli

7.1 Stimulus

A plaid pattern consists of two drifting gratings arranged such that their individual orientations are orthogonal (see illustration in the left panel of figure 6). The drifting velocities of the two gratings in one plaid are determined by the global

motion and their orientations. As the point of intersection for two superimposed gratings provides a tracking feature, the plaid stimulus reduces ambiguity in the measurement of local motion. The experimental methods were the same as those used in Experiment 2, except plaid elements were replaced by grating elements.

7.2 Results

As shown in the middle panel of figure 6, human observers exhibited the worst performance (highest thresholds) in discriminating motion directions in translation, as compared to rotation and radial motion. The right panel of figure 6 shows the model's performance, which matches human performance quantitatively. Notably, the model predicts better performance for plaid stimuli (Exp 3) than for grating stimuli (Exp 2), which is consistent with human performan.

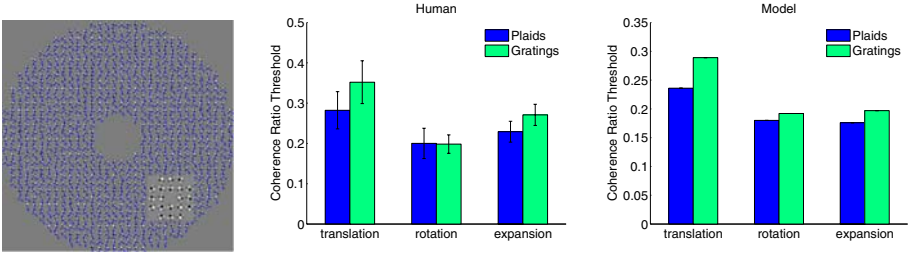


Fig. 6. Stimulus and results in plaid experiment in Experiment 3 with plaid stimuli. Left panel: illustration of plaid stimulus. Blue arrows indicate the drifting velocity of each grating. Middle panel: human coherence thresholds for different motion stimuli. Right panel: Model prediction of coherence thresholds which are consistent with human trends.

8 Experiment 4: Results for Non-rigid Rotation and Radial Motion

8.1 Stimulus

In each of the previous three experiments, all the elements were assigned the same speed, which yielded non-rigid rotation and radial motion. However, in a rigid motion pattern, the assigned speed for each element should vary according to the distance to the center of rotation and radial motion. In Experiment 4, we assessed whether rigidity could affect human motion performance in perceiving rotation and radial motion. We introduced a rigid rotation condition, in which element speeds were determined by a constant rotation velocity and the distance of the element to the rotation center. The average speed was kept the same as that used in non-rigid rotation condition. Similar manipulations were employed for the rigid radial-motion condition. The experimental methods were the same as those employed in Experiment 2 with the grating stimulus.

8.2 Results

As shown in figure 7 (left panel), human observers were not sensitive to rigid versus non-rigid rotation and radial motion (in the current experimental conditions). This result is consistent with previous findings reported in the literature [1]. The right panel in figure 7 shows the model's predictions, which qualitatively match human performance across the four conditions. Although the rotation and expansion priors were motivated by rigid rotation and expansion, the results in Experiment 4 clearly show that the priors are "robust" to non-rigid cases.

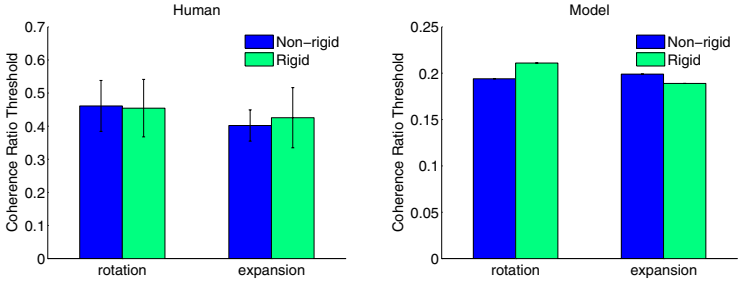


Fig. 7. Results in Experiment 4 with grating stimuli to compare rigid versus nonrigid rotation and expansion. Left panel: human coherence thresholds for rigid and non-rigid conditions as a function of different motion patterns. Right panel: Model prediction of coherence thresholds which are consistent with human trends.

9 Conclusion

We propose a computational theory for motion perception which proceeds in two stages. The first stage selects the best model to explain the data from a set of competing models. The second stage estimates the velocity field using the selected model. In this paper, we used three competing models for translation, rotation, and expansion. We defined novel priors for rotation and expansion by analogy to the slow-and-smooth prior proposed by Yuille and Grzywacz [23,24] and showed that the velocity estimation could be expressed as a linear combination of Green functions. We showed that model selection correctly selects the appropriate model for several different stimulus conditions and that the selected model estimates a plausible velocity field.

We compared our *competitive prior* theory to the performance of human subjects on threshold tasks using different types of stimuli. Our results on random dot stimuli were in general agreement with the experimental findings of Freeman and Harris [5]. We also found good agreement with the experimental findings of our group [9] for grating and plaid stimuli.

Our current work aims to extend these findings to a range of different motions (e.g. affine motion) and to use increasingly naturalistic appearance/intensity models. It is also important to determine if motion patterns to which humans are sensitive correspond to those appearing regularly in natural motion sequences.

Acknowledgments. We gratefully acknowledge funding from NSF 0736015 and 0613563.

References

1. Barraza, J.F., Grzywacz, N.M.: Measurement of angular velocity in the perception of rotation. Vision Research 42 (2002)
2. Black, M.J., Anandan, P.: The Robust Estimation of Multiple Motions: Parametric and Piecewise-Smooth Flow Fields. Computer Vision and Image Understanding 63(1), 75–104 (1996)
3. Duchon, J.: In: Schempp, W., Zeller, K. (eds.). Lecture Notes in Math., vol. 571, pp. 85–100. Springer, Berlin (1977)
4. Duffy, C.J., Wurtz, R.H.: Sensitivity of MST neurons to optic flow stimuli. I. A continuum of response selectivity to large field stimuli. Journal of Neurophysiology 65, 1329–1345 (1991)
5. Freeman, T., Harris, M.: Human sensitivity to expanding and rotating motion: effect of complementary masking and directional structure. Vision research 32 (1992)
6. Hildreth, E.C.: Computations Underlying the Measurement of Visual Motion. Artificial Intelligence 23(3), 309–354 (1984)
7. Horn, B., Schunck, B.: Determining Optical Flow. Artificial Intelligence 17 (1981)
8. Knill, D., Richards, W. (eds.): Perception as Bayesian Inference. Cambridge University Press, Cambridge (1996)
9. Lee, A., Yuille, A., Lu, H.: Superior perception of circular/radial than translational motion cannot be explained by generic priors. In: VSS 2008 (2008)
10. Lu, H., Yuille, A.L.: Ideal Observers for Detecting Motion: Correspondence Noise. In: NIPS 2005 (2005)
11. Morrone, M.C., Burr, D.C., Vaina, L.: Two stages of visual processing for radial and circular motion. Nature 376, 507–509 (1995)
12. Morrone, M., Tosetti, M., Montanaro, D., Fiorentini, A., Cioni, G., Burr, D.C.: A cortical area that responds specifically to optic flow revealed by fMRI. Nature Neuroscience 3, 1322–1328 (2000)
13. Nishida, S., Amano, K., Edwards, M., Badcock, D.R.: Global motion with multiple Gabors - A tool to investigate motion integration across orientation and space. In: VSS 2006 (2006)
14. Ramachandran, V.S., Anstis, S.M.: The perception of apparent motion. Scientific American 254, 102–109 (1986)
15. Roth, S., Black, M.J.: On the Spatial Statistics of Optical Flow. In: International Conference of Computer Vision (2005)
16. Sekuler, R., Watamaniuk, S.N.J., Blake, R.: Perception of Visual Motion. In: Pashler, H. (ed.) Steven's Handbook of Experimental Psychology, 3rd edn., J. Wiley Publishers, New York (2002)
17. Stocker, A.A., Simoncelli, E.P.: Noise characteristics and prior expectations in human visual speed perception. Nature Neuroscience 9(4), 578–585 (2006)
18. Stocker, A.A., Simoncelli, E.: A Bayesian model of conditioned perception. In: Proceedings of Neural Information Processing Systems (2007)
19. Tanaka, K., Fukada, Y., Saito, H.: Underlying mechanisms of the response specificity of expansion/contraction and rotation cells in the dorsal part of the MST area of the macaque monkey. Journal of Neurophysiology 62, 642–656 (1989)
20. Ullman, S.: The Interpretation of Structure from Motion. PhD Thesis. MIT (1979)

21. Weiss, Y., Adelson, E.H.: Slow and smooth: A Bayesian theory for the combination of local motion signals in human vision Technical Report 1624. Massachusetts Institute of Technology (1998)
22. Weiss, Y., Simoncelli, E.P., Adelson, E.H.: Motion illusions as optimal percepts. Nature Neuroscience 5, 598–604 (2002)
23. Yuille, A.L., Grzywacz, N.M.: A computational theory for the perception of coherent visual motion. Nature 333, 71–74 (1988)
24. Yuille, A.L., Grzywacz, N.M.: A Mathematical Analysis of the Motion Coherence Theory. International Journal of Computer Vision 3, 155–175 (1989)
25. Yuille, A.L., Ullman, S.: Rigidity and Smoothness of Motion: Justifying the Amoothness Assumption in Motion Measurement. In: Ullman, S., Richards, W. (eds.) Image Understanding 1989, ch. 8 (1989)

Appendix A

This appendix gives the derivations of the formulae for estimating velocity and for performing model selection.

A1. MAP Estimate of the Velocity

This subsection justifies that the MAP estimate of the velocity is of form:

$$\boldsymbol{v}(\boldsymbol{r}) = \sum_{i=1}^{N} [\alpha_i \boldsymbol{G}_1^M(\boldsymbol{r} - \boldsymbol{r}_i) + \beta_i \boldsymbol{G}_2^M(\boldsymbol{r} - \boldsymbol{r}_i)]$$

as stated in equation (16), and the $\{\alpha\}, \{\beta\}$ are given by equations (17) for the dot, grating, and plaid stimuli. This generalizes the derivation of the original slow-and-smooth model [23,24].

A1.1 Re-express the Prior by a Differential Operator $\mathcal{D}$. We use the quadratic form of the energy functions of the priors, see equations (7, 8, 9), to re-express them in terms of a differential operator $\mathcal{D}_{\mathcal{M}}$ indexed by the model M.

$$E(\{\boldsymbol{v}\}|M) = \int \boldsymbol{v} \cdot (\mathcal{D}_{\mathcal{M}} \boldsymbol{v}) d\boldsymbol{r}. \tag{25}$$

This re-expression is performed by functional integration (assuming suitable boundary conditions as $|\boldsymbol{r}| \mapsto \infty$) and yields:

$$\int |\boldsymbol{v}|^2 d\boldsymbol{r} = \int \boldsymbol{v}^T \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \boldsymbol{v} d\boldsymbol{r}$$

$$\int (|\nabla v_x|^2 + |\nabla v_y|^2) d\boldsymbol{r} = -\int \boldsymbol{v}^T \begin{pmatrix} \nabla^2 & 0 \\ 0 & \nabla^2 \end{pmatrix} \boldsymbol{v} d\boldsymbol{r}$$

$$\int 2 \frac{\partial v_x}{\partial y} \frac{\partial v_y}{\partial x} d\boldsymbol{r} = -\int \boldsymbol{v}^T \begin{pmatrix} 0 & \frac{\partial^2}{\partial x \partial y} \\ \frac{\partial^2}{\partial x \partial y} & 0 \end{pmatrix} \boldsymbol{v} d\boldsymbol{r} = \int 2 \frac{\partial v_x}{\partial x} \frac{\partial v_y}{\partial y} d\boldsymbol{r}$$

$$\int |\nabla^2 \boldsymbol{v}|^2 d\boldsymbol{r} = \int \boldsymbol{v}^T \begin{pmatrix} (\nabla^2)^2 & 0 \\ 0 & (\nabla^2)^2 \end{pmatrix} \boldsymbol{v} d\boldsymbol{r}$$

Combining the terms for the different priors $E(\boldsymbol{v}|M)$ gives the operators:

$$\mathcal{D}_t = \lambda(I - \mu \begin{pmatrix} \nabla^2 & 0 \\ 0 & \nabla^2 \end{pmatrix} + \eta \begin{pmatrix} (\nabla^2)^2 & 0 \\ 0 & (\nabla^2)^2 \end{pmatrix})$$

$$\mathcal{D}_r = \lambda(I - \mu \begin{pmatrix} \nabla^2 & \frac{\partial^2}{\partial x \partial y} \\ \frac{\partial^2}{\partial x \partial y} & \nabla^2 \end{pmatrix} + \eta \begin{pmatrix} (\nabla^2)^2 & 0 \\ 0 & (\nabla^2)^2 \end{pmatrix})$$

$$\mathcal{D}_e = \lambda(I - \mu \begin{pmatrix} \nabla^2 & -\frac{\partial^2}{\partial x \partial y} \\ -\frac{\partial^2}{\partial x \partial y} & \nabla^2 \end{pmatrix} + \eta \begin{pmatrix} (\nabla^2)^2 & 0 \\ 0 & (\nabla^2)^2 \end{pmatrix})$$

A1.2 The Euler-Lagrange Equations

The MAP estimate $\{\boldsymbol{v}\}$ for model M, $\{\boldsymbol{v}^*\} = \arg\max_{\{\boldsymbol{v}\}} P(\{\boldsymbol{v}\}|\{\boldsymbol{u}\}, M)$, is obtained by solving the Euler-Lagrange equations for the functional $E(\{\boldsymbol{u}\}|\{\boldsymbol{v}\}) + E(\{\boldsymbol{v}\}|M)$. This equation will depend on the form of the stimuli (e.g. dots and gratings).

For the dot stimuli, we express the functional in the form:

$$E(\{\boldsymbol{u}\}|\{\boldsymbol{v}\}) + E(\{\boldsymbol{v}\}|M) = \int \sum_{i=1}^{N} |\boldsymbol{v}(\boldsymbol{r}) - \boldsymbol{u}(\boldsymbol{r}_i)|^2 \delta(\boldsymbol{r} - \boldsymbol{r}_i) d\boldsymbol{r} + \int \boldsymbol{v} \cdot (\mathcal{D}_{\mathcal{M}} \boldsymbol{v}) d\boldsymbol{r} \tag{26}$$

This gives Euler-Lagrange equation:

$$\sum_{i=1}^{N} 2[\boldsymbol{v}(\boldsymbol{r}) - \boldsymbol{u}(\boldsymbol{r}_i)]\delta(\boldsymbol{r} - \boldsymbol{r}_i) + 2\mathcal{D}_{\mathcal{M}} \boldsymbol{v}(\boldsymbol{r}) = 0. \tag{27}$$

The solution can be obtained (extending [3] and [24]) by assuming a solution expressed as linear combination of vector valued Green's functions G_1^M, G_2^M:

$$\boldsymbol{v}(\boldsymbol{r}) = \sum_{i=1}^{N} [\alpha_i \boldsymbol{G}_1^M(\boldsymbol{r} - \boldsymbol{r}_i) + \beta_i \boldsymbol{G}_2^M(\boldsymbol{r} - \boldsymbol{r}_i)], \tag{28}$$

where $\{\alpha\}, \{\beta\}$ are coefficients and G_1^M, G_2^M are the solutions of

$$\mathcal{D}_M G_1(\boldsymbol{r}) = \begin{pmatrix} \delta(0) \\ 0 \end{pmatrix} = \boldsymbol{e}_1, \ \mathcal{D}_M G_2(\boldsymbol{r}) = \begin{pmatrix} 0 \\ \delta(0) \end{pmatrix} = \boldsymbol{e}_2$$

The form of the priors, and hence the Green functions, ensures that this solution is well-posed at the data points and falls off smoothly as $\boldsymbol{r} \mapsto \infty$. More specifically: (i) the second order derivative terms in the priors ensure that $G_1(0)$ and $G_2(0)$ are finite [24], and (ii) the slowness term ensures that the Green functions decrease to zero as $|\boldsymbol{r}| \mapsto \infty$ [24].

We obtain the linear equations for $\{\alpha\}, \{\beta\}$, given by equation (17), by substituting equation (28) into the Euler-Lagrange equations (27), using the Green functions from equation (9).

For the grating stimulus, we obtain similar results. The energy functional becomes:

$$E(\{\boldsymbol{u}\}|\{\boldsymbol{v}\}) + E(\{\boldsymbol{v}\}|M) = \int \sum_{i=1}^{N} |\boldsymbol{v}(\boldsymbol{r}) \cdot \hat{\boldsymbol{u}}(\boldsymbol{r}_i) - u(\boldsymbol{r}_i)|^2 \delta(\boldsymbol{r} - \boldsymbol{r}_i) d\boldsymbol{r} + \int \boldsymbol{v} \cdot (\mathcal{D}_{\mathcal{M}} \boldsymbol{v}) d\boldsymbol{r}. \tag{29}$$

This gives Euler-Lagrange equations:

$$\sum_{i=1}^{N} \{[\boldsymbol{v}(\boldsymbol{r}) \cdot \hat{\boldsymbol{u}}(\boldsymbol{r}_i)]\hat{\boldsymbol{u}}(\boldsymbol{r}_i) - u(\boldsymbol{r}_i)\}\delta(\boldsymbol{r} - \boldsymbol{r}_i) + \mathcal{D}_M \boldsymbol{v}(\boldsymbol{r}) = 0. \tag{30}$$

The solution $\boldsymbol{v}$ can be expressed in form given by equation (28) with the same Green functions from equation (9). Substituting the form of $\boldsymbol{v}$ into the Euler-Lagrange equation (30) gives the linear equation (19) for the $\{\alpha\}, \{\beta\}$.

A1.3. Solving for the Green Functions. We solve equation (9) for the Green functions by Fourier transforms. Denote $G_1(\boldsymbol{r}) = (G_{1x}(\boldsymbol{r}),\ G_{1y}(\boldsymbol{r}))^T$ (we drop the model index ^{M} to simplify notation). This gives the following solutions for the three different cases:

1. *Slow-and-smooth-translation*: We apply Fourier transforms to the equation for the Green functions:

$$\lambda \begin{pmatrix} 1 - \mu\nabla^2 + \eta(\nabla^2)^2 & 0 \\ 0 & 1 - \mu\nabla^2 + \eta(\nabla^2)^2 \end{pmatrix} \begin{pmatrix} G_{1x} \\ G_{1y} \end{pmatrix} = \begin{pmatrix} \delta(0) \\ 0 \end{pmatrix}$$

to obtain linear equations for the Fourier transforms $\mathcal{F}G(\omega_x, \omega_y)$ (where $\omega^2 = \omega_x^2 + \omega_y^2$):

$$\lambda \begin{pmatrix} 1 + \mu\omega^2 + \eta\omega^4 & 0 \\ 0 & 1 + \mu\omega^2 + \eta\omega^4 \end{pmatrix} \begin{pmatrix} \mathcal{F}G_{1x} \\ \mathcal{F}G_{1y} \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

which is solved to give

$$\begin{pmatrix} \mathcal{F}G_{1x} \\ \mathcal{F}G_{1y} \end{pmatrix} = \frac{1}{\lambda} \begin{pmatrix} \frac{1}{1+\mu\omega^2+\eta\omega^4} \\ 0 \end{pmatrix}$$

with similar solution for $\mathcal{F}G_2$.

2. *Slow-and-smooth-rotation*: we re-express the Green function equation in fourier space

$$\lambda \begin{pmatrix} 1 - \mu\nabla^2 + \eta(\nabla^2)^2 & -\mu\frac{\partial^2}{\partial x \partial y} \\ -\mu\frac{\partial^2}{\partial x \partial y} & 1 - \mu\nabla^2 + \eta(\nabla^2)^2 \end{pmatrix} \begin{pmatrix} G_{1x} \\ G_{1y} \end{pmatrix} = \begin{pmatrix} \delta(0) \\ 0 \end{pmatrix}$$

$$\lambda \begin{pmatrix} 1+\mu\omega^2+\eta\omega^4 & \mu\omega_x\omega_y \\ \mu\omega_x\omega_y & 1+\mu\omega^2+\eta\omega^4 \end{pmatrix} \begin{pmatrix} \mathcal{F}G_{1x} \\ \mathcal{F}G_{1y} \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

defining

$$d_r = \det[\begin{pmatrix} 1+\mu\omega^2+\eta\omega^4 & -\mu\omega_x\omega_y \\ -\mu\omega_x\omega_y & 1+\mu\omega^2+\eta\omega^4 \end{pmatrix}]$$

gives the solution

$$\begin{pmatrix} \mathcal{F}G_{1x} \\ \mathcal{F}G_{1y} \end{pmatrix} = \frac{1}{\lambda d_r} \begin{pmatrix} 1+\mu\omega^2+\eta\omega^4 & -\mu\omega_x\omega_y \\ -\mu\omega_x\omega_y & 1+\mu\omega^2+\eta\omega^4 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

3. *Slow-and-smooth-expansion*: this is a simple modification of the rotation case:

$$\lambda \begin{pmatrix} 1+\mu\nabla^2+\eta(\nabla^2)^2 & -\mu\frac{\partial^2}{\partial x\partial y} \\ -\mu\frac{\partial^2}{\partial x\partial y} & 1+\mu\nabla^2+\eta(\nabla^2)^2 \end{pmatrix} \begin{pmatrix} \mathcal{F}G_{1x} \\ \mathcal{F}G_{1y} \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

yields the solution

$$\begin{pmatrix} \mathcal{F}G_{1x} \\ \mathcal{F}G_{1y} \end{pmatrix} = \frac{1}{\lambda d_e} \begin{pmatrix} 1+\mu\omega^2+\eta\omega^4 & \mu\omega_x\omega_y \\ \mu\omega_x\omega_y & 1+\mu\omega^2+\eta\omega^4 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

Note that the following equalities hold for both rotation and expansion:

$$G_{1x}(x,y) = G_{2y}(y,x),\ G_{1y}(x,y) = G_{2x}(x,y)$$

A2. Model Selection

The model evidence is

$$p(\{\boldsymbol{u}\}) = \int p(\{\boldsymbol{u}\}|\{\boldsymbol{v}\})p(\{\boldsymbol{v}\})d\{\boldsymbol{v}\}$$

which we re-express in term of (A, B)

$$p(\{\boldsymbol{u}\}) = \int p(\{\boldsymbol{u}\}|A,B)p(A,B)dAdB$$

in which

$$p(\{\boldsymbol{u}\}|A,B) = \frac{1}{Z_1}\exp(-E(\{\boldsymbol{u}\}|A,B)/T),\ p(A,B) = \frac{1}{Z_2}\exp(-E(A,B)/T)$$

where, for the random dot stimuli,

$$E(\{\boldsymbol{u}\}|A,B) = \sum_{i=1}^{N} |\boldsymbol{v}(\boldsymbol{r}_i) - \boldsymbol{u}(\boldsymbol{r}_i)|^2 = \Phi^T G^2 \Phi - 2U^T G\Phi + U^T U, \tag{31}$$

where $\Phi = (AB)^T$.

We can re-express this as:

$$E(A,B) = \Phi^T G \Phi,$$

using $\boldsymbol{v}(\boldsymbol{r}) = \sum_{i=1}^{N}[\alpha_i G_1(\boldsymbol{r}-\boldsymbol{r}_i) + \beta_i G_2(\boldsymbol{r}-\boldsymbol{r}_i)]$ and $\mathcal{D}\boldsymbol{v}(\boldsymbol{r}) = \sum_{i=1}^{N}[\alpha_i \delta(\boldsymbol{r}-\boldsymbol{r}_i)e_1 + \beta_i \delta(\boldsymbol{r}-\boldsymbol{r}_i)e_2]$.

The partition functions are of form:

$$Z_1 = \int \exp(-E(\{u\}|A,B)/T)d\{u\} = \int \exp(-(U-V)^T(U-V)/T)dU = (\pi T)^N$$

$$Z_2 = \int \exp(-E(A,B)/T)dAdB = \int \exp(-\Phi^T G\Phi/T)d\Phi = \frac{(\pi T)^N}{\sqrt{\det(G)}}$$

We now re-express the model evidence $p(\{\boldsymbol{u}\})$,

$$\begin{aligned} p(\{\boldsymbol{u}\}) &= \int \frac{1}{Z_1}\exp(-E(\{\boldsymbol{u}\}|A,B)/T)\frac{1}{Z_2}\exp(-E(A,B)/T)dAdB \\ &= \frac{1}{Z_1 Z_2}\int \exp[-\frac{1}{T}(\Phi^T(G^2+G)\Phi - 2U^T G\Phi + U^T U)]d\Phi \end{aligned}$$

denoting

$$\Sigma = G^2 + G = GH,\ H = G + I,\ \Theta = \Sigma^{-1}GU = H^{-1}U$$

gives

$$\begin{aligned} p(\{\boldsymbol{u}\}) &= \frac{1}{Z_1 Z_2}\int \exp[-\frac{1}{T}(\Phi^T\Sigma\Phi - 2U^T G\Phi + U^T U)]d\Phi \\ &= \frac{1}{Z_1 Z_2}\int \exp[-\frac{1}{T}((\Phi-\Theta)^T\Sigma(\Phi-\Theta) + U^T U - \Theta^T\Sigma\Theta)]d\Phi \\ &= \frac{1}{Z_1 Z_2}\exp[-\frac{1}{T}(U^T U - \Theta^T\Sigma\Theta)]\int \exp\{-\frac{1}{T}[(\Phi-\Theta)^T\Sigma(\Phi-\Theta)]\}d\Phi \\ &= \frac{1}{(\pi T)^N}\frac{\sqrt{\det(G)}}{(\pi T)^N}\exp[-\frac{1}{T}(U^T U - \Theta^T\Sigma\Theta)]\frac{(\pi T)^N}{\sqrt{\det(\Sigma)}} \\ &= \frac{1}{(\pi T)^N\sqrt{\det(G+I)}}\exp[-\frac{1}{T}(U^T U - U^T H^{-1}U)]. \end{aligned}$$

Modification for the Grating Stimulus. In the model selection stage we modify the energy to be,

$$\begin{aligned} E(\{\boldsymbol{u}\}|A,B) &= \sum_{i=1}^{N}|\boldsymbol{v}(\boldsymbol{r}_i)\cdot\hat{\boldsymbol{u}}(\boldsymbol{r}_i) - u(\boldsymbol{r}_i)|^2 \\ &= \Phi^T\tilde{G}^T\tilde{G}\Phi - 2U^T\tilde{G}\Phi + U^T U \end{aligned}$$

$$E(A,B) = \int \boldsymbol{v}\cdot(\mathcal{D}\boldsymbol{v})d\boldsymbol{r} = \Phi^T G\Phi$$

The partition functions Z_1 and Z_2 remain the same. So

$$p(\{\boldsymbol{u}\}) = \int \frac{1}{Z_1}\exp(-E(\{\boldsymbol{u}\}|A,B)/T)\frac{1}{Z_2}\exp(-E(A,B)/T)dAdB$$
$$= \frac{1}{Z_1 Z_2}\int \exp[-\frac{1}{T}(\Phi^T(\tilde{G}^T\tilde{G}+G)\Phi - 2U^T\tilde{G}\Phi + U^TU)]d\Phi$$

denoting

$$\tilde{\Sigma} = \tilde{G}^T\tilde{G} + G,\ \tilde{\Theta} = \tilde{\Sigma}^{-1}\tilde{G}^TU$$

and (noting that $\tilde{\Sigma}$ is symmetric even though $\tilde{G}$ is not) we get:

$$\begin{aligned}
p(\{u\}) &= \frac{1}{Z_1 Z_2}\int \exp[-\frac{1}{T}(\Phi^T\tilde{\Sigma}\Phi - 2U^T\tilde{G}\Phi + U^TU)]d\Phi \\
&= \frac{1}{Z_1 Z_2}\int \exp[-\frac{1}{T}((\Phi-\tilde{\Theta})^T\tilde{\Sigma}(\Phi-\tilde{\Theta}) + U^TU - \tilde{\Theta}^T\tilde{\Sigma}\tilde{\Theta})]d\Phi \\
&= \frac{1}{Z_1 Z_2}\exp[-\frac{1}{T}(U^TU - \tilde{\Theta}^T\tilde{\Sigma}\tilde{\Theta})]\int \exp\{-\frac{1}{T}[(\Phi-\tilde{\Theta})^T\tilde{\Sigma}(\Phi-\tilde{\Theta})]\}d\Phi \\
&= \frac{1}{(\pi T)^N}\frac{\sqrt{\det(G)}}{(\pi T)^N}\exp[-\frac{1}{T}(U^TU - \tilde{\Theta}^T\tilde{\Sigma}\tilde{\Theta})]\frac{(\pi T)^N}{\sqrt{\det(\tilde{\Sigma})}} \\
&= \frac{1}{(\pi T)^N}\sqrt{\det(\frac{G}{\tilde{\Sigma}})}\exp[-\frac{1}{T}(U^TU - U^T\tilde{G}\tilde{\Sigma}^{-1}\tilde{G}^TU)]
\end{aligned}$$

Appendix B: Least-Square Fit of Rotation/Expansion

This appendix answers the question of how to estimate the coefficients of the expansion and rotation motions from the estimated velocity fields. More precisely, we need to estimate the centers (x_c, y_c) of expansion/rotation and the expansion/rotation coefficients e, ω. We perform this by least squares fitting.

B1. Rotation

We need to estimate the center (x_c, y_c) and rotation ω from a velocity field. We define

$$\boldsymbol{u}(x,y) = (\omega(y_c - y),\ \omega(x - x_c))$$

and an energy function:

$$E(x_x, y_c, \omega) = \sum_{i=1}^{N}\{[v_x(\boldsymbol{r}_i) - \omega(y_c - y_i)]^2 + [v_y(\boldsymbol{r}_i) - \omega(x_i - x_c)]^2\}$$

We estimate x_c, y_c and ω by minimizing $E(.,.,.)$. This is performed analytically by solving the equations $\frac{\partial E}{\partial x_c} = \frac{\partial E}{\partial y_c} = \frac{\partial E}{\partial \omega} = 0$. The derivatives with respect to (x_c, y_c) can be computed and set to zero yielding the equations:

$$\frac{\partial E}{\partial x_c} = 2\sum_{i=1}^{N}[v_y(\boldsymbol{r}_i) - \omega(x_i - x_c)] \cdot \omega = 2\omega N(\bar{v}_y - \omega\bar{x} + \omega x_c) = 0$$

$$\frac{\partial E}{\partial y_c} = 2\sum_{i=1}^{N}[v_x(\boldsymbol{r}_i) - \omega(y_c - y_i)] \cdot (-\omega) = -2\omega N(\bar{v}_x - \omega y_c + \omega\bar{y}) = 0$$

This gives the solutions (x_c, y_c) as functions of ω and the means $\bar{x} = 1/N \sum_{i=1}^{N} x_i$, $\bar{y} = 1/N \sum_{i=1}^{N} y_i$,$\bar{v}_x = 1/N \sum_{i=1}^{N} v_x(\boldsymbol{r}_i)$,$\bar{v}_y = 1/N \sum_{i=1}^{N} v_y(\boldsymbol{r}_i)$ of the data points and their estimated velocities. The solutions are given by:

$$\begin{cases} x_c & = \bar{x} - \frac{\bar{v}_y}{\omega} \\ y_c & = \bar{y} + \frac{\bar{v}_x}{\omega} \end{cases}$$

To solve for ω, we calculate $\frac{\partial E}{\partial \omega}$ and set it to zero. This gives

$$\begin{aligned} \frac{\partial E}{\partial \omega} &= -2\sum_{i=1}^{N}\{[v_x(\boldsymbol{r}_i) - \omega(y_c - y_i)](y_c - y_i) + [v_y(\boldsymbol{r}_i) - \omega(x_i - x_c)](x_i - x_c)\} \\ &= -2N[y_c\bar{v}_x - x_c\bar{v}_y - \overline{yv_x} + \overline{xv_y} - \omega(x_c^2 + y_c^2) + 2\omega(x_c\bar{x} + y_c\bar{y}) - \omega(\bar{x^2} + \bar{y^2})] = 0 \end{aligned}$$

Plugging in x_c and y_c yields

$$0 = (\bar{y}\bar{v}_x - \overline{yv_x}) - (\bar{x}\bar{v}_y - \overline{xv_y}) + \omega(\bar{x}^2 + \bar{y}^2) - \omega(\bar{x^2} + \bar{y^2}) \tag{32}$$

hence

$$\omega = \frac{(\overline{xv_y} - \bar{x}\bar{v}_y) - (\overline{yv_x} - \bar{y}\bar{v}_x)}{(\bar{x^2} + \bar{y^2}) - (\bar{x}^2 + \bar{y}^2)}$$

B2. Expansion

Similarly to the rotation case, define an expansion velocity field with three parameters x_c, y_c and k.

$$\boldsymbol{u}(x, y) = (k(x - x_c),\ k(y - y_c))$$

Define an energy

$$E(x_c, y_c, k) = \sum_{i=1}^{N}\{[v_x(\boldsymbol{r}_i) - k(x_i - x_c)]^2 + [v_y(\boldsymbol{r}_i) - k(y_i - y_c)]\}^2$$

The x_c, y_c and k that minimize E satisfy

$$\frac{\partial E}{\partial x_c} = \frac{\partial E}{\partial y_c} = \frac{\partial E}{\partial k} = 0$$

As before, we solve for (x_c, y_c) as functions of k and the means of the data points and their velocities.

$$\begin{aligned}\frac{\partial E}{\partial x_c} &= 2\sum_{i=1}^{N}[v_x(\boldsymbol{r}_i) - k(x_i - x_c)] \cdot k = 2kN(\bar{v}_x - k\bar{x} + kx_c) = 0 \\ \frac{\partial E}{\partial y_c} &= 2\sum_{i=1}^{N}[v_y(\boldsymbol{r}_i) - k(y_i - y_c)] \cdot k = 2kN(\bar{v}_y - k\bar{y} + ky_c) = 0 \end{aligned} \tag{33}$$

This leads to

$$\begin{cases} x_c &= \bar{x} - \frac{\bar{v}_x}{k} \\ y_c &= \bar{y} - \frac{\bar{v}_y}{k} \end{cases}$$

Then we solve for k by setting

$$\begin{aligned}\frac{\partial E}{\partial k} &= -2\sum_{i=1}^{N}\{[v_x(\boldsymbol{r}_i) - k(x_i - x_c)](x_i - x_c) + [v_y(\boldsymbol{r}_i) - k(y_i - y_c)](y_i - y_c)\} \\ &= -2N[\overline{xv_x} - x_c\bar{v}_x + \overline{yv_y} - y_c\bar{v}_y - k(\bar{x^2} + \bar{y^2}) + 2k(x_c\bar{x} + y_c\bar{y}) - k(x_c^2 + y_c^2)] = 0\end{aligned}$$

and plugging in x_c and y_c gives

$$\begin{aligned}0 = \overline{xv_x} + \overline{yv_y} - (\bar{x} - \frac{\bar{v}_x}{k})\bar{v}_x - (\bar{y} - \frac{\bar{v}_y}{k})\bar{v}_y - k(\bar{x^2} + \bar{y^2}) \\ + 2k[(\bar{x} - \frac{\bar{v}_x}{k})\bar{x} + (\bar{y} - \frac{\bar{v}_y}{k})\bar{y}] - k[(\bar{x} - \frac{\bar{v}_x}{k})^2 + (\bar{y} - \frac{\bar{v}_y}{k})^2]\end{aligned}$$

hence

$$k = \frac{(\overline{xv_x} - \bar{x}\bar{v}_x) + (\overline{yv_y} - \bar{y}\bar{v}_y)}{(\bar{x^2} + \bar{y^2}) - (\bar{x}^2 + \bar{y}^2)}$$

Derivation of Motion Characteristics Using Affine Shape Adaptation for Moving Blobs

Jorge Sanchez[1], Reinhard Klette[2], and Eduardo Destefanis[1]

[1] Universidad Tecnológica Nacional, Cordoba, Argentina
[2] The *.enpeda..* Project, The University of Auckland, New Zealand

Abstract. This chapter applies anisotropic Gaussian scale space theory for modeling affine shape modifications of moving blobs in the context of vision-based driver assistance systems. First, affine blobs are detected in an image sequence and tracked; second, their scale ratios are used for the derivation of 3D motion characteristics. For example, this also allows to estimate the navigation angles of a moving camera in 3D space. The theoretical concept is explained in detail, and illustrated by a few experiments, including an indoor experiment and also the estimation of navigation angles of a car (i.e., of the ego-vehicle) in provided test sequences. The numerical evaluations indicate the validity of the idea and advantages to vehicle vision.

1 Introduction

Studies on visual perception [25] indicate that the relative change in size of perceived image features, projected onto the retina of the human eye, is processed independently of perceived optic flow.[1] This change in size constitutes an important source of information for the perception of motion. However, this information is usually not taken into account in standard optic flow or tracking techniques.

This chapter discusses the idea of using the *size* of image 'structures' for understanding three-dimensional (3D) motion. Moreover, if one tries to analyze image patterns, moving in a broad range of scales and for different poses, such as cars moving towards or away from an observer, some special attention must be paid on possible distortions that such patterns suffer when being viewed from arbitrary angles.

Booth problems (i.e., the formalization of the concept of *scale* and the proper handling of local image distortions) can be treated using *scale-space* theory. This theory became a very valuable approach in computer vision in general; see, for example, [2,6,9,11,29].

The basic concept behind the presented approach is to track not just the spatial position, but also scale changes of local image features. More precisely,

[1] Tracking methods for estimating a dense 2D motion field are known in computer vision as *optic flow techniques*.

D. Cremers et al. (Eds.): Visual Motion Analysis, LNCS 5604, pp. 259–279, 2009.

the work focuses on affine blobs, which can (roughly) be understood as being connected (say, convex) regions in an image that are either brighter or darker than the surrounding areas in that image. This is motivated by comparative studies using different types of feature detectors, as presented in [21].

The chapter presents a modified version of the algorithm of [21], which uses the normalized Laplacian operator [11] both for scale selection as for spatial localization of an affine shape. The resulting algorithm allows that the shape of image features can be correctly estimated and tracked.

For tracked blobs, a measure is derived that relates the change in size on the image plane, and the corresponding translation in the 3D-space to each-other. This measure is used to conclude the direction of motion of the camera. This is closely related to the concept of *scene-flow* as estimated, e.g., in [26] for the case of 'non-sparse' representations.

Instead of affine blobs, [24] used characteristic scales assigned to KLT-tracked points. The assigned scales were identified by curve maxima in isotropic Gaussian scale space. This chapter applies anisotropic Gaussian scale space, thus also allowing to track moving blobs which undergo some affine shape transform within the recorded image sequence.

With respect to applications, this chapter is motivated by vision-based driver assistance systems. For example, see [4] for a recent monograph, or [8,23,27] for examples of recent publications on vehicle vision.

This chapter is structured as follows: Section 2 discusses briefly the Gaussian scale-space, illustrating fundamental results that form the theoretical basis of concepts presented in the following sections. In Section 3 the affine shape adaptation algorithm is discussed, recalling a common affine region detector. Improvements are proposed for scale detection and normalization steps of the transformation matrix, as well as the inclusion of some degree of robustness against changes in lighting conditions.

Section 4 specifies then the proposed technique for extracting scene flow vectors from given scale characteristics, also allowing to estimate navigation angles. Section 5 reports about experiments and some evaluations of obtained results based on synthetic sequences (with accompanying ground truth). The chapter concludes with Section 6.

2 Scale-Space Theory and Automatic Scale Selection

Scale-space theory provides a formal treatment of the multiscale nature of images; see [2,6,9,11,29]. In the traditional scale space formulation, an image is represented as a 3D entity for which, besides its spatial coordinates, also various *scales* of the image are represented.

2.1 Linear Scale-Space Representations

For the case of a single channel images, a *linear scale-space* representation can be obtained by means of convolution of its intensity function $f : \mathbb{R}^2 \to \mathbb{R}$ with

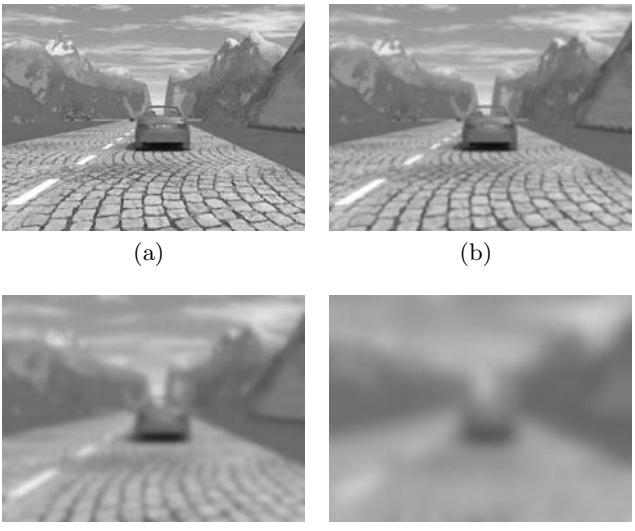


Fig. 1. Scale space representation of the image shown in (a), for scales (b) $\sigma = 1.2^3$, (c) $\sigma = 1.2^5$, and (d) $\sigma = 1.2^7$

Gaussian kernels of the form

$$g_a(\mathbf{q}) = g(\mathbf{q}; a) = \frac{1}{2\pi a} \exp\left(-\frac{\mathbf{q}^T\mathbf{q}}{2a}\right)$$

Note that the limit value $a = 0$ corresponds to the original image. Figure 1 illustrates this representation for one of the images from the synthetic sequence in Set 2 on [5], also used in our tracking experiments later on.

It can be shown, that the linear scale-space representation of an image is invariant under uniform rescaling of the spatial domain [11]. This motivated the formulation of feature detectors and descriptors that are invariant under similarities (rigid 2D Euclidean transformations and scale), see [14,16].

Dealing with real images, shape distortions that arise when a scene is viewed from different viewpoint angles can be approximated by means of 2D projective models. In order to reduce the dimensionality of the model, one common assumption is that 3D scene patches can be locally well approximated by an affine model. That is, the surfaces are smooth enough so that they can be locally considered as being planar. The *affine Gaussian scale-space* [11] extends the (common) isotropic Gaussian scale-space in order to account for a broader class of transformation of the image domain.

As before, let $f : \mathbb{R}^2 \to \mathbb{R}$ denote the intensity function of a gray-level image. Let $\mathcal{M}_{SPD}$ be the family of all symmetric positive definite (SPD) 2×2 matrices. For $\Sigma \in \mathcal{M}_{SPD}$, the affine Gaussian scale space representation of f,

$$L : \mathbb{R}^2 \times \mathcal{M}_{SPD} \to \mathbb{R}$$

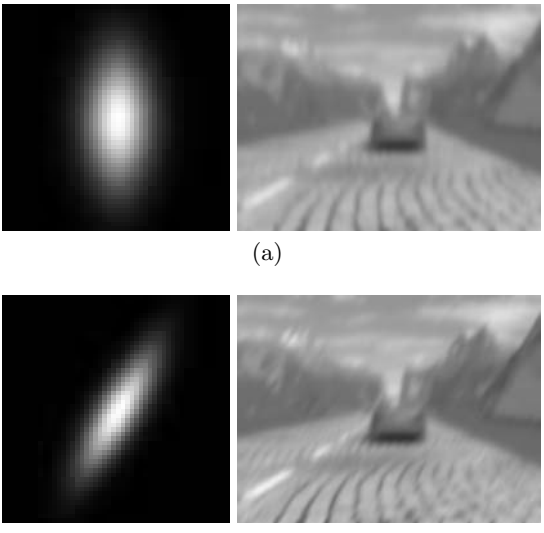


Fig. 2. Two examples of anisotropic smoothing (images on the right), for two different Gaussian kernels (shown on the left)

can be obtained by means of convolutions of the form

$$L(\mathbf{q};\Sigma) = \int_{\xi\in\mathbb{R}^2} g(\mathbf{q}-\xi;\Sigma) f(\xi)\ \mathrm{d}\xi \tag{1.1}$$

for $\mathbf{q} \in \mathbb{R}^2$, where $g(\mathbf{q}-\xi;\Sigma)$ corresponds to the non-uniform Gaussian kernel

$$g(\mathbf{x};\Sigma) = \frac{1}{2\pi\sqrt{\det(\Sigma)}} \exp\left(-\frac{\mathbf{x}^T\Sigma^{-1}\mathbf{x}}{2}\right)$$

for $\mathbf{x} \in \mathbb{R}^2$. Matrix Σ is called the *anisotropic scale*. Figure 2 illustrates two examples of anisotropic Gaussian smoothing.

An important result arises when considering the case of invertible linear transformations of the spatial domain. Let B be such a transformation and $f(\xi) = h(B\xi)$ the transformed image under B. Let L_f and L_h, be the affine-scale space representations of images f and h respectively. It can be shown that, given a scale matrix Σ_f for the first representation, there exist always a matrix Σ_h for which the affine scale space representations of images f and h are identical. This matrix is given by $\Sigma_h = B\Sigma_f B^T$ (see [11] for details).

2.2 Normalized Derivatives and Scale Selection

In order to infer motion characteristics from perceived changes in the size of intensity patterns, we have to estimate corresponding *scale ratios* between them in some way.

In the context of the linear Gaussian scale space, Lindeberg [11] proposed a method for automatic scale selection based on (possibly non-linear) combinations of normalized derivatives. The idea behind this principle is that for any point $\mathbf{q}$, the scale at which the response of a (possibly non-linear) combination of such derivatives takes a local extrema, may be assumed to reflect the *characteristic scale* of the image data surrounding point $\mathbf{q}$.

One commonly used operator for scale selection is the normalized Laplacian, due to experimental evidence that this operator provides a maximum number of extrema compared to other choices [20]. This operator is also the base of the popular *Difference of Gaussians* (DoG) pyramid used by the SIFT detector [16]. It is defined by the following:

$$\Delta_{norm} L(\mathbf{q}; a) = a \cdot \left| (D_x^2 L)(\mathbf{q}; a) + (D_y^2 L)(\mathbf{q}; a) \right| \tag{1.2}$$

where D_x^2 and D_y^2 are the second order derivatives of L (at scale $\sqrt{a}$). Note that the variance of the kernel is introduced as a normalizing factor. This compensates for the decreasing magnitudes (with scale) of the Gaussian derivatives.

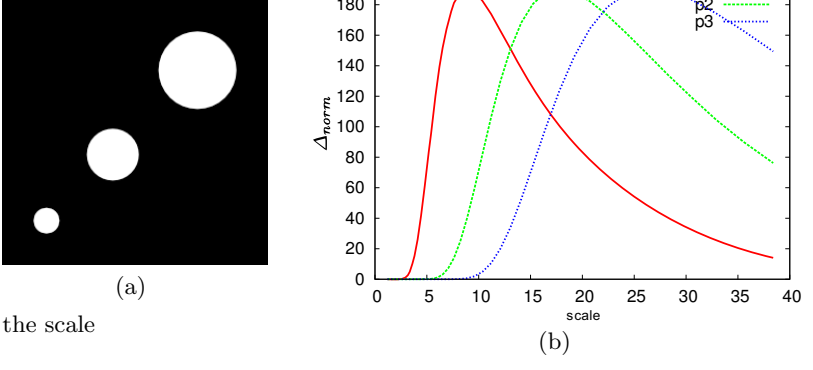


Fig. 3. (a) White disks of radius 25, 50 and 75 pixel. (b) Evolution on the Laplacian for center points $\mathbf{p}_i$, for $i = 1, 2, 3$.

We use Figure 3 for visualizing the scale selection principle. In this simple example, the image on the left contains three white disks with center points $\mathbf{p}_1$, $\mathbf{p}_2$, and $\mathbf{p}_3$. On the right, the figure shows the scale evolution of the function in Equation (1.2) for those three center points. In this example, the ratio between the scales, identified by maxima of the scale characteristics of those three center points, equals the ratio of areas of the corresponding white disks.

In the case of the affine scale-space, the normalization is given in terms of the determinant of the scale matrix Σ.

3 Shape Adaptation

The scale detection principle described in the previous section assumes that image patches are isotropic, with the Laplacian operator acting as a shape-matching kernel. However, this assumption might be a too strong for image sequences taken in common traffic scenarios.

3.1 Weighted Second Moment Matrix

In the context of Shape-from-Texture, [10] proposed the so-called *weighted second moment matrix* as a statistical measure of the local affine distortion of an image patch. Let $L_f : \mathbb{R}^2 \times \mathcal{M}_{SPD} \to \mathbb{R}$ be the scale-space representation of image f, and let $\nabla L_f(\cdot; \Sigma)$ be its spatial gradient, computed at the anisotropic scale Σ. The weighted second moment matrix $\mu : \mathbb{R}^2 \times \mathcal{M}_{SPD} \to \mathcal{M}_{SPD}$, being given by

$$\mu(\cdot; a\Sigma, b\Sigma) = g(\cdot; b\Sigma) * \left(\nabla L_f(\cdot; a\Sigma)(\nabla L_f(\cdot; a\Sigma))^T \right)$$

where $a < b$ describes the relative size of the area covered by such Σ-matrices, which have been both chosen in such a way that $\det(\Sigma) = 1$. Note that this matrix resembles also the structure tensor, as described in [28].

For a given point $\mathbf{q}$, the μ-matrix can be understood as the matrix of second moments of normalized gradients, with a density function given by a zero-mean anisotropic Gaussian of covariance $\Sigma_a = a\Sigma$ around that point $\mathbf{q}$.

Following the notation introduced in Section 2, it can be shown [12] that under an invertible linear transformation, this matrix behaves as

$$\mu_f(\mathbf{q}; a\Sigma_f, b\Sigma_f) = B^T \cdot \mu_h(\mathbf{p}; a\Sigma_h, b\Sigma_h) \cdot B \tag{1.3}$$

To see the importance of this property, consider that the covariance matrices are computed in such a way that the following two conditions

$$\begin{aligned} \Sigma_{a,f} &= aM_f^{-1} \\ \Sigma_{b,f} &= bM_f^{-1} \end{aligned} \tag{1.4}$$

are met, with $M_f = \mu_f(\mathbf{q}; \Sigma_{a,f}, \Sigma_{b,f})$. Considering a linear transformation of the spatial domain, and using the relation (1.3) and conditions (1.4), we obtain that

$$\begin{aligned} \Sigma_{a,h} &= B\Sigma_{a,f}B^T \\ &= aBM_f^{-1}B^T \\ &= a\left(B^{-T}M_fB^{-1}\right)^{-1} \\ &= aM_h^{-1} \end{aligned}$$

as result for the scale matrices. This shows that the fixpoint conditions (1.4) are preserved under a linear transformation (defined by B).

The existence of a fixpoint allows the formulation of an iterative algorithm for the estimation of the second moment descriptor. Lindeberg [14] proposed an algorithm where the shapes of the smoothing Σ-matrices are kept proportional to the second moment matrix (in each iteration step). The size of the integration scale matrix is modified according to the extrema of the (normalized) differential entities used for scale selection. The size of the local (derivation) scale matrix is held proportional to the integration scale, or selected according to an isotropy measure formulated in terms of the components of the second moment matrix. For details of the complete procedure, see [14].

The restriction on the shape of the smoothing kernels reduces the search space, but the still needed convolutions are with non-uniform Gaussians, which do not obey (in general) the separability property of their uniform counterparts; thus they do not allow for efficient recursive implementations [3,30].

An alternative option is to transform the image domain such that a circularly symmetric Gaussian corresponds to an anisotropic smoothing kernel in the original domain, as used in [1,19].

3.2 Iterative Algorithm for Shape Adaptation

Given an image $f : \mathbb{R}^2 \rightarrow \mathbb{R}$ and an initial point with detected location $\mathbf{x}$ and scale a, the algorithm in Figure 4 provides a basic outline of the *Harris affine region detector* and the *Hessian affine region detector*, see [19]. This algorithm calculates an estimate of the affine distortion of the associated structure in an incremental manner.

Algorithm 1. *Estimation of affine distortion.*

Initialize $U^{(0)} = \mathbb{I}$.
Normalize the window $W(\mathbf{x}_w) = f(\mathbf{x})$, *where* $U^{(k-1)}\mathbf{x}_w^{(k-1)} = \mathbf{x}^{(k)}$.
{*i.e., bring the anisotropic (elliptical) shape estimated at step* $(k-1)$ *to an isotropic (circular) one*}
Select the integration scale $a^{(k)}$ *based on the extrema in the evolution of scales of the differential operator used to scale selection.*
Select the local scale $b^{(k)} = \gamma a^{(k)}$ *which maximizes an isotropy measure, given in terms of the eigenvalues* (λ_{min},λ_{max}) *of the second moment matrix.*
Refine the spatial location of the interest point on the transformed frame and update the location of the interest point $\mathbf{x}^{(k)}$.
Compute the second moment matrix for the updated $\mathbf{x}_w^{(k)}$, $a^{(k)}$ *and* $b^{(k)}$ *and compute their inverse square root* $\mu_{adapt}^{(k)} = \mu^{-\frac{1}{2}}(\mathbf{x}_w^{(k)}, a^{(k)}\mathbb{I}, b^{(k)}\mathbb{I})$.
Update the shape adaptation matrix $U^{(k)} = \mu_{adapt}^{(k)} U^{(k-1)}$ *and normalize it in such a way that* $\lambda_{max}(U^{(k)}) = 1$, *ensuring that the image patch will be enlarged in the direction of* $\lambda_{min}(U^{(k)})$.
if *the affine shape is not yet close enough to a perfectly isotropic structure* ***then***
    goto Step 2
end if

Fig. 4. Basic outline of an affine region detector [19]

As already said, this adaptation algorithm works in the transformed image domain. Here, $\mathbf{x}_w$ are the coordinates of a point on an (adapted) window W, which is centered at point $\mathbf{x}$ on the original image domain. The final adaptation matrix U is defined by the concatenation of those matrices estimated in all the previous iteration runs.

Consider the scale selection step (Step 3) and a given estimate of the integration scale at iteration run $(k-1)$. The updated characteristic scale at iteration run (k) is obtained as an extrema in a neighborhood of $a^{(k-1)}$. This is due to the fact that the shape adaptation procedure changes the location of such scale-extrema, making such a refinement necessary.

In order to reduce the number of iterations needed for convergence, the local scale is selected as the one which maximizes the eigenvalue ratio $\mathcal{Q} = \lambda_{min}(\mu)/\lambda_{max}(\mu)$. It is easy to see that $\mathcal{Q} \in [0,1]$ where $\mathcal{Q} = 1$ corresponds to the perfect isotropic case.

A refinement of spatial location (Step 5) is needed in order to compensate for the displacement of local extrema (due to Gaussian smoothing) in the spatial selection response. This step is supposed to guarantee that initial points, which belong to the same local structure, converge to the same point if detected for slightly different scales.

Note that the term *local structure* does not correspond to a particular (say, physical) shape. This term corresponds to the extension of the equivalent kernel that results when applying the normalized Laplacian operator on smoothed copies of an image (i.e., its scale space representation). Thus it is possible that multiple maxima are detected (at different scale levels) for the same local structure. Figure 5 illustrates this by means of a (very) simple example of a white elliptical blob on a black background. The green circles represent the initially detected (spatial) maxima on the normalized Laplacian response. Note that maxima are detected at multiple scales for the same local structure (in this case, the white ellipse). The red ellipses (there are actually four convergent solutions) illustrate the estimated affine shape.

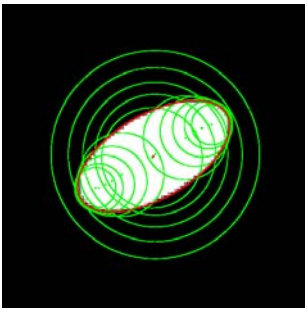


Fig. 5. Initially detected maxima of the Laplacian response (green) and estimated shapes (red). The sizes of the shown circles/ellipses are 1.41 times the characteristic scale of the corresponding point.

3.3 Shape Adaptation for Motion Cues Estimation

Some particular issues arises when considering the case of affine shapes aimed to be used in the context of affine tracking and motion estimation. This leads to a modified version of the algorithm described in the previous section, that was originally formulated in the context of feature detection and matching. Having this algorithm as reference for discussion, the following considerations has to be pointed.

Scale Selection Step. The scale selection step of the shape adaptation algorithm estimates – for the current iteration run – the characteristic scale of the brightness pattern associated to the given point with coordinates $\mathbf{x}^{(k)}$. This is justified by the fact that during each iteration run, the image pattern is *deformed* in order to compensate for the (current estimate of) the affine distortion, changing the relative size of the pattern, thus making a re-detection of the integration scale (on the current normalized frame) necessary.

Such an estimation is done by looking for local extrema on the response of the scale-space operator used for scale selection, in a local neighborhood (in scale) of the current estimate $a^{(k)}$. [19] used a range of $a^{(k)} = b^2 a^{(k-1)}$, with $b = 0.7, \ldots, 1.4$.

At each step of the algorithm, the warping matrix $U^{(k)}$ is normalized by ensuring $\lambda_{max}(U^{(k)}) = 1$. As pointed out in [20], this has the effect of expanding the image pattern in the direction of the minimum eigenvalue of the U-matrix. This expansion increases the area of the uniform region that represents the actual blob, increasing this way also their associated scale.

In the case where the initially detected blob undergoes a significant affine deformation, the corresponding warping of the region may have the impact that scale extrema fall outside of the scale search interval (i.e., we are loosing the point due to a 'no detection of integration scale').

At this moment, and after the $U^{(k)}$-matrix is estimated, it is important to correct the actual estimate of $a^{(k)}$ for the mentioned change in size, before a further estimation of characteristic scale takes place.

One way to introduce such a correction factor is by means of the area ratio of the ellipses associated to the quadratic form

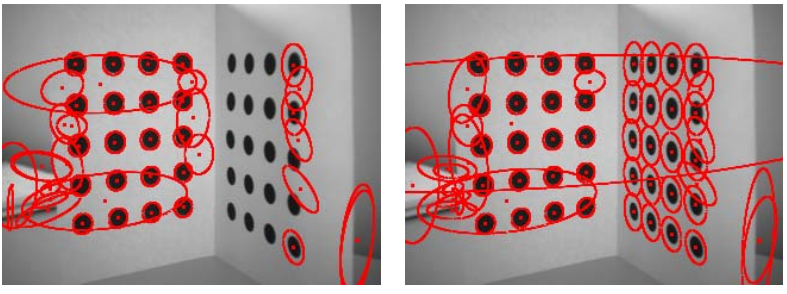
$$\mathbf{x}^T \mu^{(\nu)} \mathbf{x} = 1$$

with $\nu \in \{k, k-1\}$. In this case, the integration scale should be corrected by the factor

$$c_k = \frac{\lambda_{min}(\mu^{(k-1)}) \cdot \lambda_{max}(\mu^{(k-1)})}{\lambda_{min}(\mu^{(k)}) \cdot \lambda_{max}(\mu^{(k)})}$$

prior to the normalization step. Figure 6 shows a real example of the inclusion of such a correction into the affine adaptation procedure.

Figure 6(a) illustrates a detection obtained from the 'standard' algorithm, as described in Section 3.2. The test pattern corresponds to two planes with circular black blobs. These two planes are approximately orthogonal to each other. It



(a) (b)

Fig. 6. LoG-affine detector. Correction of the integration scale prior to the window normalization step.

can be seen that for most of the blobs on the right plane, the algorithm fails to converge. The shown ellipses correspond to detected blobs, and are displayed in such a way that their area is $\sqrt{2}$ times the detected scale, and their shape is given by the matrix $U^T U$.

Figure 6(b) illustrates a detection obtained by correcting the integration scale in each iteration run of the algorithm, as described above. In all the cases, the normalized Laplacian is used both for scale selection and spatial localization. It can be seen that, besides of correct detections, the estimates of relative scales of blobs differ between the left and the right plane. This is due to the fact that the correction is done in order to reduce the number of points which are lost by non-detections of characteristic scales.

As mentioned before, the normalization of the U-matrix by $\lambda_{max}(U^k)$ produces a displacement of the characteristic scales towards larger values. With patterns undergoing important viewpoint changes, that could be a serious problem if one relies on the detected scale for further processing.

***U*-Normalization at Important Viewpoint Changes.** As shown in Figure 6, affine normalization induces an overestimation of the characteristic scale if the imaged blobs undergo essential deformations. This is produced by the change in relative size that occurs during the U-normalization step. In order to overcome this, a possible alternative is to normalize the $U^{(k)}$ matrix in such a way that the *area* of blobs remains constant during iteration runs. This corresponds to a normalization of the U matrix by the square root of the product of their eigenvalues, $U^{(k)}_{norm} = U^{(k)} / \sqrt{\lambda_{min}(U^{(k)}) \lambda_{max}(U^{(k)})}$.

Figure 7 shows results when applying such a U-normalization in the affine adaptation procedure. Note that resulting scales and shapes of the final estimates correspond now to the actual (physical) blobs, even if such blobs are imaged under quite different viewpoints.

It is important to note that, with this normalization of the U-matrix, there is no need to correct the integration scale, and we may use factor $c_k = 1$ during all the iteration runs.

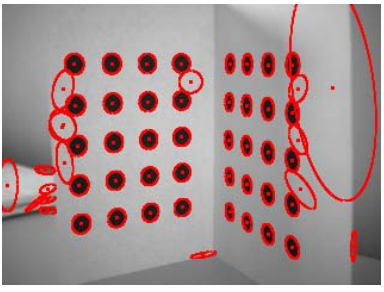


Fig. 7. LoG-affine detector. Normalization of the U-matrix to enforce constant area.

Lighting Conditions: Centered Second Moment Matrix. Besides some modifications in the adaptation scheme, some degree of robustness against changes in lighting conditions is also desirable. Regarding the shape adaptation matrix, [13] proposed the so-called *centered second moment matrix*, defined as follows:

$$\nu(\cdot;\Sigma_a,\Sigma_b) = g(\cdot;\Sigma_b) * \left(\nabla L(\cdot;\Sigma_a)(\nabla L(\cdot;\Sigma_a))^T\right) - (\overline{\nabla L})(\overline{\nabla L})^T$$

where

$$\overline{\nabla L} = g(\cdot;\Sigma_b) * \nabla L(\cdot;\Sigma_a) \tag{1.5}$$

The centered second moment matrix has the same behavior as the non-centered counterpart, as both behave as in Equation (1.3) for a linear transformation of the spatial domain. This centering makes shape adaptation invariant with respect to linear changes in image brightness.

3.4 Transformation Properties of Corresponding Regions

Consider (as before) two images f and h. Let us suppose that they are images of a surface that can be locally approximated by a plane. Under this condition, the images can be related to each-other by an unknown affine transformation.

Let us also suppose that, for corresponding points on those images, the shape adaptation matrix was independently computed. Under the transformation property of Equation (1.3), and using the decomposition $\mu = \mu^{\frac{1}{2}}\mu^{\frac{1}{2}}$, where $\mu^{\frac{1}{2}}$ is a symmetric matrix, it follows that[2]

$$\mu_f^{\frac{1}{2}}\mu_f^{\frac{1}{2}} = B^T\mu_h^{\frac{1}{2}}\mu_h^{\frac{1}{2}}B$$

and

$$I = (\mu_h^{\frac{1}{2}}B\mu_f^{-\frac{1}{2}})^T(\mu_h^{\frac{1}{2}}B\mu_f^{-\frac{1}{2}}) = W^TW$$

[2] Arguments of μ-matrices have been omitted for the sake of simplicity.

where $W = \mu_h^{\frac{1}{2}} B \mu_f^{-\frac{1}{2}}$ is an orthogonal matrix. Finally, the linear transformation B can be recovered (up to an arbitrary orthogonal matrix) from the shape adaptation matrices as follows:

$$B = \mu_h^{-\frac{1}{2}} W \mu_f^{\frac{1}{2}} \tag{1.6}$$

The rotation can be recovered in a robust manner based on orientation histograms (e.g., see the method in [16] used for orientation assignments), but is computed on affine normalized patches. Also, for a given image structure, the change in *scale* of corresponding regions (under an assumed affine transformation) is given by the determinant

$$\det(B) = \sqrt{\frac{\det(\mu_f)}{\det(\mu_h)}} \tag{1.7}$$

of the transformation matrix of Equation (1.6).

4 Derivation of Motion Characteristics

Ideas presented so far may be used (like a guide) to infer some additional motion characteristics, usually not taken into account in standard optic flow techniques.

4.1 Moving Surface Patch

We consider a projective camera of focal length l. A point $\mathbf{X} = (X, Y, Z)^T$ in the 3D space is projected onto a point $\mathbf{x} = (x, y)^T = \left(l\frac{X}{Z}, l\frac{Y}{Z}\right)^T$ on the 2D image plane.

As in [7], be Ω a surface patch that moves relatively to the camera, and Ω_t and Ω_{t+1} their projections onto the image plane at times t and $t+1$, respectively; see Figure 8. Let us suppose that Ω can be locally approximated to be incident with the plane $\mathbf{E} : Z = pX + qY + d$ located in the three-dimensional space. Let be A_t and A_{t+1} the area measures of Ω_t and Ω_{t+1} respectively. It follows that

$$A_t = \int_{\Omega_t} \mathrm{d}x \; \mathrm{d}y = \int_{\Omega} \det(J_t) \; \mathrm{d}X \; \mathrm{d}Y \tag{1.8}$$

with J_t the Jacobi matrix of the given transformation at time t. Using the expressions for the imaged point $\mathbf{x}_t$ and the plane $\mathbf{E}$, results into

$$J = \frac{\partial(x, y)}{\partial(X, Y)} = \partial\left(l\frac{X}{pX + qY + d}, l\frac{Y}{pX + qY + d}\right) \Big/ \partial(X, Y)$$

It follows that

$$\det(J) = l^2 \frac{d}{Z^3} \tag{1.9}$$

and thus for the area measure (for a fixed focal length l) that

$$A_t = l^2 \int_{\Omega} \frac{d}{Z^3} \; \mathrm{d}X \; \mathrm{d}Y \tag{1.10}$$

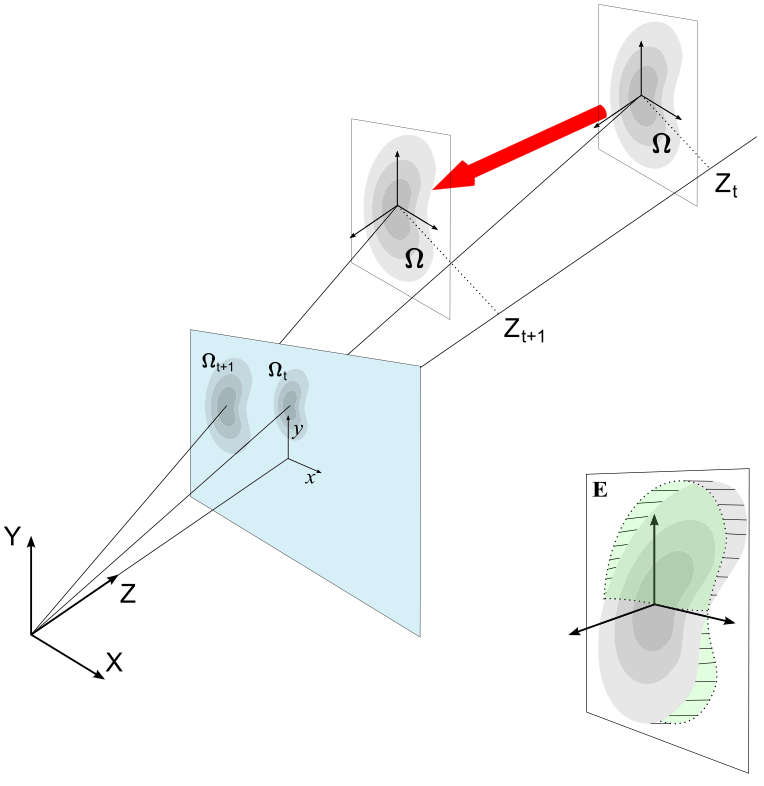


Fig. 8. A moving surface patch Ω, projected into Ω_t

Let us assume that the planar patch is located parallel to the image plane or, in a similar way to the case of a *weak-perspective* projection, consider a projection of the surface patch onto a plane which is parallel to the image plane, and incident with the centroid of the patch; see the green region at the bottom right of Figure 8. Let $\tilde{Z}$ be the mean of Z-coordinates of the patch. In that case, $d = \tilde{Z}$ and the relation between the projected area and the Z coordinate of the plane is of the form $A_t \sim 1/\tilde{Z}^2$.

Let λ_z be the ratio between area measures at times t and $t+1$. It follows that

$$\lambda_z = \frac{\sqrt{A_t}}{\sqrt{A_{t+1}}} = \frac{\tilde{Z}_{t+1}}{\tilde{Z}_t} \tag{1.11}$$

Now consider two subsequent images f_t and f_{t+1} of a recorded sequence, at times t and $t+1$, respectively, and, as before, a surface patch that moves in 3D relatively to the camera between the time slots when frames t and $t+1$ were taken. Assume that this patch is visible in both frames, say with centroids at

$\mathbf{x}_t$ and $\mathbf{x}_{t+1}$. For the ratio between the other two coordinate components of a moving point we have that

$$\lambda_x = \frac{X_{t+1}}{X_t} = \frac{Z_{t+1}}{Z_t} \cdot \frac{x_{t+1}}{x_t} = \lambda_z \frac{x_{t+1}}{x_t} \tag{1.12}$$

$$\lambda_y = \frac{Y_{t+1}}{Y_t} = \frac{Z_{t+1}}{Z_t} \cdot \frac{y_{t+1}}{y_t} = \lambda_z \frac{y_{t+1}}{y_t} \tag{1.13}$$

In a more compact form, this may be expressed as $\mathbf{X}_{t+1} = \Lambda \mathbf{X}_t$, where $\Lambda = \text{diag}(\lambda_x, \lambda_y, \lambda_z)$, and

$$\Delta\mathbf{X} = \mathbf{X}_{t+1} - \mathbf{X}_t = (\Lambda - \mathbb{I})\mathbf{X}_t \tag{1.14}$$

is the absolute spatial displacement, specifying a *scene flow vector*, positioned at $\mathbf{X}_t$.

4.2 Estimation of Navigation Angles

Consider a mobile platform (e.g., a car) moving on a planar surface (e.g., a patch of a road surface of limited area), as illustrated in Figure 9. From (1.14) and letting $\Delta\mathbf{X} = (\Delta X, \Delta Y, \Delta Z)$, it follows that the ratios

$$\frac{\Delta X}{\Delta Z} = \left(\frac{\lambda_x - 1}{\lambda_z - 1}\right) \frac{X_t}{Z_t} = \left(\frac{\lambda_x - 1}{\lambda_z - 1}\right) \frac{x_t}{l} \tag{1.15}$$

$$\frac{\Delta Y}{\Delta Z} = \left(\frac{\lambda_y - 1}{\lambda_z - 1}\right) \frac{Y_t}{Z_t} = \left(\frac{\lambda_y - 1}{\lambda_z - 1}\right) \frac{y_t}{l} \tag{1.16}$$

are the tangents of the *navigation angles* Φ_{zx} and Φ_{zy} (see Figure 9). Those angles represent the *3D direction of motion* (between two subsequent frames) for a tracked 3D point.

Now suppose that two shape adaptation matrices were obtained for two corresponding points $\mathbf{x}_t$ and $\mathbf{x}_{t+1}$, in images at times t and $t+1$ (as described in Section 3.4 for a single point in one image only).[3] Let us call these two matrices μ_t and μ_{t+1}. Recalling Equation (1.7), it follows that

$$\lambda_z = \sqrt{\det(B)} = \sqrt{\frac{\det(\mu_t)}{\det(\mu_{t+1})}}$$

given an estimation of λ_z, the navigation angles, θ_{zx} and θ_{zy}, are given as arcus tangent of the ratios provided by Equations (1.15) and (1.16). This estimation is done independently for each of the corresponding features, obtained from frames at times t and $t+1$.

[3] Note that, if the shape adaptation matrix has been obtained with the full iterative algorithm, then it can be used as an initial estimate for the corresponding point, in order to reduce the number of iterations.

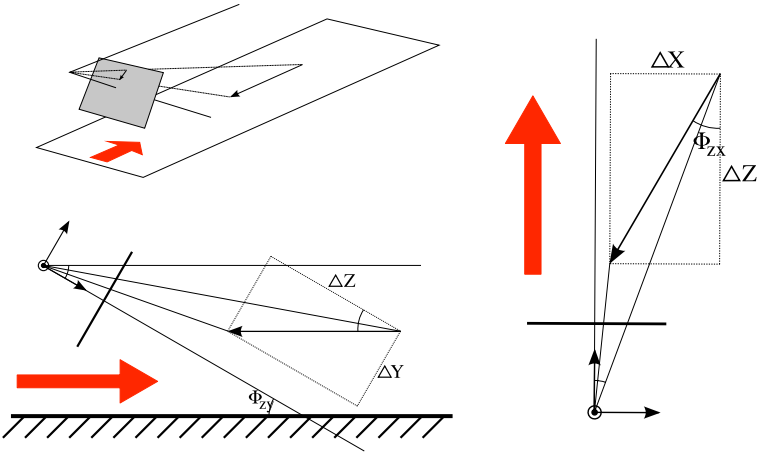


Fig. 9. A tilted camera translating along a plane (top left), motion angles (tilt) on the ZY-plane (bottom left), and on the ZX-plane (yaw; right)

After the estimation of both navigation angles (for all pairs of tracked points), histograms of these two values are computed. This allows to calculate one *summarizing 3D direction*, for all the detected 3D directions between images f_t and f_{t+1}. For this summarizing 3D direction, we decided for

$$\hat{\phi}_{zx} = \arg\max_\theta h(\theta_{zx})$$
$$\hat{\phi}_{zy} = \arg\max_\theta h(\theta_{zy})$$

where $h(\theta_{zx})$ and $h(\theta_{zy})$ are the histograms of the navigation angles.

To summarize, given a pair of consecutive images, f_t and f_{t+1}, the estimation of the navigation angles, ϕ_{zx} and ϕ_{zy}, is accomplished by the algorithm of Figure 10.

The estimation of navigation angles relies on the scale ratio of the tracked blobs; thus, an estimation of such characteristic scales at *sub-scale* level is desirable. Assume that the detection of characteristic scales for an image point is based on the local maxima over scales of the response of some operator (see, for example, Section 2.2). One possibility is then to fit a parabola to three points that determine a local maxima (a characteristic scale) at sub-scale resolution.

For the computation of such an interpolated extrema, one must take into account that the scale-axis of the scale-space representation is not sampled linearly. Instead, an exponential sampling is used, where the scale of the Gaussian kernel that generates the scale level (n) is obtained from the scale at scale level $(n-1)$ as follows:

$$a_n = k a_{n-1} = k^n a_0 \quad \text{for} \quad n = 1, 2, \ldots, N-1 \tag{1.17}$$

Obviously, logarithms of those a values must be used for the computation of an interpolating parabola.

Algorithm 2. *Estimation of navigation angles.*

1: Extract from image f_t affine blobs, as discussed in Section 3.3.
2: ***for*** *each adapted blob obtained from the previous step* ***do***
3: Set the size of the tracking window proportional to the characteristic scale of the corresponding blob.
4: Track them from image f_t to f_{t+1} as described in Section 4.3.
5: Affine-adapt the tracked point in order to refine their shape/scale estimate.
6: Estimate the λ-ratios, as described in Section 4.
7: Estimate the navigation angles for the current blob, as described in Section 4.2, and update their histograms.
8: ***end for***
9: Obtain a final estimate of ϕ_{zx} and ϕ_{zy} by means of the peak on the constructed histograms.

Fig. 10. Algorithm for the estimation of the navigation angles

Exponential sampling (1,17) also allows the use of the DoG operator as a fast approximation technique for the Laplacian of Gaussian; see [15].

4.3 Multiscale Tracking

The determination of those navigation angles depends upon the detection of projected motion (of tracked points) in the image plane.

Actually, the used tracking method is not essential for presenting the basic idea of our approach for estimating 3D directions; however, it is, of course, important for obtaining reliable results of the proposed approach.

A popular approach for feature tracking is the algorithm of Lucas-Kanade [17], for which pyramidal implementations are available at [22]. Here, one parameter to be selected is the size of the tracking window, which is usually treated as a value to be manually adjusted. In the case of blob tracking, this parameter cannot be kept constant for every feature because of the variable size of the image blobs, and, more importantly, because of that image blobs correspond in general to uniform regions of brightness patterns; places where the *information content* is low (i.e., textureless regions) lead to problems as known from the aperture problem.

For the presented model, the relative size of image features may be selected automatically by detecting extrema in scale-space (see, for example, [24]). Thus, it is reasonable to set a tracking window proportional to a detected characteristic scale (of the feature to be tracked). This specifies a *multiscale affine blob tracking* method, which proved to be well suited for this task (and possibly others).

5 Experiments

Figure 11 illustrates one particular experiment for affine blob tracking, following the approach as described above. A camera moves freely away from a black disk.

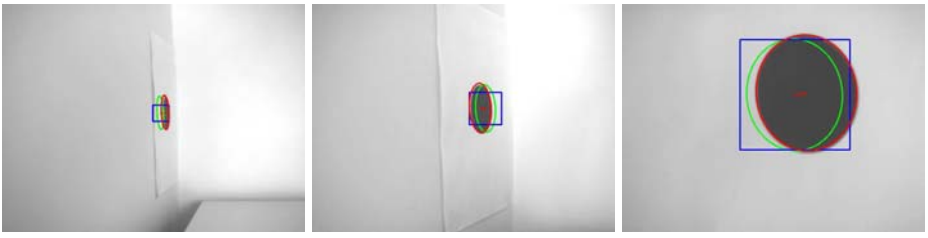


Fig. 11. Detection of an affine blob (black disk), and tracking results for viewpoint and scale changes

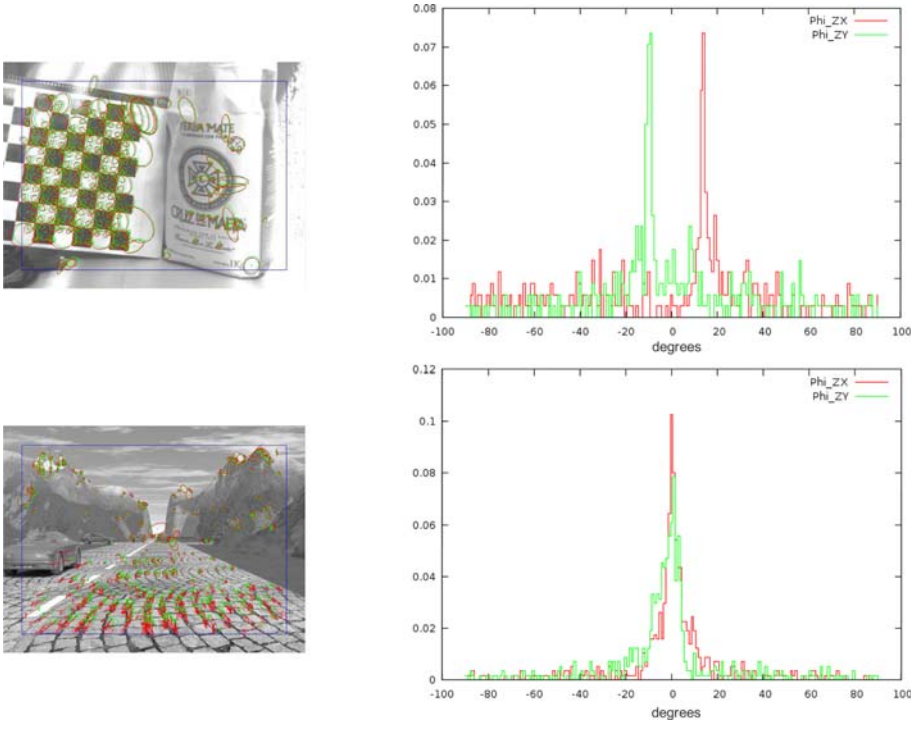


Fig. 12. Images showing tracked blobs and histograms computed for the estimation of angles ϕ_{zx} (red) and ϕ_{zy} (green)

We observed that that the black disk is correctly tracked even if there are 'rapid' changes in viewpoint or scale. In this figure, green and red ellipses indicate shape (scale) estimations at times t and $t+1$, respectively, while the blue box shows the window used for tracking.

Figure 12 shows two generated histograms, obtained either (bottom) for the synthetic sequence of Set 2 on [5], or (top) the desktop sequence.

A square window of size $(2N+1) \times (2N+1)$ was used for tracking, with N set as twice the characteristic scale of the blob detected at time t.

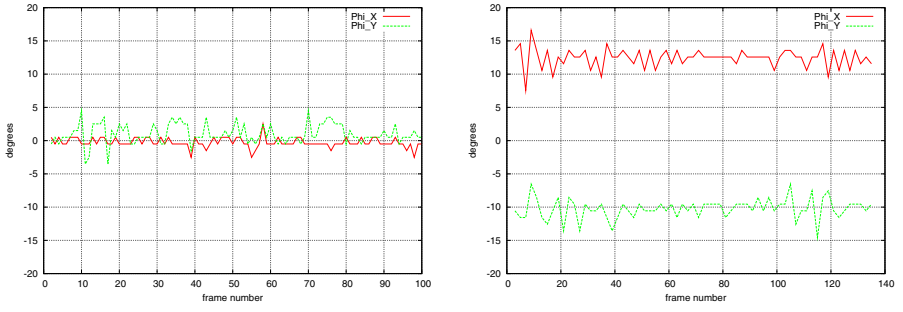


Fig. 13. Estimation of navigation angles for the (a) synthetic and (b) desktop sequence

Such estimates at time t can be visualized by diagrams for the whole image sequence. Figure 13 shows such results along the used test sequences. (Note that this allows a new quality of analysis compared to short sequences, such as, for example, currently available on [18].)

The accuracy of estimated values depends in some way on the magnitude of the relative motion. For points that remain almost static, and where the factor λ_z is close to 1, the quotients in Equations (1.15) and (1.16) are not well defined, due to noisy measurements.

The synthetic sequence was generated with corresponding navigation angles of $\phi_{zx} = 0°$ and $\phi_{zy} = 0°$. The desktop sequence was generated with a calibrated camera, with translational motion on a rail with fixed navigation angles of approximately $\phi_{zx} = 12°$ and $\phi_{zy} = -10°$.

The experiments carried out over the synthetic sequence gave a mean value of $0.13°$ and a standard deviation of $0.57°$ for ϕ_{zx}. For ϕ_Z, those values were $0.15°$ and $0.93°$, respectively.

In the case of the desktop sequence, mean and standard deviation of ϕ_{zx} are equal to $12.42°$ and $1.43°$, respectively. For ϕ_{zy}, those values are equal to $-10.243°$ and $1.52°$, respectively, taken over the entire sequence.

Figure 14 shows the estimation of the navigation angles for two real-world sequences (Daimler sequences 3 and 7, available in Set 1 on [5]). For these sequences, there is no ground truth available for navigation angles, and they are only used as a qualitative (visual) reference. The figure illustrates the instantaneous estimation of the navigation angles and, superimposed, results after applying a sliding mean (5 backward, current, and 5 forward values) filter.[4] It also shows some images of the corresponding sequences.

In the case of Sequence 7 (bottom), the estimated directions Φ_{zx} correspond to the steering of the car over the sequence. In Sequence 3 (top), we observed a low frequency oscillation in the value of Φ_{zy}, starting about at frame 180, when the 'squirrel' (actually, a cat) crossed the street and the car made a breaking maneuver.

[4] This sliding mean filter is certainly only a demonstration of filter opportunities; the design of an appropriate Kalman filter would be an option to ensure real-time analysis.

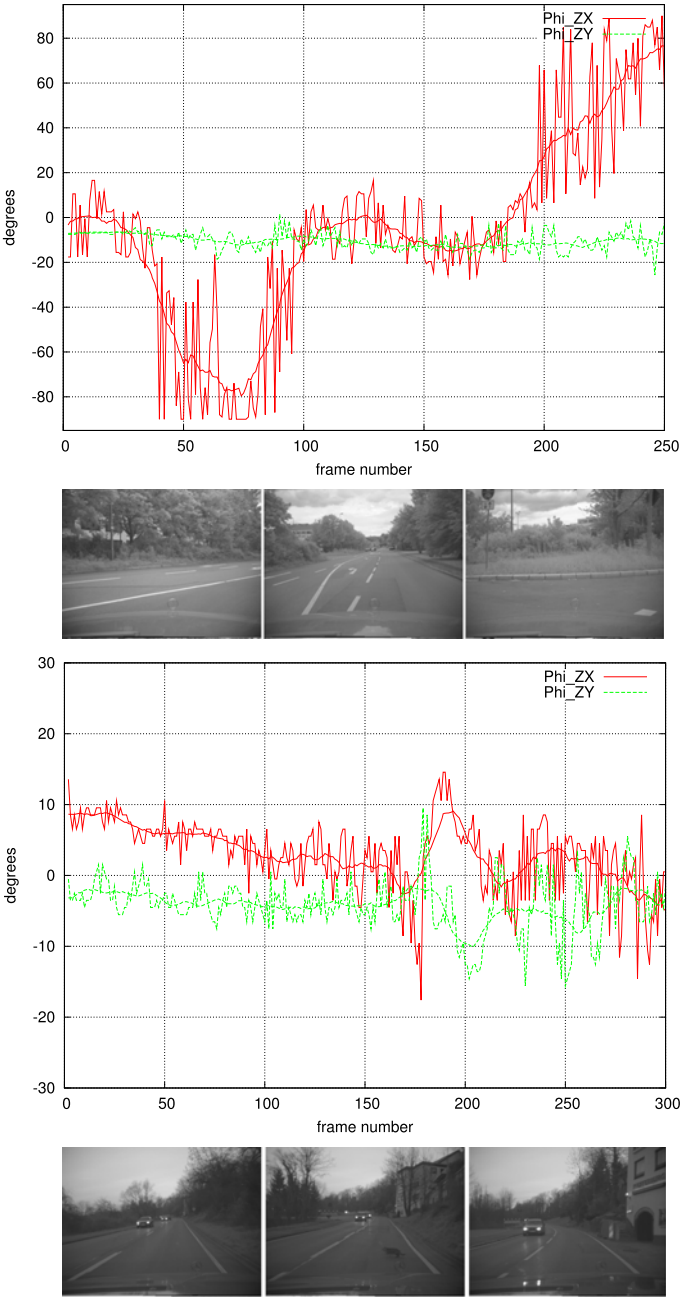


Fig. 14. Results on Sequences 3 (top) and 7 (bottom) of Set 1 on [5]

The proposed (non-run-time-optimized) algorithm runs at approximately 0.1 fps on a 3.0 GHz Intel© Core 2 Duo CPU.

6 Conclusions

This chapter proposes a method for the instantaneous (frame to frame) estimation of the 3D direction of motion; the method was studied based on the determination of scale ratios between tracked blobs. Results may be used for ego-motion correction of video sequences recorded by the ego-vehicle (by compensating for estimated tilt and roll angles).

Besides some poor estimations for some frames, the proposed method may be recommended as a possible approach for the use of perceptually very important spatio-temporal cues, induced on images as an observer (camera) moves relatively to the scene. The extracted information has the advantage of being local, allowing robustness in the case of multiple moving objects. The same principle of scale-ratio estimation could also be used for motion segmentation, or to add new constraints to multiple-view approaches of 3D motion estimation, thus further contributing to the already known coherence between optic flow vectors and image disparities.

The overall run-time of the algorithm can be significantly improved by the use of dedicated hardware (FPGA or ASICs). Here, the bottle-neck remains in the computation of the scale-space representations of the given images. This is done, as mentioned previously, by means of convolution with bi-dimensional Gaussians of variable width. This type of kernels allows for efficient implementations in terms of separable 1D Gaussians (which allow recursive implementations; see [3,30]).

Acknowledgement. The authors thank the three anonymous reviewers for their valuable comments.

References

1. Baumberg, A.: Reliable feature matching across widely separated views. In: Proc. Conf. Computer Vision Pattern Recognition, vol. I, pp. 774–781 (2000)
2. Burt, P., Adelson, T.: The Laplacian pyramid as a compact image code. IEEE Trans. Communications 9, 532–540 (1983)
3. Deriche, R.: Recursively implementing the Gaussian and its derivatives. In: Proc. Int. Conf. Image Processing, pp. 236–267 (1992)
4. Dickmanns, E.D.: Dynamic Vision for Perception and Control of Motion. Springer, New York (2007)
5. enpeda. Image Sequence Analysis Test Site, http://www.mi.auckland.ac.nz/EISATS
6. Florack, L.M.J.: Image Structure. Computational Imaging, vol. 10. Kluwer, Dordrecht (1997)
7. Klette, R.: Shape from area and centroid. In: Proc. Int. Conf. Artificial Intelligence Information-Control Systems Robots, pp. 309–314 (1995)
8. Klette, R.: Stereo-vision-support for intelligent vehicles - the need for quantified evidence. In: Wobcke, W., Zhang, M. (eds.) AI 2008. LNCS (LNAI), vol. 5360, pp. 1–17. Springer, Heidelberg (2008)

9. Koenderink, J.J.: The structure of images. Biological Cybernetics 50, 363–370 (1984)
10. Lindeberg, T., Garding, J.: Shape from texture from a multi-scale perspective. In: Proc. Int. Conf. Computer Vision, pp. 683–691 (1993)
11. Lindeberg, T.: Scale-Space Theory in Computer Vision. Kluwer Academic Publishers, Norwell (1994)
12. Lindeberg, T., Garding, J.: Shape-adapted smoothing in estimation of 3-d depth cues from affine distortions of local 2-d brightness structure. In: Eklundh, J.-O. (ed.) ECCV 1994. LNCS, vol. 800, pp. 389–400. Springer, Heidelberg (1994)
13. Lindeberg, T.: A scale selection principle for estimating image deformations. In: Proc. Int. Conf. Computer Vision, pp. 134–141 (1995)
14. Lindeberg, T.: Feature detection with automatic scale selection. Int. J. Computer Vision 30, 77–116 (1998)
15. Lowe, D.: Object recognition from local scale-invariant features. In: Proc. Int. Conf. Computer Vision, pp. 1150–1157 (1999)
16. Lowe, D.: Distinctive image features from scale-invariant keypoints. Int. J. Computer Vision 60, 91–210 (2004)
17. Lucas, B., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: Proc. IJCAI, pp. 674–679 (1981)
18. Middlebury Optical Flow website, http://vision.middlebury.edu/flow/
19. Mikolajczyk, K., Schmid, C.: An affine invariant interest point detector. In: Heyden, A., Sparr, G., Nielsen, M., Johansen, P. (eds.) ECCV 2002. LNCS, vol. 2350, pp. 128–142. Springer, Heidelberg (2002)
20. Mikolajczyk, K.: Detection of local features invariant to affine transformations. PhD thesis, INRIA (2002)
21. Mikolajczyk, K., Schmid, C.: Scale and affine invariant interest point detectors. Int. J. Computer Vision 60, 63–86 (2004)
22. Open Source Computer Vision Library, http://www.intel.com/research/mrl/research/opencv/
23. Ottlik, A., Nagel, H.-H.: Initialization of model-based vehicle tracking in video sequences of inner-city intersections. Int. J. Computer Vision 80, 211–225 (2008)
24. Sánchez, J., Klette, R., Destefanis, E.: Estimating 3D flow for driver assistance applications. In: Wada, T., Huang, F., Lin, S. (eds.) PSIVT 2009. LNCS, vol. 5414, pp. 237–248. Springer, Heidelberg (2009)
25. Schrater, P., Knill, D., Simoncelli, E.: Perceiving visual expansion without optic flow. Nature 410, 816–819 (2001)
26. Wedel, A., Vaudrey, T., Rabe, C., Brox, T., Franke, U., Cremers, D.: Decoupling motion and position of image flow with an evaluation approach to scene flow. Chapter XX in this book
27. Wedel, A., Rabe, C., Vaudrey, T., Brox, T., Franke, U., Cremers, D.: Efficient dense scene flow from sparse or dense stereo data. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part I. LNCS, vol. 5302, pp. 739–751. Springer, Heidelberg (2008)
28. Weickert, J., Hagen, H. (eds.): Visualization and Processing of Tensor Fields. Springer, New York (2006)
29. Witkin, A.P.: Scale-space filtering. In: Proc. Int. Joint Conf. Art. Intell., pp. 1019–1022 (1983)
30. Young, I.T., van Vliet, L.J.: Recursive implementation of the Gaussian filter. Signal Processing 44, 139–151 (1995)

Comparison of Point and Line Features and Their Combination for Rigid Body Motion Estimation

Florian Pilz[1], Nicolas Pugeault[2,3], and Norbert Krüger[3]

[1] Department of Medialogy and Engineering Science
Aalborg University Copenhagen, Denmark
fpi@imi.aau.dk
[2] School of Informatics
University of Edinburgh, United Kingdom
nicolas.pugeault@edinburgh.ac.uk
[3] The Maersk Mc-Kinney Moller Institute
University of Southern Denmark, Denmark
norbert@mmmi.sdu.dk

Abstract. This paper discusses the usage of different image features and their combination in the context of estimating the motion of rigid bodies (RBM estimation). From stereo image sequences, we extract line features at local edges (coded in so called multi-modal primitives) as well as point features (by means of SIFT descriptors). All features are then matched across stereo and time, and we use these correspondences to estimate the RBM by solving the 3D-2D pose estimation problem. We test different feature sets on various stereo image sequences, recorded in realistic outdoor and indoor scenes. We evaluate and compare the results using line and point features as 3D-2D constraints and we discuss the qualitative advantages and disadvantages of both feature types for RBM estimation. We also demonstrate an improvement in robustness through the combination of these features on large data sets in the driver assistance and robotics domain. In particular, we report total failures of motion estimation based on only one type of feature on relevant data sets.

1 Introduction

The knowledge about the egomotion of the camera or the motion of objects in a scene is crucial for many applications such as driver assistance systems, object recognition, collision avoidance and motion capture in animation. In case we deal with a rigid object, such a motion can be understood as a 'Rigid Body Motion' (RBM), which is defined as a continuous motion preserving the distance between any two points of the object. The mathematical structure of this motion is well known and has been studied for more than 100 years (see e.g., [1]).

The estimation of such a motion from images faces two sub-problems. First, based on a certain mathematical formulation of the RBM, constraint equations

D. Cremers et al. (Eds.): Visual Motion Analysis, LNCS 5604, pp. 280–304, 2009.

need to be defined using correspondences between image features. This problem can become complex, in particular when correspondences of different features (e.g., line and point correspondences) become mixed. The state of the art in this field is described in Section 1.1. Second, correspondences between image features need to be computed from image sequences. Here, different kinds of local image structures are observable that lead to constraints of different structure and strength (e.g., point and line correspondences). Point correspondences are more valuable since they lead to stronger constraint equations. However, in general there are much fewer point correspondences compared to line correspondences available in natural scenes due to the dominance of edges (see, e.g., [2]). Work addressing the correspondence problem for different feature types is discussed in Section 1.2.

The intention of this paper is to analyze the consequences of using two different kinds of correspondences (i.e., point and line correspondences) as well as their combination for motion estimation on a large set of indoor and outdoor sequences, with large variety across the sequences as well as within the sequences. We evaluate and compare the results of line features (primitives) and point features (SIFT) as constraints for RBM estimation. We discuss the qualitative advantages and disadvantages of both feature types for RBM estimation and also demonstrate an improvement in robustness through the combination of these features on large data sets in the driver assistance and robotics domain. In particular, we report total failures of motion estimation based on only one type of feature on relevant data sets stressing the importance of their combination for robust motion estimation. Hence, our main contribution is giving empirical evidence to argue that for stable motion estimation both kinds of features need to be used. As a consequence, we state that (1) a rich feature representation needs to be provided by the visual processing[1] and (2) a mathematical framework needs to be used that allows for such a mixing of correspondences.

The paper is structured as followed. First, we discuss related work on the formalization of the RBM estimation problem in Section 1.1 and about the finding of feature correspondences in Section 1.2. Section 2 describes all methods used in this work, including the formulation of the class of RBM. Furthermore, the section briefly describes the process of feature extraction and matching, and the 3D-2D pose estimation algorithm. Section 3 is concerned with the settings under which all methods are applied and presents the results. Section 4 discusses the results and briefly summarizes the contribution of this paper.

1.1 Mathematical Aspects of RBM Estimation

Motion estimation in different scenarios has been approached successfully (e.g. [4,5,6]) and a recent overview on monocular rigid pose estimation can be found in [7]. The different methods for RBM estimation that have been proposed can be separated into feature based (see, e.g., [8, 9]), optic flow based (see, e.g., [10,11]) and direct methods where no explicit representations as features

[1] In this context we coined the term 'early cognitive vision' in [3].

or optic flow vectors are used but image intensities are directly matched (see, e.g., [12,13,14,15]). A summary of the comparison of optic flow and feature based methods can be found in [16]. Solutions for feature based methods can be divided into linear and iterative algorithms. Linear algorithms use a limited number of feature correspondences (usually $n \geq 4, 5$). Proposed solutions include the use of 3 points [17], 6 points [18], 4 points in general position [19] and 3 lines in general position [20]. Iterative algorithms (see, e.g. [9,21,22]) make use of large correspondence sets, where closed form solutions no longer perform efficiently. The main difference to linear algorithms is that nonlinear algorithms require a starting point (an initial guess of the motion parameters) and need multiple iterations to find the motion parameters. Furthermore, feature based motion estimation algorithms also differ in the usage of different mathematical representations and error functions to be minimized. An example using points is proposed in [21], using 2D to 2D, 2D to 3D and 3D to 3D point correspondences. Other examples make use of line correspondence as in [4,5].

All feature based solutions mentioned above require the extraction and matching of visual features, most commonly point or line features. However, as we will show, the RBM estimation problem can not be completely solved based on one feature type alone, since there are cases where standard methods fail due to certain particularities in the data set. In addition, since different feature types are localized with different accuracy, the precision of the estimated motion depends on the feature types used. As a consequence, we argue that a general, precise and robust motion estimation algorithm makes use of a set of features as complete as possible for describing the scene.

The joint use of point and line correspondences requires a mathematical framework that allows for such a combination. This is mathematically non-trivial since a system of constraint equations on different entities need to be defined and, indeed, most algorithms are based on point correspondences only (see, e.g., [9,21,19,18]). In recent years, some linear solutions have been proposed for the pose estimation problem making use of n points or n lines [6,23]. These solutions provide algorithms and mathematical formulations allowing to use either points or lines, but not a combination of both mathematical representations within the same system of constraint equations.

However, there exist some algorithms which allow for a combination of correspondences of different types of constraints (see, e.g., [24,25]). For our work, we chose the algorithm [24], since, in addition of being able to deal with different visual entities, it does optimization on 3D constraint equations, using a twist formulation (see, e.g., [26,4]). This formulation directly acts on the parameters of rigid-body motions (i.e., $SE(3)$), avoiding the use of additional non-linear constraints (e.g. enforcing rotation quaternions) to make sure that the found solution actually is in $SE(3)$.

A similar approach is [27], which uses a non-linear least squares optimization to optimize the 2D re-projection error (instead of a 3D error as in [24]). As shown in [28], this approach performs better than the POSIT (**P**ose from **O**rthography

and **S**caling with **IT**erations) algorithm [29] in tracking scenarios. The work [27] extends the original publication [30] by the possibility to include 3D-point to 2D-line correspondences.

Note that some recent work, including [31,32], proposed monocular, multiple view evaluation of a constrained motion (line or conic section). In this work we are interested in the unconstrained case. Recent work in a similar driving environment as the outdoor sequences used in this work include concerning the estimation of a three dimensional velocity vector using stereo data [33]. This proposed approach uses two consecutive image pairs in stereo sequences, where the main concept is the decoupling of the position estimation and velocity estimation, allowing both the use of sparse and dense disparity estimation algorithms. Similar work in indoor robotic environments include the work of [34], addressing the inverse kinematics problem, providing a model for tracking kinematic chains with restriced degrees of freedom.

1.2 Computing Feature Correspondences

Relevant research does not only concern the mathematical aspects of RBM estimation, but also the extraction and matching of features. Visual scenes contain different kinds of local image structures (e.g., texture, junctions, local edges, homogeneous image patches) with different properties with regards to the correspondence problem, in particular line structures suffer from the aperture problem. The statistical distribution of these structures can be efficiently represented by making use of the concept of intrinsic dimensionality (see [2,35]). In general, it can be stated that approximately 10% of the local image structures correspond to edges, 1% to junction-like structures, and the remaining 90% to texture or homogeneous areas. There is no strict delimitation, but a continuum between texture and homogeneous image areas since recording noise on homogeneous areas already represents some weak texture (that of course is unsuitable for correspondence finding) and also most textured areas have very low contrast making them unsuitable for reliable correspondence finding. The distribution of occurrences of different image structures can vary significantly across images. Texture and junctions lead to point correspondences, which result in a stronger mathematical constraint (two constraint equations) than edge/line correspondences (one single constraint equation). However, in particular in man-made environments, edge features can be dominating. SIFT features [36] and their derivatives (for a review see [37]) describe semi-local textures and allow for very robust (but not necessarily precise, see below) correspondence finding, resulting in point correspondences that provide two constraint equations each. In contrast to that, edges are frequently occurring features that allow for robust and precise line–correspondences, that are however in mathematical terms 'weak', since they result in only one constrain equation.

Besides the frequency of occurrence, there exist also differences in the precision with which these features can be localized. We observed problems in the precision of SIFT correspondences in particular for sequences in the driver assistance domain where motion blur and other sources of noise (such as rain drops, fog, etc)

influence the recording process. In contrast to SIFT features, edge/line features represent more simple structures for which sub-pixel accuracy can be achieved with higher precision. Moreover, we observed that for scenes taken in an indoor environment, there occur cases where not enough SIFT features could be extracted, leading to an underconstrained estimation problem[2]. This also occurred in some (although very few) cases in the outdoor scenario.

Since feature matching is a fundamental problem in computer vision, many feature detectors and techniques for matching have been proposed over the years (for an overview see [37]). The term of 'feature detector' covers the process of extracting of wide range of interest points, where each of them usually defines a location in an image with large gradients in several directions. The approach of using a corner detector for stereo matching was initially proposed by Moravec [38] and later improved by Harris and Stephens [39], and has since been used for a wide range of computer vision applications involving point matching. Harris also showed the value of corner features for recovering structure from motion [40]. Zhang [41] showed successful matching of Harris corners over large image ranges and motions by using a correlation window for identifying potential correspondences. Other examples of feature points used for motion estimation include junctions [42,43] and SIFT features [44]. As an alternative to point features, line features have been used for motion estimation [45,23].

Regardless of the type of feature or constraints introduced to matching, finding correspondences remains a real challenge due to the change of the 3D viewpoint, resulting in the perspective distortions of features. The Harris corner detector, for example, is very sensitive to changes in image scale. In the field of RBM estimations, where objects move and thereby change size, Harris corners will encounter difficulties for temporal matching. Therefore, a good feature descriptor for matching is invariant towards change in rotation and scale. The SIFT descriptor [36] provides these properties, and is therefore used in this evaluation (for a discussion of different descriptors in the context of correspondence finding, we refer again to [37]).

2 Methods

In this section, we explain the different parts of the motion algorithm. In Section 2.1, we briefly explain the mathematical formulation of the RBM estimation problem. In Section 2.2, we describe the features and their extraction. The stereo matching as well as the temporal matching is described in Section 2.3. Finally, in Section 2.4, we describe the complete motion estimation process. In this section, we will write 2D entities in lower case e, 3D entities in upper case E, predicted entities as $\hat{e}$ and the matched ones as $\check{e}$. When relevant, we denote in which image 2D entities belong to using subscript, e_l corresponding to a 2D entity in the left

[2] We are confident that the reason is not a bad parameter choice of the SIFT processing, since another group confirmed our experience on the specific data set using their motion estimation algorithm.

image and e_r to a 2D entity in the right image. Furthermore, when relevant we make use of superscript e^t for describing the instant of time t at which a given entity was observed.

2.1 Rigid Body Motion

A Rigid Body Motion $\mathcal{M}$ consisting of a translation $\boldsymbol{t}$ and a rotation $\boldsymbol{r}$ is described by six parameters, three for the translation $\boldsymbol{t} = (t_1, t_2, t_3)$ and three for the rotation axis $\boldsymbol{r} = (r_1, r_2, r_3)$ (see Figure 1). This allows for the formulation of the transformation between a visual entity according to this motion.

$$\mathcal{M}^{(\boldsymbol{t},\boldsymbol{r})}(E) = \hat{E} \tag{1.1}$$

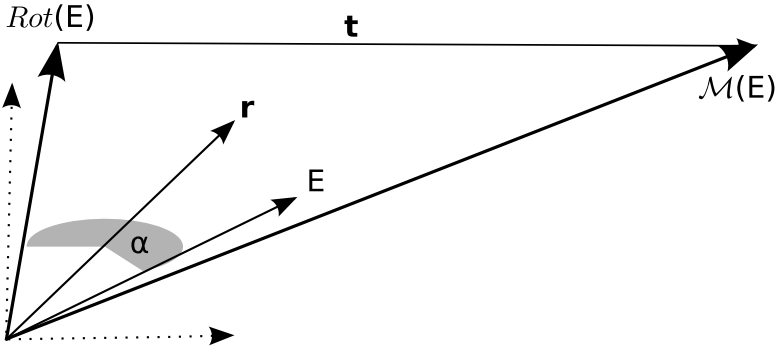


Fig. 1. Representations of a Rigid Body Motion by a combination of rotation (coded in axis-angle representation) and translation. First a rotation $\mathrm{Rot}(E)$ is performed around the axis $\boldsymbol{r}$. Then the 3D entity E is moved according to the translation vector $\boldsymbol{t}$. This allows for the formulation of the transformation between a visual entity in one frame, and the same entity in the next frame. The norm of this rotation axis codes the angle of rotation $\alpha = \|\boldsymbol{r}\|$.

The problem of computing the RBM from correspondences between 3D objects and 2D image entities is referred as 3D-2D pose estimation [46,47]. The 3D entity (3D object information) needs to be associated to a 2D entity (2D correspondence of the same object in the next image) according to the perspective projection $\mathcal{P}$.

$$\mathcal{P}(\mathcal{M}^{(\boldsymbol{t},\boldsymbol{r})}(E)) = \hat{e} \tag{1.2}$$

There exist approaches (in the following called projective approaches) that formalize constraints directly on equation (1.2) (see e.g., [30]). An alternative is, instead of formalizing the pose estimation problem in the image plane, to associate a 3D entity to each 2D entity: a 2D image point together with the optical center of the camera spans a 3D line (see figure 2a) and an image line together with the optical center generates a 3D plane (see figure 2b). In case of

a 2D point $\check{\boldsymbol{x}}$, we denote the 3D line that is generated in this way by $\boldsymbol{L}(\check{\boldsymbol{x}})$. The RBM estimation problem can be formulated for 3D entities as:

$$\mathcal{M}^{(\boldsymbol{t},\boldsymbol{r})}(\boldsymbol{X}) \in \boldsymbol{L}(\check{\boldsymbol{x}}) \tag{1.3}$$

where $\boldsymbol{X}$ is the 3D point and $\check{\boldsymbol{x}}$ the 2D point it is matched with. Such a formulation in 3D has been applied by, e.g., [48,47], coding the RBM estimation problem in a twist representation that can be computed iteratively on a linearized approximation of the RBM. For that, we want to formulate constraints between 2D image entities and 3D object entities, where a 2D image point together with the optical center of the camera spans a 3D-line (see Figure 2a) and an image line together with the optical center generates a 3D-plane (see Figure 2b).

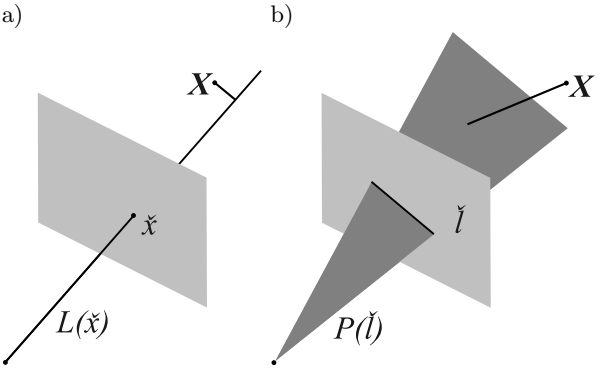


Fig. 2. Geometric interpretation of constraint equations (assuming the camera geometry is known): **a)** From an image point and the camera optical center, a 3D-line can be generated The 3D-point 3D-line constraint realizes the shortest Euclidean distance between the 3D-point and the 3D-line. **b)** From an image line and the camera optical center, a 3D-plane can be generated. The 3D-point 3D-plane constraint realizes the shortest Euclidean distance between the 3D-point and the 3D-plane.

A Rigid Body Motion $\mathcal{M}^{(\boldsymbol{t},\boldsymbol{r})}$ can be formalized in different ways, describing the same motion in different formulations. One of them is the formulation of *twists* [26], which is used in this work and described briefly in the following. Twists have a straightforward linear approximation (using a Taylor series expansion) and lead to a formalization that searches in the six dimensional space of RBMs (i.e, SE(3)). In the twist formulation, an RBM is understood as a rotation of angle α around a line $\boldsymbol{L}$ in 3D space with direction $\boldsymbol{w} = (\boldsymbol{w_1}, \boldsymbol{w_2}, \boldsymbol{w_3})$ and moment $\boldsymbol{w} \times \boldsymbol{q}$, where $\boldsymbol{q} = (q_1, q_2, q_3)$ is a point on that line. The vectors $\boldsymbol{w}$ and the cross-product of $\boldsymbol{w} \times \boldsymbol{q}$ are referred to as Plücker coordinates. In addition to the rotation, a translation with magnitude λ along the line $\boldsymbol{L}$ is performed. Then, an RBM can be represented as follows:

$$\hat{E} = e^{\alpha\tilde{\xi}}(E) = \mathcal{M}^{(\boldsymbol{t},\boldsymbol{r})}(E) \tag{1.4}$$

with

$$e^{\alpha\tilde{\xi}} = \sum_{n=0}^{\infty} \frac{1}{n!}(\tilde{\xi}\alpha)^n \tag{1.5}$$

where $\tilde{\xi}\alpha$ being the 4×4 matrix with 6 motion parameters to be estimated

$$\tilde{\xi}\alpha = \begin{pmatrix} 0 & -\alpha w_3 & \alpha w_2 & \alpha v_1 \\ \alpha w_3 & 0 & -\alpha w_1 & \alpha v_2 \\ -\alpha w_2 & \alpha w_1 & 0 & \alpha v_3 \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{1.6}$$

with

$$\begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix} = \begin{pmatrix} w_3 q_2 - w_2 q_3 + \lambda w_1 \\ w_1 q_3 - w_3 q_1 + \lambda w_2 \\ w_2 q_1 - w_1 q_2 + \lambda w_2 \end{pmatrix} \tag{1.7}$$

By using the exponential representation in Equation 1.5, a straightforward linearization is given by:

$$e^{\tilde{\xi}\alpha} \approx I_{4\times4} + \tilde{\xi}\alpha \tag{1.8}$$

A 3D-line $\boldsymbol{L}$ can be expressed as two 3D vectors $(\boldsymbol{\nu}, \boldsymbol{\mu})$. The vector $\boldsymbol{\nu}$ describes the direction and $\boldsymbol{\mu}$ describes the moment which is the cross product of a point $\boldsymbol{X}$ on the line and the direction $\boldsymbol{\mu} = \boldsymbol{X} \times \boldsymbol{\nu}$. The null space of the equation $\boldsymbol{X} \times \boldsymbol{\nu} - \boldsymbol{\mu} = \boldsymbol{0}$ is the set of all points on the line, and can be expressed in matrix form as follows:

$$\mathcal{F}^{\boldsymbol{L}}(\boldsymbol{X}) = \begin{pmatrix} 0 & \nu_3 & -\nu_2 & -\mu_1 \\ -\nu_3 & 0 & \nu_1 & -\mu_2 \\ \nu_2 & -\nu_1 & 0 & -\mu_3 \end{pmatrix} \begin{pmatrix} X_1 \\ X_2 \\ X_3 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \tag{1.9}$$

Combining the above formulation of a 3D-line with a 3D-point $\boldsymbol{X}$ allows the creation of the 3D-point/3D-line constraint using the linearization from Equation 1.8 as:

$$\mathcal{F}^{\boldsymbol{L}(\check{x})}\left((I_{4\times4} + \alpha\tilde{\xi})\boldsymbol{X}\right) = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \tag{1.10}$$

Here, the value $||\mathcal{F}^{\boldsymbol{L}(\check{\boldsymbol{x}})}(\hat{\boldsymbol{X}})||$ can be interpreted as the Euclidian distance between the moved point $\hat{\boldsymbol{X}}$ and the closest point on the line $\boldsymbol{L}$ [49,47]. Note that, although we have 3 equations for one correspondence the matrix is of rank 2 resulting in only 2 constraints.

A 3D-plane $\boldsymbol{P}$ can be expressed by defining the components of the unit normal vector $\boldsymbol{n}$ and a scalar (Hesse distance) δ_h. The null space of the equation

$\boldsymbol{n} \cdot \boldsymbol{X} - \delta_h = \boldsymbol{0}$ is the set of all points on the plane, and can be expressed in matrix form as follows:

$$\mathcal{F}^{\boldsymbol{P}(\check{\boldsymbol{l}})}(\boldsymbol{X}) = \begin{pmatrix} n_1 & n_2 & n_3 & -\delta_h \end{pmatrix} \begin{pmatrix} X_1 \\ X_2 \\ X_3 \\ 1 \end{pmatrix} = 0 \tag{1.11}$$

Combining the above formulation of a 3D-plane together with a 3D-point $\boldsymbol{X}$ allows for the creation of the 3D-point/3D-plane constraint using the linearization from Equation 1.8:

$$\mathcal{F}^{\boldsymbol{P}(\check{\boldsymbol{l}})} \left((I_{4\times 4} + \alpha\tilde{\xi})\boldsymbol{X} \right) = 0 \tag{1.12}$$

Note that the value $|\mathcal{F}^{\boldsymbol{P}}(\hat{\boldsymbol{X}})|$ can be interpreted as the Euclidean distance between the moved point $\hat{\boldsymbol{X}}$ and the closest point on the plane $\boldsymbol{P}$ (see [47]). These 3D-point/3D-line and 3D-point/3D-plane constraints result in a system of linear equations, the solution of which is found by iterative optimization (for details see [50]).

2.2 Feature Extraction

Visual Primitives form a feature based image representation that has been developed in the European project ECOVISION [51], which was focused on the modeling of early cognitive vision [3]. We believe that this representation provides robust means for finding correspondences across stereo and time, which are necessary for addressing the problem of RBM estimation.

Visual Primitives are extracted at image points representing edge structures, encoded as values of different visual modalities: position $\boldsymbol{m}$, orientation θ, phase ω, color $\boldsymbol{c}$ and local optic flow $\boldsymbol{f}$. Consequently, a multi-modal primitive is described by the following vector:

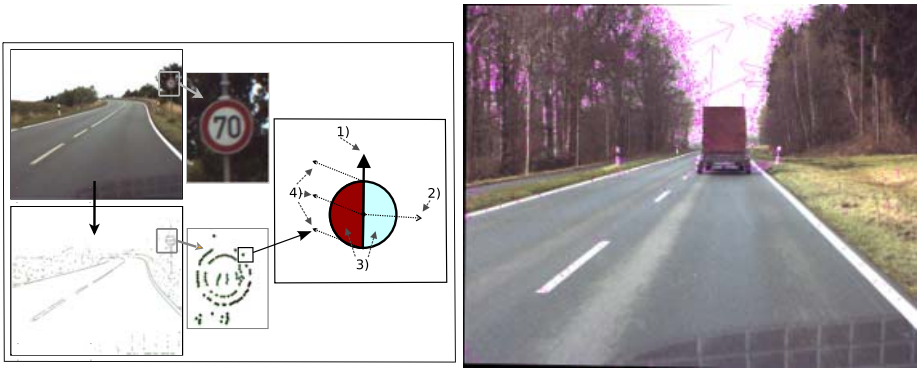
$$\boldsymbol{\pi} = (\boldsymbol{m}, \theta, \omega, \boldsymbol{c}, \boldsymbol{f}, p)^T \tag{1.13}$$

where p is the size of the image patch represented by the primitive (see Figure 3a). The problem of matching primitives was discussed in [52,53], and we make use of the same criteria in the present evaluation. The matching criterion over all modalities is defined as:

$$d(\boldsymbol{\pi}_i, \boldsymbol{\pi}_j) = \sum_k w_k d_k(\boldsymbol{\pi}_i, \boldsymbol{\pi}_j) \tag{1.14}$$

where w_k is the relative weighting of the modality k and d_k being the distance of the modality k between the two primitives $\boldsymbol{\pi}_i$ and $\boldsymbol{\pi}_j$.

Scale-Invariant Feature Transform feature extraction provides a set of robust features invariant to scaling and rotation [36]. Moreover, SIFT features are also very resilient to the effects of image noise. These properties make SIFT features widely used in many vision application involving the task of feature matching. SIFT features are extracted in a four step process [36].



a) b)

Fig. 3. The feature descriptors used in this work. **a)** Extracted visual primitives, modeling line features using multiple visual modalities, where: 1. stands for the orientation (θ), 2. for the phase (ω), 3. for the color ($\boldsymbol{c}$), and 4. for the optic flow ($\boldsymbol{f}$). **b)** Features extracted using the Scale Invariant Feature Transform (SIFT).

Scale-space extrema detection: The first stage of computation searches over all scales and image locations by using a difference-of-Gaussian function to identify potential interest points that are invariant to scale and orientation.

Keypoint localization: During the second step a detailed model is fit to determine location and scale for each candidate location, then keypoints are selected based on measures of their stability.

Orientation assignment: One or more orientations are assigned to each keypoint location based on local image gradient directions.

Keypoint descriptor: The local image gradients are measured at the selected scale in the region around each keypoint. These are transformed into a representation that allows for significant levels of local shape distortion and change in illumination.

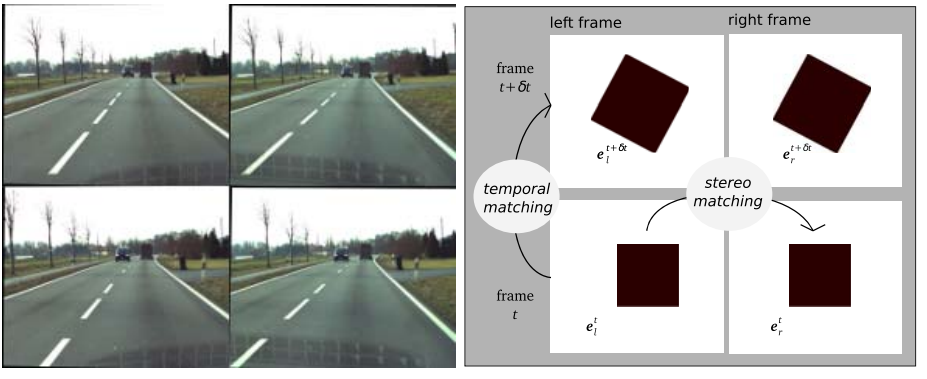
During orientation assignment, the gradient orientation histogram is computed in the neighborhood of the feature point location. This serves as the feature-descriptor defined as a vector containing the values of all the orientation histogram entries, corresponding to the lengths of the arrows shown in Figure 3b. All the properties of the feature point are measured relative to its dominant orientation, providing invariance to rotation. Matchings are found by identifying the nearest neighbor from the set of SIFT features, defined as the keypoint with minimum Euclidean distance for the invariant descriptor vector.

2.3 Feature Matching

Having defined the process of feature extraction and metrics for matching, we now want to apply these to the image sequence used in this paper. In the context of RBM estimation from stereo sequences, the correspondence problem is twofold: first stereo correspondences have to be found, then temporal

correspondences (as described in a later subsection). Note that the temporal correspondence problem suffers from higher ambiguity than stereo matching, since the epipolar constraint is not directly applicable, and need to be replaced by a neighborhood search.

One of the sequences used is shown in Figure 4a, where the left column (resp. right) contains the images obtained from the left (resp. right) camera, while the rows show the stereo images taken at different time steps. For image acquisition, a calibrated stereo rig was used and the captured images have been undistorted and rectified. The task of finding correspondences consists of two steps illustrated in Figure 4b is briefly described in the next two subsections.



a) b)

Fig. 4. Feature matching: **a)** Example frames from one of the sequences, columns showing images from the left (resp. right) camera, and the rows depicting images from different frames over times. **b**) The two tasks of feature matching required in the context of 3D-2D pose estimation. First stereo matches have to be found (at time t, see bottom row). If a stereo match was found for a given feature, the corresponding feature in next frame (time $t + \delta t$) has to be identified (illustrated in the first column).

Stereo Matching: Considering a feature e_l in the left image, the corresponding feature e_r in the right image has to be found, as illustrated by the bottom row in Figure 4b. Since the epipolar geometry is known and the images are rectified (see Figure 4a), the search space can be limited to horizontal scan lines. Primitives are matched using the similarity constraint defined by their modalities (see [52] for details). The matching of SIFT features uses a nearest neighbors search using a k-d tree [54] and an approximation algorithm, called the Best-Bin-First (BBF) [55]. This is an approximation algorithm in the sense that it returns the closest neighbor with a high probability. For an evaluation of the matching performance, we refer the reader to [36] for SIFT features, and [52] for primitives. An example of stereo matching from SIFT features is illustrated in Figure 5.

Having found stereo correspondences of the different features (representing lines or points), a 3D point is computed for each correspondence. The 3D point is regarded as valid if it is located in front of the camera.

Temporal Matching: Temporal matching involves finding correspondences between the 3D-points reconstructed from stereo at time t and the 2D features (primitives and SIFT) extracted at time $t + \delta t$ (upper row in Figure 4a). This is achieved by matching all e_l^t with all $e_l^{t+\delta t}$ for which there exists a valid 3D point. Temporal matching is done in the very same manner as stereo-matching with the exception that the epipolar constraint is not applicable. Therefore, the computational complexity is significantly higher than for stereo matching. Fortunately, the number of features for temporal matching has already been considerably reduced. Furthermore, we constrain the search of temporal matches within a neighborhood of the feature's previous position. The size of this neighborhood is called temporal disparity threshold. For the experiments, we both use a maximum and minimum threshold to this disparity. The minimum disparity threshold disregards temporal feature matches which do not move sufficiently in the image. These normally correspond to structures very distant to the camera, which would not serve as valuable contributions to the set of temporal features matches. During the experiments a threshold of minimum 2 and maximum of 150 pixels have been found adequate for all outdoor sequences, allowing to match nearby features at speed over 100km/h.

For the indoor sequence (see Figure 6d) where an object is moving, rather than the camera, a segmentation of background and the object becomes necessary. Using a minimum disparity threshold solves this problem for all point features (including SIFT), since the background is static, and therefore all non-moving temporal matches can be disregarded. For primitives the problem is different. Since primitives often are part of a global edge structure, temporal matches for a given primitive may be found at any point along the global edge structure, since primitives located on a global edge have similar attributes. Therefore, we constrain the temporal matching of primitives to a region around the robot, reducing the number of incorrect matched primitives.

After temporal matching, each correspondence contains a left entity in the current frame e_l^t, and the entity matched in the next left frame $e_l^{t+\delta t}$, where e_l^t has an associated 3D point from stereopsis (matched with e_r^t). An example of temporal matching of SIFT features is illustrated in Figure 5.

2.4 Applying the Pose Estimation Algorithm

Having 3D-2D feature correspondences we can now apply the pose estimation algorithm for estimating the motion in stereo sequences. The first step is to project the 2D temporal correspondence matches from time $t + \delta t$ in the image plane to 3D coordinates, by using the information from camera calibration. The corresponding 2D-points (SIFT) or lines (primitives) from the next left frame generate 3D-lines or 3D-planes, respectively. So, from the 3D-2D correspondences, we derive constraints describing the motion between two 3D entities (see Equation 1.10, 1.12). A 3D-point/2D-line correspondence (primitives) leads to one independent constraint equation and a 3D-point/2D-point correspondence (SIFT)

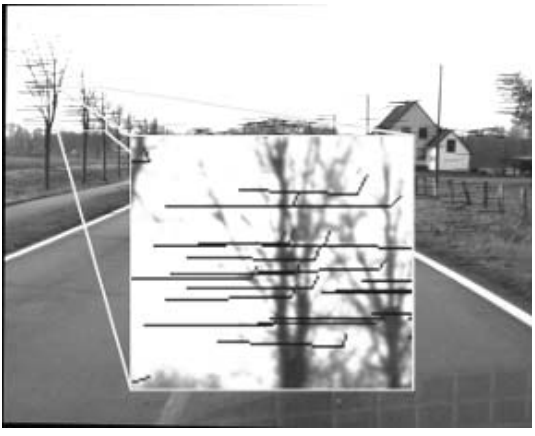


Fig. 5. Results for the matching of SIFT features (corresponding to the left-top frame in figure 4a), depicting stereo and temporal matches. The disparity of stereo correspondences between e_l^t and e_r^t are depicted as the horizontal lines. The disparity of temporal correspondences between e_l^t and $e_l^{t+\delta t}$ are shown as the almost vertical lines.

leads to two independent constraint equations [50]. Earlier work has shown [56] that 3D-point/2D-point correspondences produce more accurate estimates with the same number of correspondences.

The constraints derived from Equation 1.10 and 1.12 result in a system of linear equations for which the solution is found iteratively (for details see [24,50]).

Eliminating outliers using RANSAC. Another challenge is the selection of correspondences resulting in correct estimates, referred to as inliers. Erroneous correspondences, referred to as outliers, are introduced due to the inherent ambiguity of stereo and temporal matching of local features, and should be neglected. By applying the Random Sample Consensus [17] (RANSAC) approach, we identify a set of inliers resulting in more accurate estimation results. The algorithm explained below uses a correspondence set containing either SIFT, primitives, or a combination of both.

1. The set of correspondences is divided into a generation set and an evaluation set, where the evaluation set is only to be used for performance statistics.
2. From the generation set, a random set of correspondences is selected, corresponding to 8 independent constraints.
3. The pose estimation algorithm is run with this random set.
4. The inverse of the computed motion is applied to all remaining 3D entities reprojected in the generation set and back into the image. The distance between the original and re-projected 2D feature, referred to as the *deviation*, serves as measure of accuracy for the estimated motion.
5. A consensus set is formed from all correspondences of the generation set, using correspondences with a deviation below a given threshold τ.

6. If the size of the consensus set is above a certain percentage ξ of the of size of the generation (referred to as the consensus threshold), the estimated motion is regarded as correct and the algorithm continues, otherwise the algorithm goes back to step 2.
7. The pose estimation algorithm is re-run with the whole consensus set, which is considered to only contain inliers.

During the experiments a deviation threshold of $\tau = 1$ pixel for SIFT an $\tau = 2$ pixels for primitives have been found as adequate. The consensus threshold of $\xi = 70\%$ has shown to result in precise motion estimates while at the same time succeeding for most of the frames.

3 Results

The four stereo image sequences used are recorded with a calibrated camera system and contain three outdoor scenes and one indoor scene (see Figure 6). Note that the outdoor sequences s_03a_01#000 and s_06a_01#000 are available as Set 3 on the web site `http://www.mi.auckland.ac.nz/EISATS/`. On this website, initiated by Reinhard Klette and colleagues (see, e.g., [57]), data sets recorded in the driver assistance domain are provided for comparison of computer vision algorithms. The indoor sequence is provided together with calibration matrices and ground truth on the web site `http://www.mip.sdu.dk/covig/Data/PACO`.

a) b) c) d)

Fig. 6. Example frames from each of the four sequences: a) Lippstadt-Town2, b) s_03a_01#000, c) s_06a_01#000, d) PACO-5

Table 1. Average and minimal numbers (in italics) of extracted and matched (stereo and temporal) features per frame

Image Sequence	number of frames	Primitives extracted	SIFT extracted	Primitives matched	SIFT matched
Lippstadt-Town2	437	1679(*1427*)	4818(*2500*)	665(*194*)	572(*136*)
s_03a_01#000	1180	1223(*708*)	1347(*70*)	669(*49*)	395(*0*)
s_06a_01#000	848	779(*250*)	1182(*89*)	449(*291*)	216(*0*)
PACO-5	60	1857(*1565*)	442(*335*)	394(*239*)	15(*6*)

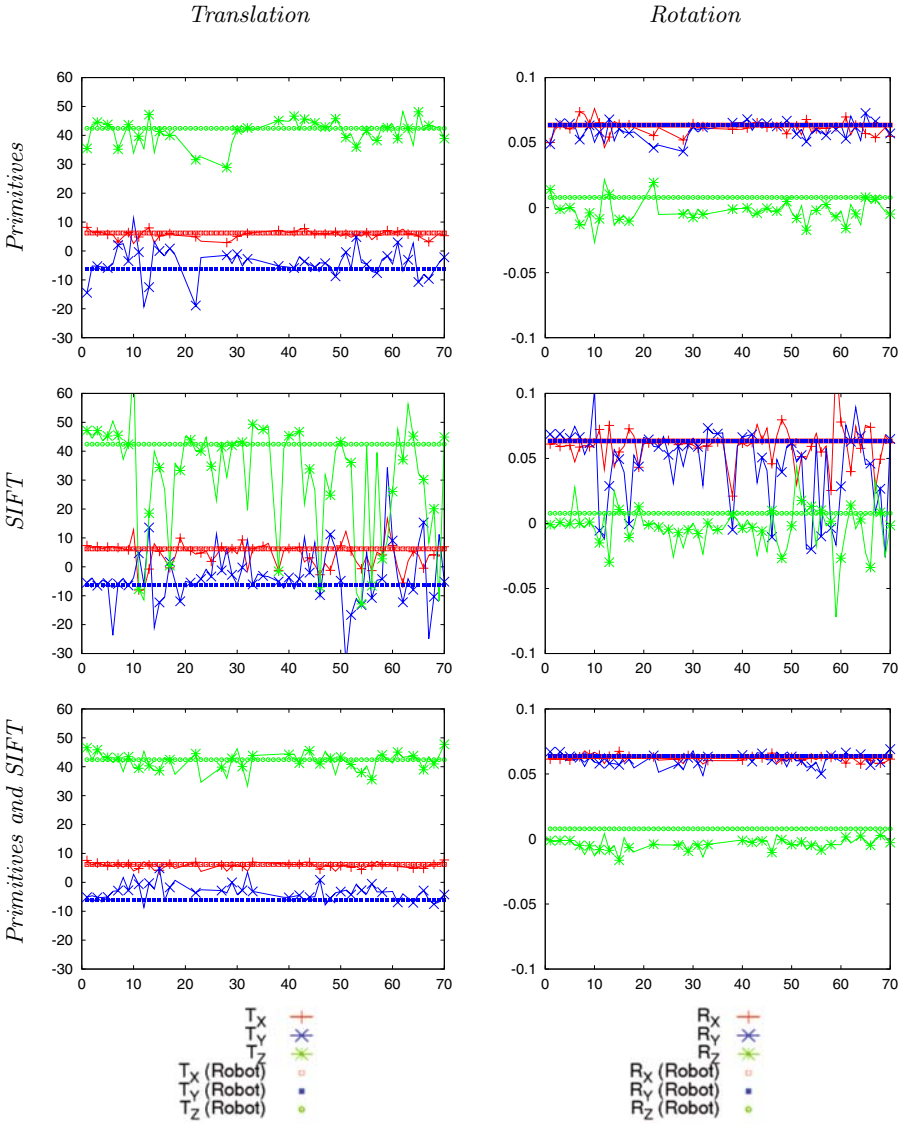


Fig. 7. Results for motion estimates for indoor sequence PACO-5. Left column contains plots translation in mm and right column contains the plots displaying rotation angle in radians. Each row corresponds to a different feature set: primitives, SIFT and a combination of primitives and SIFT.

As illustrated in Figure 6a-c the outdoor scene consists of images recorded through a car's windscreen. For these outdoor sequences, logging of car data by an onboard computer provides information about velocity and steering angle.

Table 2. Mean error for image sequence PACO-5

Feature Set	Tx (mm)	Ty (mm)	Tz (mm)	Rx (rad)	Ry (rad)	Rz (rad)
Primitives	1.0	4.1	3.2	0.003	0.005	0.005
SIFT	3.2	6.9	14.1	0.009	0.018	0.008
Primitives + SIFT	0.8	3.0	2.5	0.002	0.003	0.004

Hence, this data does not describe the full motion (only translation and rotation around the Y-axis) and is subject to noise. Therefore, it can only be used as reference and should not be considered as actual ground truth. The car data is shown in Figure 8. The left column of plots shows the car's translation in mm. and the right column shows the car's rotation angle in radians. In these figures, the rotation along the Y-axis corresponds to steering input and the translation on Z-axis represents the forward motion of the car. During turns, the car moves forward, sideways, and rotates along the Y-axis. Notice the slight correlation between rotation (right columns) and translation along the X-axis (see left column). For the Lippstadt-Town2 sequences no rotation data is available and the forward translation (T_Z) is equal to the velocity provided by the car's onboard computer.

The indoor scene depicts a robot holding and moving a blue vase, and consists of a sequence of 60 stereo-frames. The camera is situated in front of the robot and, throughout the entire sequence, the robot rotates the vase 360 degrees around one axis. In this experiment, we used an industrial robot, meaning that the robot's controller provides high accuracy ground truth data, plotted alongside with the motion estimation results (see Figure 7)

Since the motion estimation uses statistical methods for removing outliers, the number of extracted features is directly related to the robustness of the estimated motion. If too few features are extracted and matched, RANSAC will no longer be applicable. The number of extracted and matched features is shown in Table 1. The reason for the low number of correspondences in the PACO-5 sequence is explained by the relatively small size of the moving object in the images, whereas all outdoor sequences undergo ego-motions and therefore displays an apparent world motion (apart from other moving cars). Furthermore we see significant changes in the number of extracted features between the different outdoor sequences. As depicted in Table 1, the number of extracted SIFT features is significantly higher in an urban environment (Lippstadt-Town2) than on a countryroad (s_03a_01#000 and s_06a_01#000).

Indoor sequence

Results are shown in Figure 7, depicting both ground truth and estimated motion. Furthermore, the mean error over entire sequences is recorded for all three feature sets in Table 2. In this table, the primitives lead to more accurate motion estimates than SIFT features, and the combination of the two lead to the best results. In particular, the motion estimation based on SIFT features shows large deviation from the true motion for many frames (see second row in

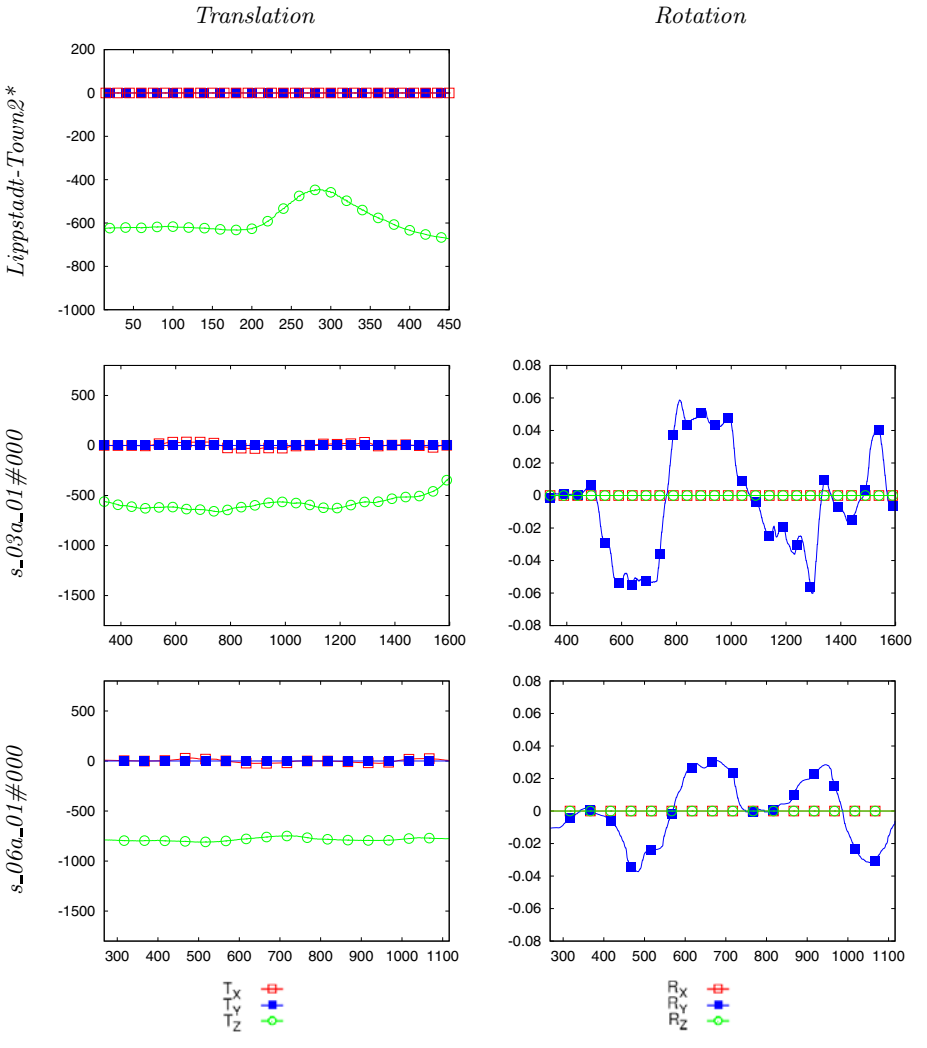


Fig. 8. Reference data computed from car data for all outdoor sequences. Left column contains plots for the car translation in mm. and right column contains the plots displaying rotation angle in radians. Each row corresponds to a different reference sequence. * For the Lippstadt-Town2 sequence (first row) no rotation data is available and T_Z is computed directly from the car's velocity.

Figure 7), due to the comparatively small amount of matched SIFT features (see Table 1). The estimation would basically fail in terms of most applications that make use of the estimated motion data. The reason for this is the small number of extracted and matched SIFT features (see Tab.1 and Figure 12).

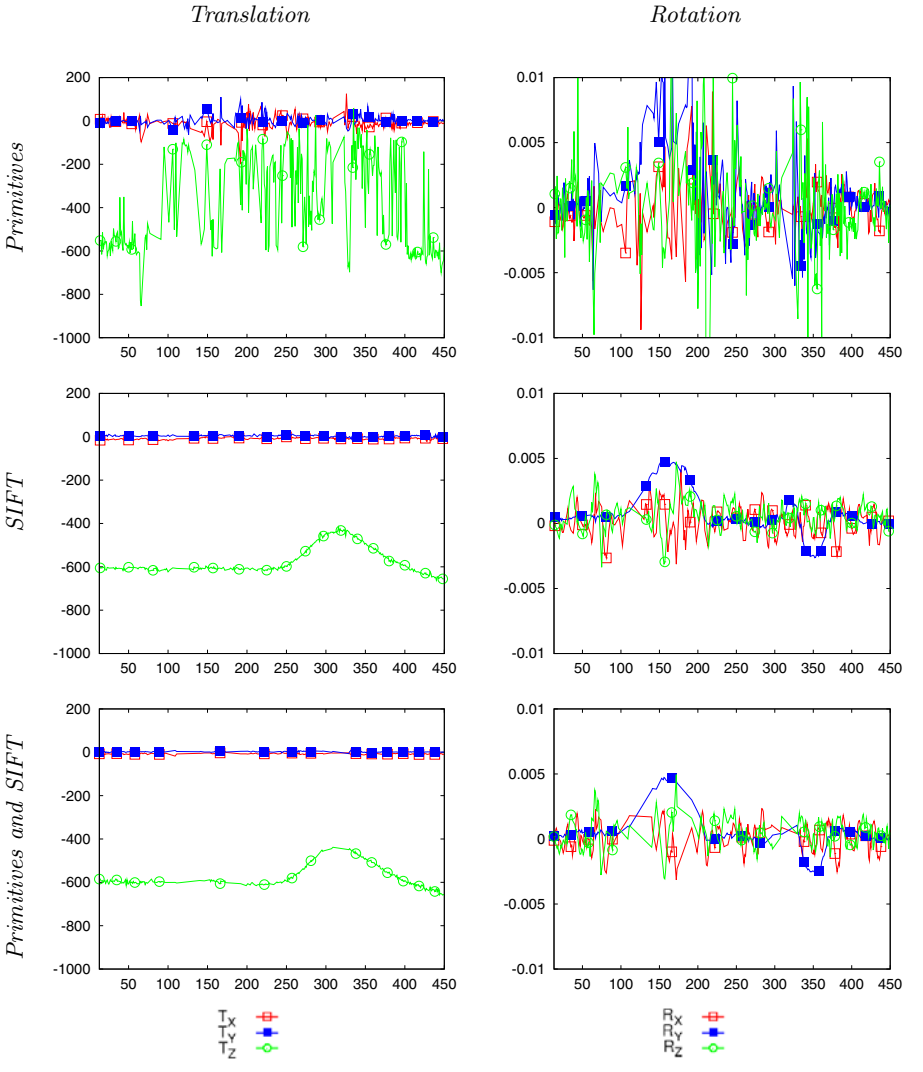


Fig. 9. Results of ego-motion estimates for the outdoor sequence 'Lippstadt-Town2' over 437 frames in a city environment. The left column shows translation estimates, in mm., and the right column shows rotation angle estimates, in radians.

Outdoor Sequences

For the outdoor sequences with a sufficient number of extracted and matched features (see Table 1), SIFT features alone lead in general to better or similar results than primitives alone (compare the top to the middle rows in Figure 9, 10 and 11). This is explained by the fact that line correspondences fail to constrain the motion in all directions. Especially, in the outdoor scenes the lane structure

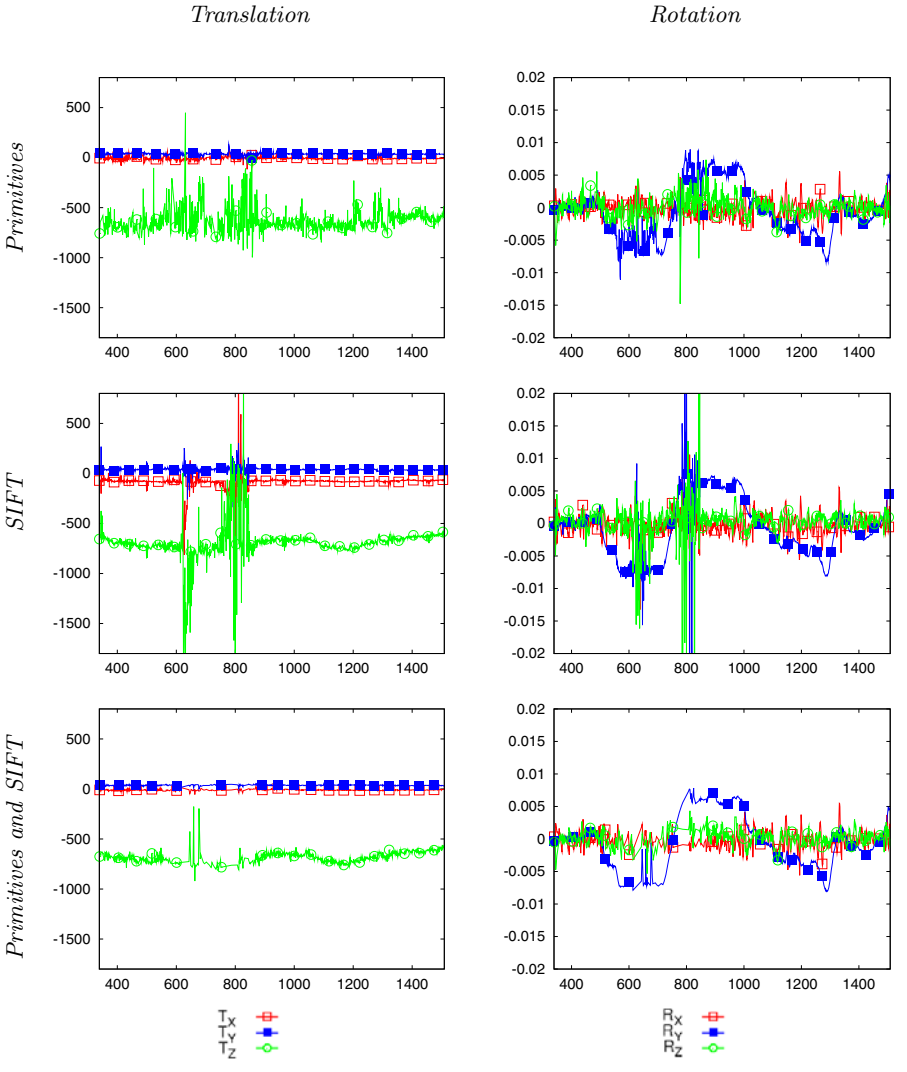


Fig. 10. Results of ego-motion estimates for outdoor sequence 's_03a_01#000' over 1180 frames. The left column shows translation estimates, in mm., and the right column shows rotation angle estimates, in radians. Each row corresponds to a different feature set.

(in particular the lane markers) dominate the scenes, and due to their particular orientation (i.e., nearly radial from the car's heading), they do not constrain the ego-motion in the z–direction. Therefore, the estimation results of the forward translation, based on primitives alone (see first row in Figure 9, 10 and 11) are very unprecise. However, it can be seen that primitive correspondences constrain effectively the 5 other motion parameters, and even in a slightly better way than

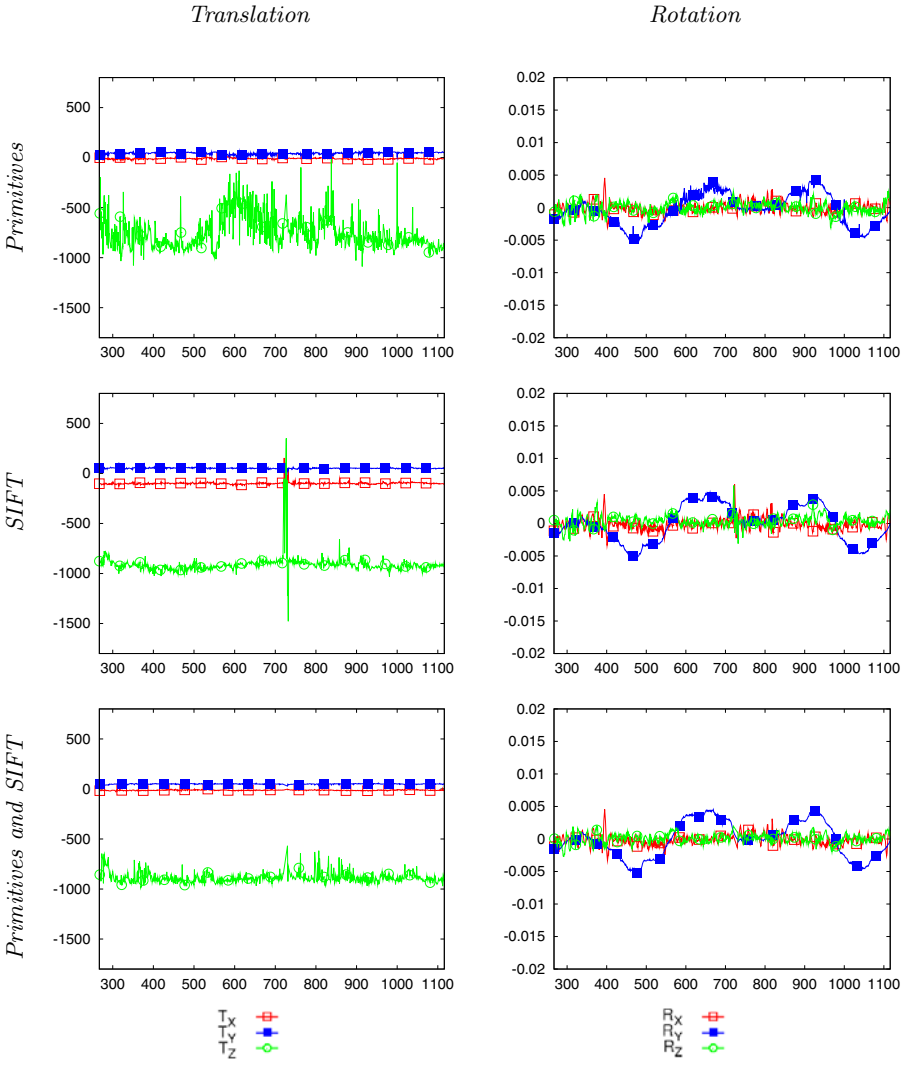
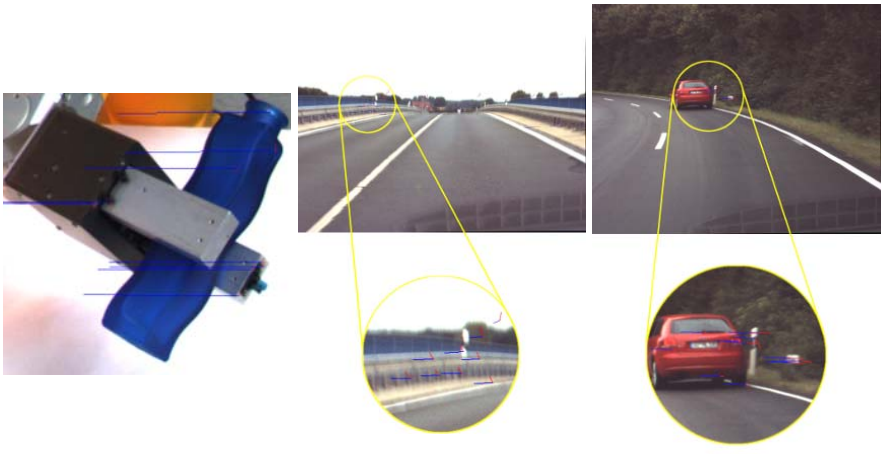


Fig. 11. Results for ego-motion estimates for outdoor sequence 's_06a_01#000' over 848 frames. The left column shows translation estimates, in mm., and the right column shows rotation angle estimates, in radians. Each row corresponds to a different feature set.

the SIFT correspondences. We assume that this is caused by the higher precision of the primitive localization compared to SIFT.

Furthermore we observe that usage of SIFT features in an urban environment, provides very accurate estimations, when comparing the reference data with the estimated motion (first row in Figure 8 and second row in Figure 9). Combining



a) b) c)

Fig. 12. Examples of problematic image data: **a)** for the indoor sequence only few SIFT features are extracted and matched. **b)** Some frames from the outdoor sequences (as in 's_06a_01#000', frame 723) are blurred due to bumps in the road, resulting in large estimation errors (spikes) as shown in Figure 11. **c)** Independently moving objects (e.g., other vehicles on the road) provide temporal matches which are not coherent with the ego motion. This becomes especially a problem if the scene contains few matched features (as in 's_03a_01#000', frame 600-800), resulting in erroneous motion estimates (see Figure 10).

SIFT with additional primitive correspondences does not further improve the estimates (see third row in Figure 9), since the great number of extracted and matched SIFT features already serves as a sufficient correspondence set.

However in the other environmets, some frames do not contain enough SIFT features, due to the small amount of texture within the scene (see Figure 12c). Then, few spurious features can very quickly result in large errors in the estimated RBM. Another source of outliers in this type of scenario are pot holes or speed bumps (see Figure 12b). These sudden and violent vertical motions result in blurred images, which make feature extraction and matching a difficult task. In such cases, we observe that a combination of SIFT and primitives make the estimation more stable to outliers. Over all sequences, we observe that a combination of features (see bottom row in Figure 10 and 11) consistently improves the robustness. Note that, we did not make use of any temporal regularization (which of course would improve the results further) since we were interested in investigating the different properties and consequences of different kinds of correspondences. In this context, temporal regularization would effectively hide the occasional failures of the estimation, that are relevant indicators of the estimation's robustness.

4 Conclusion

We have evaluated point and line correspondences on a set of sequences with large variations across and within the sequences. There are a number of issues involved in such an investigation, namely (1) the extraction process of features, (2) the correspondence finding process, (3) the mathematical formulation of constraints allowing for their mixing and (4) the efficient handling of outliers.

For the process of finding correspondences, an evaluation of matching needs to be done, using ground truth data in form of disparity maps.

From a mathematical point of view, line features represent weaker correspondences than point features as they provide only one constraint equation, against two for the point constraints. Because of this, pathologic scene structures, like the road markers in the outdoor sequences can lead to an ill-definition of the motion estimation problem. On the other hand, point features such as SIFT lead to an ill-definition in the case of the indoor scene, due to the lack of texture in the scene, whereas edge features give stable results. The same holds true for certain sub-parts of the outdoor sequences which were dominated by sky and edge structures. Besides the avoidance of severe outliers, we also observed that the additional use of edge-features increases the precision of the estimates due to their frequent occurrence in visual scenes as well as their more accurate localization.

As a consequence, besides the mathematical properties of the constraints, we need to take the actual distribution of features in images into account. Here, we can observe that this distribution changes significantly across as well as within sequences. Hence, a robust motion estimation must rely on the combination of point and line features and hence relies on a *rich feature processing* as well as on a formulation of the motion estimation problem that allows for the mixing of both kinds of correspondences.

SIFT features become extracted at local textured structures. In our current research, we investigate the use of junctions as an additional feature type which is extractable with high local precision and rich semantic. We expect a further improvement of stability and precision by such further enrichment of our representations.

Acknowledgement

This work has been supported by the European Commission - FP6 Project DRIVSCO (IST-016276-2). We also would like to thank Daniel Grest and Emre Başeski for fruitful discussions.

References

1. Ball, R.: The theory of screws. Cambridge University Press, Cambridge (1900)
2. Zetzsche, C., Barth, E.: Fundamental limits of linear filters in the visual processing of two dimensional signals. Vision Research 30, 1111–1117 (1990)

3. Krüger, N., Hulle, M.V., Wörgötter, F.: Ecovision: Challenges in early-cognitive vision. International Journal of Computer Vision 72, 5–7 (2007)
4. Bregler, C., Malik, J.: Tracking people with twists and exponential maps. In: IEEE computer Society conference on Computer Vision and Pattern Recognition, pp. 8–15 (1998)
5. Christy, S., Horaud, R.: Iterative pose computation from line correspondences. Comput. Vis. Image Underst. 73, 137–144 (1999)
6. Ansar, A., Daniilidis, K.: Linear pose estimation from points or lines. In: Heyden, A., Sparr, G., Nielsen, M., Johansen, P. (eds.) ECCV 2002. LNCS, vol. 2350, pp. 282–296. Springer, Heidelberg (2002)
7. Lepetit, V., Fua, P.: Monocular model-based 3d tracking of rigid objects. Found. Trends. Comput. Graph. Vis. 1, 1–89 (2005)
8. Roach, J., Aggarwall, J.: Determining the movement of objects from a sequence of images. IEEE Transactions on Patterm Analysis and Machine Intelligence 2, 554–562 (1980)
9. Lowe, D.G.: Three–dimensional object recognition from single two images. Artificial Intelligence 31, 355–395 (1987)
10. Bruss, A., Horn, B.: Passive navigation. Computer Vision, Graphics, and Image Processing 21, 3–20 (1983)
11. Horn, B.: Robot Vision. MIT Press, Cambridge (1994)
12. Waxman, A., Ullman, S.: Surface structure and 3-D motion from image flow: A kinematic analysis. International Fournal of Robot Research 4, 72–94 (1985)
13. Negahdaripour, S., Horn, B.: Direct passive navigation. IEEE Transactions on Pattern Analysis and Machine Intelligence 9, 168–176 (1987)
14. Steinbach., B.G.E.: An image-domain cost function for robust 3-d rigid body motion estimation. In: 15th International Conference on Pattern Recognition (ICPR 2000, vol. 3, pp. 823–826 (2000)
15. Steinbach, E.: Data driven 3-D Rigid Body Motion and Structure Estimation. Shaker Verlag (2000)
16. Torr, P.H.S., Zisserman, A.: Feature based methods for structure and motion estimation. In: ICCV 1999: Proceedings of the International Workshop on Vision Algorithms, London, UK, pp. 278–294. Springer, Heidelberg (2000)
17. Fischler, R., Bolles, M.: Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. Communications of the ACM 24, 619–638 (1981)
18. Schaffalitzky, F., Zisserman, A., Hartley, R.I., Torr, P.H.S.: A six point solution for structure and motion. In: Vernon, D. (ed.) ECCV 2000. LNCS, vol. 1842, pp. 632–648. Springer, Heidelberg (2000)
19. Horaud, R., Conio, B., Leboulleux, O., Lacolle, B.: An analytic solution for the perspective 4-point problem. Comput. Vision Graph. Image Process. 47, 33–44 (1989)
20. Dhome, M., Richetin, M., Lapreste, J.T.: Determination of the attitude of 3d objects from a single perspective view. IEEE Trans. Pattern Anal. Mach. Intell. 11, 1265–1278 (1989)
21. Haralick, R., Joo, H., Lee, C., Zhuang, X., Vaidya, V., Kim, M.: Pose estimation from corresponding point data. Systems, Man and Cybernetics, IEEE Transactions on 19, 1426–1446 (1989)
22. Liu, Y., Huang, T., Faugeras, O.: Determination of camera location from 2-d to 3-d line and point correspondence. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), vol. 12, pp. 28–37 (1989)

23. Phong, T., Horaud, R., Yassine, A., Tao, P.: Object pose from 2-D to 3-D point and line correspondences. International Journal of Computer Vision 15, 225–243 (1995)
24. Rosenhahn, B., Granert, O., Sommer, G.: Monocular pose estimation of kinematic chains. In: Dorst, L., Doran, C., Lasenby, J. (eds.) Applied Geometric Algebras for Computer Science and Engineering, pp. 373–383. Birkhäuser, Basel (2001)
25. Bretzner, L., Lindeberg, T.: Use your hand as a 3-D mouse, or, relative orientation from extended sequences of sparse point and line correspondences using the affine trifocal tensor. In: Burkhardt, H.-J., Neumann, B. (eds.) ECCV 1998. LNCS, vol. 1406, pp. 141–157. Springer, Heidelberg (1998)
26. Murray, R., Li, Z., Sastry, S.: A mathematical introduction to robotic manipulation. CRC Press, Boca Raton (1994)
27. Grest, D., Herzog, D., Koch, R.: Monocular body pose estimation by color histograms and point tracking. In: DAGM-Symposium, pp. 576–586 (2006)
28. Grest, D., Petersen, T., Krüger, V.: A Comparison of Iterative 2D-3D Pose Estimation Methods for Real-Time Applications. In: Salberg, A.-B., Hardeberg, J.Y., Jenssen, R. (eds.) SCIA 2009. LNCS, vol. 5575, pp. 706–715. Springer, Heidelberg (2009)
29. Dementhon, D.F., Davis, L.S.: Model-based object pose in 25 lines of code. International Journal of Computer Vision 15, 123–141 (1995)
30. Araujo, H., Carceroni, R., Brown, C.: A fully projective formulation to improve the accuracy of lowe's pose–estimation algorithm. Computer Vision and Image Understanding 70, 227–238 (1998)
31. Wolf, L., Shashua, A.: Lior wolf and a. shashua. on projection matrices $p^k - > p^2, k = 3, ., 6$, and their applications in computer vision. In: Proceedings of the 8th International Conference on Computer Vision, pp. 412–419. IEEE Computer Society Press, Los Alamitos (2001)
32. Avidan, S., Shashua, A.: Trajectory triangulation: 3d reconstruction of moving points from a monocular image sequence. IEEE Transactions on Pattern Analysis and Machine Intelligence 22, 348–357 (2000)
33. Wedel, A., Rabe, C., Vaudrey, T., Brox, T., Franke, U., Cremers, D.: Efficient dense scene flow from sparse or dense stereo data. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part I. LNCS, vol. 5302, pp. 739–751. Springer, Heidelberg (2008)
34. Rosenhahn, B., Brox, T., Cremers, D., Seidel, H.P.: Modeling and tracking line-constrained mechanical systems. In: Sommer, G., Klette, R. (eds.) RobVis 2008. LNCS, vol. 4931, pp. 98–110. Springer, Heidelberg (2008)
35. Felsberg, M., Kalkan, S., Krüger, N.: Continuous dimensionality characterization of image structures. Image and Vision Computing (accepted for publication in a future issue)
36. Lowe, D.G.: Distinctive Image Features from Scale-Invariant Keypoints. International Journal of Computer Vision 2, 91–110 (2004)
37. Mikolajczyk, K., Schmid, C.: A performance evaluation of local descriptors. IEEE Transactions on Pattern Analysis and Machine Intelligence 27, 1615–1630 (2005)
38. Moravec, H.: Obstacle avoidance and navigation in the real world by a seeing robot rover. Technical Report CMU-RI-TR-3, Carnegie-Mellon University, Robotics Institute (1980)
39. Harris, C.G., Stephens, M.: A combined corner and edge detector. In: 4th Alvey Vision Conference, pp. 147–151 (1988)
40. Harris, C.G.: Geometry from visual motion. MIT Press, Cambridge (1992)

41. Zhang, Z., Deriche, R., Faugeras, O., Luong, Q.T.: A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry. Artificial Intelligence 87, 87–119 (1995)
42. Kalkan, S., Shi, Y., Pilz, F., Krüger, N.: Improving junction detection by semantic interpretation. In: VISAPP (1), pp. 264–271 (2007)
43. Pollefeys, M., Koch, R., van Gool, L.: Automated reconstruction of 3D scenes from sequences of images. ISPRS Journal of Photogrammetry and Remote Sensing 55, 251–267 (2000)
44. Lowe, D.G.: Robust model-based motion tracking through the integration of search and estimation. Int. J. Comput. Vision 8, 113–122 (1992)
45. Krüger, N., Jäger, T., Perwass, C.: Extraction of object representations from stereo imagesequences utilizing statistical and deterministic regularities in visual data. In: DAGM Workshop on Cognitive Vision, pp. 92–100 (2002)
46. Grimson, W. (ed.): Object Recognition by Computer. The MIT Press, Cambridge (1990)
47. Rosenhahn, B., Sommer, G.: Adaptive pose estimation for different corresponding entities. In: Van Gool, L. (ed.) DAGM 2002, vol. 2449, pp. 265–273. Springer, Heidelberg (2002)
48. Rosenhahn, B., Perwass, C., Sommer, G.: Cvonline: Foundations about 2d-3d pose estimation. In: Fisher, R. (ed.) CVonline: On-Line Compendium of Computer Vision (2004), http://homepages.inf.ed.ac.uk/rbf/CVonline/
49. Selig, J.: Some remarks on the statistics of pose estimation. Technical Report SBU-CISM-00-25, South Bank University, London (2000)
50. Krüger, N., Wörgötter, F.: Statistical and deterministic regularities: Utilisation of motion and grouping in biological and artificial visual systems. Advances in Imaging and Electron Physics 131, 82–147 (2004)
51. ECOVISION: Artificial visual systems based on early-cognitive cortical processing (EU–Project) (2001–2003), http://www.pspc.dibe.unige.it/ecovision/project.html
52. Pugeault, N., Krüger, N.: Multi–modal matching applied to stereo. In: Proceedings of the BMVC 2003, pp. 271–280 (2003)
53. Krüger, N., Felsberg, M.: An explicit and compact coding of geometric and structural information applied to stereo matching. Pattern Recognition Letters 25(8), 849–863 (2004)
54. Freidman, J.H., Bentley, J.L., Finkel, R.A.: An algorithm for finding best matches in logarithmic expected time. ACM Trans. Math. Softw. 3, 209–226 (1977)
55. Beis, J.S., Lowe, D.G.: Shape indexing using approximate nearest-neighbour search in high-dimensional spaces. In: Proc. IEEE Conf. Comp. Vision Patt. Recog, pp. 1000–1006 (1997)
56. Pilz, F., Shi, Y., Grest, D., Pugeault, N., Kalkan, S., Krüger, N.: Utilizing semantic interpretation of junctions for 3d-2d pose estimation. In: Bebis, G., Boyle, R., Parvin, B., Koracin, D., Paragios, N., Tanveer, S.-M., Ju, T., Liu, Z., Coquillart, S., Cruz-Neira, C., Müller, T., Malzbender, T. (eds.) ISVC 2007, Part I. LNCS, vol. 4841, pp. 271–280. Springer, Heidelberg (2007)
57. Hermann, S., Klette, R.: A study on parameterization and preprocessing for semi-global matching. Technical report, Computer Science Department, The University of Aukland, New Zealand (2008), http://citr.auckland.ac.nz/techreports/2008/CITR-TR-221.pdf

The Conformal Monogenic Signal of Image Sequences*

Lennart Wietzke[1], Gerald Sommer[1], Oliver Fleischmann[1], and Christian Schmaltz[2]

[1] Institute of Computer Science, Chair of Cognitive Systems, Christian-Albrechts-University, 24118 Kiel, Germany
[2] Mathematical Image Analysis Group, Faculty of Mathematics and Computer Science, Building E1 1, Saarland University, 66041 Saarbrücken, Germany

Abstract. Based on the research results of the Kiel University Cognitive Systems Group in the field of multidimensional signal processing and Computer Vision, this book chapter presents new ideas in 2D/3D and multidimensional signal theory. The novel approach, called the *conformal monogenic signal*, is a rotationally invariant quadrature filter for extracting i(ntrinsic)1D and i2D *local* features of any curved 2D signal - such as lines, edges, corners and circles - without the use of any heuristics or steering techniques. The *conformal monogenic signal* contains the *monogenic signal* as a special case for i1D signals - such as lines and edges - and combines monogenic scale space, local energy, direction/orientation, both i1D and i2D phase and curvature in one unified algebraic framework. The *conformal monogenic signal* will be theoretically illustrated and motivated in detail by the relation of the 3D Radon transform and the generalized Hilbert transform on the sphere. The main idea of the *conformal monogenic signal* is to lift up 2D signals by *stereographic projection* to a higher dimensional conformal space where the local signal features can be analyzed with more degrees of freedom compared to the flat two-dimensional space of the original signal domain. The philosophy of the *conformal monogenic signal* is based on the idea to make use of the direct relation of the original two-dimensional signal and abstract geometric entities such as lines, circles, planes and spheres. Furthermore, the *conformal monogenic signal* can not only be extended to 3D signals (image sequences) but also to signals of any dimension.

The main advantages of the *conformal monogenic signal* in practical applications are the completeness with respect to the intrinsic dimension of the signal, the rotational invariance, the low computational time complexity, the easy implementation into existing Computer Vision software packages and the numerical robustness of calculating exact local curvature of signals without the need of any derivatives.

* We acknowledge funding by the German Research Foundation (DFG) under the projects *SO 320/4-2* and *We 2602/5-1*.

D. Cremers et al. (Eds.): Visual Motion Analysis, LNCS 5604, pp. 305–322, 2009.

1 Introduction

Low level two-dimensional image analysis is often the first step of many Computer Vision tasks. Therefore, local signal features such as gray value or color information, gradient, curvature, orientation and phase determine the quality of subsequent higher level processing steps. It is important not to lose or to merge any of the original information within the local neighborhood of the test point[1]. The constraints of local signal analysis are: to span an orthogonal feature space (split of identity) and to be robust against stochastic and deterministic deviations between the actual signal and the model.

This book chapter is organized as follows: First the fundamental 1D and 2D local signal models with specific geometric and structural features are defined. The feature set of the assumed 2D signal model contains energy, phase, direction/orientation and curvature. Regarding those features, already known and closely related phase based and rotationally invariant quadrature filter signal processing approaches such as the monogenic signal, the structure multivector and the monogenic curvature tensor are being analyzed. Their possibilities and limitations are shown by means of the generalized Hilbert transform in Euclidean space and its relation to the Radon transform, which enables explicit extraction of the features by all different approaches in one framework. The monogenic signal, the structure multivector and the monogenic curvature tensor make use of generalized Hilbert transforms of different order in Euclidean space. Whereby the monogenic signal is limited to the first order generalized Hilbert transform and the structure multivector and the monogenic curvature tensor are extended to the second and third order generalized Hilbert transforms. The limitations of the nth-order generalized Hilbert transforms concerning 2D signal analysis are being shown and a novel 2D signal approach in conformal space called the *conformal monogenic signal* is presented.

Image signals $f \in L^2(\Omega;\mathbb{R})$ with $\Omega \subset \mathbb{R}^2$ will be locally analyzed on a low level. 2D signals are classified into local regions $N \subseteq \Omega$ of different intrinsic dimension (see figure 1)

$$\mathrm{i0D} = \left\{ f \in L^2(\Omega;\mathbb{R}) : f(\mathbf{x}_i) = f(\mathbf{x}_j) \ \forall \mathbf{x}_i, \mathbf{x}_j \in N \right\} \tag{1}$$

$$\mathrm{i1D} = \left\{ f \in L^2(\Omega;\mathbb{R}) : f(x,y) = g(x\cos\theta + y\sin\theta) \ \forall (x,y) \in N \right\} \setminus \mathrm{i0D} \tag{2}$$

$$\mathrm{i2D} = L^2(\Omega;\mathbb{R}) \setminus (\mathrm{i0D} \cup \mathrm{i1D}) \ . \tag{3}$$

The assumed local signal model is defined as a curve which can be locally approximated by a circle with arbitrary orientation and curvature

$$f(x,y) = a \ \cos\left(\left\| \begin{bmatrix} x \\ y \end{bmatrix} - \frac{1}{\kappa} \begin{bmatrix} \cos\theta \\ \sin\theta \end{bmatrix} \right\| + \phi \right) \in \mathrm{i1D} \cup \mathrm{i2D} \tag{4}$$

with $a \in \mathbb{R}$ as the local amplitude, $\phi \in [0, 2\pi)$ as the local phase, $\kappa \in \mathbb{R}$ as the local curvature and $\theta \in [0, 2\pi)$ as the local direction/orientation of the signal

[1] There is no method of signal analysis which is universal in respect of any arbitrary local 2D structure. Hence, it is necessary to formulate a model of local signal structure as basis of the analysis. The great challenge is the search for a most general model which can cope with as much as possible variants of local signal structure.

for $\kappa \neq 0$. For the special case of $\kappa = 0$ the curved 2D signal degrades to an i1D function. Therefore the task is to solve an *inverse problem*, i.e. to determine local features such as amplitude a, phase ϕ, orientation θ and curvature κ of any curved signal such as lines, edges, corners and junctions. One important local structural feature is the phase ϕ which can be calculated by means of the Hilbert transform [10]. Furthermore all signals will be analyzed in *monogenic scale space* [7] since the Hilbert transform can only be interpreted for narrow banded signals

$$f(x, y; s_s) = \mathcal{P}(x, y; s_s) * f(x, y) \tag{5}$$

with $*$ as the convolution operator and s_s as the scale space parameter. The Poisson kernel of the applied low pass filter reads

$$\mathcal{P}(\mathbf{x}) = \mathcal{P}(\mathbf{x}; s_s) = \frac{s_s}{2\pi \left(s_s^2 + \|\mathbf{x}\|^2\right)^{\frac{n+1}{2}}}, \quad n \in \mathbb{N},\ \mathbf{x} \in \mathbb{R}^n\ . \tag{6}$$

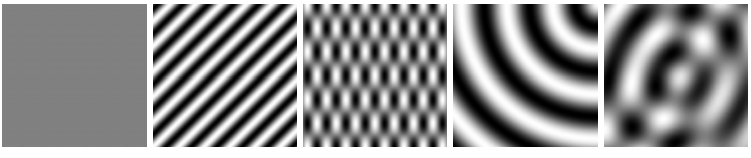


Fig. 1. From left to right: a constant signal (i0D), an arbitrary rotated 1D signal (i1D) and an i2D checkerboard signal consisting of two simple superimposed i1D signals. A curved i2D signal and two superimposed curved i2D signals. Note that all signals displayed here preserve their intrinsic dimension globally.

1.1 Related Work

Phase and energy of 1D signals can be analyzed by the *analytic signal* [10]. The generalization of the analytic signal to multidimensional signal domains has been done by the *monogenic signal* [6]. In case of 2D signals the *monogenic signal* delivers local phase, orientation and energy information restricted to the set of i1D signals. This book chapter presents the generalization of the *monogenic signal* for 2D signals to analyze both i1D and i2D signals in one unified framework. The *conformal monogenic signal* delivers local phase, orientation, energy and curvature for i1D and i2D signals with the *monogenic signal* as a special case. The *monogenic signal* replaces the classical 1D Hilbert transform of the analytic signal by the generalized Hilbert transform [3]

$$R\{f\}(\mathbf{x}) = (\mathcal{Q} * f)(\mathbf{x}) = (h_n * \mathcal{P} * f)(\mathbf{x}), \quad \mathbf{x} \in \mathbb{R}^n,\ n \in \mathbb{N} \setminus \{1\} \tag{7}$$

with $\mathcal{Q}$ as the conjugate Poisson kernel and h_n as the generalized Hilbert transform kernel

$$h_n(\mathbf{x}) = \frac{2}{A_{n+1}} \frac{\mathbf{x}}{\|\mathbf{x}\|^{n+1}}, \quad \mathbf{x} \in \mathbb{R}^n,\ n \in \mathbb{N} \setminus \{1\} \tag{8}$$

with A_{n+1} as the surface area of the unit sphere $\mathbb{S}^n$ in Euclidian space $\mathbb{R}^{n+1}$. To enable interpretation of the generalized Hilbert transform, its relation to the

Radon transform is the key [20]. The generalized Hilbert transform can be expressed by a concatenation of the Radon transform, the inverse Radon transform and the well known classical 1D Hilbert transform. Note that the relation to the Radon transform is required solely for interpretation and theoretical results. Neither the Radon transform nor its inverse are ever applied to the signal in practice. Instead the generalized Hilbert transformed signal will be determined by convolution in spatial domain and the signal features can be extracted in a rotationally invariant way.

2 Generalized Hilbert Transforms in Conformal Space

The feature space of the 2D *monogenic signal* is spanned by phase, orientation and energy information. This restriction correlates to the dimension of the associated 2D Radon space [20]. Therefore, our main idea is that the feature space of the 2D signal can only be extended by lifting up the original signal to higher dimensions. This is one of the main ideas of the *conformal monogenic signal*. In the following the 2D monogenic signal will be generalized to analyze also i2D signals by embedding the 2D signal into the 3D conformal space [15]. The 2D generalized Hilbert transform can be expressed by the 2D Radon transform

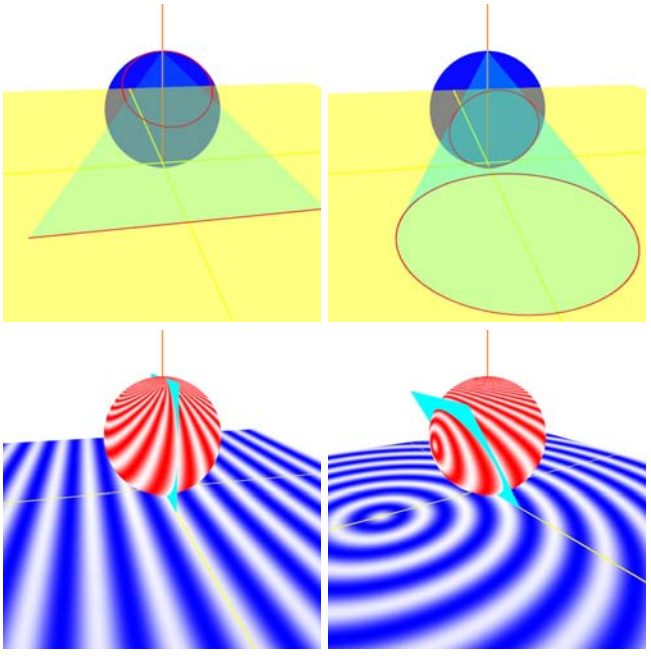


Fig. 2. Lines and circles of the 2D image domain are both mapped to circles on the sphere. Each circle on the sphere is uniquely defined by its parameterized intersection plane in conformal space. The third figure illustrates the monogenic signal as a special case.

which integrates all function values on lines [17]. This restriction to lines is one of the reasons why the 2D monogenic signal is limited to i1D signals (such as lines and edges) and can not be applied to corners and general curves. To analyze also i2D signals and to measure curvature $\kappa = \frac{1}{\rho}$, a 2D Radon transform which integrates on curved lines (i.e. local circles with radius ρ) is preferable. In the 3D domain the Radon transform integrates on planes, although at first sight 3D planes are not related to 2D signals. But the idea is that circles form the intersection of a sphere (with center at $\left[0, 0, \frac{1}{2}\right]$ and radius $\rho = \frac{1}{2}$) and planes passing through the origin $(0, 0, 0)$ of 3D space. Since the generalized Hilbert transform can be extended to any dimension [4] and the 3D generalized Hilbert transform can be expressed by the 3D Radon transform, the 2D signal coordinates must be mapped appropriately to the sphere. This mapping must be conformal (i.e. angle preserving), so that angular feature interpretation of the 3D generalized Hilbert transform in conformal space is still reasonable. Analogous to the line parametrization by $(t, \theta) \in \mathbb{R} \times [0, \pi)$ (where $t \in \mathbb{R}$ is the minimal distance of the 2D line to the origin and $\theta \in [0, \pi)$ is the orientation of the line) of the 2D Radon transform [20], the planes of the 3D Radon transform are uniquely defined by the parameters $(t, \theta, \varphi) \in \mathbb{R} \times [0, 2\pi) \times [0, \pi)$ where t is the minimal distance of the plane to the origin of the 3D space and the angles θ and φ determine the orientation of the plane in 3D space (see figure 3). This new parametrization truly extends the interpretation space of the monogenic signal by one dimension. In contrast to the well known Monge patch embedding known from differential geometry [5], the original 2D signal will now be embedded into the conformal space.

2.1 The Conformal Space

The main idea is that the concept of lines in 2D Radon space becomes the concept of planes in 3D Radon space and the more abstract concept of hyperplanes in multidimensional space. These planes determine circles on the sphere in conformal space. Since lines and circles of the 2D signal domain are mapped to circles [15] on the sphere (see figure 2), the integration on these circles determines points in the 3D Radon space. The projection $\mathcal{C}$ known from complex analysis [15] maps the original 2D signal domain to the sphere and can be inverted by $\mathcal{C}^{-1}$

$$\mathcal{C}(x, y) = \frac{1}{x^2 + y^2 + 1} \begin{bmatrix} x \\ y \\ x^2 + y^2 \end{bmatrix}, \quad \mathcal{C}^{-1}(\xi_1, \xi_2, \xi_3) = \frac{1}{1 - \xi_3} \begin{bmatrix} \xi_1 \\ \xi_2 \end{bmatrix}. \tag{9}$$

This mapping has the property that the 2D origin $(0, 0)$ of a local coordinate system will be mapped to the south pole $(0, 0, 0)$ of the sphere in conformal space and both $-\infty, +\infty$ will be mapped to the north pole $(0, 0, 1)$ of the sphere. Lines and circles of the 2D signal domain will be mapped to circles on the sphere and can be determined uniquely by planes in 3D conformal space. The integration on these planes corresponds to points (t, θ, φ) in the 3D Radon space.

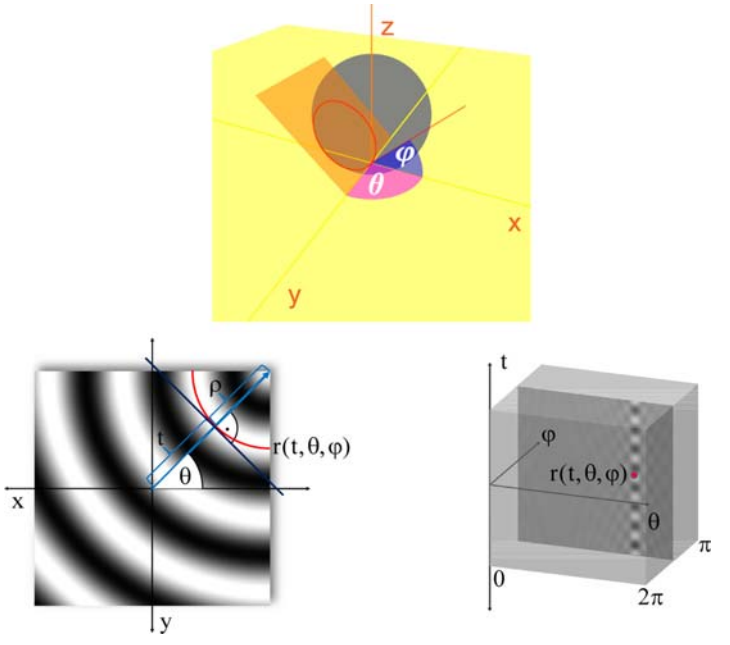


Fig. 3. Top row: Each with the triple (t, θ, φ) with $t = 0$ parameterized plane can be determined exactly by the generalized Hilbert transforms on the sphere. The interpretation of this parameter set delivers the features such as direction, phase and curvature of the original signal without any steering. Bottom row left figure: Curved i2D signal with orientation θ and curvature $\kappa = \frac{1}{\rho}$. Bottom row right figure: Corresponding 3D Radon space representation of the i2D signal spanned by the parameters t, θ and φ. Since the Radon transform on circles directly on the plane of the original 2D signal is not possible, the Radon transform has to be done in higher dimensional 3D conformal space where circles correspond to planes.

2.2 3D Radon Transform in Conformal Space

To interpret the *conformal monogenic signal*, the relation to the 3D Radon transform in conformal space must be taken into account. The 3D Radon transform is defined as the integral of all function values on the plane (see figure 2) defined by

$$\mathcal{R}\{c\}(t,\theta,\varphi) = \int_{\mathbf{x}\in\mathbb{R}^3} c(\mathbf{x})\delta_0(\mathbf{x}\begin{bmatrix}\sin\varphi\cos\theta\\ \sin\varphi\sin\theta\\ \cos\varphi\end{bmatrix} - t)d\mathbf{x} \tag{10}$$

with δ_0 as the Dirac distribution and the mapping c defined in equation (12). Since the signal is mapped on the sphere and all other points of the conformal space are set to zero, the 3D Radon transform actually sums up all points lying on the intersection of the plane and the sphere. For all planes this intersection

can either be empty or a circle. The concept of circles in the conformal 3D Radon transform can be compared with the concept of lines known from the 2D Radon transform. Since lines in the 2D signal domain are also mapped to circles, the *conformal monogenic signal* can analyze i1D as well as curved i2D signals in one single framework. Recall the very important fact that every corner or curve can be locally approximated by a circle. The inverse 3D Radon transform exists and differs from the 2D case such that it is a *local* transformation [2].

$$\mathcal{R}^{-1}\{r\}(0,0,0) = -\frac{1}{8\pi^2}\int_{\theta=0}^{2\pi}\int_{\varphi=0}^{\pi}\frac{\partial^2}{\partial t^2}\, r(t,\theta,\varphi)|_{t=0}\, d\varphi\, d\theta\,. \tag{11}$$

That means the generalized Hilbert transform at $(0,0,0)$ is completely determined by all planes passing the origin (i.e. $t=0$). In contrast, the 2D monogenic signal requires all integrals on all lines (t,θ) to reconstruct the original signal at a certain point and is therefore called a *global* transform. This interesting fact turns out from the definition of the inverse 3D Radon transform $\mathcal{R}^{-1}\{\cdot\}$. Therefore, the local features of i1D and i2D signals can be determined by the *conformal monogenic signal* at each test point of the original 2D signal without knowledge of the whole 3D Radon space.

2.3 The 2D Conformal Monogenic Signal

To give the generalized Hilbert transform more degrees of freedom for signal analysis, the original 2D signal will be embedded in a applicable subspace of the 3D conformal space by the mapping

$$c(x,y,z) = \begin{cases} f(\mathcal{C}^{-1}(x,y,z)^T; s_s)\,, x^2+y^2+\left(z-\frac{1}{2}\right)^2 = \frac{1}{4} \\ 0 \qquad\qquad\qquad\qquad\quad , \text{else} \end{cases}\,. \tag{12}$$

Thus, the 3D generalized Hilbert transform can be applied to all points on the sphere. The center of convolution in spatial domain is the south pole $(0,0,0)$ where the test point of the 2D signal domain meets the sphere. At this point the 3D generalized Hilbert transform will be performed at the origin $(\mathbf{0})$ of the applied local coordinate system for each test point separately. The *conformal monogenic signal* [19,18] is defined as

$$f_{\mathrm{CMS}}(\mathbf{0}) = \left[c(\mathbf{0}), R_x\{c\}(\mathbf{0}), R_y\{c\}(\mathbf{0}), R_z\{c\}(\mathbf{0})\right]^T \tag{13}$$

and can be expressed by the classical 1D Hilbert transform kernel $h_1(\tau) = \frac{1}{\pi\tau}$ [10], the 3D Radon transform and its inverse analogous to the monogenic signal in 2D [20]

$$\begin{bmatrix} R_x\{c\}(\mathbf{0}) \\ R_y\{c\}(\mathbf{0}) \\ R_z\{c\}(\mathbf{0}) \end{bmatrix} = \left[\mathcal{R}^{-1}\left\{\begin{bmatrix}\sin\varphi\cos\theta \\ \sin\varphi\sin\theta \\ \cos\varphi\end{bmatrix} h_1(t) * \mathcal{R}\{c\}(t,\theta,\varphi)\right\}(0,0,0)\right] \tag{14}$$

with $*$ as the 1D convolution operator. Compared to the 2D monogenic signal the *conformal monogenic signal* performs a 3D generalized Hilbert transformation in conformal space.

2.4 Interpretation

Analogous to the interpretation of the *monogenic signal* in [20], the parameters of the plane within the 3D Radon space determine the local features of the curved i2D signal (see figure 2). The *conformal monogenic signal* can be called the generalized monogenic signal for i1D and i2D signals, because the special case of lines and edges can be considered as circles with zero curvature. These lines are mapped to circles passing through the north pole in conformal space. The parameter θ will be interpreted as the orientation in i1D case and naturally deploys to direction $\theta \in [0, 2\pi)$ for the i2D case

$$\theta = \operatorname{atan2}\left(R_y\left\{c\right\}(\mathbf{0}), R_x\left\{c\right\}(\mathbf{0})\right) \ . \tag{15}$$

The energy of the signal is defined by

$$E = a^2 = c^2(\mathbf{0}) + R_x^2\left\{c\right\}(\mathbf{0}) + R_y^2\left\{c\right\}(\mathbf{0}) + R_z^2\left\{c\right\}(\mathbf{0}) \ . \tag{16}$$

The i1D and i2D curvature phase is defined by

$$\phi = \operatorname{atan2}\left(\sqrt{R_x^2\left\{c\right\}(\mathbf{0}) + R_y^2\left\{c\right\}(\mathbf{0}) + R_z^2\left\{c\right\}(\mathbf{0})}, c(\mathbf{0})\right) \ . \tag{17}$$

Note that all proofs are analogous to those for the 2D monogenic signal shown in [20].

2.5 Local Curvature

The parameter φ of the 3D Radon space corresponds to the isophote curvature κ [14] known from differential geometry

$$\varphi = \arctan \frac{\sqrt{R_x^2\left\{c\right\}(\mathbf{0}) + R_y^2\left\{c\right\}(\mathbf{0})}}{R_z\left\{c\right\}(\mathbf{0})} \tag{18}$$

$$\kappa = \frac{-f_{xx}f_y^2 + 2f_xf_yf_{xy} - f_{yy}f_x^2}{\left(f_x^2 + f_y^2\right)^{\frac{3}{2}}} \tag{19}$$

Proof:
Let be

$$\gamma(t) = \left[\rho(\cos\theta + \cos t), \rho(\sin\theta + \sin t)\right]^T \tag{20}$$

with $t \in [0, 2\pi)$ a parametrization of a circle in the 2D plane touching the origin $(0, 0)$ with radius ρ and tangential orientation θ. This circle will be the model for the osculating circle touching the isophote curve of the 2D signal f at the origin $(0, 0)$ of the local coordinate system for each test point. Therefore,

$$f(\gamma(t_1)) = f(\gamma(t_2)) \ \forall t_1, t_2 \in [0, 2\pi) \ . \tag{21}$$

Define $\gamma_S(t) = \mathcal{C}(\gamma(t))$ as the projection of γ to the sphere

$$\mathbb{S}^2(m_S, \rho_S) = \left\{v \in \mathbb{R}^3 : \|v - m_S\| = \rho_S\right\} \tag{22}$$

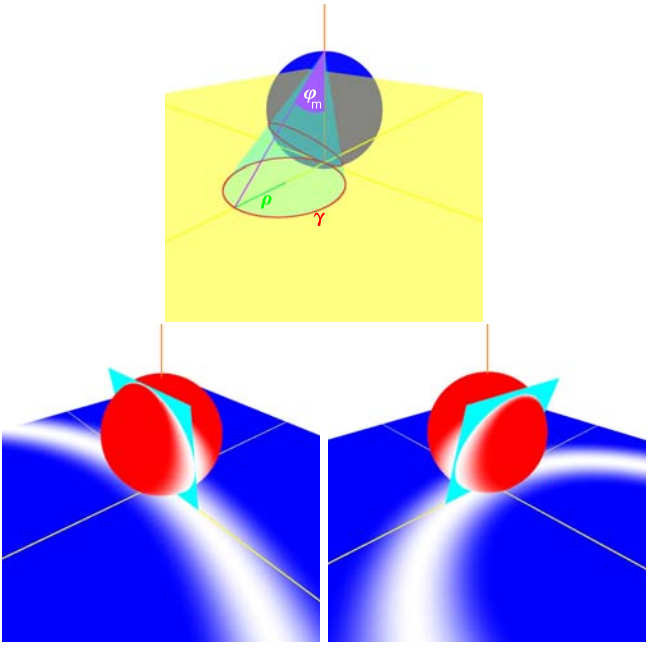


Fig. 4. Top row: Visualization of the circle described by γ projected to $\mathbb{S}^2(m_S, \rho_S)$. This figure illustrates geometrically the relation of the curvature κ to the orientation parameter φ of the hyperplane in 3D Radon space $\frac{1}{\kappa} = \rho = \tan\varphi_m$.

with the center $m_S = \left[0, 0, \frac{1}{2}\right]^T$ and the radius $\rho_S = \frac{1}{2}$. Furthermore define

$$f_S(\gamma_S(t)) = f(\mathcal{C}^{-1}(\gamma_S(t))) \; . \tag{23}$$

The conjugate Poisson kernel in $\mathbb{R}^{n+1}_+$ reads

$$\mathcal{Q}(x) = [\mathcal{Q}_x(x), \mathcal{Q}_y(x), \mathcal{Q}_z(x)]^T = (h_3 * \mathcal{P})(x) \tag{24}$$

with

$$R\{f_S\}(\mathbf{x}) = (\mathcal{Q} * f_S)(\mathbf{x}) \; . \tag{25}$$

The radius ρ of the osculating circle described by the parameterized curve γ reads

$$\rho = \frac{2R_z\{f_S\}(\mathbf{0})}{\sqrt{R_x^2\{f_S\}(\mathbf{0}) + R_y^2\{f_S\}(\mathbf{0})}} \; . \tag{26}$$

Since the values of $f_S(x)$ will only be nonzero for $x \in \mathbb{S}^2(m_S, \rho_S)$, the integration can be restricted to the ball

$$\mathbb{B}^2(m_S, \rho_S) = \left\{v \in \mathbb{R}^3 : \|v - m_S\| \leq \rho_S\right\} \; . \tag{27}$$

Furthermore $f_S(x)$ is only nonzero for the circle projected on the sphere

$$M = \{\gamma_S(t) : t \in [0, 2\pi)\} \ . \tag{28}$$

Now let $\mathbb{S}^2(m, \rho)$ be the sphere whose intersection with $\mathbb{S}^2(m_S, \rho_S)$ results in M. Then the set M is a circle on the surface of both $\mathbb{S}^2(m_S, \rho_S)$ and $\mathbb{S}^2(m, \rho)$. The integration over the volumes of $\mathbb{B}^2(m, \rho)$ and $\mathbb{B}^2(m_S, \rho_S)$ will be the same.

$$\int\limits_{x \in \mathbb{R}^3_+} \mathcal{Q}(x) f_S(x) \, dx = \int\limits_{x \in \mathbb{B}^2(m_S, \rho_S)} \mathcal{Q}(x) f_S(x) \, dx = \int\limits_{x \in \mathbb{B}^2(m, \rho)} \mathcal{Q}(x) f_S(x) \, dx \tag{29}$$

According to the results from harmonic analysis [1] the convolution of a function in $\mathbb{R}^n$ with the Poisson kernel $\mathcal{P}$ in upper the half space $\mathbb{R}^{n+1}_+$ results in a harmonic function in $\mathbb{R}^{n+1}_+$. Therefore, $\mathcal{Q}$ is harmonic in $\mathbb{R}^{3+1}_+$. Using the *mean value theorem* for harmonic functions it follows that

$$\int\limits_{x \in \mathbb{B}^2(m, \rho)} \mathcal{Q}(x) \, dx = k \ \mathcal{Q}(m) \tag{30}$$

with the components of $\mathcal{Q}$ written in spherical coordinates

$$\mathcal{Q}(m) = \begin{bmatrix} \mathcal{Q}_x(m) \\ \mathcal{Q}_y(m) \\ \mathcal{Q}_z(m) \end{bmatrix} = \frac{1}{\left[\|m\|^2 + s_s^2\right]^2} \begin{bmatrix} \sin\varphi_m \cos\theta_m \\ \sin\varphi_m \sin\theta_m \\ \cos\varphi_m \end{bmatrix} \tag{31}$$

with s_s as the scale space parameter. Since f_S is the signal model for the isophote curve of a signal in the plane, it is a curve consisting of constant values. Therefore, $f_S(x)$ will be constant for all $x \in M$ which results in

$$\int\limits_{x \in \mathbb{B}^2(m, \rho)} \mathcal{Q}(x) f_S(x) \, dx = f_c \int\limits_{x \in \mathbb{B}^2(m, \rho)} \mathcal{Q}(x) \, dx = f_c \ k \ \mathcal{Q}(m) \ . \tag{32}$$

With equation (26) it is now possible to determine $\frac{\sin\varphi_m}{\cos\varphi_m}$. Figure 4 illustrates that this is exactly $\frac{\rho}{2\,\rho_S}$. Since $\rho_S = \frac{1}{2}$ it follows that the radius of the local curvature can be determined by

$$\rho = \frac{\sin\varphi_m}{2\cos\varphi_m} = \frac{\sqrt{\mathcal{Q}_x^2(m) + \mathcal{Q}_y^2(m)}}{2\mathcal{Q}_z(m)} \ . \tag{33}$$

2.6 Phase Congruency of the Conformal Monogenic Signal

Since the local phase of the *conformal monogenic signal* is independent of the local signal amplitude a_{s_s}, it thus has the advantage of being not sensitive to local illumination changes. Hence, detecting i1D and i2D key points can be done by searching for points of *stationary phase* [12,11,16,21] in monogenic scale-space.

This approach is called *phase congruency* ϕ_{PC} and is based on comparisons of the local phase at certain distinct scales $s_s > 0$

$$\phi_{\mathrm{PC}} = \frac{\sum\limits_{s_s \in I} W \left\lfloor a_{s_s}(\cos(\phi_{s_s} - \overline{\phi}) - \| \sin(\phi_{s_s} - \overline{\phi})\| - T) \right\rfloor}{\varepsilon + \sum\limits_{s_s \in I} a_{s_s}} \tag{34}$$

with s_s as the scale-space parameter within the interval $I \subset [s_f, \cdots, s_c]$ with $\|I\| \in \mathbb{N}$, $W \in \mathbb{R}$ as a weighting factor for frequency spread. $a_{s_s}^2 = e_{s_s} = e = a^2$ is the local signal amplitude of the *conformal monogenic signal* at the scale-space parameter s_s and ϕ_{s_s} is the local phase of the *conformal monogenic signal* at the scale-space parameter s_s. $\overline{\phi} = \frac{1}{\|I\|} \sum_{s_s \in I} \phi_{s_s}$ is the mean local phase for all scale-space parameters $s_s \in I$ and ε is a constant because of numerical reasons to avoid division by zero,

$$\lfloor x \rfloor = \begin{cases} x, \, x > 0 \\ 0, \, \text{else} \end{cases} \tag{35}$$

The estimated noise influence value T is a constant to ensure that only energy values that exceed T are taken into account. The phase congruency measure can be directly applied to detect rotationally invariant i1D and i2D image structures. Any test point with a phase congruency greater than a certain threshold can be marked as a i1D or i2D point respectively.

The so far discussed phase congruency approach is based on comparisons of local phase at certain distinct scales. Nevertheless, there exist some drawbacks. Since local features are relatively to the scale, an algorithm using distinct scales has to contain heuristics to judge whether the structure is present or not if the phase is only congruent in some of the considered scales. Besides, it is not straightforward, how to map at different scales estimated phases to a certainty measure. Hence, the *differential phase congruency* [7] detects i1D and i2D structures in a more simple and efficient way. The points in monogenic scale-space, where the differentials of their phase vector $\Phi(s_s)$ are zero, are called points of differential phase congruency. They have to be identified as i1D or i2D structures. The scale derivative of the phase vector reads

$$\frac{\partial}{\partial s_s}\Phi(s_s) = \frac{c(\mathbf{0}; s_s) \begin{bmatrix} \frac{\partial}{\partial s_s} R_x\{c\}(\mathbf{0}; s_s) \\ \frac{\partial}{\partial s_s} R_y\{c\}(\mathbf{0}; s_s) \\ \frac{\partial}{\partial s_s} R_z\{c\}(\mathbf{0}; s_s) \end{bmatrix} - \frac{\partial}{\partial s_s} c(\mathbf{0}; s_s) \begin{bmatrix} R_x\{c\}(\mathbf{0}; s_s) \\ R_y\{c\}(\mathbf{0}; s_s) \\ R_z\{c\}(\mathbf{0}; s_s) \end{bmatrix}}{c(\mathbf{0}, s_s)^2 + \|R\{c\}(\mathbf{0}; s_s)\|^2} \tag{36}$$

Test points where $\frac{\partial}{\partial s_s}\Phi(s_s) = 0$ are of differential phase congruency and hence considered as i1D or i2D structures. To find these points, the zeros of the three components of the numerator in equation (36) have to be found. These zeros can be easily obtained with subpixel accuracy by a *linear regression*. The differential phase congruency is quite useful since it yields a higher accuracy and a significant speedup of the derivative computation compared to a finite difference approximation.

2.7 Experimental Results

On synthetic signals with known ground truth the average error of the feature extraction converges to zero with increasing refinement of the convolution mask size. Under the presence of noise the *conformal monogenic signal* curvature performs more robust than e.g. the gradient based Sobel detector (see figure 5). The curvature feature delivered by the novel *conformal monogenic signal* performs better in dense optical flow applications with an average angular error (AAE) of 1.99° compared to [22] (with $AAE = 2.67°$) on the cloudy Yosemite sequence (see figure 5). Since the *conformal monogenic signal* combines all intrinsic dimensions in one framework it could be an interesting alternative for the gradient or the Laplace operator.

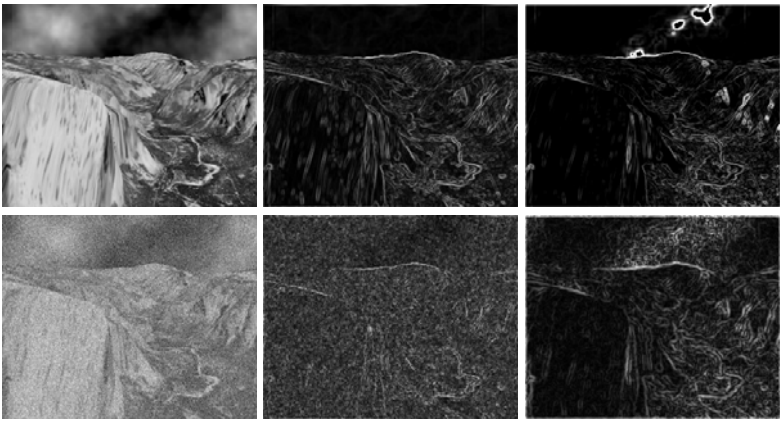


Fig. 5. Top row from left to right: Original Yosemite image, Sobel detector output and *conformal monogenic signal* curvature which delivers much more structural information (see cloudy sky). Bottom row from left to right: Noise degraded image (SNR=10dB), blurred Sobel output and *conformal monogenic signal* curvature. Convolution mask size: 7×7 pixels.

3 The 3D Conformal Monogenic Signal

In case of *visual motion analysis* a three dimensional isotropic quadrature filter is needed [13,8]. The *conformal monogenic signal* of a 3D signal $f \in L^2(\Omega; \mathbb{R})$ with $\Omega \subset \mathbb{R}^3$ delivers energy, 3D orientation, phase and curvature. For image sequences (3D signals) the concept of planes in 3D Radon space becomes the more abstract concept of hyperplanes in 4D Radon space. These 4D hyperplanes determine 3D spheres on the 4D hypersphere in 4D conformal space. Since 3D planes and 3D spheres of the three-dimensional signal domain are mapped to 3D spheres on the 4D hypersphere, the integration on these 3D spheres determines

points in the 4D Radon space. The general stereographic projection for any dimension $n \in \mathbb{N}$ which maps the Euclidian space $\mathbb{R}^n$ to the conformal space $\mathbb{R}^{n+1}$ reads

$$\mathcal{C}(x_1, x_2, \cdots, x_n) = \frac{1}{1 + \sum_{i=1}^{n} x_i^2} \begin{bmatrix} x_1 \\ x_2 \\ \cdots \\ x_n \\ \sum_{i=1}^{n} x_i^2 \end{bmatrix} \in \mathbb{R}^{n+1} . \tag{37}$$

The stereographic projection maps the Euclidian space $\mathbb{R}^n$ to the hypersphere in $\mathbb{R}^{n+1}$ with radius $\frac{1}{2}$ and the south pole of the hypersphere touching the origin $(\underbrace{0, \cdots, 0}_{n}) \in \mathbb{R}^n$ of the Euclidian space $\mathbb{R}^n$ and the north pole of the hypersphere with coordinates $(\underbrace{0, 0, \cdots, 0}_{n}, 1) \in \mathbb{R}^{n+1}$. For the signal dimension $n = 3$ the stereographic projection $\mathcal{C}$ known from complex analysis [9] maps the 3D signal domain to the hypersphere

$$\mathcal{C}(x, y, z) = \frac{1}{1 + x^2 + y^2 + z^2} \begin{bmatrix} x \\ y \\ z \\ x^2 + y^2 + z^2 \end{bmatrix} . \tag{38}$$

This projection is conformal and can be inverted by the general formula

$$\mathcal{C}^{-1}(\xi_1, \xi_2, \cdots, \xi_n, \xi_{n+1}) = \frac{1}{1 - \xi_{n+1}} \begin{bmatrix} \xi_1 \\ \xi_2 \\ \cdots \\ \xi_n \end{bmatrix} . \tag{39}$$

The inversion $\mathcal{C}^{-1}$ for all elements of the hypersphere reads

$$\mathcal{C}^{-1}(\xi_1, \xi_2, \xi_3, \xi_4) = \frac{1}{1 - \xi_4} \begin{bmatrix} \xi_1 \\ \xi_2 \\ \xi_3 \end{bmatrix} \tag{40}$$

with $\xi = (\xi_1, \xi_2, \xi_3, \xi_4)$. This mapping has the property that the origin $(0, 0, 0)$ of the 3D signal domain will be mapped to the south pole $\mathbf{0} = (0, 0, 0, 0)$ of the hypersphere and both $-\infty, +\infty$ will be mapped to the north pole $(0, 0, 0, 1)$ of the hypersphere. 3D planes and spheres of the 3D signal domain will be mapped to spheres on the hypersphere and can be determined uniquely by hyperplanes in 4D Radon space. The integration on these hyperplanes corresponds to points $(t, \theta_1, \theta_2, \varphi)$ in the 4D Radon space.

Since the signal domain $\Omega \subset \mathbb{R}^3$ is bounded, not the whole hypersphere is covered by the original signal. Anyway, all hyperplanes corresponding to spheres on the hypersphere remain unchanged. That is the reason why the *conformal*

monogenic signal models 3D planes and all kinds of curved 3D planes which can be locally approximated by spheres. To give the Riesz transform more degrees of freedom, the original three-dimensional signal will be embedded in an applicable subspace of the conformal space by the so called *conformal signal* $c \in \mathbb{R}^{(\mathbb{R}^4)}$ of the original 3D signal f:

$$c(\xi) = \begin{cases} f(\mathcal{C}^{-1}(\xi_1, \xi_2, \xi_3, \xi_4)^T) & , \xi_1^2 + \xi_2^2 + \xi_3^2 + \left(\xi_4 - \frac{1}{2}\right)^2 = \frac{1}{4} \\ 0 & , \text{else} \end{cases} \tag{41}$$

Thus, the 4D Riesz transform $R\{\cdot\}$ can be applied to all points on the hypersphere. The center of convolution in spatial domain is the south pole $(0, 0, 0, 0)$ where the origin of the 3D signal domain meets the hypersphere. At this point the 4D Riesz transform $R\{\cdot\}(\mathbf{0})$ will be evaluated in spatial domain by convolution

$$R\{c\}(\xi)|_{\xi=(0,0,0,0)} = \frac{2}{A_5} \text{P.V.} \int\limits_{x \in \mathbb{R}^4} \frac{x}{\|x\|^5} \, c(x - \xi) \, dx \tag{42}$$

with P.V. as the Cauchy principal value and A_5 as the surface area of the unit sphere $\mathbb{S}^4$. The *conformal monogenic signal* f_{CM} for 3D signals is defined by the even part and the four odd parts of the 4D Riesz transform in conformal space

$$f_{\text{CM}}(0, 0, 0) = \begin{bmatrix} c_e(\mathbf{0}) \\ c_{o_1}(\mathbf{0}) \\ c_{o_2}(\mathbf{0}) \\ c_{o_3}(\mathbf{0}) \\ c_{o_4}(\mathbf{0}) \end{bmatrix} = \begin{bmatrix} c(\mathbf{0}) \\ R_1\{c\}(\mathbf{0}) \\ R_2\{c\}(\mathbf{0}) \\ R_3\{c\}(\mathbf{0}) \\ R_4\{c\}(\mathbf{0}) \end{bmatrix} = \begin{bmatrix} c(\mathbf{0}) \\ R\{c\}(\mathbf{0}) \end{bmatrix} . \tag{43}$$

Note that the coordinates $(0, 0, 0)$ are relative to the local coordinate system for each test point of the original 3D signal and $\mathbf{0} = (0, 0, 0, 0)$ are the corresponding relative coordinates in conformal space, i.e. this is no restriction.

The Riesz transform of the 3D signal embedded in the conformal space can also be written in terms of the 4D Radon transform and its inverse

$$R\{c\}(\mathbf{0}) = \mathcal{R}^{-1} \left\{ \begin{bmatrix} \cos\varphi \sin\theta_1 \sin\theta_2 \\ \sin\varphi \sin\theta_1 \sin\theta_2 \\ \cos\theta_1 \sin\theta_2 \\ \cos\theta_2 \end{bmatrix} h_1(t) * \mathcal{R}\{c\}(t, \theta_1, \theta_2, \varphi) \right\} (\mathbf{0}) . \tag{44}$$

This representation of the Riesz transform is essential for the subsequent interpretation of the *conformal monogenic signal*. Remember that without loss of generality the signal will be analyzed at the origin $\mathbf{0} = (0, 0, 0)$ of the local coordinate system of the test point of local interest. Compared to the 2D monogenic signal the *conformal monogenic signal* of 3D signals is based on a 4D Riesz transformation in conformal space.

Analogous to the interpretation of the monogenic signal in [20], the parameters of the hyperplane within the 4D Radon space determine the local features of the curved 3D signal. The norm of the 4D Riesz transform of the *conformal signal* is defined by

$$\|R\{c\}(\mathbf{0})\| = \sqrt{R_1^2\{c\}(\mathbf{0}) + R_2^2\{c\}(\mathbf{0}) + R_3^2\{c\}(\mathbf{0}) + R_4^2\{c\}(\mathbf{0})}\,. \tag{45}$$

The *conformal monogenic signal* can be called the generalized monogenic signal for 3D signals, because the special case of planes in the original 3D signal can be considered as spheres with zero curvature. These planes are mapped to spheres passing through the north pole in conformal space. The 3D curvature corresponds to the parameter φ of the 4D Radon space,

$$\varphi = \arctan \frac{R_2^2\{c\}(\mathbf{0})}{R_1^2\{c\}(\mathbf{0})}\,. \tag{46}$$

Besides, the curvature of the *conformal monogenic signal* naturally indicates the intrinsic dimension of the signal. The parameters (θ_1, θ_2) will be interpreted as the orientation of the signal in the original 3D space

$$\theta_1 = \arcsin \frac{\sqrt{R_1^2\{c\}(\mathbf{0}) + R_2^2\{c\}(\mathbf{0})}}{R_4\{c\}(\mathbf{0})} \tag{47}$$

and

$$\theta_2 = \operatorname{atan2}\left(\sqrt{R_1^2\{c\}(\mathbf{0}) + R_2^2\{c\}(\mathbf{0}) + R_3^2\{c\}(\mathbf{0})}, R_4\{c\}(\mathbf{0})\right)\,. \tag{48}$$

The energy of the signal is defined by

$$e = c^2(\mathbf{0}) + \|R\{c\}(\mathbf{0})\|^2\,. \tag{49}$$

The phase for curved 3D signals is defined by

$$\phi = \operatorname{atan2}\left(\|R\{c\}(\mathbf{0})\|, c(\mathbf{0})\right)\,. \tag{50}$$

In all different intrinsic dimensions the phase indicates a measure of parity symmetry. Note that all proofs are analogous to those for the 2D monogenic signal shown in [20].

3.1 Implementation

The implementation of the *conformal monogenic signal* of 3D signals such as images sequences is analogous to the 2D signal case. The computational time complexity is in $O(n^3)$ with n as the convolution mask size in one dimension.

```
//Input:  double Image3D(double x,double y,double z)
//Input:  double x,y,z (Local pixel test point for analysis)
//Input:  double Coarse > Fine > 0 (Bandpass filter parameters)
//Input:  double Size > 0 (Convolution mask size)
//Output: Direction1, Direction2, Phase, Curvature, Energy

double Coarse=2,Fine=0.1; int Size=5;//e.g.
double rp=0,r1=0,r2=0,r3=0,r4=0;
for(double cx = -Size;cx <= Size;cx += 1)
for(double cy = -Size;cy <= Size;cy += 1)
for(double cz = -Size;cz <= Size;cz += 1)
{
    //Map points (cx,cy,cz) to conformal space (x1,x2,x3,x4)
    double d = pow(cx,2)+pow(cy,2)+pow(cz,2)+1;
    double x1 = cx    / d;
    double x2 = cy    / d;
    double x3 = cz    / d;
    double x4 = (d-1) / d;
    //Generalized Hilbert transform in conformal space
    double a  = pow(x1,2)+pow(x2,2)+pow(x3,2)+pow(x4,2);
    double pf = pow(pow(Fine  ,2) + a,-2.5);
    double pc = pow(pow(Coarse,2) + a,-2.5);
    double f  = Image3D(x + cx,y + cy,z + cz);
    double c  = f * (pf - pc);
    rp += f * (Fine*pf - Coarse*pc);
    r1 += x1 * c; r2 += x2 * c; r3 += x3 * c; r4 += x4 * c;
}
Curvature = atan(r2/r1);
Direction1 = asin(sqrt(pow(r1,2)+pow(r2,2))/r4);
Direction2 = atan2(sqrt(pow(r1,2)+pow(r2,2)+pow(r3,2)),r4);
Phase = atan2(sqrt(pow(r1,2)+pow(r2,2)+pow(r3,2)+pow(r4,2)),rp);
//For energy a DC free convolution kernel must be used instead
Energy = pow(rp,2)+pow(r1,2)+pow(r2,2)+pow(r3,2)+pow(r4,2);
```

4 Conclusion

In this book chapter a new fundamental idea for locally analyzing 2D curved signals such as lines, edges, corners, arcs and circles in one unified framework has been presented. It has been shown that the feature space of the nth-order 2D Riesz transform is much too flat for analyzing 2D signals and extracting real i2D phase information. Generalized Hilbert transforms in Euclidean space lack from the restriction to the classical 1D phase information for all 2D signals. In such a case arbitrary 2D signals can be modeled by a superposition of individual i1D signals. The resulting system of equations to separate these i1D signals can not be solved in the most general case. The two-dimensional Riesz transforms of any order are always limited to the related 2D Radon space which gives direct access to the feature space. To extend the dimension of the related feature space

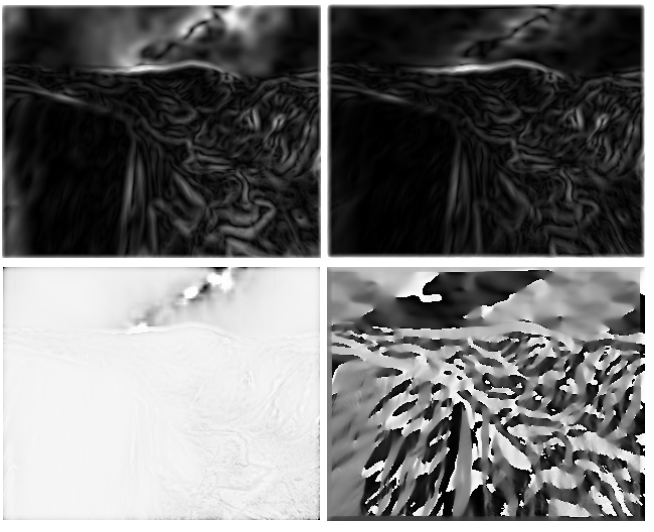


Fig. 6. The 3D *conformal monogenic signal* delivers four local features which can be used for image sequence analysis such as optical flow and motion analysis. First row shows from left to right: Curvature and phase information. Second row: Two parts of the orientation information. 3D convolution mask size $5 \times 5 \times 5$ pixels.

to analyze i1D and i2D signals in one framework, this problem can be solved by embedding 2D signals in higher dimensional conformal spaces in which the original 2D signal can be analyzed by generalized Hilbert transforms with more degrees of freedom. Without steering and in a rotationally invariant way, local signal features such as energy, phase, orientation/direction and curvature can be determined in spatial domain by 2D convolution. The *conformal monogenic signal* can be computed efficiently and can be easily implemented into existing low level image processing steps of Computer Vision applications. Furthermore, exact curvature can be calculated with all the advantages of rotationally invariant local phase based approaches (robustness against brightness and contrast changes) and without the need of any partial derivatives. Hence, lots of numerical problems of partial derivatives on discrete grids can be avoided. All results can be proved mathematically as well as by experiments. More applications of the *conformal monogenic signal* such as object tracking [14] on three-dimensional data will be part of our future work. The *conformal monogenic signal* shows the direct relation of the original image domain and geometric entities such as lines, circles, planes and spheres. For further results the reader is advised to have a look on our website `http://www.ks.informatik.uni-kiel.de/`

References

1. Axler, S., Bourdon, P., Ramey, W.: Harmonic Function Theory (Graduate Texts in Mathematics), vol. 137. Springer, Heidelberg (2002)

2. Bernstein, S.: Inverse Probleme. Technical report, TU Bergakademie Freiberg (2007)
3. Brackx, F., De Knock, B., De Schepper, H.: Generalized multidimensional Hilbert transforms in Clifford analysis. International Journal of Mathematics and Mathematical Sciences (2006)
4. Delanghe, R.: Clifford analysis: History and perspective. Computational Methods and Function Theory 1(1), 107–153 (2001)
5. do Carmo, M.P.: Differential Geometry of Curves and Surfaces. Prentice-Hall, Englewood Cliffs (1976)
6. Felsberg, M.: Low-level image processing with the structure multivector. Technical Report 2016, Kiel University, Department of Computer Science (2002)
7. Felsberg, M., Sommer, G.: The monogenic scale-space: A unifying approach to phase-based image processing in scale-space. Journal of Mathematical Imaging and Vision 21, 5–26 (2004)
8. Grau, V., Becher, H., Alison Noble, J.: Phase-based registration of multi-view real-time three-dimensional echocardiographic sequences. In: Larsen, R., Nielsen, M., Sporring, J. (eds.) MICCAI 2006. LNCS, vol. 4190, pp. 612–619. Springer, Heidelberg (2006)
9. Gürlebeck, K., Habetha, K., Sprössig, W.: Funktionentheorie in der Ebene und im Raum (Grundstudium Mathematik). Birkhäuser, Basel (2006)
10. Hahn, S.L.: Hilbert Transforms in Signal Processing. Artech House Inc., Boston (1996)
11. Kovesi, P.: Phase congruency detects corners and edges. In: The Australian Pattern Recognition Society Conference, pp. 309–318 (2003)
12. Kovesi, P., Videre, A.: Image features from phase congruency. Journal of Computer Vision Research 1(3) (1999)
13. Krause, M., Sommer, G.: A 3D isotropic quadrature filter for motion estimation problems. In: Proc. Visual Communications and Image Processing, Beijing, China, vol. 5960, pp. 1295–1306. The International Society for Optical Engineering, Bellingham (2005)
14. Lichtenauer, J., Hendriks, E.A., Reinders, M.J.T.: Isophote properties as features for object detection. In: CVPR, vol. (2), pp. 649–654 (2005)
15. Needham, T.: Visual Complex Analysis. Oxford University Press, Oxford (1997)
16. Reisfeld, D.: The constrained phase congruency feature detector: Simultaneous localization, classification and scale determination 17(11), 1161–1169 (1996)
17. Toft, P.: The Radon Transform - Theory and Implementation. PhD thesis, Technical University of Denmark (1996)
18. Wietzke, L., Fleischmann, O., Sommer, G.: 2D image analysis by generalized Hilbert transforms in conformal space. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part II. LNCS, vol. 5303, pp. 638–649. Springer, Heidelberg (2008)
19. Wietzke, L., Sommer, G.: The conformal monogenic signal. In: Rigoll, G. (ed.) DAGM 2008. LNCS, vol. 5096, pp. 527–536. Springer, Heidelberg (2008)
20. Wietzke, L., Sommer, G., Schmaltz, C., Weickert, J.: Differential geometry of monogenic signal representations. In: Sommer, G., Klette, R. (eds.) RobVis 2008. LNCS, vol. 4931, pp. 454–465. Springer, Heidelberg (2008)
21. Zang, D., Sommer, G.: Detecting intrinsically two-dimensional image structures using local phase. In: Franke, K., Müller, K.-R., Nickolay, B., Schäfer, R. (eds.) DAGM 2006. LNCS, vol. 4174, pp. 222–231. Springer, Heidelberg (2006)
22. Zang, D., Wietzke, L., Schmaltz, C., Sommer, G.: Dense optical flow estimation from the monogenic curvature tensor. In: Sgallari, F., Murli, A., Paragios, N. (eds.) SSVM 2007. LNCS, vol. 4485, pp. 239–250. Springer, Heidelberg (2007)

Author Index