Xiaoyi Jiang Olga Regina Pereira Bellon
Dmitry Goldgof Takeshi Oishi (Eds.)

LNCS 7854

Advances in Depth Image Analysis and Applications

International Workshop, WDIA 2012
Tsukuba, Japan, November 2012
Revised Selected and Invited Papers

Springer

Lecture Notes in Computer Science 7854

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

Xiaoyi Jiang Olga Regina Pereira Bellon
Dmitry Goldgof Takeshi Oishi (Eds.)

Advances in Depth Image Analysis and Applications

International Workshop, WDIA 2012
Tsukuba, Japan, November 11, 2012
Selected and Invited Papers

Volume Editors

Xiaoyi Jiang
University of Münster, Department of Mathematics and Computer Science
48149 Münster, Germany
E-mail: xjiang@uni-muenster.de

Olga Regina Pereira Bellon
Universidade Federal do Paraná, Centre Politecnico-Jardim das Americas
81531-980 Curitiba, PR, Brazil
E-mail: olgarpbellon@gmail.com

Dmitry Goldgof
University of South Florida, Department of Computer Science and Engineering
Tampa, FL 33620, USA
E-mail: goldgof@cse.usf.edu

Takeshi Oishi
The University of Tokyo, Institute of Industrial Science
Tokyo, Japan
E-mail: oishi@cvl.iis.u-tokyo.ac.jp

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-40302-6 e-ISBN 978-3-642-40303-3
DOI 10.1007/978-3-642-40303-3
Springer Heidelberg New York Dordrecht London

Library of Congress Control Number: 2013945556

CR Subject Classification (1998): I.5, I.4, I.2.10, I.2, J.3

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

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

Three-dimensional depth data have turned out to be a key information source for solving a large number of challenging applications. In the past, substantial advances have been demonstrated in acquiring, processing, analyzing, and interpreting depth data. Today, depth data play a vital role in a variety of application areas including biometrics, cultural heritage applications, human action recognition, and 3DTV (e.g., depth-based image rendering). Through the recent development in consumer depth cameras, in particular the low-cost Kinect, a new era of depth data analysis emerged. Affordable depth cameras are changing the landscape of computer vision and related research fields, with profound impact far beyond consumer electronics. The primary objective of this book is to address the challenges in advanced depth acquisition techniques, processing and analyzing depth data, as well to consider novel and challenging applications.

The book comprises the proceedings of the International Workshop on Depth Image Analysis (WDIA 2012):

`http://cvpr.uni-muenster.de/WDIA2012/`

held in conjunction with ICPR 2012, in Tsukuba, Japan. The workshop intended to bring together researchers from multiple subfields to discuss the major research problems and opportunities of the emerging RGB-D camera revolution. A total of 27 papers were submitted to the workshop. After a careful review by an international Program Committee, 16 submissions were selected for the workshop program. The workshop attracted about 45 participants from 17 countries, including researchers working in the field of depth data processing/analysis and researchers working on 3D vision applications. To put together this book, authors of the accepted workshop papers were invited to submit their extended versions that should take into account the discussion at the workshop. Several other researchers were also invited to submit their papers covering additional important aspects of depth image analysis that were not part of the workshop. These submissions went through another round of reviews. Finally, a total of 19 papers are included in this book.

The collected papers are divided into four parts: Acquisition and Modeling of Depth Data, Processing and Analysis of Depth Data, Applications, and ICPR 2012 Contests. The last part contains two papers that present a short summary of two depth data-related contests run at ICPR 2012. Many of these articles have strong practical relevance. For instance, one article deals with the impact of thermal and environmental conditions on the Kinect sensor and another one presents on-going work about an augmented reality system for training how to play violin.

We thank Springer for giving us the opportunity to publish this book in the LNCS series. Our gratitude goes to the members of the WDIA 2012 Program Committee who participated in our stringent reviewing process.
Fabian Gigengack kindly supported us in the final editing phase.

Finally, to the readers of this book: Enjoy it!

April 2013

Xiaoyi Jiang
Olga Bellon
Dmitry Goldgof
Takeshi Oishi

Organization

Program Commitee

Atsuhiko Banno (Japan)
Kim Boyer (USA)
Robert Fisher (UK)
Patrick Flynn (USA)
Paulo Gotardo (USA)
Joachim Hornegger (Germany)
Yumi Iwashita (Japan)
Chandra Kambhamettu (USA)
Reinhard Koch (Germany)
Andreas Kolb (Germany)
Seong-Whan Lee (Korea)
Wen-Nung Lie (Taiwan)
Takeshi Masuda (Japan)
Gerard Medioni (USA)
Ryusuke Sagawa (Japan)
Sudeep Sarkar (USA)
Luciano Silva (Brazil)
Yuan-Fang Wang (USA)
Yunhong Wang (China)

Additional Reviewers

Karl Aguero
Fabian Benitez
Kester Duncan
Sergiy Fefilatyev
Leonardo Gomes
Gerry Hernandez
Di Huang
Andreas Jordt
Maik Keller
Abhishek Kolagunda
Ravikiran Krishnan
Martin Lambers
Damien Lefloch
Qingjie Liu
Vincent Ly
Bin Ma
Rohith Mv
Samuel Rivera
Philip Saponaro
Mauricio Pamplona Segundo
Matthew Shreve
Scott Sorenson

Table of Contents

Acquisition and Modeling of Depth Data

Processing and Analysis of Depth Data

Applications

ICPR2012 Contests

Optimal Decoding of Stripe Patterns with Window Uniqueness Constraint

Shuntaro Yamazaki and Masaaki Mochimaru

Digital Human Research Center
National Institute of Advanced Industrial Science and Technology
{shun-yamazaki,m-mochimaru}@aist.go.jp

Abstract. We propose the optimal algorithm of decoding color stripe patterns generated from a pseudo-random sequence (PRS). One-dimensional correspondence is solved globally by a variant of dynamic programming matching (DPM) that imposes the window uniqueness of the PRS as a hard constraint. Our algorithm runs in linear time complexity with respect to the size of projected pattern and acquired image, which is as efficient as the conventional DPM. The performance of our method is demonstrated qualitatively and quantitatively using simulation data with known ground truth and real data.

1 Introduction

Rapid depth acquisition is the fundamental task for many applications including recognition, tracking, and geometric modeling. Various approaches to this problem have been proposed such as, real-time laser scanning [1], time-of-flight camera [2], and stereo vision systems [3]. Coded light is a variant of active stereo methods where one of cameras is replaced by a projector, and has been widely adopted in many practical applications for its accuracy and capability of rapid depth acquisition.

In this paper we propose a method of decoding spatially-coded light patterns generated from pseudo-random sequence (PRS) for one-shot depth acquisition. PRS has *window uniqueness property*: Each subsequence of a certain length occurs at most once, and therefore the correspondences can be determined uniquely from the partial observation. Surprisingly, however, conventional methods of decoding the PRS from acquired images do not consider the window uniqueness property, and therefore suffer from the ambiguity in correspondence when the projected stripes are partly missing due to occlusion or low reflectance. Our decoding algorithm is independent of the underlying PRS, and therefore can be applied to different kinds of stripe patterns.

2 Color Stripes for One-Shot Depth Acquisition

The strategies of coded light are comprehensively surveyed and systematized by Salvi et al [4]. Stripe-based approach has an advantage that spatial resolution

X. Jiang et al. (Eds.): WDIA 2012, LNCS 7854, pp. 1–8, 2013.

is lost in only one dimension compared to other techniques of spatially-coded light for one-shot depth acquisition. In this section, we review some existing techniques of spatially-coded light using stripe patterns generated from PRS.

2.1 Pseudo-random Sequences

De Bruijn sequence is commonly used to generate color stripe patterns. Hügli and Maître propose sparse color stripes generated from the de Bruijn sequence of 3-bit alphabets [5]. Zhang et al. embed the sequence into the transition of colors to localize the stripes with sub-pixel accuracy [6]. The recurrence of alphabets in the de Bruijn sequence is solved by inserting dark separators [7] or eliminating recurring alphabets [8]. The robustness to imaging error is improved by considering the partial blending of adjacent colors [9]. Our reconstruction algorithm is independent of PRS from which color stripes are generated, and therefore can be applied to these stripe patterns as shown in Section 4.

2.2 Decoding Algorithms

Global methods try to achieve the best reconstruction by maximizing the cost function for decoding stripe codes. Belief propagation [10], graph cut [11], and optimization via linear system [12] have been utilized in prior work. They have high computational cost and typically take one to several seconds to reconstruct a single depth frame, making it hard to integrate the scanning component into interactive systems.

semi-global methods, on the other hand, solve the correspondence by iteratively applying local optimization. Many of existing techniques for one-shot depth acquisition rely on one-dimensional pattern matching by dynamic programming matching (DPM) [5,6,9]. Forster propagates the results of adjacent reconstruction to improve the coherence between scanlines [13]. Christoph and Angelopoulou propose an incremental method where local reconstructions are iteratively concatenated, and report that their semi-global method outperformed a global method due to the high complexity of the problem [11].

In general, spatially-coded patterns have the fundamental tradeoff between discriminability and robustness. Our method is the combination of color stripes and reconstruction by a variant of DPM algorithm, aiming at the dense pattern projection and efficient depth recovery. The goal of this paper is to propose a novel algorithm of decoding color stripes to achieve accurate reconstruction in a computationally efficient manner.

3 Solving Optimal Correspondence

In the rest of paper, we assume that the projector and camera are geometrically and radiometrically calibrated. The images are rectified so that each line of acquired images has one-to-one correspond to a one-dimensional color stripe pattern.

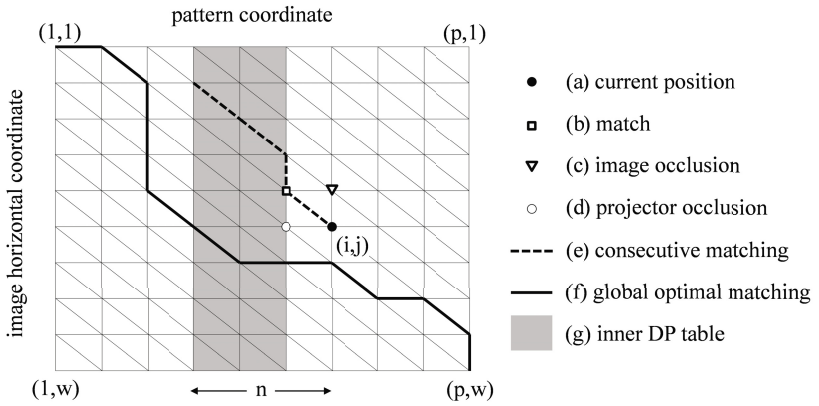


Fig. 1. DPM for a single image line: In conventional DPM, the optimal correspondence at a grid point (a) is estimated from the score at (a), and the previous results obtained at (b), (c), and (d). In our algorithm, the matching cost (b) is replaced with the optimal consecutive matches indicated by (e), which is obtained by solving another DPM using a table indicated by (g).

3.1 Dynamic Programming Matching — Review

When the projector and camera are geometrically calibrated, the depth recovery is reduced to a one-dimensional matching along epipolar lines. The matching between the projected patterns and acquired images can then be solved by DPM, on the assumption of the monotonicity of observed stripes along epipolar lines. The correspondence problem is solved independently for each line, using a two-dimensional table illustrated in Fig. 1. Please refer to prior work [6] for the detail of DPM.

DPM algorithm uses a two-dimensional table on which subproblems are solved recursively. The horizontal and vertical axes respectively correspond pattern coordinates $i \in \{1, \ldots, p\}$ and image line coordinates $j \in \{1, \ldots, w\}$. Given the score $c(i, j)$ of the matching between the pattern at i and the image at j is stored in a table, the correspondence is solved sequentially from left-top to right-bottom, by estimating the best cumulative score $M(i, j)$ according to the following formula:

$$M(i,j) \leftarrow \max\{M(i-1,j-1) + c(i,j), M(i,j-1), M(i-1,j)\} \qquad (1)$$

where $M(i, j) = 0$ if $i < 1$ or $j < 1$. The three terms in the right side of Equation (1) correspond to respectively (b), (c), and (d) in Fig. 1. The optimal matching for the entire sequences is obtained by backtracking the matches that yields $M(p, w)$.

DPM has several advantages including linear complexity and global optimality. Naïve DPM algorithms, however, try to maximize the number of matches, resulting in discontinuous correspondence which is likely incorrect. These incorrect

correspondence may be suppressed by penalizing short matches. For instance, Mei et al. propose an algorithm of a semi-global optimization suitable for the implementation on parallel computer [14]. We propose a more structured way to obtain reliable correspondences without introducing any parameters.

3.2 Two-Step Dynamic Programming Matching

Our optimization algorithm is a variant of the conventional DPM, and capable of considering the window uniqueness property of color stripes as a hard constraint. Specifically, the algorithm guarantees the following properties:

- The solution maximizes the sum of matching scores globally.
- The matching consists of consecutive subsequences of at least length n in the pattern coordinates.

Here, n is the size of window uniqueness of the underlying PRS. The second property is crucial because otherwise the algorithm tries to maximize the number of matching pairs, resulting in numerous isolated correspondences which is usually incorrect.

To enforce the minimum length of consecutive matches, we solve the DPM for each column (i.e. each pattern coordinate) in a two-step optimization. In the first step, the score of n consecutive matches is calculated in an inner DPM of size $(n-1) \times w$ indicated by a dark color in Fig. 1. The inner DPM is solved in a similar way to the conventional DPM except that all pattern colors are matched to image colors to enforce consecutive matching. The cumulative score of the inner DPM, $m(i', j)$, is updated sequentially from left-top to right-bottom in the inner table , according to the following formula:

$$m(i', j) \leftarrow \max \{m(i' - 1, j - 1) + c(i', j), m(i', j - 1)\} \tag{2}$$

where $\max \{0, i - n\} < i' < i$, and $m(i', j) = M(i', j)$ if $i' \leq i - n$. In the second step, the cumulative score $M(i, j)$ is updated by the formula in Equation (1), replacing the cumulative score $M(i-1, j-1)$ with $m(i-1, j-1)$. The pseudo-code of the two-step DPM algorithm is presented in Fig. 2.

The computational complexity of our DPM algorithm is $O(npw)$ where n is the size of window uniqueness, p is the length of a color pattern, and w is the length of an epipolar line in the image. Because n is smaller than p and w by a factor of two (e.g. $n = 4$, $p \approx 100$, $w \approx 1000$ in our experiments), our algorithm is considered as efficient as the conventional DPM [6] in practice.

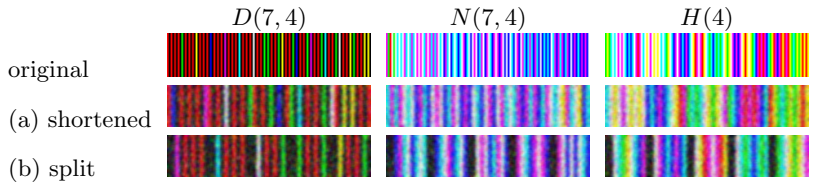
4 Results

We applied our method to four state-of-the-art techniques of color stripe patterns based on the de Bruijn sequence $B(k, n)$ where k is the number of alphabets and n is the size of window uniqueness: Direct color pattern $D(k, n)$ encodes each alphabet into non-black 8-bit colors separated by black [5]. Color transition

Input: $c(i,j)$ for $i \in \{1,\ldots,p\}$ and $j \in \{1,\ldots,w\}$
Output: $M(i,j) = \max \sum c(i'_k, j'_k)$ s.t. $x<y \Rightarrow i_x<i_y \leq k \wedge j_x<j_y \leq k$

for all $i = n$ **to** p **do**
  for all $i' = i-n+1$ **to** $i-1$ **do** ; inner DPM
    for all $j = 1$ **to** w **do**
      $m_1 \leftarrow m(i'-1, j-1) + c(i', j)$
      $m_2 \leftarrow m(i', j'-1)$
      $m(i', j') \leftarrow \max\{m_1, m_2\}$
    end for
  end for
  for all $j = 1$ **to** w **do** ; outer DPM
    $M_1 \leftarrow m(i-1, j-1) + c(i, j)$
    $M_2 \leftarrow M(i, j-1)$
    $M_3 \leftarrow M(i-1, j)$
    $M(i,j) \leftarrow \max\{M_1, M_2, M_3\}$
  end for
end for

Fig. 2. Pseudo-code of Two-step dynamic programming matching



pattern	$D(7,4)$				$N(7,4)$				$H(4)$			
distortion	(a)		(b)		(a)		(b)		(a)		(b)	
algorithm	[5]	ours	[5]	ours	[8]	ours	[8]	ours	[9]	ours	[9]	ours
miss %	1.2	2.7	0.5	1.2	0.6	3.1	0.8	3.0	0.2	0.5	0.8	0.9
error %	38.4	7.3	34.5	0.0	77.3	13.0	63.9	0.0	17.7	0.6	3.8	0.0
false %	0.0	0.0	19.7	19.5	0.0	0.0	0.4	0.2	0.0	0.0	0.2	0.1

Fig. 3. Error in the reconstruction by the conventional DPM and our algorithm: The images of size 250×50 are generated using $D(7,4)$, $N(7,4)$, and $H(4)$. The patterns are (a) shortened or (b) split, and then damaged by random noise and blurring. The table shows from top to bottom, the ratio of pixels that are not reconstructed, reconstructed with more than one pixel error, and reconstructed wrongly at empty pixels.

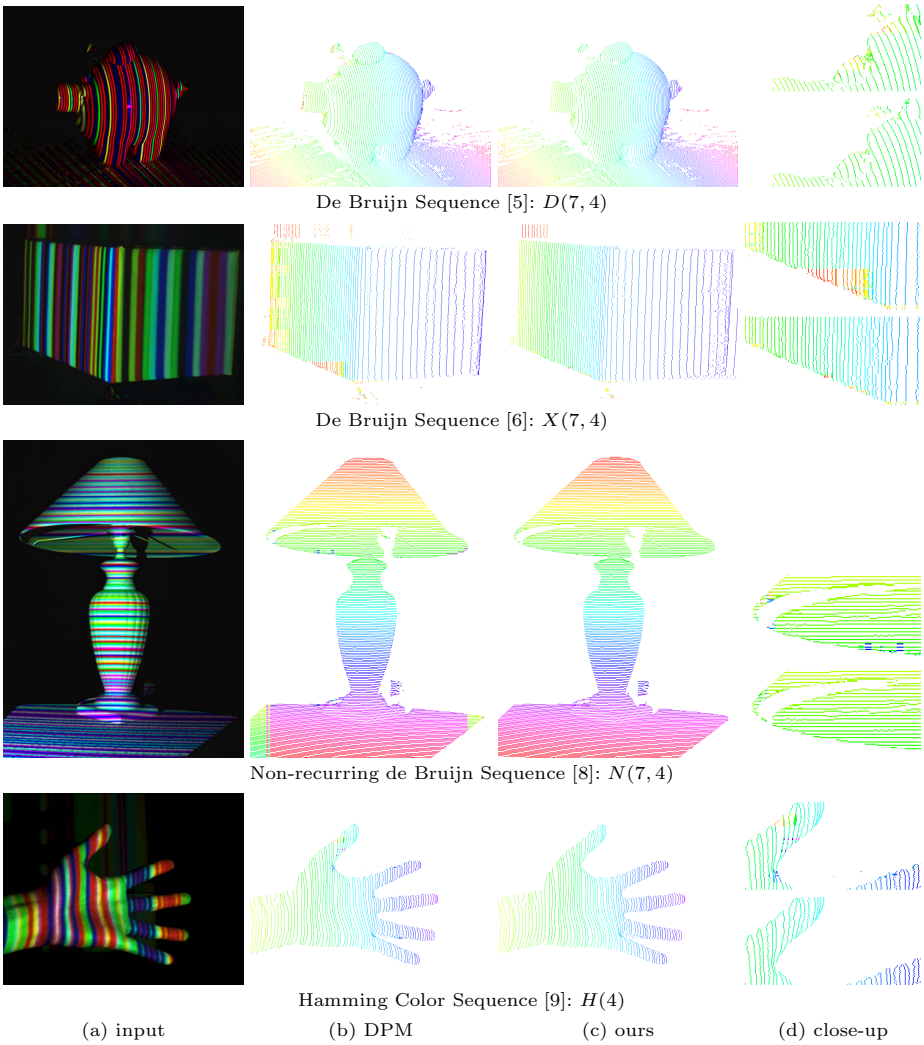


Fig. 4. Results of correspondence recovery: (a) Images of coded light projection, (b) correspondence recovered by conventional DPM, (c) our proposed algorithm encoded in hue, and (d) close-up comparisons of the results obtained by conventional (top) and proposed (bottom) algorithms. Please refer to the online version for the color figures. The conventional DPM tends to yield erroneous reconstruction around depth boundary, while our proposed algorithm can successfully eliminates spurious matching by enforcing the window uniqueness constraint that the projected patterns maintain.

pattern $X(k,n)$ is generated by modulating colors by an exclusive-or operator [6]. Other two are the non-recurring de Bruijn sequence $N(k,n)$ [8] and Hamming color sequence $H(n)$ [9]. We used $k = 7$ and $n = 4$ in our experiments. The matching score $c(i,j)$ is those proposed in the original work.

4.1 Simulation Data

The performance of our method was evaluated using simulation data for $D(7,4)$, $N(7,4)$, and $H(4)$ as shown in Fig. 3. Two types of distortion, (a) pattern shortening and (b) pattern splitting, were applied to the original patterns to simulate the coded light images in the practical situation of depth recovery. The shortening occurs when an illuminated surface is partly unobserved due to occlusion, which is simulated by randomly deleting several stripes from an input pattern. The splitting is observed when the illumination is partly occluded or object does not exist, and is simulated by randomly inserting black stripes into the original patterns. In both cases, we restricted the number and position of the deletion or insertion so that the each consecutive pattern maintains the windows uniqueness. Finally, the pattern is scaled, perturbed by random color noise, and then blurred by a Gaussian filter.

The error of the reconstruction is compared between conventional and our DPM, as presented in Fig. 3. The performance is evaluated by the ratio of erroneous pixels in the reconstructed correspondences of size 250×50. The error is calculated by the number of pixels that are not reconstructed (miss %), reconstructed with more than one pixel error (error %), and wrongly reconstructed at empty pixels (false %). Our algorithm successfully reduces the number of erroneous reconstruction, and almost perfectly suppresses false reconstruction.

4.2 Real Data

Fig. 4 summarizes the results of experiments using real data. Acquired images are presented in column (a). The correspondences recovered by conventional and proposed DPM are presented in respectively column (b) and (c), where one-dimensional coordinates are encoded in hue. Please refer to the online version of the paper for the color. The close-up of the differences between two results are presented in column (d). The acquired images do not contain significant noise, and hence, the correspondence reconstructed by the conventional DPM is accurate on the smooth surfaces. However, incorrect contiguous matches are found in many places as it attempts to maximize the number of correspondence with non-negative scores. This problem is successfully resolved by our algorithm.

5 Conclusion

We have proposed the optimal method of decoding for color stripe patterns generated from pseudo-random sequence that has window uniqueness property. We extended a conventional DPM algorithm so that the window uniqueness is imposed as a hard constraint, and demonstrated the significant improvement in the reconstruction quality using simulated and read data. The computational complexity of the proposed algorithm is comparable to that of the conventional DPM. The coherence across image epipolar lines are not considered in our current implementation, and can be improved by prior method of global or semi-global methods [6,10,11]. We are also interested in implementing the our proposed algorithm

on graphics processing unit (GPU), based on our previous work on parallel DPM on GPU [9] to accomplish robust one-shot shape recovery in real-time.

Acknowledgment. This work was supported by JSPS KAKENHI Grant No. 22700190. The first author also acknowledges support of JSPS Postdoctoral Fellowships for Research Abroad.

References

1. Livingstone, F.R., King, L., Beraldin, J.A., Rioux, M.: Development of a real-time laser scanning system for object recognition, inspection, and robot control. In: Telemanipulator Technology and Space Telerobotics, pp. 454–461 (1993)
2. Iddan, G.J., Yahav, G.: Three-dimensional imaging in the studio and elsewhere. In: SPIE Conference, vol. 4298, pp. 48–55 (2001)
3. Scharstein, D., Szeliski, R.: A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. International Journal of Computer Vision 47, 7–42 (2002)
4. Salvi, J., Pages, J., Batlle, J.: Pattern codification strategies in structured light systems. Pattern Recognition 37(4), 827–849 (2004)
5. Hügli, H., Maître, G.: Generation and use of color pseudo random sequences for coding structured light in active ranging. In: Industrial Inspection, vol. 1010, pp. 75–82 (1989)
6. Zhang, L., Curless, B., Seitz, S.M.: Rapid shape acquisition using color structured light and multi-pass dynamic programming. In: The 1st International Symposium on 3D Data Processing, Visualization, and Transmission, pp. 24–36 (2002)
7. Pagès, J., Salvi, J., Forest, J.: A new optimised de bruijn coding strategy for structured light patterns. In: International Conference on Pattern Recognition, vol. 4, pp. 284–287 (2004)
8. Lim, J.: Optimized projection pattern supplementing stereo systems. In: International Conference on Robotics and Automation, pp. 2823–2829 (2009)
9. Yamazaki, S., Nukada, A., Mochimaru, M.: Hamming color code for dense and robust one-shot 3D scanning. In: British Machine Vision Conference 2011, pp. 96.1–96.9 (2011)
10. Ulusoy, A.O., Calakli, F., Taubin, G.: Robust one-shot 3D scanning using loopy belief propagation. In: CVPR Workshop on Applications of Computer Vision in Archaeology, pp. 15–22 (2010)
11. Schmalz, C., Angelopoulou, E.: A graph-based approach for robust single-shot structured light. In: Projector-Camera Systems, pp. 80–87 (2010)
12. Sagawa, R., Ohta, Y., Yagi, Y., Furukawa, R., Asada, N., Kawasaki, H.: Dense 3D reconstruction method using a single pattern for fast moving object. In: International Conference on Computer Vision, pp. 1779–1786 (2009)
13. Forster, F.: A high-resolution and high accuracy real-time 3D sensor based on structured light. In: 3D Data Processing, Visualization, and Transmission, pp. 208–215 (2006)
14. Mei, X., Sun, X., Zhou, M., Jiao, S., Wang, H., Zhang, X.: On building an accurate stereo matching system on graphics hardware. In: ICCV Workshop on GPU in Computer Vision Applications, pp. 467–474 (2011)

A 3D Impression Acquisition System for Forensic Applications

Ruwan Egoda Gamage, Abhishek Joshi, Jiang Yu Zheng, and Mihran Tuceryan

Indiana University Purdue University Indianapolis (IUPUI), Indianapolis IN 46202, USA
{rjegodag,abhjoshi,jzheng,tuceryan}@cs.iupui.edu

Abstract. This paper presents a method with which 3D images of tire track and footprint impressions at crime scenes can be captured with high fidelity, while capturing high resolution 2D color texture images simultaneously. The resulting device is portable, easy to use, is non-destructive of the evidence, and saves time at crime scenes. The same technique can also be used in the laboratory to create 3D depth images of suspect tires or shoe soles. Computer-based pattern matching technology can be used to assist in matching and comparison tasks. The device produces better quality data at a close range obtained in a larger field compared to existing devices. It avoids problems related to occlusions by using two lasers and can digitize long spans of impressions in one scan. The method includes a calibration method which is integrated into the scanning process on site, thus avoiding problems with pre-calibrated configurations becoming stale during transportation and setup.

1 Introduction

In crime scene investigations it is necessary to capture images of impression evidence such as tire track or shoe impressions. Currently, such evidence is captured by taking two-dimensional (2D) color photographs or making a physical cast of the impression in order to capture the three-dimensional (3D) structure of the information [1,5,7]. The 2D photographs, under the right illumination conditions, may highlight feature details in the evidence, but do not provide metric depth measurement information for such features. Obtaining a 3D physical cast of the impression may destroy the evidence in the process. Therefore, the use of a 3D imaging device which can capture the details of such impression evidence can be a useful addition to the toolkit of the crime scene investigators (CSI). In this paper, we present the design of such an impression imaging device which includes a calibration method for obtaining the 3D image with the proper metric information. The method can provide a depth resolution of around 0.5mm and high resolution color image.

Related Work: The normal process of imaging impression evidence requires that the camera's optical axis be perpendicular to the ground at the site of the track. Also, there is a requirement for proper oblique lighting in order to see details created by varying depths of impression as intensity variations in the photographs. In the case of tire tracks, where the impression may cover several linear feet, the ground may not be level and camera distances may lead to confusing readings [1,5,7]. The requirements for imaging

X. Jiang et al. (Eds.): WDIA 2012, LNCS 7854, pp. 9–20, 2013.

such evidence are specified by the Scientific Working Group for Shoe print and Tire Tread Evidence (SWGTREAD) guidelines [8]. A device based on similar principles has been built before in order to scan relics excavated from archeological sites and construct their 3D computer models [10,11]. The current work modifies it to satisfy the special requirements in the field of forensic imaging. Buck et al. present the use of existing commercial 3D imaging devices for footwear impressions [2]. Khoshelham et al. shows consumer-grade range camera such as Kinect sensor has random error of depth measurement ranges from few millimeters to 4cm at the maximum range of the sensor [6]. The existing devices do not satisfy some of the imaging requirements (e.g., resolution in depth) in forensics applications. They do not work very well outdoors on long track impressions with a single scan. They usually require multiple short scans which need to be stitched together.

2 Design of the 3D Imaging System

2.1 Hardware Setup

The device for digitizing the impression evidence consists of a motorized rail (actuator) with a HD video camera and two line laser lights, each with a different color as shown in Figure 1.

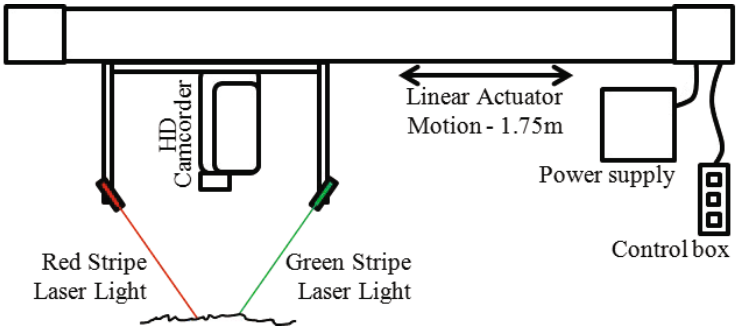
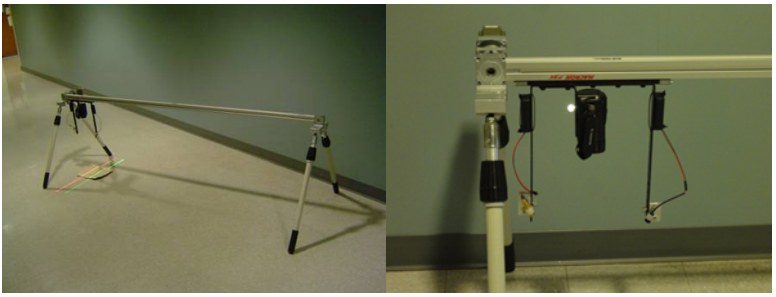


Fig. 1. The design for the device prototype built

A physical prototype of the device is shown in Figure 2. To build the device, we used the following components:

1. Linear actuator rail.
2. Servo motor with a gearbox.
3. Power supply, USB interface cable, and programming interface.
4. Camcorder.
5. Two Laser stripe lights.

We designed and built a bracket and a leg assembly on the two sides of the actuator rail that can be taken off in order to package and carry the device to the field. We also designed and built a bracket to mount the camcorder and the laser lights onto the rail



(a) (b)

Fig. 2. The prototype of the device built and a close-up of the laser and camcorder assembly

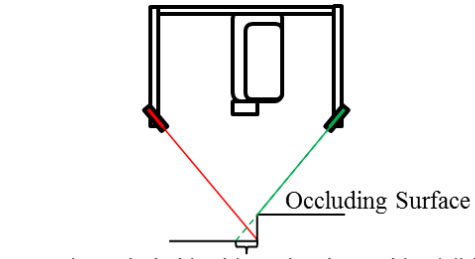


Fig. 3. Two lasers are used to eliminate the blind spots due to occlusion. The different laser colors solve the correspondence problems for calculating the triangulation.

with a fixed geometric configuration. The final physical prototype of the device is shown in Figure 2.

The two different colored lasers are used to eliminate the blind spots due to occluding surfaces as shown in Figure 3. Also, the two different colored lasers are used in order to distinguish the two different light sources, and, thus, solve the correspondence problem more easily.

The device has a total weight of 11 lb. (about 5kg) excluding the legs so that it is portable by one person and can be loaded into a normal van or a pick-up truck. Its legs can also easily be dismantled for easy transportation. By setting the device at the measuring site properly, a user can push one of the four buttons to operate the device: (a) home, (b) stop, (c) short scan, or (d) long scan. Short scan and long scan buttons move the laser-camera assembly by one of the preset distances (50cm, 170cm). This provides an appropriate distance along the rail direction according to the user's requirement. The home button moves the laser-camera assembly to a preset home position and the stop button allows for an emergency stop and/or selects a new starting position for scanning short length ground scenes.

During an outdoor tire and footprint scan, the power can be drawn from a nearby vehicle or from a charged portable battery unit. The camera focus and exposure are set to manual mode and fixed after auto-focus and auto-exposure are obtained at the

beginning of the scan. This eliminates possible changes to the optical parameters of the camera during the scanning process due to changes in height in the scanned surface.

A user interface for controlling the device has also been developed on a PC as well as on a mobile tablet using Android platform. This interface allows the field technician to control an extended set of scan parameters such as rail speed, start and stop positions, and/or run length in a more customized way than the preset buttons. A more advanced user interface in the future may also allow the capability to change the preset values of these buttons.

The video of two laser stripes are recorded onto a flash memory card. After the scan of the entire site is done, the memory card is moved to a PC in the crime lab and the video clips are copied to the hard disc of PC for image processing, 3D reconstruction, and further analysis.

2.2 Robust Detection of Laser Stripes in Video Sequence and Texture Acquisition

Laser stripe detection For robust laser stripe detection, we have implemented a slightly modified method based on an adaptively constructed RGB lookup table [9]. This approach yielded better results compared to our previous method [4]. This approach contains four image processing steps.

- *Stripe edge detection:* Edge detection is done using a Sobel filter. The red channel of the image is used in edge detection for red laser. Similarly, green channel is used for green laser image. The resultant image has pixels highlighting horizontal edges of the laser stripe. This goes through another step with a high and low thresholds to eliminate noise and low response edges. All pixels in between a negative and a positive edge are considered as pixels belong to the laser stripe.
- *RGB lookup table based color validation:* To validate the pixels in a laser stripe, two lookup tables — one for each of the green and red laser images — are generated from every 100th frame of the video. The lookup table consists of color values belonging to the *background color subspace*. These values are generated from pixels c_i outside the region of interest(ROI) in which laser stripe is searched in order to capture the color characteristics of the background colors. Here $c_i = (R, G, B) \in ([0, 255], [0, 255], [0, 255])$, where the square brackets indicate ranges in the three color bands. Let's consider the lookup table: bcs_{red} for red laser. First, we construct a complement lookup table, $\overline{bcs}_{red}(R, G, B)$ as follows:

$$\overline{bcs}_{red}(R, G, B) = \begin{cases} 1 & \text{if } \exists i : (R, G, B) \in c_i \text{ or } (R = 0) \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

Then, we use this to construct the actual lookup table $bcs_{red}(R, G, B)$ that will be used for detecting the laser pixels:

$$bcs_{red}(R, G, B) = \begin{cases} 1 & \text{if } \exists R' \geq R : \overline{bcs}_{red}(R', G, B) = 1 \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

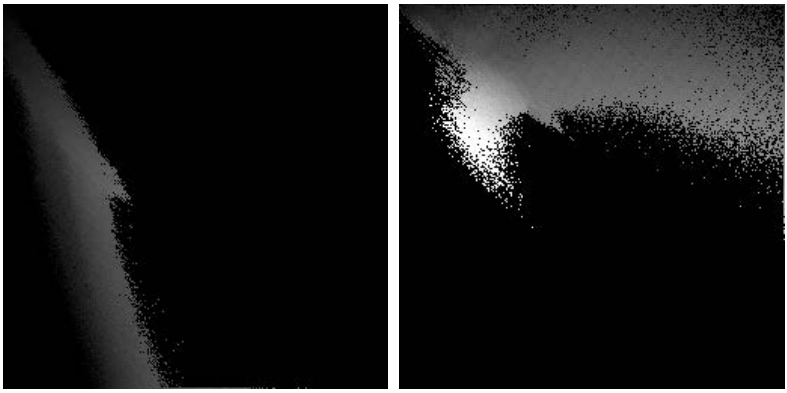
The lookup table bcs is constructed with the assumption that laser stripe color values are suppose to have higher red components than the background (Figure 4).

A given color pixel $x_i = (r, g, b)$ is classified as a red laser pixel, if $bcs_{red}(r, g, b) = 0$. The lookup table $bcs_{green}(R, G, B)$ is constructed in a similar manner to the red one and used in the same way for the green laser pixel detection. If a stripe from the edge detection step contains more than 95% of laser pixels, it is considered a laser stripe.

- *Stripe width validation:* Each stripe is validated again whether it is between a higher and a lower threshold. The lower threshold is set to 2 pixels and higher threshold is set to 20 pixels. These numbers provided more signal to noise ratio during testing.
- *Peak estimation:* To determine the peak position of stripe with sub-pixel accuracy, we used center of mass method [3]:

$$y_{peak} = \frac{\sum_{i=m}^{m+w} iI(i)}{\sum_{i=m}^{m+w} I(i)} \tag{3}$$

where i is the starting row of the laser stripe and w is the width of the detected laser stripe and $I(i)$ is the intensity of that particular color channel at pixel i within each column.



(a) (b)

Fig. 4. Visualization of RGB lookup tables for an indoor scan (a) Green lookup table. A 255x255 matrix, red and blue channels are represented by rows and columns respectively. Green channel is represented by the element value. (b) Red lookup table. A 255x255 matrix, green and blue channels are represented by rows and columns respectively. Red channel is represented by the element value.

Color texture image extraction: Simultaneously with the laser stripe detection, we also extract a high resolution color image of the impression evidence. This is done by extracting the pixels along the $y = 0$ line of each video frame having the origin at the center of the image. The color image captured by this process has the following properties: (i) The spatial resolution of the image along the scan direction is dependent on the speed with which the linear actuator moves. The slower the actuator moves, the higher the image resolution in the scan direction because the video is being captured at a fixed 30fps—the distance between the scan lines of successive video frames will be smaller as the actuator moves more slowly; (ii) The image formed along the scan direction is an orthographic projection determined by the scan motion.

In the direction perpendicular to the scan motion, the resolution is a perspective projection with the spatial resolution determined by the highest resolution of the HD camcorder. The size of the highest resolution video frame is 1920×1080 pixels. In order to maximize the resolution of the resulting image, the camcorder is oriented such that the highest resolution dimension of the video frames (i.e., 1920 pixels) is perpendicular to the actuator motion.

3 System Calibration

We have integrated the calibration of the geometry into the scanning and data collection process. This eliminates the risk that a pre-calibrated configuration can be invalidated during the transportation and set-up of the device. The only requirement in the field is that the criminalist places the calibration object in the scene for at least one scan.

We use an L-shaped calibration object (shown in Figure 5) with known dimensions to calibrate the geometric configuration of the laser beams and the camera in order to compute the height map image. The system captures the calibration object in at least one scan.

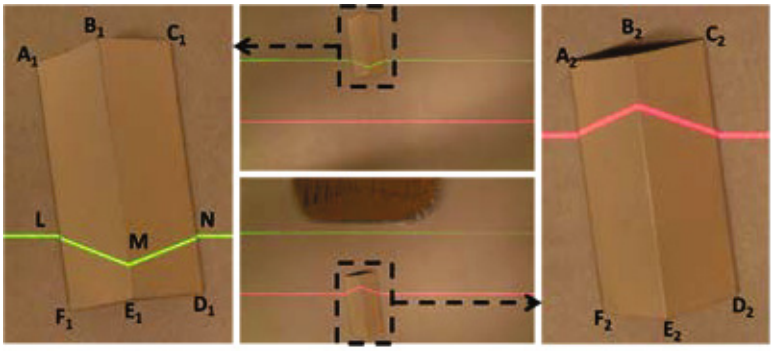


Fig. 5. Frames are captured when each laser scan over the calibration object at time t_1 and t_2

Currently, we operate the system in its slowest speed of 1.3138 mm/s in order to obtain highest scan resolution along the y axis. We perform a calibration of the video camera in order to correct for radial lens distortion at widest zoom settings, and we computed a single focal length f for this setting to be 2110 in pixel units along both x and y directions. We correct for this radial distortion in every captured frame and this corrected image is used for all subsequent depth calculations.

We use a vanishing point method to find the pose of the camera with respect to the rail and the rail motion direction. Then we use the calibration object (Figure 5) to calculate the orientations of the red and green laser planes. Everything, including the points on the evidence surface is eventually calculated in a single coordinate system which is defined by the rail. The details of each of these calculations are described below.

Coordinate System Definitions: We use three main coordinate systems (see Figure 6)

1. *o-xyz*: original camera coordinate system. Coordinates are measured in pixel units.
2. *O-XYZ*: ideal image coordinate system is still camera centered, but corrected for the roll and tilt of the camera with respect to the rail coordinates. The Y axis is parallel to the rail motion and the Z axis points downwards where the camera points. Coordinates are in pixel units.
3. *O-*$X'Y'Z'$: rail coordinate system. The coordinate system is aligned with the ideal image coordinate system orientation and related to it with a translation. The coordinates are in metric units.

The calibration procedure: Ideally, the camera should be connected to the rail, looking downward and the y-axis of the image aligned perfectly with the rail's direction of motion. Achieving this physically, however, is not realistic. Therefore, we assume the camera is put on the rail roughly as described above and the small misalignments are accounted for via a calibration procedure. This calibration calculates the exact pose of the camera with respect to the rail coordinate system. The calibration object is placed in the scene roughly pointing in the direction of the rail motion. Assume the camera is perfectly aligned with the rail coordinate system. If we translate the camera along the rail and take two images, one before and one after the translation, corresponding points on the calibration object in the two images will form parallel lines. This will result in a vanishing point formed by the lines at infinity. In reality, the camera is not perfectly aligned with the rail system resulting in the vanishing point to be finite. We use this fact to estimate the camera pose with respect to the rail coordinate system from the calculated vanishing point.

We capture two frames as seen in Figure 5, one at $t = t_1$ when the green laser is projected onto the calibration object and the second at $t = t_2$ when the red laser is projected onto the calibration object. We mark the corners of the calibration object ($A_i, B_i, C_i, D_i, E_i, F_i$, for $i = 1, 2$). This is done via an interactive interface developed in the software that lets the person doing the computations in the crime lab pick these points. The following are the steps for achieving this calibration.

First, we calculate the vanishing point from the corresponding points in two frames as described above. Let this vanishing point be (x_v, y_v).

Second, we compute the pose of the camera with respect to the rail from this vanishing point (for *O-XYZ* transformation). The camera roll, θ, (around its optical axis) is given by $\theta = \tan^{-1}(x_v/y_v)$.

The camera tilt, α, between the optical axis and the Z axis in the ideal coordinate system *O-XYZ* is given by$(y_v/|y_v|)\tan^{-1}\left(f/\sqrt{x_v^2 + y_v^2}\right)$.

Next, the calculated roll and the tilt of the camera is used to obtain the transformation from the image coordinates (*o-xyz*) to the ideal coordinates (*O-XYZ*). This transformation is given by $\mathbf{T} = \mathbf{R}_x(\alpha)\mathbf{R}_z(\theta)$, where the $\mathbf{R}_x(\alpha)$ and $\mathbf{R}_z(\theta)$ are the rotation transformations around the x and z axes, respectively. Note that because the linear motion is along the y axis, we do not need to calculate the third rotation angle, pan.

Computing rail coordinates of a point on the calibration object: After we apply the roll and tilt correction, the transformed coordinates of the points A_i may not be parallel

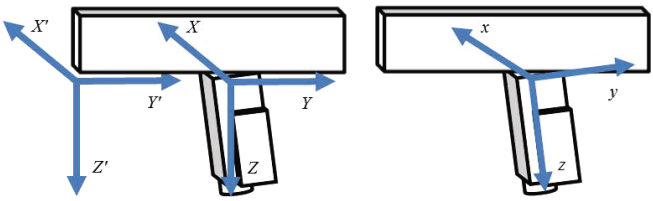


Fig. 6. Various 3D coordinate systems in our system

to the motion direction. Therefore, we project these points to a plane parallel to the motion direction. We use the $Z = f$ plane. This is done by projecting a ray from the origin (camera's optical center) via the point in the ideal image coordinate system. Considering the first frame, i.e., when $t = t_1$, we obtain $Z_{a_1}/f = Y_{a_1}/Y$ and $Z_{a_1}/f = X_{a_1}/X$. Therefore, the projection of A_1 onto the $Z = f$ plane is obtained by $A_{1f} = (X_{a_1}f/Z_{a_1}, Y_{a_1}f/Z_{a_1}, f)$. Similarly, the projection of A_2 onto the $Z = f$ plane is given by $A_{2f} = (X_{a_2}f/Z_{a_2}, Y_{a_2}f/Z_{a_2}, f)$.

Finding the Z value in rail coordinate system: Assume the distance traveled between the first frame and the second frame is d (in metric units). Then by similar triangles, we obtain

$$Z' = \frac{d}{(Y_{a_2}/Z_{a_2} - Y_{a_1}/Z_{a_1})} \tag{4}$$

Finding the point A_1 in rail coordinate system at time $t = t_1$: Considering the edges of a triangle, we obtain $Z_{a_1}/Z' = X_{a_1}/X'$ and $Z_{a_1}/Z' = Y_{a_1}/Y'$. Therefore the ideal coordinates of the point A_1 at time $t = t_1$ is given by $A'_1 = (X_{a_1}Z'/Z_{a_1}, Y_{a_1}Z'/Z_{a_1}, Z')$. Points corresponding to B_1, C_1, D_1, E_1, and F_1 are similarly computed in the rail coordinate system.

Finding the laser plane: Let's consider the green laser plane. First, we transform the pixel locations of L, M, and N (in Figure 5) to the ideal image coordinate system. We project rays, starting from optical center, through all transformed points. Next we compute the intersection of the ray and calibration object edges to find the points where laser touches the calibration object. Since edges are in rail coordinate system, we are getting laser points in rail coordinate system. Finally, using these points, green laser plane is computed.

We find the red laser plane using the same method with the red laser points analogous to L, M, and N. We perform all the steps above when $t = t_2$ coordinates systems.

Assume a point on the laser plane is $\mathbf{P}_a$, and its surface normal is $\mathbf{N}$, using the vectors $\overrightarrow{LM}$ and $\overrightarrow{MN}$ we have the equation of the laser plane normal given as the cross-product:

$$\mathbf{N} = \overrightarrow{LM} \times \overrightarrow{MN} \tag{5}$$

And the equation of the laser plane, for any point $\mathbf{P}$ on it, is given by the dot-product:

$$\mathbf{N} \cdot (\mathbf{P} - \mathbf{P}_a) = 0 \tag{6}$$

4 Computing the 3D Impression Image

Each detected laser pixel (x, y) in a frame, is transformed to the ideal image coordinate system. Through that point we project a ray starting from the optical center. The ideal image coordinate system and the rail coordinate system share the same axes, but they may be at a different scale. Therefore, finding the intersection of the ray and a laser plane gives a rail coordinate of the laser pixel directly.

By applying the offset between red and green laser coordinate systems, i.e. the translation transformation is along y axis — d as in Eq 4, we bring the red laser plane to $t = t_1$ rail coordinate system. This way, a ray and laser plane intersections always provide registered results for both red and green lasers points. This makes the later fusion of the two height maps easier.

For any point $\mathbf{P}$ on the laser plane (lit by the laser stripe), its 3D position satisfies:

$$\mathbf{N} \cdot (\mathbf{P} - \mathbf{P}_a) = 0$$

$$\mathbf{N} \cdot \left(\left(\frac{X(t)Z'}{f}, \frac{Y(t)Z'}{f}, Z' \right) - \mathbf{P}_a \right) = 0 \tag{7}$$

where $\mathbf{P}_a$ is a point on the laser plane. From this, Z' can be computed as

$$Z' = f \frac{\mathbf{N} \cdot \mathbf{P}_a}{\mathbf{N} \cdot (X(t), Y(t), f)} \tag{8}$$

and

$$X' = X(t) \frac{\mathbf{N} \cdot \mathbf{P}_a}{\mathbf{N} \cdot (X(t), Y(t), f)} \tag{9}$$

$$Y' = Y(t) \frac{\mathbf{N} \cdot \mathbf{P}_a}{\mathbf{N} \cdot (X(t), Y(t), f)} + tV \tag{10}$$

where V is the camera translation speed and t is time. The depth calculation is performed in a lookup table so that the 3D transformation from the image coordinates can be performed much faster.

Finally the height map is constructed by choosing the z value as the intensity. In the results section, the z value is mapped into 0-255 range of 8-bit gray level intensity in order to allow viewing the results in common image viewing software such as Adobe Photoshop or Gimp. The depth map has a resolution of (7500×7500) where each pixel corresponds to 0.1mm in size. One intensity level in the depth map corresponds to 0.1mm, and the coarsest level for looking at the global elevation changes starts from 10mm.

The prototype software, however, stores the real heights as real number for each pixel (in double data type) in a binary data file. These data files can be used for further processing, such as pattern matching, using the actual metric information.

The color texture map of the scanned part has also the same resolution, stored in another image with points exactly corresponding to the ones in the depth map. The user can compare depth map and the color texture map images to find the shapes of the impression unrevealed in the texture map due to the color and lighting on the ground, and confirm the depth on strange shapes by examining the color texture image.

5 Experimental Results

We used following components in the device.

1. Linear actuator rail: Motion Dynamic belt drive linear actuator MSA-PSC, 1771mm long.
2. Myostat Cool Muscle (CM1 Series) Integrated Servo motor with a planetary gearbox 8:1.
3. 24VDC Power supply, USB interface cable, and programming interface.
4. Canon VIXIA HF M30, HD camcorder.
5. Laser stripe lights.
   (a) 5mWatt, 532nm green module.
   (b) 3mWatt, 650nm red module.

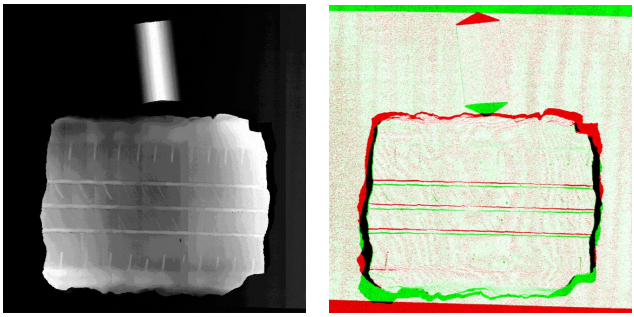
To test the system we have scanned tire and shoe impressions. Figure 7a shows the computed impression image for one such scan. The scan video file consisted of a total of 11,146 frames captured in 371.533 seconds. The total size of the video is approximately 1.03 GB and each frame in the video has a resolution of 1920×1088 pixels.

Figure 7a shows the fused result of the two independently processed images with the red and green laser. Figure 7b shows the contributions of each laser to the final result color coded. Over most of the image the two computed height maps from the two lasers agree. Where one image has data and the other does not due to occlusions, the fusion process fills in these pixels with the appropriate value. Figure 8a shows a longer scan of shoe print impressions. Figure 8c and 8d show the fused depth image for a shoe print.

We also captured some impressions in different materials such as snow, mud and sand. In mud and sand we got good results. However, in snow we had difficulties to detect laser due to reflectivity properties. We also experienced difficulties to detect laser in outdoor scanning due to strong lighting conditions. It was necessary to provide a shade for controlled lighting.

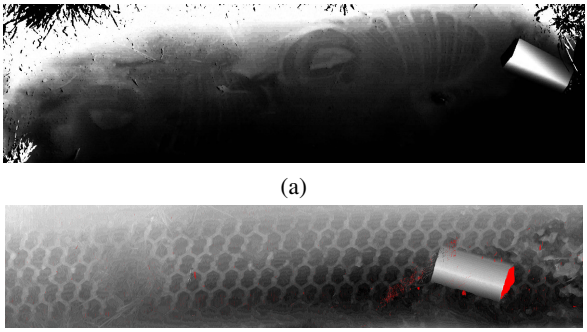
We scanned some 3D images using range scanners that are commercially available on the market. We scanned shoeprint impressions using the Kinect sensor, the Konica Minolta Vivid 910fw 3D Laser Scanner and the 3dMD Scanner. In all cases, we found that the resulting accuracy was not sufficient to detect sub-millimeter features in the impression.

Accuracy and Resolution Measurements: We have determined the following accuracy measurements for the system: (i) Rail speed = 1.3138 mm/s, fps = 30, $f = 2110$; (ii) Resolution along Y' axis = 1.3138 / 30 = 0.0438mm; (iii) Resolution along X' axis (at a Z' distance of 500mm) = Z'/f = 500 / 2110 = 0.2369 mm; and (iv) Empirically, we have observed that we can resolve 0.5mm in Z' values as reflected in detectable differences of at least 1 pixel in the disparity image computed. (v) The device takes approximately 20 minutes to scan a 1.75m long surface. Note that even though 20 minutes seems like a long time, this is a great improvement for the current practices of obtaining impression evidence in the field. Currently, if the impression evidence is captured by making physical casts, not only can the process take longer, but it could also destroy the evidence in the process.



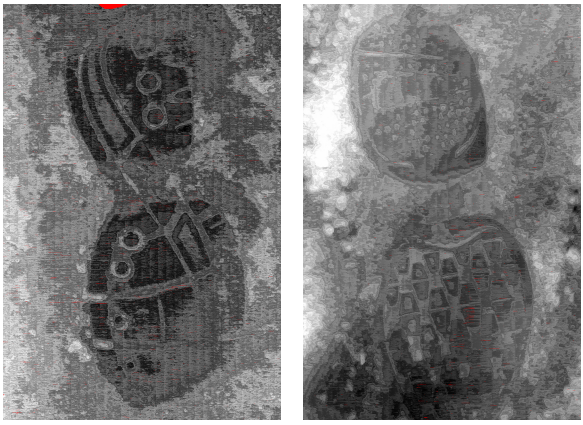
(a) (b)

Fig. 7. Computed impression image results: (a) Fused height map for a tire impression; (b) Contributions of each laser for this fusion



(a)

(b)



(c) (d)

Fig. 8. Computed impression image results: (a) A long scan of two shoe prints. (b) A long scan of tire track. (c) & (d) Scans of shoe prints.

6 Conclusion

In summary, we have developed an inexpensive high resolution 3D impression device for digitizing shoe and tire impressions in crime scenes. The device can also be used for many other objects such as ancient calligraphy on stones. We used two laser modules to eliminate occlusions and improved performance. The calibration method we used is integrated in the scanning process and eliminates the requirement of pre-calibrating the system which can become stale in the field due to the pre-calibrated configuration being changed during transportation and setup. Compared to current practices in forensics, the device can greatly improve and speed up the process of collecting impression evidence in the field. Moreover, currently, in order to scan long tire tracks, multiple photographs need to be taken along the track and stitched together. Our device can capture a 3D impression of such long tire tracks in one single scan.

Acknowledgements. This project was supported by Award No. 2010-DN-BX-K145, awarded by the US National Institute of Justice, Office of Justice Programs, U.S. Department of Justice. The opinions, findings, and conclusions or recommendations expressed in this publication/program/exhibition are those of the author(s) and do not necessarily reflect those of the Department of Justice.

References

1. Bodziak, W.: Footwear impression evidence: detection, recovery, and examination. CRC Press (1999)
2. Buck, U., Albertini, N., Naether, S., Thali, M.J.: 3D documentation of footwear impressions and tyre tracks in snow with high resolution optical surface scanning. Forensic Science International 171(2-3), 157–164 (2007), http://www.sciencedirect.com/science/article/pii/S0379073806006712
3. Fisher, R.B., Naidu, D.K.: A comparison of algorithms for subpixel peak detection. In: Image Technology, Advances in Image Processing, Multimedia and Machine Vision, pp. 385–404. Springer (1996)
4. Gamage, R.E., Joshi, A., Zheng, J.Y., Tuceryan, M.: A High Resolution 3D Tire and Footprint Impression Acquisition for Forensics Applications. In: Proceedings of IEEE Workshop on the Applications of Computer Vision (WACV), Clearwater Beach, FL (in press, January 2013)
5. Houck, M., Siegel, J.: Fundamentals of forensic science, 2nd edn. Academic Press (2010)
6. Khoshelham, K., Elberink, S.O.: Accuracy and resolution of kinect depth data for indoor mapping applications. Sensors 12(2), 1437–1454 (2012), http://www.mdpi.com/1424-8220/12/2/1437
7. McDonald, P.: Tire imprint evidence. CRC Press (1992)
8. Scientific Working Group for Shoeprint and Tire Tread Evidence: SWGTREAD Guidelines, http://www.swgtread.org/
9. Strobl, K., Sepp, W., Wahl, E., Bodenmuller, T., Suppa, M., Seara, J., Hirzinger, G.: The DLR multisensory Hand-Guided Device: the Laser Stripe Profiler. In: Proceedings of the 2004 IEEE International Conference on Robotics and Automation, ICRA 2004, April 26-May 1, vol. 2, pp. 1927–1932 (2004)
10. Zheng, J.: A flexible laser range sensor based on spatial-temporal analysis. In: Proceedings of the 15th International Conference on Pattern Recognition, vol. 4, pp. 740–743 (2000)
11. Zheng, J., Zhang, Z.L.: Virtual recovery of excavated relics. IEEE Computer Graphics and Applications 19(3), 6–11 (1999)

Impact of Thermal and Environmental Conditions on the Kinect Sensor

David Fiedler and Heinrich Müller

Department of Computer Science VII, Technische Universität Dortmund,
Otto-Hahn-Straße 16, 44227 Dortmund, Germany
{fiedler,mueller}@ls7.cs.tu-dortmund.de

Abstract. Several approaches to calibration of the Kinect as a range sensor have been presented in the past. Those approaches do not take into account a possible influence of thermal and environmental conditions. This paper shows that variations of the temperature and air draft have a notable influence on Kinect's images and range measurements. Based on these findings, practical rules are stated to reduce calibration and measurement errors caused by thermal conditions.

Keywords: Kinect Sensor, Calibration, Thermal Influence.

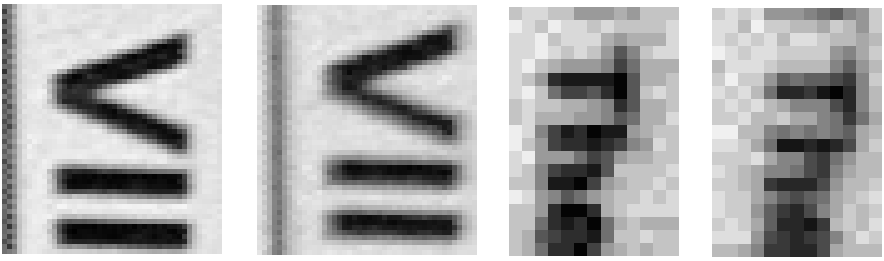
1 Introduction

Many applications utilize the Kinect [16], originally an input device of the Microsoft Xbox video game console, as a range sensor, e.g. [3,4,6]. Several comparisons of accuracy between Kinect's depth data and other range systems, like laser range scanners [5], Time-of-Flight cameras [1] or PMD cameras [7], have been evaluated. All these works perform geometric (intrinsic and distortion parameters) and depth (range) calibration to increase accuracy. Some works also involve Kinect's internal RGB camera within the depth calibration process [1,2,11] to gain accuracy. But all of them do not consider thermal and environmental conditions, neither during the calibration phase, nor during the measurement or evaluation phase.

This paper experimentally demonstrates that variations of temperature as well as air draft significantly affect the range measurement of the Kinect. Air draft can cause changes of the depth values up to 21 mm at a total distance of 1.5 m, and temperature variations cause changes up to 1.88 mm per 1°C difference. The necessary warm-up time to rule out temperature-induced errors is up to 60 minutes. Depending on its thermal state, Kinect's RGB camera shows a shift of projected objects up to 6.7 pixels measured with an optical flow approach. This observation is also important for range calibration since, as mentioned before, many approaches involve the RGB camera. The findings are transferred into rules which help to reduce measurement errors.

The following section gives a brief survey of related work. Section 3 is devoted to the influence of different thermal states to the optical lens system of both

X. Jiang et al. (Eds.): WDIA 2012, LNCS 7854, pp. 21–31, 2013.



(a) RGB cold state (b) RGB warm state (c) IR cold state (d) IR warm state

Fig. 1. Close-up views of small regions that were cropped out from the image of the texture-rich poster shown in Fig. 2(a)

internal cameras (RGB and IR). Section 4 presents several experiments based on different approaches of distance measurements and different environmental conditions. Conclusions are given in section 5.

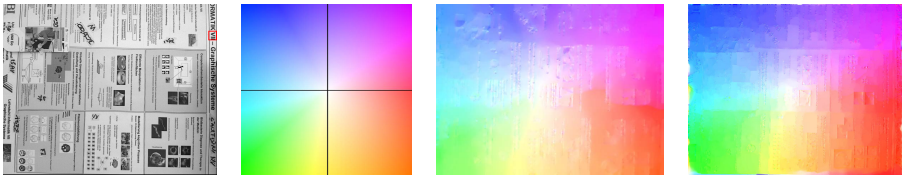
2 Related Work

The influence of temperature in the context of imaging and range sensors has been studied in the past. In the field of aerial mapping, lenses were put into large refrigerators to simulate the temperature in high altitudes and to calibrate them under these conditions [18]. The robustness of different calibration methods and their influence to the calibration accuracy was tested under temperature variations in [20]. In [9] a method for temperature calibration of an infra-red camera using warm water was proposed. The effect of temperature variations on intrinsic parameters of SLR-cameras has been studied in [19]. For the Swiss Ranger SR-2 Time-of-Flight camera the offset drift caused by self-induced heating of the sensor and the drift due to changes of the environmental temperature were analyzed in [8]. However, the influence of temperature and other environmental conditions has not been investigated for the Kinect sensor so far.

3 Thermal Influence on Kinect's Optical Camera Systems

To determine the thermal influence on the optical systems of both internal cameras, we tested the Kinect at two different thermal conditions (heat states). In one case (called cold state) the Kinect was cooled down by an externally mounted fan (cf. Fig. 3(c)) which slowly streams air through the Kinect's body and cools down its internal components to the environmental temperature of 27.6°C. In the other case (called warm state) the fan was deactivated and the Kinect was warmed up just by processing the color and depth image-stream for 45 minutes. The fan was always accelerated smoothly to prevent motion of the Kinect.

Image Based Comparison. A texture-rich poster (cf. Fig. 2(a)) was captured by both cameras at both heat states. Comparing pictures taken in different heat



(a) Poster (b) Color code (c) IR opt. flow (d) RGB opt. flow

Optical Flow	Image resolution	Maximal flow	Horizontal Range	Vertical Range
RGB	1280x1024	6.7338	-6.440 .. 4.897	-5.044 .. 3.678
IR	640x480	1.7853	-1.666 .. 1.581	-1.461 .. 1.115

Fig. 2. Texture-rich poster (a) observed by both cameras in both heat states. In the color coding scheme (b), hue indicates direction and saturation indicates magnitude. White color indicates no movement, strong colors indicate large movements. The results of the dense optical flow from the cold to the warm state are shown in (c) for the IR and in (d) for the RGB camera. The table summarizes quantitative details.

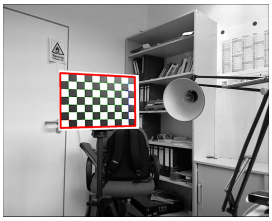
Table 1. Comparison of calibration parameters of the RGB camera (upper table) and the IR camera (lower table) for the cold and warm state

Parameter	f_x	f_y	c_x	c_y	p_1	p_2	p_3	q_1	q_2	Error
RGB										
Cold	1041.60	1043.29	656.70	520.56	0.18314	-0.51989	0.45196	0.00012	0.00122	0.07146
Warm	1046.25	1048.14	656.87	523.67	0.19178	-0.55455	0.50113	0.00069	0.00144	0.06770
Difference	4.65	4.85	0.17	3.10	0.00864	-0.03466	0.04917	0.00057	0.00022	-0.00376
IR										
Cold	586.02	586.78	321.27	239.21	-0.10137	0.46481	-0.6413	-0.00112	-0.00009	0.02704
Warm	587.14	588.04	322.70	238.15	-0.11278	0.5179	-0.71455	-0.00213	-0.00013	0.02438
Difference	1.12	1.25	1.43	-1.06	-0.01141	0.05309	-0.07325	-0.00101	-0.00004	-0.00266

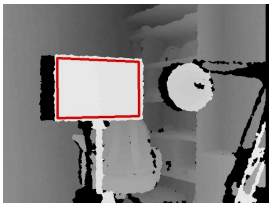
states, two changes could be observed for the warm state: the pictures were more blurred and the poster appeared slightly magnified. Although the cropped areas of the close-up views in Fig. 1 had the same size and pixel-position for both heat states, a shift of the letters and a loss of sharpness can be noticed.

Comparison Based on Dense Optical Flow. A dense optical flow approach [10] has been applied to image pairs of the poster taken in both heat states. For visualization, the color-code proposed in [12] was used (cf. Fig. 2(b)). The magnitude of the optical flow was small near the image center and large at its margins (cf. Fig. 2(c) and 2(d)). The maximal flow was 6.7 pixels for the RGB and 1.8 pixels (note the smaller IR image resolution) for the IR camera. Regarding the color distribution and the magnitude, the observations can be interpreted as a zoom-in effect.

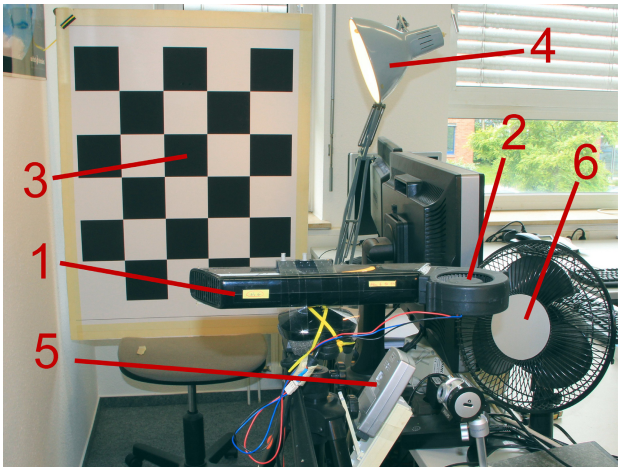
Comparison Based on Calibration Parameters. If the previous observations are caused by a thermally dependent deformation or shift of the optical lens system, we expect a change in the parameters of the camera model and the lens distortion model. A calibration plane with checker pattern was placed in front of the Kinect at 23 different orientations and captured simultaneously by



(a) RGB image.



(b) Depth image.



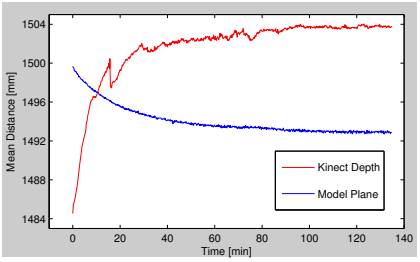
(c) Experimental setup in a static scene.

Fig. 3. The region of interest (red box) within the RGB camera image (converted to grayscale) and the depth image are shown in (a) and (b). The experimental setup consists of the following items: Kinect (1), mounted fan (2), large planar checkerboard (3), fluorescent lamp (4), thermometer (5), table fan (6).

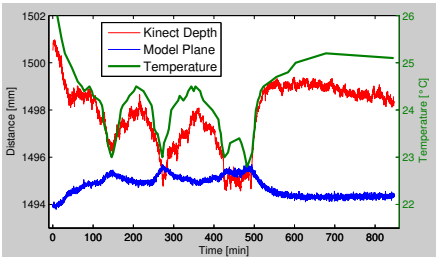
both cameras. Repeating this for both heat states, we got two sets of images for each camera. The IR projector was blocked to prevent detection errors of checkerboard corners by the structured light. Each set of images was used to determine the parameters using the MATLAB camera calibration toolbox [15]. Table 1 shows the results. For both cameras the focal length increases in the warm state, which is consistent with the zoom-in effect revealed by the optical flow approach. No significant changes of the back-projection error [14] could be observed. Thus we can assume that the employed camera model fits well for both heat states. Note that both cameras are sensitive to thermal changes, especially the RGB camera. This is important for calibration approaches that involve the RGB camera within their range calibration like in [1]. We conclude the following rule: *Camera calibration and subsequent measurements should be performed at the same thermal conditions.*

4 Thermal Influence on Range Measurement

The experimental evaluation of thermal influences on range measurements used mean distances to a checkerboard of size 0.75×1.0 m at a distance of 1.5 m placed in front of a Kinect mounted with a fan, cf. Fig. 3(c). The mean distances were determined in two ways. Both Kinect cameras were calibrated in advance, using sets of images generated in the cold state, to obtain intrinsic as well as radial and tangential distortion parameters. Furthermore, a stereo calibration of the cameras was performed simultaneously as described in [13].



(a) Evaluation after power-on.



(b) Influence of temperature variations.

Power-On	Start	End	Diff.	90%
Kinect Depth[mm]	1484.54	1504.03	19.49	41.0 min
Model Plane[mm]	1499.68	1492.71	-6.97	62.9 min

Temp. influence	Min	Max	Diff.	mm/°C
Temperature [°C]	22.8	26.2	3.40	-
Kinect Depth[mm]	1494.57	1500.96	6.39	1.88
Model Plane[mm]	1493.79	1495.69	1.90	-0.56

Fig. 4. Plots of $\mathcal{D}^{\mathcal{D}}$ (red) and $\mathcal{D}^{\mathcal{MP}}$ (blue) as well as the environmental temperature (green) over time. The erratic change in (a) at minute 15.9 will be discussed in Sec. 4.5. Each table beneath the plots summarizes quantitative details.

Mean Distance Calculation Based on the Model Plane. The mean distance $\mathcal{D}^{\mathcal{MP}}$ from the RGB camera to the checkerboard is determined as follows:

- Detect the checkerboard within the current image (see the red box in Fig. 3(a) where the region of interest (ROI_{RGB}) is marked).
- Construct a 3D-model of the checkerboard (denoted as model plane).
- Calculate the 3D rotation matrix $\mathbf{R}_m$ and the translation vector $\mathbf{t}_m$ of the model plane relatively to the camera, so that the back-projection error is minimized (see [14] for details).
- Define 3D-rays from the camera center $\mathbf{c}_0$ to every pixel within the ROI_{RGB}.
- Finally, calculate the mean of all single distances between $\mathbf{c}_0$ and the intersection point of each 3D-ray with the model plane.

Mean Distance Calculation Based on Depth. The checker pattern was not visible in the depth image. Thus the checkerboard model, whose position in the coordinate frame of the RGB camera was given by $\mathbf{R}_m$ and $\mathbf{t}_m$, was transformed to the coordinate frame of the IR camera using the rotation matrix $\mathbf{R}_s$ and the translation vector $\mathbf{t}_s$ between the frames of both cameras available from the stereo calibration. Then the checkerboard model was projected onto the image plane of the IR camera to get the ROI_{IR}. According to [1], there is a pure shift between the IR and the depth image of three pixels in x- and y- direction. Thus, we just shifted the ROI_{IR} to get the ROI_D within the depth image (cf. Fig. 3(b)). For each pixel within the ROI_D we calculated the corresponding 3D-point in space using the OpenNI framework [17]. Finally, the mean of all magnitudes of these 3D-points is the desired mean distance $\mathcal{D}^{\mathcal{D}}$ based on depth data.

4.1 Tracking Distances After Power-On

The Kinect was disconnected for three hours to cool it down to the room temperature of 27.7°C before starting the tracking of both distances during the warm-up

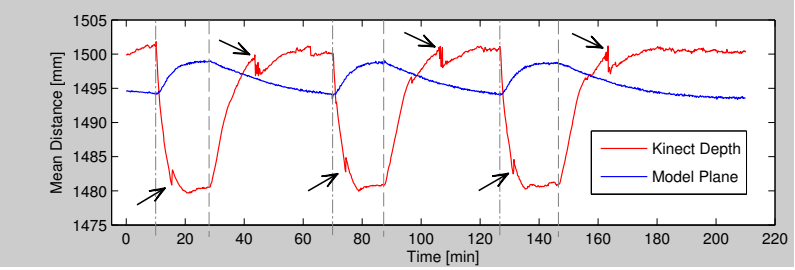


Fig. 5. Track of distances in alternating phases with and without cooling using a mounted fan. Dot-dashed lines mark time points of fan activation, dashed lines indicate deactivations. Arrows mark time points where erratic distance changes occurred.

phase. Fig. 4(a) shows the plots within 135 minutes. A decrease of -6.97 mm was observed for $\mathcal{D}^{\mathcal{MP}}$. 90% thereof took 60 minutes of warm-up time. We suppose that this change is a direct consequence of the zoom-in effect (cf. Sec. 3): the checkerboard appears to have moved towards the camera and thus the measurement of $\mathcal{D}^{\mathcal{MP}}$ outputs a smaller checkerboard distance, although the scene was not changed. The measured change in $\mathcal{D}^{\mathcal{D}}$ was an increase of 19.49 mm. 90% thereof occurred within 41 minutes. Note that a similar behavior was reported for the SR2 time-of-flight camera in [8], where the measured distance also increased approx. 12 mm within the first minutes after the sensor activation. In a second experiment at a room temperature of 22.5°C we could observe comparable results. The determined changes were 19.68 mm (90% within 42 minutes) for $\mathcal{D}^{\mathcal{D}}$ and -6.12 mm (90% within 56 minutes) for $\mathcal{D}^{\mathcal{MP}}$. Since the effect of lens deformation of the IR camera was smaller compared to the RGB camera (cf. Sec. 3), the change in $\mathcal{D}^{\mathcal{D}}$ is not explainable only by this effect. However, for typical indoor scenarios we can state the rule: *A warm-up time of up to 60 minutes is necessary to reach stable measurement conditions.*

4.2 Distance Changes between Thermal States

The following experiment was performed at a constant room temperature of 27.5°C. We used the mounted fan and alternated between phases with and without fan cooling to change the heat state and tracked again the distances (cf. Fig. 5). The experiment was repeated three times. The cool down was completed within 10 and 18 minutes for $\mathcal{D}^{\mathcal{D}}$ and $\mathcal{D}^{\mathcal{MP}}$, respectively. The longest warm-up period was finished after 61 minutes regarding $\mathcal{D}^{\mathcal{MP}}$. This is comparable to the results in Sec. 4.1. Regarding $\mathcal{D}^{\mathcal{D}}$, the warm-up took 33 minutes. The ventilation had a strong impact on the measurements, although the room temperature was stable. $\mathcal{D}^{\mathcal{MP}}$ increased by 5.67 mm while $\mathcal{D}^{\mathcal{D}}$ decreased by -22.76 mm during fan cooling.

The arrows in Fig. 5 mark the points in time where erratic changes between 2 and 4 mm occurred in the plot of $\mathcal{D}^{\mathcal{D}}$. At the same time a rapid and global change in the values of the corresponding depth image could be noticed. Erratic changes

Table 2. Summary of the measured values and the noise by the standard deviation σ of the distance measurements in both heat states in stable environmental conditions

Noise measurement (23.7°C)	Min	Max	Diff.	σ
Cold state				
Kinect Depth[mm]	1457.07	1457.58	0.51	0.103
Model Plane[mm]	1487.49	1487.81	0.32	0.058
Warm state				
Kinect Depth[mm]	1474.12	1474.86	0.74	0.134
Model Plane[mm]	1482.50	1482.72	0.22	0.038

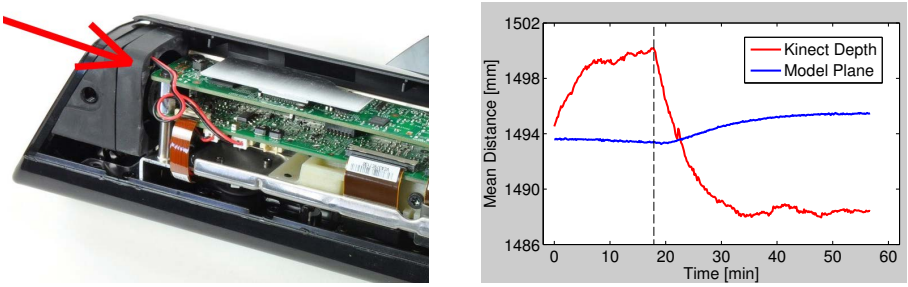
occurred in situations of rapid temperature variations of Kinect's internal components. In a decreasing phase they performed an upward correction and vice versa.

4.3 Noise in Distance Measurements

The noise in the distance measurements $\mathcal{D}^{\mathcal{D}}$ and $\mathcal{D}^{\mathcal{MP}}$ within the known static scene was evaluated by investigating the standard deviation. The room temperature was 23.7°C during the complete experiment. The standard deviation was calculated from over 660 measurement points within 15 minutes for each thermal state. The evaluation of the cold state was performed after a cool-down time of 30 minutes using the mounted fan. The warm state was evaluated after 60 minutes warm-up time without fan activity. Table 2 compiles the numerical results. Regarding the transition from the cold to the warm state, noise reduction of 34.5% for $\mathcal{D}^{\mathcal{MP}}$ was determined. We suppose that this is due to the blur observed in section 3, that is comparable to a low-pass filter and that reduces the noise in the determination of the model plane within the RGB image. For $\mathcal{D}^{\mathcal{D}}$, the noise has increased by 30.1%.

4.4 Distance Changes Caused by Kinect's Internal Fan

Within the Kinect's body a small integrated fan (cf. Fig. 6(a)) is used to prevent damage of internal components by overheating. There is no interface to control the internal fan but it is activated automatically if the temperature exceed a certain threshold. In our experiments the fan was activated if the environmental temperature exceed 30.5°C. This point in time was reached at minute 18 and is marked by the dashed line in the plot shown in Fig. 6(b). During the experiment the temperature increased from 30°C to 31°C. The impact on the measurements after this activation was a change of -12.19 mm and 2.38 mm regarding $\mathcal{D}^{\mathcal{D}}$ and $\mathcal{D}^{\mathcal{MP}}$, respectively. Since the internal fan is less powerful, the impact is smaller than in the experiment in section 4.2, but it is significant anyway. Thus we derive the following rule: *It is necessary to keep the fan activation in mind if operating the Kinect at an environmental temperature near the activation threshold.*



(a) Internal fan (www.geek.com) (b) Measurements after fan activation

Fig. 6. The automatic activation of the internal fan (a) and its impact on distance measurements (b) at minute 18 (dashed line). The erratic change at minute 22 is discussed in Sec. 4.5.

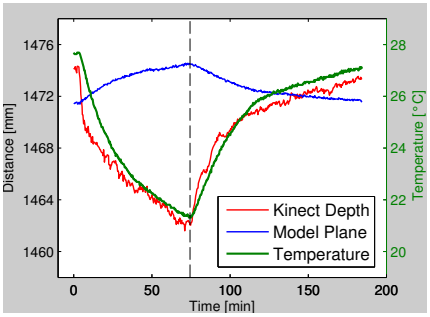
4.5 Distance Changes Caused by Air Draft

During this experiment the room temperature was constantly 27.5°C. To simulate air draft, we used a standard table fan (cf. Fig. 3(c)). It was blowing sideways at the Kinect in the warm state at a distance of 30 cm for only 10 seconds. The changes in $\mathcal{D}^{\mathcal{MP}}$ were insignificant. However, regarding $\mathcal{D}^{\mathcal{D}}$, even this very short period of time caused a change of -3.22, -3.14, and -3.18 mm for three repetitions of this experiment. The necessary warm-up time to reach the initial distance values took between 5 and 6 minutes in each repetition. Due to the described high sensitivity, the following rule can be established: *Try to avoid air draft while using the Kinect in the warm state.*

4.6 Tracking Temperature and Distance Changes

We demonstrate Kinect's sensitivity to naturally occurring temperature changes in an everyday scenario. The investigations took place in a room of size 5×3×3 m with a digital thermometer mounted 10 cm beneath the Kinect (cf. Fig 3(c)). The door and windows were closed before the experiment started. At the beginning, the room temperature was 26.2°C and we opened a window. Air draft was prevented by blinding the window and keeping the door closed. Weather changes (mix of sun and rain) caused indoor temperature variations. At minute 497 the window was closed and the room temperature increased. In Fig. 4(b), a negative correlation between temperature (green) and $\mathcal{D}^{\mathcal{MP}}$ (blue) can be observed. This conforms to the zoom-in effect (cf. Sec. 3). In contrast to that, a positive correlation between temperature and $\mathcal{D}^{\mathcal{D}}$ (red) can be noticed. The maximal temperature difference was 3.4°C. This caused a maximal change of -1.9 mm in $\mathcal{D}^{\mathcal{MP}}$ and 6.39 mm in $\mathcal{D}^{\mathcal{D}}$, what means a change of -0.56 and 1.88 mm per 1°C. The latter compares well with an increase by 8 mm per 1°C as reported for the SR2 time-of-flight camera [8]. The table in Fig. 4(b) shows quantitative details.

(a) Temperature sensor (left) attached at Kinect's ventilation slot.



(b) Plot of temperature and distances before and after internal fan activation.

Temp. influence	Min	Max	Diff.	mm/°C	Correl. with Temp.
Temperature [°C]	21.31	27.69	6.38		
Kinect Depth[mm]	1461.60	1474.32	12.72	1.99	0.96852
Model Plane[mm]	1471.39	1474.56	3.17	-0.50	-0.99562

Fig. 7. The plot (b) illustrates the dependancy of the measured distances to the temperature determined by the sensor mounted at Kinect's ventilation slot (a). The table summarizes quantitative results.

4.7 Correlation between Temperature and Measured Distances

In this experiment we confirm the results of Sec. 4.6 within more controlled variations of the temperature. We used a small temperature sensor that was mounted directly at Kinect's ventilation slot, cf. Fig. 7(a). In contrast to the measurements performed by the room temperature sensor (cf. Fig. 3(c)), the measured temperature corresponds more to Kinect's internal temperature, since the warm air directly passes the sensor at the ventilation slot. Note, that there was no fan activity. At the beginning of the experiment the Kinect was warmed up, the environmental temperature was 22.0°C and the temperature at the ventilation slot was 27.1°C. The outdoor temperature of 11.0°C was significantly lower. During minutes 5 to 75 the window was opened to cool down the room. At minute 75 the window was closed and the room was warmed up again. Fig. 7(b) shows the plot of the temperature and the measured distances. The dashed line indicates the point in time where the window was closed.

A strong correlation could be confirmed by a correlation coefficient of 0.96852 between temperature and $\mathcal{D}^{\mathcal{D}}$, while a negative correlation of -0.99562 could be determined between temperature and $\mathcal{D}^{\mathcal{MP}}$. The maximal temperature variation was 6.34°C. A change of 1.99 mm and -0.5 mm per 1°C could be observed for $\mathcal{D}^{\mathcal{D}}$ and $\mathcal{D}^{\mathcal{MP}}$, respectively. This is comparable to the results in Sec. 4.6. Quantitative results are summarized in the table in Fig. 7.

4.8 Distance Changes after Stand-By and USB Disconnection

In these experiments the environmental conditions were constant (constant temperature, no air draft, no fan, closed door and windows). Before starting, the

Kinect was on-line (streaming depth and RGB images) to warm up. The amount of changes caused by two types of working interruptions within a long-term use of the Kinect will be examined in the following. This investigation has practical relevance because such interruptions are typical while working with the Kinect or developing software with interleaved testing phases.

Type 1: Disconnection from USB or Power. Some pretests revealed that the power supply as well as the USB disconnection produced the same results. This is traceable since the Kinect stopped power consumption if USB was disconnected. Thus, there was no heating of internal components in both cases. The Kinect was disconnected for 2, 5, and 10 minutes resulting in a change of -6.12, -10.38, and -14.66 mm in $\mathcal{D}^{\mathcal{D}}$. Regarding $\mathcal{D}^{\mathcal{MP}}$, 0.32, 1.21, and 2.36 mm were observed. These small values are valid since $\mathcal{D}^{\mathcal{D}}$ and $\mathcal{D}^{\mathcal{MP}}$ base on mean distances with a low noise level below $\sigma_{\mathcal{D}^{\mathcal{D}}} = 0.14$ and $\sigma_{\mathcal{D}^{\mathcal{MP}}} = 0.06$ mm in stable environmental conditions in both heat states, cf. Sec. 4.3. After 10 minutes of disconnection, 18 and 57 minutes were needed to reach stable values again for $\mathcal{D}^{\mathcal{D}}$ and $\mathcal{D}^{\mathcal{MP}}$, respectively.

Type 2: Stand-By Mode. If the Kinect was not streaming any data (OpenNI XnSensorServer is shut down) but connected to USB and power, it stayed in a stand-by mode (green LED was still flashing). Regarding an application using the Kinect, this is the typical mode between the application's termination and the next access to the Kinect. After warm-up, the stand-by mode was entered for 15 minutes before returning to the on-line mode. The changes in $\mathcal{D}^{\mathcal{D}}$ and $\mathcal{D}^{\mathcal{MP}}$ were -3.09 and 0.73 mm. To determine maximal changes, the stand-by mode was entered for 10 hours. We determined -5.95 and 1.67 mm regarding $\mathcal{D}^{\mathcal{D}}$ and $\mathcal{D}^{\mathcal{MP}}$. This corresponds to approx. 25% of the changes compared to the power-on and the fan cooling scenario (cf. Sec. 4.1 and 4.5). This smaller change is due to Kinect's power consumption, that was comparable in the stand-by and the on-line mode. This prevented a cooling of Kinect's internal components. Thus the last rule is: *Try to keep the Kinect always in the on-line mode. If this is not possible, leaving it in the stand-by mode is the best alternative.*

5 Conclusion

The analysis of several combinations of environmental and thermal conditions (stable and varying temperature, air draft, usage of fans, power disconnection etc.) has shown that they have a strong impact on the Kinect's output. Based on the findings temperature-related rules have been established which may help to reduce errors in the calibration and measurement process of the Kinect. Future work will include finding a model which describes the depth error in relation to the temperature, and developing a correction function based on this model.

References

1. Smisek, J., Jancosek, M., Pajdla, T.: 3D with Kinect. In: International Conference on Computer Vision Workshops (ICCV Workshops, IEEE), pp. 1154–1160 (2011)
2. Zhang, C., Zhang, Z.: Calibration between Depth and Color Sensors for Commodity Depth Cameras. In: International Workshop on Hot Topics in 3D, in conjunction with ICME 2011, Barcelona, Spain (2011)

3. Stowers, J., Hayes, M., Bainbridge-Smith, A.: Quadrotor Helicopter Flight Control Using Hough Transform and Depth Map from a Microsoft Kinect Sensor. In: Conference on Machine Vision Applications, MVA 2011, Nara, Japan (2011)
4. Tran, J., Ufkes, A., Fiala, M., Ferworn, A.: Evaluation of an Inexpensive Depth Camera for Passive In-Home Fall Risk Assessment. In: International Symposium on Safety, Security, and Rescue Robotics (SSRR), pp. 161–166. IEEE (2011)
5. Khoshelham, K., Elberink, S.O.: Accuracy and resolution of Kinect depth data for indoor mapping applications. Sensors: Journal on the Science and Technology of Sensors and Biosensors, 1437–1454 (2012)
6. Berger, K., Ruhl, K., Brümmer, C., Schröder, Y., Scholz, A., Magnor, M.: Markerless Motion Capture using multiple Color-Depth Sensors. In: Proc. Vision, Modeling and Visualization (VMV), pp. 317–324 (2011)
7. Weinmann, M., Wursthorn, S., Jutzi, B.: Semi-automatic image-based coregistration of range imaging data with different characteristics. PIA11 - Photogrammetric Image Analysis. The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Science 38(3)/W22, 119–124 (2011)
8. Kahlmann, T., Remondino, F., Ingensand, H.: Calibration for increased accuracy of the range imaging camera Swissranger. In: ISPRS Commission V Symposium Image Engineering and Vision Metrology, Desden, vol. XXXVI, Part 5, pp. 25–27 (2006)
9. Bower, S., Kou, J., Saylor, J.: A method for the temperature calibration of an infrared camera using water as a radiative source. Review of Scientific Instruments 80(9) (2009)
10. Bruhn, A., Weickert, J., Schnoerr, C.: Lucas/Kanade meets Horn/Schunk: combining local and global optical flow methods. International Journal of Computer Vision (IJCV) 61(3), 211–231 (2005)
11. Herrera, C., Kannala, J., Heikkilä, J.: Accurate and Practical Calibration of a Depth and Color Camera Pair. In: Real, P., Diaz-Pernil, D., Molina-Abril, H., Berciano, A., Kropatsch, W. (eds.) CAIP 2011, Part II. LNCS, vol. 6855, pp. 437–445. Springer, Heidelberg (2011)
12. Baker, S., Scharstein, D., Lewis, J.P., Roth, S., Black, M.J., Szeliski, R.: A Database and Evaluation Methodology for Optical Flow. In: Proc. Eleventh IEEE International Conference on Computer Vision (ICCV 2007), Rio de Janeiro, Brazil (2007)
13. Hartley, R., Zisserman, A.: Multiple View Geometry in Computer Vision, 2nd edn. Cambridge University Press, New York (2003)
14. Zhang, Z.: A Flexible New Technique for Camera Calibration. Technical Report, MSR-TR-98-7, Microsoft Research Microsoft Corporation, One Microsoft Way, Redmond, WA 98052 (1998)
15. Bouguet, J.Y.: Camera Calibration Toolbox for Matlab, `http://www.vision.caltech.edu/bouguetj/calib_doc/index.html`
16. Microsoft Corporation, Kinect for Xbox 360, `http://www.xbox.com/de-DE/Kinect`
17. OpenNI - Open Natural Interaction, `http://openni.org/`
18. Hothmer, J.: Possibilities and limitations for elimination of distortion in aerial photographs. Photogrammetric Record 2(12), 426–445 (1958)
19. Smith, M.J., Cope, E.: The effect of temperature variation on single-lens-reflex digital camera calibration parameters. International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences 28 (2010)
20. Podbreznik, P., Potocnik, B.: Infence of Temperature Variations on Calibrated Cameras. World Academy of Science, Engineering and Technology, WASET 41 (2008)

A Prior-Based Approach to 3D Face Reconstruction Using Depth Images

Donny Tytgat, Sammy Lievens, and Erwin Six

Alcatel-Lucent Bell Labs,
Copernicuslaan 50, 2018 Antwerp, Belgium
{donny.tytgat,sammy.lievens,erwin.six}@alcatel-lucent.com

Abstract. We present a system for 3D face reconstruction that is based on live depth information combined with face priors. The system includes a stereo matching method that employs prior information for limiting the search space and introduces an offline scanned 3D model that is animated by means of 2D morphing techniques in order to match the live facial expressions. The resulting live- and synthetic depth images are combined and a live 3D mesh is generated. Computational aspects are taken into account for every part of the system, enabling a live face reconstruction system with the potential for real-time execution.

Keywords: Depth image, 3D, Face reconstruction, Real-time.

1 Introduction

A live 3D face reconstruction system can offer new video communication solutions by decoupling the capturing device from the visualized data. Virtual cameras can be placed anywhere around the face, enabling a more compact and clear depiction of the conversation dynamics or establishing better gaze alignment. In addition, the reconstructed model can be visualized on 3D displays.

A system is presented that employs prior information, both generic and personalized, in order to create a 3D face reconstruction system that could be applied in a real-time video communication system. The system uses depth images as an intermediary format for data aggregation. A prior-aided stereo matching method is shown that uses a sparse set of correspondences in order to guide the dense stereo matching process. This is followed by a method that employs 2D morphing techniques for generating an approximation of an animated, personalized 3D model that is represented by multiple depth images. A surface reconstruction method is furthermore presented that produces a triangulated model from an arbitrary number of depth images. These three components are combined into a system that can produce a live 3D face mesh.

2 Related Work

3D face reconstruction is an active field in the domains of computer vision and computer graphics. Reconstruction from a single image is in its most generic form

X. Jiang et al. (Eds.): WDIA 2012, LNCS 7854, pp. 32–41, 2013.

an ill-posed problem. As such prior information is required in order to solve it. A survey of this field is discussed in [8]. Prior information such as light sources (shape from shading) and geometric/albedo models (analysis by synthesis) are often used. Results are however rarely satisfactory for realistic rendering due to the limited live information which poses a strict dependence on the priors.

A different approach involves the use of alternative inputs such as depth sensors in order to fit geometric models to the live data. A Kinect sensor is used in [7] for fitting a user-specific expression model to the data (their goal is expression transfer however, and not face reconstruction). This can offer satisfying results, however the approach is limited by the expressiveness of the model.

The use of more camera viewpoints can also increase the confidence of the results. Sparse stereo information is used in [6] for fitting a generic 3D model to the live data. Geometric details are limited due to the sparsity of the features in combination with a generic model. Models that are reconstructed using a multitude of cameras such as in [1] can offer excellent reconstruction quality at the expense of applicability and computational performance. Such methods generally require fewer priors and are thus less sensitive to relaxations of the contextual assumptions.

The hybrid approach that is presented here attempts to find a compromise between the convenient use of live data and flexible use of prior information.

3 Prior-Aided Stereo Matching

For the stereo matching process, prior information is ingested in the form of AAM (Active Appearance Models [3]) features along with a linear interpolation model on top of these features. AAM is applied to both rectified stereo images, thus introducing a sparse correspondence set S of features between the two images. In total 68 correspondences are used. These are triangulated in accordance with the face anatomy. The result is a lower- and upper bound (dmin; dmax) for the disparity search range in each triangle. In order to facilitate for small errors in the correspondence set, a fixed value δ is used to widen the search range. The dense stereo matching algorithm is based on [9] and employs a locally adaptive window in order to accommodate to the local texture variations. Search ranges are limited using the local lower- and upper bounds. Integral images are used in order to accelerate the computation of the similarity score over variable rectangular window shapes.

The traditional winner-takes-all approach for selecting the final disparity value is replaced by an iterative approach where a per-pixel Gaussian Mixture Model (GMM) is used for including temporal- and neighborhood influences. First of all the temporal influence is modeled for each pixel by adding the corresponding GMM from the previous time instance. A Gaussian that models the stereo matching result is then added, and is followed by a number of iterations that consist out of combining each GMM with those of its neighbors and reducing these GMMs in order to limit the number of embedded Gaussians. This is illustrated in Figure 1. GMM reduction is done closed-form by means of least-squares parabolic fitting in the log-domain.

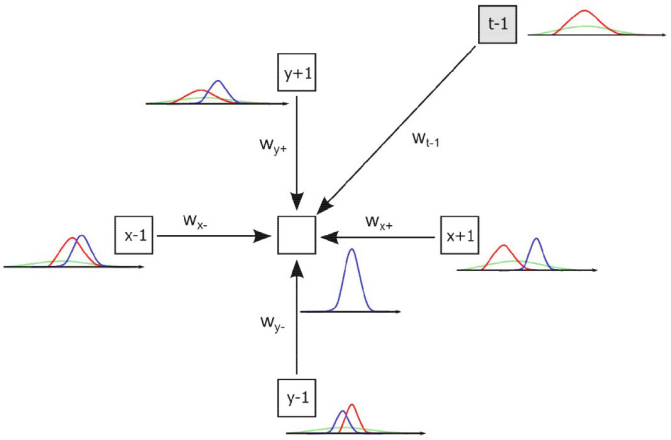
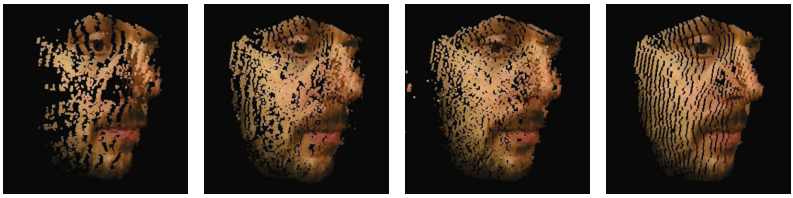


Fig. 1. Updating the per-pixel GMM

Figure 2 illustrates some of the results with varying stages of the method in use. The effective resolution of the shown stereo-matched face is about 110x130 (bounding box around the AAM features). The method, which uses the stereo images along with the accompanying AAM features as input, runs at 22 frames per second for the shown results on an Intel Core i7 2820QM processor.



(a) (b) (c) (d)

Fig. 2. Incrementally improving stereo matching results by (a) restricting computation to the facial region, (b) additionaly constraining the disparity domain based on the sparse AAM correspondence set and increasing precision to 1/2 pixel resolution, (c) adding the temporal factor and (d) also including the neighborhood influence.

4 3D Model Animation by 2D Morphing

The live stereo matched depth image from the previous section is not enough to reconstruct the complete face due to occlusions and incomplete stereo matched data. To this purpose, a personalized 3D scan is available as a prior and needs to be adapted in order to reflect the actual facial expression state. The same AAM features as used in the previous section are applied here in order to transfer the

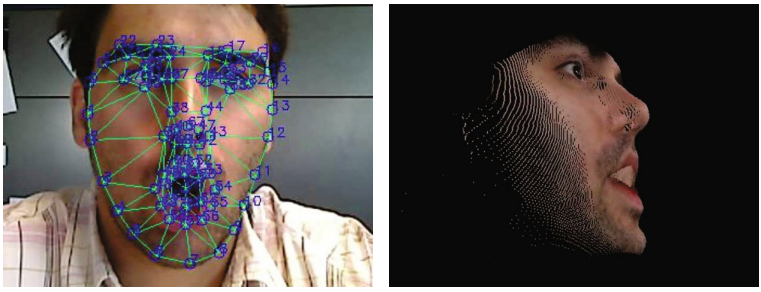
live expressions to the 3D model. The resulting animated model consists out of a number of depth images rather than a mesh model. As will become clear in the next section, this is not necessarily a disadvantage.

The model animation consists out of the following steps. First of all, the live AAM features need to be transferred to the 3D model; in other words the 3D coordinates of the AAM features are estimated. Once this is done, the 3D data is adapted in order to adhere to these new AAM coordinates. This is done in the 2D domain by projecting the 3D model a number of times, and doing 2D depth morphing from the original AAM coordinates to the new ones.

A common reference frame is constructed between the live- and offine data by means of matching the virtual camera to the live camera. The intrinsic camera parameters are directly transferred from the live camera whereas the extrinsic camera parameters are estimated by aligning three selected AAM points. Now the AAM features of the live data can be transferred by simply copying the 2D coordinates. The 3D coordinates of these transferred AAM points are then found by estimating the depth for the points, and back-projecting them back into 3D space. In this case the depth of each AAM point is taken to be the same as that of the relevant original AAM point.

Now that the new 3D coordinates of the AAM features are known, we can transform the model by employing 2D morphing on an arbitrary number of generated depth images. The generated images should cover the complete face. The data is morphed by using a coarse mesh that spans the head and which is a superset of the AAM-based mesh. The AAM points are placed at their actual location, and the other points are linearly morphed in relation to the mesh triangle they belong to.

Figure 3 shows an input frame that has been augmented by AAM features and the triangular model, and a morphed 2D+depth map that has been back-projected. Note the coarse animation of the mouth (the mouth is closed in the source 3D model).



(a) AAM model on live data (b) Animated model (1 view)

Fig. 3. 3D model animation

5 3D Surface Reconstruction

The previous two sections discussed ways of generating depth images of the model that is to be reconstructed. These are essentially point clouds (the associated calibration matrices are available), and can thus not be readily rendered in a flexible manner. The generation of a 3D surface model will remedy this.

The approach is based on [4] where an implicit surface function called the truncated signed distance function (TSDF) is constructed using a number of range images. This TSDF contains a signed distance metric d and a reliability value r for each sample. The distance metric indicates the nearest distance to the surface that is reconstructed; the sign defining whether it is located inside or outside of the object. In order to make the method scalable, an adaptive octree is used for sampling the implicit function. An additional iterative filter on the implicit surface function furthermore refines the results. This allows for a high quality surface reconstruction combined with real-time computational performance.

At first, the considered 3D space is uniformly sampled at a low sampling rate. This sampling rate is chosen in function of the minimum object size one wants to reconstruct. The diagonal of the cube in between samples cannot be larger than the diagonal of the minimal inner sphere of the reconstructed object. The sampling space is then refined in an adaptive manner by evaluating the implicit surface function at each sample location. A cube is refined when it contains an edge in which the TSDF of the endpoints have opposite signs. This is done until the maximum octree depth is reached. As this process does not necessarily produce a uniform sampling frequency along the surface boundaries, an additional post-processing step is needed for refining the octree. Note that the maximum octree depth introduces an implicit scalability parameter for the method. By increasing the value one gets a higher detailed model at the expense of computational resources.

The TSDF at a certain 3D point is estimated by projecting the point to each depth image, and selecting the one with the best score. This score is based on the distance and its reliability. Note that the reliability of the measurement is decreased in relation with the distance inside the object. Indeed, when a measurement indicates that a point is behind the surface, it does not necessarily mean that the point is actually inside the object; the object can merely be occluding a free point in space.

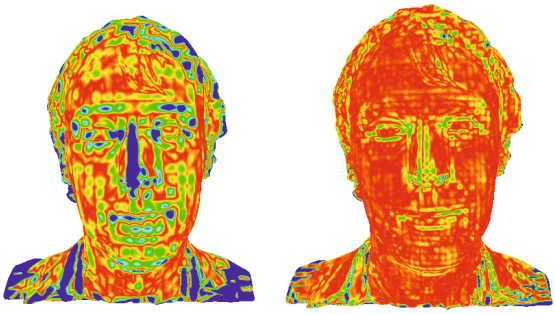
A filtering phase is introduced on the TSDF in order to reduce the influence of noise or missing values in the depth images. The filter models the neighboring sample influence on the assumption that the implicit surface is a plane:

$$D'(p) = R(p)^{\alpha} D(p) + (1 - R(p)^{\alpha}) D_n(p) \tag{1}$$

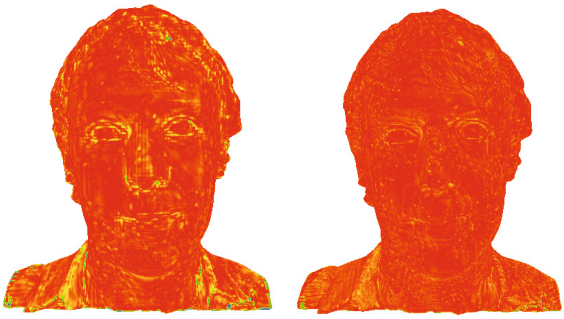
$$D_n(p) = \frac{\sum\limits_{(x,y)\in N(p)} (R(x) + R(y)) \dfrac{(D(x) + D(y))}{2}}{\sum\limits_{(x,y)\in N(p)} (R(x) + R(y))} \tag{2}$$

(a) Textured (b) Wireframe

Fig. 4. 3D surface reconstruction



(a) Level 5 (b) Level 6

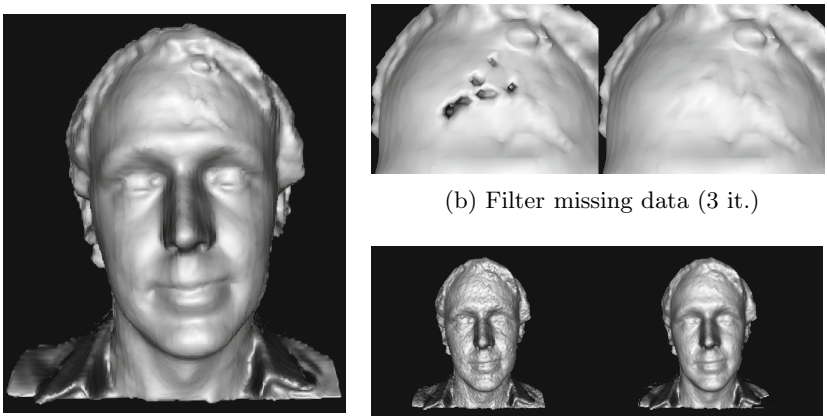


(c) Level 7 (d) Level 8

Fig. 5. Surface reconstruction error - ground truth annotation. Red: 0; blue: $\geq$ 2mm.

where $D(x)$ and $R(x)$ are the distance and reliability components of the TSDF, α is a parameter to control the neighbor influence and $N(x)$ is the set of valid neighboring pairs in each dimension ($max(\|N(x)\|) = 3$). $R'(p)$ is calculated in a similar way with $R_n(p)$ being the average reliability of the valid neighbors.

The last step involves the generation of an explicit surface from the implicit model. This is done by generating a triangle mesh using the Marching Cubes algorithm [5].



(a) Original

(b) Filter missing data (3 it.)

(c) Filter Gaussian noise (10 it.)

Fig. 6. Implicit surface filter

Table 1. Surface reconstruction statistics

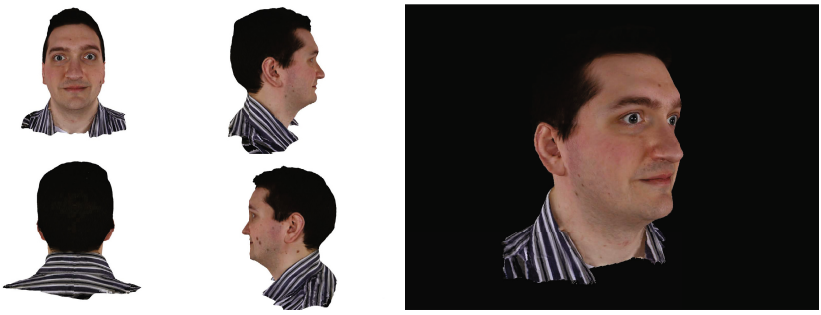
Model	Triangle count	Average error *mm (% of diag.)*	RMS *mm (% of diag.)*	Frames per second
Reference	876069	-	-	-
Level 5	6720	0.74692 (0.147%)	1.31329 (0.258%)	20.78
Level 6	28212	0.25322 (0.050%)	0.42232 (0.083%)	9.15
Level 7	117024	0.11558 (0.023%)	0.17305 (0.034%)	3.02
Level 8	484072	0.08722 (0.017%)	0.11662 (0.023%)	0.93

Figure 4 shows a closed mesh model that was reconstructed using this method. Five synthetically generated depth images have been used as an input and it consists out of 27.3k triangles. Table 1 contains the reconstruction errors and frame rate at 4 different maximum octree levels. No filtering was used for these measurements. Figure 5 shows the spatial distribution of the errors over the front of the model. Red indicates the smallest error; blue the largest (clipped to 2mm). The error statistics and images were created using Metro [2].

Filtering performance is illustrated in Figure 6 and considers two cases: gap filling and noise reduction. Gap filling is illustrated by adding unknowns to the depth values in the depth image. For demonstrating noise reduction, Gaussian white noise has been added to the depth image (the reliability noise is also modeled using Gaussian white noise on the reliability reduction that is proportional to the introduced error). A single depth image was used for reconstruction.

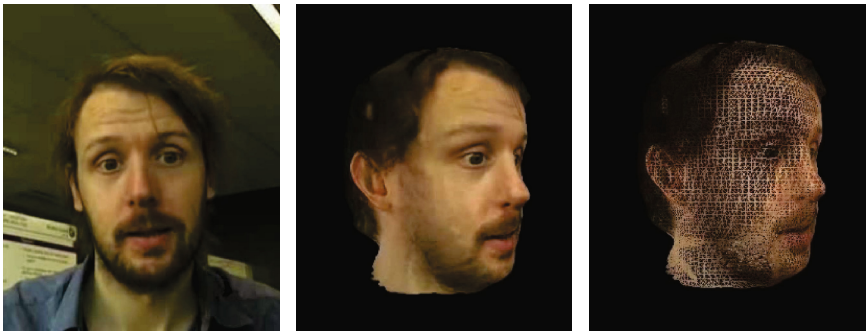
6 Live 3D Face Reconstruction

This final step brings the three previous sections together. As stated before, the goal involves the aggregation of live data and offine scanned data in order to produce a live 3D face mesh. The live depth images are produced by means of prior-aided stereo matching, the synthetic depth images are generated by 3D model animation using 2D morphing and all are aggregated using the 3D surface reconstruction method. Registration between the live- and synthetic depth images is done based on the 3D locations of the AAM points. One important aspect for the aggregation is the choice of reliability metric for the depth images. The metric that is used for the stereo matched depth images is related to the variance within each resulting GMM. A wide variance implies a less reliable value. For the synthetic depth images this metric is statically determined.



(a) Source textures (b) Resulting render

Fig. 7. Projective texturing



(a) Live input (stereo pair) (b) Textured mesh (c) Wireframe mesh

Fig. 8. Live 3D face reconstruction

Up to this point, the discussion has focused on the reconstruction of the model geometry. In order to produce a convincing render, one also needs to address the appearance however. With traditional texture mapping, one needs to assign texture coordinates to every vertex of the mesh. Due to the implicit nature on how the geometry is built, this can be quite a cumbersome process. Instead, a technique called 'projective texturing' is used. In projective texturing, one or more textures are projected on an arbitrary geometry. Note that one needs to know the camera (or projector) matrices in order to do this. Figure 7 illustrates this concept. The source textures are shown to the left, whereas the render is shown to the right. In the live 3D face reconstruction system, 5 textures are used that are sourced from the animated model and 1 texture is used that represents the live video feed.

A result that uses one live color- and depth image combined with 5 animated color- and depth images is shown in Figure 8. Due to the unavailability of ground truth data, only a qualitative assessment can be given. The method performs well in regions where the animated depth images coarsely agree with the stereo matched data. The linear model that is used for animating the 3D model has some deficiencies however, especially in the region of the cheeks. Despite this, the combination of a well chosen reliability metric and implicit surface filtering allows for a high quality 3D reconstruction using live data.

Real-time rates are not fully achieved for the complete system at the time of writing. This can be attributed to implementation details however, and not due to inherent computational constraints.

7 Conclusions

A method was presented that employs depth images from different sources in order to generate a live 3D mesh of a known face at potentially real-time rates. Prior information is used at different stages of the method in order to enhance quality and reinforce reliability. The generated mesh is the result of combining information from live stereo matched data and an offline scanned animated mesh in a way that is computationally tractable and resilient to errors.

Future work includes the replacement of the coarse animated mesh prior by a more accurate model; e.g. by means of a 3D morphable model in a face/expression space. We believe that the combination of such a model with real-time data could relieve this model from the disadvantages thereof. In addition, dynamically predicting the model reliability according to the current circumstances will also be investigated.

Acknowledgments. This research was carried out as part of the IBBT/IWT iCocoon project. This project is funded by the Interdisciplinary Research Institute IBBT and by the Agency for Innovation by Science and Technology (IWT-Flanders).

References

1. Beeler, T., Hahn, F., Bradley, D., Bickel, B., Beardsley, P., Gotsman, C., Sumner, R.W., Gross, M.: High-quality passive facial performance capture using anchor frames. ACM Trans. Graph. 30, 75:1–75:10 (2011), http://doi.acm.org/10.1145/2010324.1964970
2. Cignoni, P., Rocchini, C., Scopigno, R.: Metro: measuring error on simplified surfaces. Tech. rep., Paris, France (1996)
3. Cootes, T.F., Edwards, G.J., Taylor, C.J.: Active appearance models. IEEE Transactions on Pattern Analysis and Machine Intelligence 23(6), 681–685 (2001)
4. Curless, B., Levoy, M.: A volumetric method for building complex models from range images. In: Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1996, pp. 303–312. ACM, New York (1996), http://doi.acm.org/10.1145/237170.237269
5. Lorensen, W.E., Cline, H.E.: Marching cubes: A high resolution 3d surface construction algorithm. SIGGRAPH Comput. Graph. 21(4), 163–169 (1987), http://doi.acm.org/10.1145/37402.37422
6. Park, U., Jain, A.K.: 3d face reconstruction from stereo video. In: Proceedings of the the 3rd Canadian Conference on Computer and Robot Vision, CRV 2006, p. 41. IEEE Computer Society, Washington, DC (2006), http://dx.doi.org/10.1109/CRV.2006.1
7. Weise, T., Bouaziz, S., Li, H., Pauly, M.: Realtime performance-based facial animation. ACM Trans. Graph. 30(4), 77:1–77:10 (2011), http://doi.acm.org/10.1145/2010324.1964972
8. Widanagamaachchi, W.N., Dharmaratne, A.T.: 3d face reconstruction from 2d images. In: Proceedings of the 2008 Digital Image Computing: Techniques and Applications, DICTA 2008, pp. 365–371. IEEE Computer Society, Washington, DC (2008), http://dx.doi.org/10.1109/DICTA.2008.83
9. Zhang, K., Lu, J., Lafruit, G.: Scalable stereo matching with locally adaptive polygon approximation. In: ICIP, pp. 313–316 (2008)

View-Invariant Method for Calculating 2D Optical Strain

Matthew Shreve, Sergiy Fefilatyev, Nestor Bonilla,
Gerry Hernandez, Dmitry Goldgof, and Sudeep Sarkar

Department of Computer Science and Engineering
University of South Florida
Tampa, Florida
mshreve@mail.usf.edu

Abstract. Two-dimensional optical strain maps have been shown to be a useful feature that describes a bio-mechanical property of facial skin tissue during the non-rigid motion that occurs during facial expressions. In this paper, we propose a method for accurately estimating and modeling the three-dimensional strain impacted onto the face and demonstrate its robustness at different depth resolutions and views. Experimental results are given for a publically available dataset that contains high depth resolutions of facial expressions, as well as a new dataset collected using the Microsoft Kinect synchronized with two HD webcams.

Keywords: 3D, Optical Strain, Optical Flow.

1 Introduction

Optical strain has been shown to be an effective feature for several applications in expression spotting [1], biometrics [2], as well as medical analysis [3]. Due to the recent releases of several feasible solutions for real-time three-dimensional imaging, extending this method to take advantage of 3-D data has become practical and could potentially lead to improvements in each of these areas. In this work, we demonstrate a method for calculating the 3-D strain incurred on a subject's face based on the non-rigid facial motion observed during a facial expression.

Some key advantages of this method over traditional two-dimensional strain methods are the following: (i) horizontal motions that occur along the sides of the face are often projected as smaller displacements due to parallax projection. Our method of calculating three-dimensional strain has the advantage of reconstructing these vectors in order to represent a more accurate displacement; (ii) motion perpendicular to the camera axis is not factored in to two-dimensional strain maps, however this is captured with three-dimensional strain as an additional normal strain component.

The method is based on the observation that the captured depth images of the surface of the face exhibit 2-manifold qualities, i.e., local regions of the surface can be accurately estimated using two dimensional planar equations. Hence, we take advantage of this by estimating the three-dimensional correspondences using established two-dimensional optical flow methods.

X. Jiang et al. (Eds.): WDIA 2012, LNCS 7854, pp. 42–49, 2013.

Methods for calculating non-rigid three-dimensional disparity has found extensive application in the entertainment industry, where they are used to animate faces of humanoid avatars in movies and games. Most current systems require special makeup or markers [4-6] due to low texture variability of the skin. Such markers, however, provide only a limited number of anchor points and are not sufficient to capture fine expression details at all points on the face. Other approaches [7] use direct intensity of the skin to track the displacements between the frames by imposing constraints on non-rigid motion. Approaches for 3D reconstruction of the scene itself can be broadly categorized into three groups. The first group, *motion stereo*, requires a multi-view camera setup of a single scene, where 3D information is obtained through triangulation of 3D points from multiple view [7-10]. Most of the prior work for estimation of non-rigid 3-D disparity falls into this category. The second category, called *monocular sequence,* uses a single-camera setups, but exploits *a priori* knowledge about the reconstructed scene by developing a model that constrains and develops the observed 2D motion [11]. The last category, referred to as a *dynamic depth map* approach [12,13], assumes a monocular setup for both depth and color information using inexpensive sensors such as Microsoft Kinect. Our approach for 3D optical strain estimation is based on the monocular setup with direct intensity skin tracking.

2 3-D Optical Strain

2.1 Optical Flow

Optical Flow is a well-known motion estimation technique that has two constraints: (i) the smoothness constraint, i.e., points within a small region move at some level of uniformity, and (ii) the brightness constraint, i.e. the intensity of a point in the image does not change over time. Optical flow is typically represented by the following equation.

$$(\nabla \mathrm{I})^T \mathbf{p} + I_\mathrm{t} = 0, \tag{1}$$

where $I(x,y,t)$ is the image intensity as a spatial and temporal function, x and y are the image coordinates and t is time. ∇I and I_t are the spatial and temporal gradients of the intensity function. $\mathbf{p} = \left[p = \frac{dx}{dt}, q = \frac{dy}{dt}\right]^T$ denotes horizontal and vertical motion.

After experimenting with several versions of optical flow, we decided to use an implementation of the Horn-Schunck [14] method found in the computer vision toolbox for Matlab 2012. The Horn-Schunk method consists of re-writing equation (1) as a global energy function that is constrained by a smoothness parameter $\alpha \in (0,1]$, and is optimized over k iterations. In general a lower alpha allows for less smooth flow fields (good for small motion), while a larger alpha restricts neighboring motions to be more uniform (good for large motions). For our experiments we used a low value for the smoothness constraint ($\alpha = .05$) to allow for the small, non-rigid motion inherent with facial expressions, and chose an iteration count of k=200.

2.2 Optical Strain

Considering a three dimensional surface of a deformable object, its motion can be described by a three-dimensional displacement vector $\boldsymbol{u} = [u, v, w]^T$. Next, if we assume both a small region and small motion for a point P, we can define the strain tensor:

$$\varepsilon = \frac{1}{2}[\nabla \boldsymbol{u} + (\nabla \boldsymbol{u})^T], \tag{2}$$

or in an expanded form:

$$\varepsilon = \begin{bmatrix} \varepsilon_{xx} = \frac{\partial u}{\partial x} & \varepsilon_{yx} = \frac{1}{2}\left(\frac{\partial v}{\partial x} + \frac{\partial u}{\partial y}\right) & \varepsilon_{zx} = \frac{1}{2}\left(\frac{\partial u}{\partial z} + \frac{\partial w}{\partial x}\right) \\ \varepsilon_{xy} = \frac{1}{2}\left(\frac{\partial v}{\partial x} + \frac{\partial u}{\partial y}\right) & \varepsilon_{yy} = \frac{\partial v}{\partial y} & \varepsilon_{zy} = \frac{1}{2}\left(\frac{\partial w}{\partial y} + \frac{\partial v}{\partial z}\right) \\ \varepsilon_{xz} = \frac{1}{2}\left(\frac{\partial w}{\partial x} + \frac{\partial u}{\partial z}\right) & \varepsilon_{yz} = \frac{1}{2}\left(\frac{\partial w}{\partial y} + \frac{\partial v}{\partial z}\right) & \varepsilon_{zz} = \frac{\partial w}{\partial z} \end{bmatrix} \tag{3}$$

where $(\varepsilon_{xx}, \varepsilon_{yy,}, \varepsilon_{zz})$ are normal strain components, $(\varepsilon_{xy}, \varepsilon_{zy}, \varepsilon_{zx})$ are shear strain components, and u,v,w are the displacements in the x,y,z directions.

Since strain is defined with respect to the displacement vector (u,v,w) in continuous space, we make the following 2-D approximation from the optical flow data (p,q):

$$p = \frac{dx}{dt} \doteq \frac{\Delta x}{\Delta t} = \frac{u}{\Delta t}, u = p\Delta t, \tag{4}$$

$$q = \frac{dy}{dt} \doteq \frac{\Delta y}{\Delta t} = \frac{v}{\Delta t}, v = q\Delta t, \tag{5}$$

$$r = \frac{dz}{dt} \doteq \frac{\Delta z}{\Delta t} = \frac{w}{\Delta t}, w = r\Delta t. \tag{6}$$

where Δt is the elapsed time between two image frames.

If we compute the optical flow and strain using a fixed frame interval throughout a particular video sequence, we can treat Δt as a constant and estimate the partial derivatives as follows:

$$\frac{\partial u}{\partial x} = \frac{\partial p}{\partial x}\Delta t, \quad \frac{\partial u}{\partial y} = \frac{\partial p}{\partial y}\Delta t, \quad \frac{\partial u}{\partial z} = \frac{\partial p}{\partial z}\Delta t, \tag{7}$$

$$\frac{\partial v}{\partial x} = \frac{\partial q}{\partial x}\Delta t, \quad \frac{\partial v}{\partial y} = \frac{\partial q}{\partial y}\Delta t, \quad \frac{\partial v}{\partial z} = \frac{\partial q}{\partial z}\Delta t, \tag{8}$$

$$\frac{\partial w}{\partial x} = \frac{\partial r}{\partial x}\Delta t, \quad \frac{\partial w}{\partial y} = \frac{\partial r}{\partial y}\Delta t, \quad \frac{\partial w}{\partial z} = \frac{\partial r}{\partial z}\Delta t, \tag{9}$$

The above computation scheme can then be implemented by using any spatial derivate over a finite number of points such as the forward difference method, central difference method, or the Richardson extrapolation method. We chose the central difference method due to its accuracy and efficiency.

$$\frac{\partial u}{\partial x} = \frac{u(x+\Delta x)-u(x-\Delta x)}{2\Delta x} \doteq \frac{p(x+\Delta x)-p(x-\Delta x)}{2\Delta x} \tag{10}$$

$$\frac{\partial v}{\partial y} = \frac{v(y+\Delta y)-v(y-\Delta y)}{2\Delta y} \doteq \frac{q(y+\Delta y)-q(y-\Delta y)}{2\Delta y} \tag{11}$$

$$\frac{\partial w}{\partial z} = \frac{w(z+\Delta z)-w(z-\Delta z)}{2\Delta z} \doteq \frac{r(z+\Delta z)-r(z-\Delta z)}{2\Delta z} \tag{12}$$

where $(\Delta x, \Delta y, \Delta y)$ are preset distances of 2-3 pixels.

Under the uniform stress, large strain values correspond to low elastic moduli and vice versa. Therefore, elastograms based on the absolute strain value or relative strain ratio can be used to reveal underlying elastic property changes. For this purpose, we compute a strain magnitude as follows:

$$\varepsilon_m = \sqrt{\varepsilon_{xx}^2 + \varepsilon_{yy}^2 + \varepsilon_{zz} + \varepsilon_{xy}^2 + \varepsilon_{yx}^2 + \varepsilon_{zx}^2 + \varepsilon_{yz}^2} \ . \tag{13}$$

3 Results

3.1 Feasibility at Multiple Depth Resolutions

In order to test the feasibility getting useful strain calculations at multiple depth resolutions, we developed an experiment that subsamples high resolution depth data at different rates. This was done on a publically available 3D dataset released from Binghamton University [15]. Fig. 1 contains an example subject from this dataset.

We selected 20 subjects performing two expressions (smile, surprise) for a total of 40 sequences. The cropped face resolutions are approximately 700 x 700 pixels in dimension, with approximately every 3x3 pixel window containing a single depth value. We sampled the depth values at a 1:1, 1:2, 1:3, and 1:4 ratio and then used bilinear interpolation to scale the values back up to 700x700. Fig. 2 Contains some example strain maps calculated at each scale.

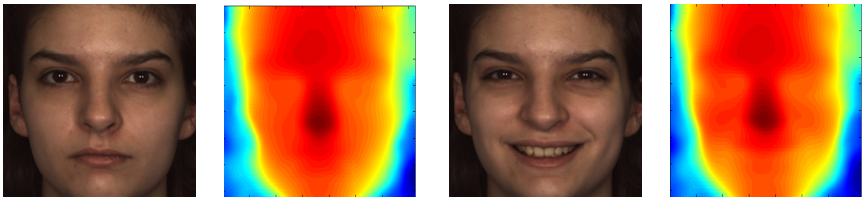


Fig. 1. Example data from BU dataset showing face image and corresponding depth map. (red=closest, blue = farthest)

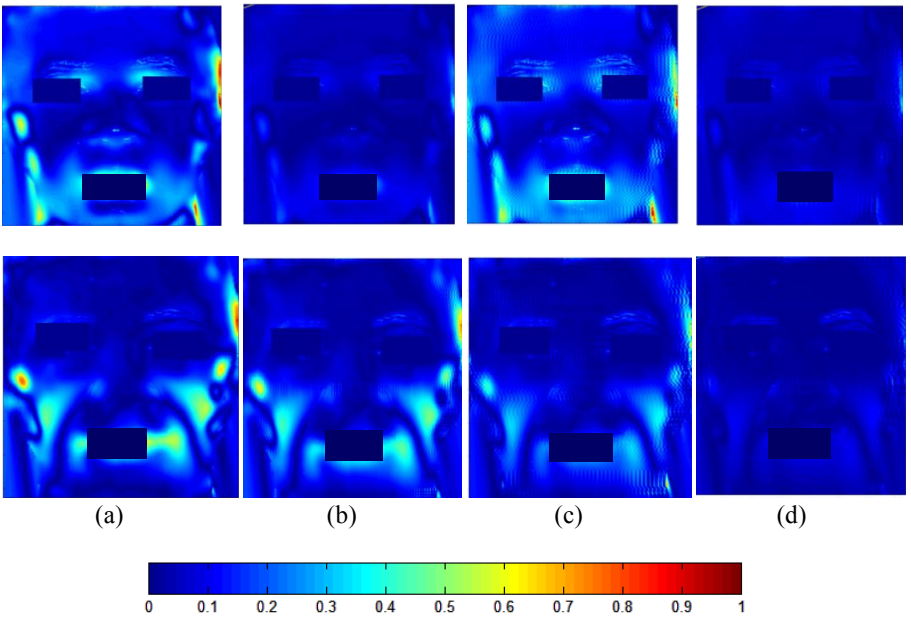


Fig. 2. Example normalized 3-D strain maps calculated for two subjects corresponding to the surprise and smile expressions (each row). Depths were sub-sampled at ratios of (a) 1:1, (b) 1:2, (c) 1:3, and (d) 1:4 (each row) resulting in depth resolutions of approximately 200x200, 100x100, 66x66 and 50x50.

In order to measure the similarity between strain maps calculated at several different depth resolutions, the correlation coefficient was used (Table 1). Several observations can be made. First, at least an 80% similarity is maintained for both expressions even when using approximately 70x70 of 200x200 (a third) of available depth points.

Table 1. Correlation coefficients for 40 expression (20 smile, 20 surprise) after subsampling at the given ratios and compared with a 1:1 sampling ratio

Exp. / Ratio	1:2	1:3	1:4
Smile	.90±.11	.80±.12	.69±.18
Surprise	.95±.02	.89±.05	.79±.15
Both	.93±.08	.85±.10	.74±.17

3.2 View Invariance

To demonstrate the view invariance of the method and give further evidence of the methods stability at low resolutions, we collected several subjects using the Microsoft Kinect sensor that was synchronized with an additional two additional HD webcams at approximately 30 degree angles to the face (see Fig. 3). We then registered each webcam to the automatically calibrated image provided by the Kinect using manually

labeled eye coordinates. The Kinect sensor provides depth imaging at an image resolution of 640x480 and a depth resolution of 320x280. However, due to mechanical restrictions on the minimum distance allowed to the sensor (roughly 3 feet), the face image is typically 175x175 in image resolution with a depth resolution of roughly 90 x 90. Some examples images and depth maps for this dataset can be found in Fig. 4.

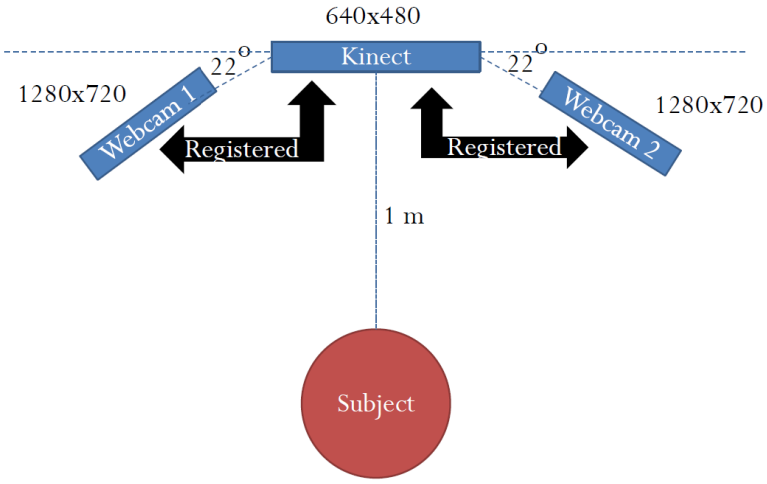


Fig. 3. Experimental setup for demonstrating view invariance

Fig. 4. Example images captured from approx. 30 degree angles with depth map (red=closest, blue = farthest) captured from Kinect synchronized with two webcams

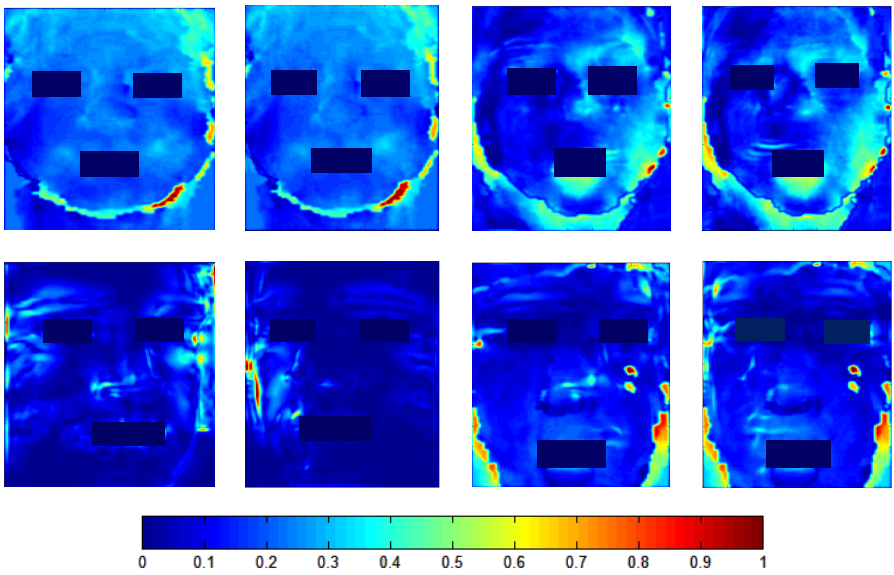


Fig. 5. Example strain maps calculated at two views roughly 45 degrees apart, for two subjects (each row). The first two pairs of columns are for the smile expression, the second pair of columns are for the surprise expression.

Due to the amount of noise in the depth values provided by the Kinect, smoothing was required as pre-processing step. It is worth noting that we tested several kernel sizes and standard deviations, and a 3x3 Gaussian kernel with a standard deviation of .1 led to the best trade-off between keeping the three-dimensional structure of the face intact and minimizing erroneous noisy depth gradients. Some example 3-D strain maps can be found in Figure 4 for two different views that demonstrate the view-invariance of the method.

4 Conclusions

Optical strain maps calculated during facial expression are a bio-mechanical feature that has been shown to have broad significance in facial motion analysis. The goal of this paper is to use cost-effective 3-D imaging to extend this method to a more robust 3-D strain map. However, due to the current limitations and optics of some depth sensors, direct 3-D flow correspondences may not be feasible. Hence, we have proposed a method for calculating view-invariant strain maps based on projecting the 2-D motion obtained from high resolution images on to a low resolution 3-D surface. Therefore, the correspondence issue is solved using 2-D displacements, which are then updated using rough 3-D estimations. We have demonstrated that the method is robust at several different depth resolutions and views by first giving results on the high resolution BU dataset sampled at different resolutions. The view-invariance of

the method was demonstrated using a low-resolution Kinect that was synchronized with two HD webcams that were roughly 45 degrees apart.

References

1. Shreve, M., Godavarthy, S., Goldgof, D.: Macro- and micro-expression spotting in long videos using spatio-temporal strain. In: Proceedings of Int. Conference on Automatic Face and Gesture Recognition, pp. 51–56 (2011)
2. Shreve, M., Manohar, V., Goldgof, D., Sarkar, S.: Face recognition under camouflage and adverse illumination. In: Proceedings of International Conference on Biometrics: Theory Applications and Systems, pp. 1–6 (2010)
3. Shreve, M., Jain, N., Goldgof, D., Sarkar, S., Kropatsch, W., Tzou, C.-H.J., Frey, M.: Evaluation of facial reconstructive surgery on patients with facial palsy using optical strain. In: Real, P., Diaz-Pernil, D., Molina-Abril, H., Berciano, A., Kropatsch, W. (eds.) CAIP 2011, Part I. LNCS, vol. 6854, pp. 512–519. Springer, Heidelberg (2011)
4. Bickel, B., Botsch, M., Angst, R., Matusik, W., Otaduy, M., Pfister, H., Gross, M.: Multi-scale capture of facial geometry and motion. ACM Transactions on Graphics 29(3), 33 (2007)
5. Blanz, V., Basso, C., Poggio, T., Vetter, T.: Reanimating faces in images and video. Computer Graphics Forum 22(3), 641–650 (2003)
6. Lin, I., Ouhyoung, M.: Mirror MoCap: Automatic and efficient capture of dense 3D facial motion parameters. Visual Computer 21(6), 355–372
7. Bradley, D., Heidrich, W., Popa, T., Sheffer, A.: High resolution passive facial performance capture. ACM Transactions on Graphics 29(4), 41 (2010)
8. Furukawa, Y., Ponce, J.: Dense 3D motion capture from synchronized video streams. In: Image and Geometry Processing for 3-D Cinematography, 193–211 (2010)
9. Furukawa, Y., Ponce, J.: Dense 3D motion capture for human faces. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp. 1674–1681 (2009)
10. Pons, J., Keriven, R., Faugeras, O.: Multi-view stereo reconstruction and scene flow estimation with a global image-based matching score. International Journal of Computer Vision 72(2), 179–193 (2007)
11. Penna, M.: The Incremental Approximation of Nonrigid Motion. Computer Vision, Graphics, and Image Processing 60(2), 141–156 (1994)
12. Hadfield, S., Bowden, R.: Kinecting the dots: particle based scene flow from depth sensors. In: Proceedings of International Conference on Computer Vision, pp. 2290–2295 (2011)
13. Weise, T., Bouaziz, S., Li, H., Pauly, M.: Realtime Performance-Based Facial Animation. ACM Transactions on Graphics 30(4), 77 (2011)
14. Horn, B., Schunck, B.: Determining optical flow. Artificial Intelligence 17, 185–203 (1981)
15. Neumann, J., Aloimonos, Y.: Spatio-temporal stereo using multi-resolution subdivision surfaces. International Journal of Computer Vision 47(1-3), 181–193 (2002)
16. Horn, B.K.P., Schunck, B.G.: Determining optical flow. Artificial Intelligence 17, 185–203 (1981)
17. Yin, L., Chen, X., Sun, Y., Worm, T., Reale, M.: A High-Resolution 3D Dynamic Facial Expression Database. In: International Conference on Automatic Face and Gesture Recognition (2008)

Removing Moving Objects from Point Cloud Scenes

Krystof Litomisky and Bir Bhanu

University of California, Riverside
krystof@litomisky.com, bhanu@ee.ucr.edu

Abstract. Three-dimensional simultaneous localization and mapping is a topic of significant interest in the research community, particularly so since the introduction of cheap consumer RGB-D sensors such as the Microsoft Kinect. Current algorithms are able to create rich, visually appealing maps of indoor environments using such sensors. However, state-of-the-art systems are designed for use in static environments, which severely limits the application space for such systems. We present an algorithm to explicitly detect and remove moving objects from multiple views of a scene. We do this by finding corresponding objects in two views of a scene. If the position of an object with respect to the other objects changes between the two views, we conclude that the object is moving and should therefore be removed. After the algorithm is run, the two views can be merged using any existing registration algorithm. We present results on scenes collected around a university building.

Keywords: SLAM, 3D Mapping, RGB-D sensors, Kinect.

1 Introduction

Although point clouds and sensors that provide point cloud data have been around for several decades, the introduction in 2010 of the Microsoft Kinect RGB-D (RGB color + per-pixel depth) sensor reinvigorated the field. One popular area of research has been using RGB-D sensors for Simultaneous Localization and Mapping (SLAM) in primarily indoor environments. State of the art systems have achieved impressively accurate results, producing visually-appealing maps of moderately-sized indoor environments [1]. Such algorithms typically rely on the Iterative Closest Point (ICP) for point cloud registration, and also incorporate loop-closing techniques to detect when an agent has returned to a previously visited area [2].

In published work on SLAM using RGB-D sensors in indoor environments, such as [1], [3], and [4], the maps are created in static environments: there are no moving objects, and, in particular, no people walking around. In more dynamic environments, not addressing moving objects can lead to maps that contain moving objects as permanent features, inconsistent maps, or even registration failure. Our work addresses this issue by introducing a novel preprocessing step to explicitly detect and remove moving objects from point cloud frames prior to registration. A sample result of our algorithm is in Fig. 1, and an overview of our approach is in Fig. 2.

X. Jiang et al. (Eds.): WDIA 2012, LNCS 7854, pp. 50–58, 2013.

To identify moving objects, we compare two frames with significant overlap (i.e. some camera motion is allowed). The viewpoint between the two frames can change, and some amount of time must elapse before the second frame is captured so that the position of moving objects changes between the frames. For each frame, we segment out individual clusters, and find which cluster from the first frame corresponds to each cluster from the second. We then analyze the spatial relationship of each cluster in a frame to all other clusters in the frame. If this relationship changes from one frame to the next, we conclude that the cluster in question must be moving and remove it. Having done this, we can apply any existing registration algorithm to register, align, and merge the two clouds.

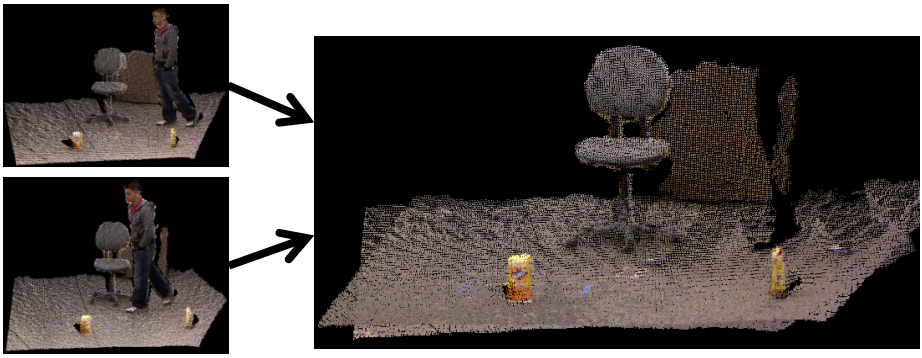


Fig. 1. A sample result of our algorithm. The top row shows two input point clouds. Our algorithm identified the person as a moving object and removed him from each point cloud. We then aligned and merged the clouds to produce the cloud in the bottom row.

2 Related Work

3D SLAM has been an area of immense research interest. Over the years, a number of approaches using different technologies have been developed, including range scans [3, 4], stereo cameras [7], monocular cameras [8], and recently also consumer RGB-D cameras [1][3][4]. However, explicitly identifying and removing moving objects from point cloud data has not been a topic of great interest in the research community.

In their RGB-D mapping work, Henry et al. [1] do not address moving objects, and do not present any results from environments with moving objects. However, the use of a surfel representation [9] allows them to deal with some instances of moving objects. Because surfels have an area, they permit reasoning about occlusion. This ultimately enables this representation to remove moving objects by eliminating surfels which occlude other surfels. However, this approach will fail to eliminate moving objects when these objects do not occlude other known objects. Furthermore, converting all data to surfels is computationally inefficient.

Similarly, the Kinect Fusion work [10], which implements a real-time version of ICP that runs on CUDA-enabled GPUs, deals with moving objects implicitly by using a volumetric representation. Kinect Fusion is not primarily a SLAM algorithm; as a

result, the approach does not scale to larger environments. In particular, with current GPUs, the approach cannot be used for environments larger than a moderately small room (approximately a 5m x 5m x 5m volume).

Other volumetric approaches, such as the Octree-based OctoMap [6], also identify moving objects only implicitly. The OctoMap's probabilistic nature means that it may require a substantial amount of measurements until a moving object is discarded.

Unlike previous work, which deals with moving objects in only an implicit and incomplete way, this paper presents a focused and systematic approach for removing moving objects from point cloud scenes captured from a moving platform. An advantage of our approach is that it deals directly with the point cloud data. This means that after eliminating moving objects with our approach, any existing algorithm for point cloud registration – or any other application – can be applied straightforwardly.

3 Technical Approach

A high-level overview of our approach is given in Fig. 2. For notational clarity, we will refer to the first point cloud we process as the "source" cloud, and to the second cloud we process as the "target" cloud. Our approach is symmetric, however, and which cloud is designated as "source" does not affect our results. For some of the point cloud manipulation tasks, we rely on the Point Cloud Library (PCL) [11].

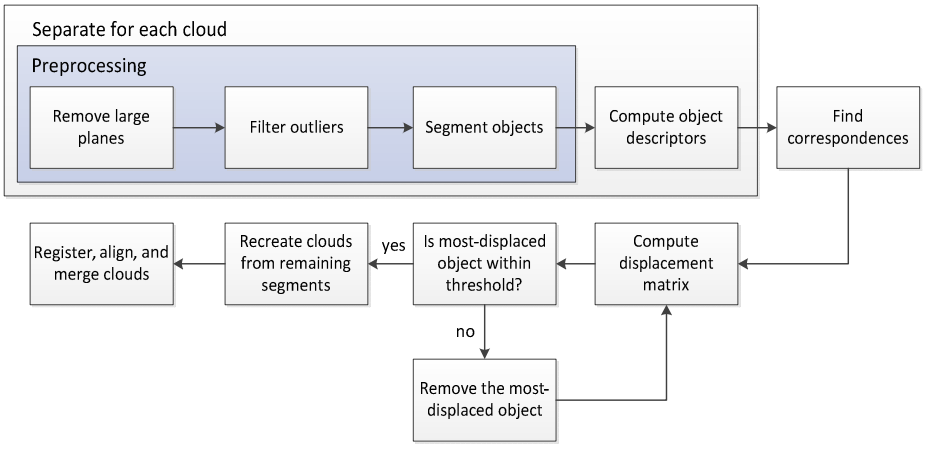


Fig. 2. An overview of our system

3.1 Preprocessing

Before identifying moving clusters, we temporarily remove large planes, filter outlier points, and run segmentation to identify individual clusters.

We identify planes using a Random Sample Consensus (RANSAC) algorithm. The reason for removing large planes is twofold: first, we can safely assume that large planes are not parts of moving objects, so we do not need to consider them in subsequent steps of the algorithm. Second, removing large planes – in particular, the

floor – improves the performance of our segmentation algorithm. Having completed this step, we filter out outlier points to remove artifacts due to sensor noise.

We now use a Euclidean Cluster Extraction algorithm to get individual clusters. This algorithm places points that are less than the cluster tolerance apart in the same cluster. Since large planar objects as well as statistical outliers have already been removed from the clouds at this point, this is an acceptable approach. Due to quantization of the depth data, which is inevitable with consumer RGB-D sensors, we need to use a somewhat large cluster tolerance value. We settled on 15 cm, meaning two points will be in the same cluster if they are less than 15 cm apart. We use a k-d tree to speed up the cluster extraction process.

Each point cloud can now be represented as a set of clusters. Let $C_s = \{s_1, s_2, \ldots, s_{n_s}\}$ be the source point cloud, and $C_t = \{t_1, t_2, \ldots, t_{n_t}\}$ be the target point cloud. Here, s_i is the set of points representing the i^{th} cluster in the source cloud, and t_i is the set of points representing the i^{th} cluster of the target cloud.

3.2 Cluster Descriptors and Identifying Correspondences

We use the Viewpoint Feature Histogram (VFH) proposed by Rusu et al. [12] as our feature descriptor. The VFH has good object classification performance, outperforming spin images for these tasks [12]. The VFH stores the relationships between the pan, tilt, and yaw angles between pairs of normals in a cluster, as well as the relationships between the viewpoint direction and the surface normals.

We now find cluster correspondences, or which cluster from the target cloud corresponds to each cluster from the source cloud. We calculate the distance between a pair of clusters in feature space in two ways and compare the performance in Section 4.

The first distance measure is the sum of absolute differences between two clusters' VFH descriptors, while the second distance measure comes from an insight on the ordering of the VFH: the bins for each angle "category" are consecutive. For example, if the yaw angle between two points changed slightly, the pair would move from one bin into a bin immediately next to it. As a result, small object deformations (such as a person's pose changing as he walks) as well as changes in sensor position cause nonlinear local deformations to the object's histogram. Fig. 3 illustrates the local deformations of a part of the VFH due to sensor motion.

Such issues have been addressed in the literature with Dynamic Time Warping [13]. Dynamic time warping finds a nonlinear, monotonic mapping from one sequence to the other, allowing for small shifts and deformations between the series. To limit the amount of warping, plausible values, we use the Sakoe-Chiba band [14] to limit warping to 2% of the total histogram length, corresponding to the shift in histogram resulting from a camera pan of about 45 degrees.

To identify correspondences, we iteratively take the closest pair of clusters (s_i, t_j) in feature space as corresponding to each other, until there are no clusters left in at least one cloud. Due to sensor and/or object motion, there may be leftover objects, which we remove from their cloud. This may remove objects that could provide useful data, but this is an acceptable tradeoff to ensure the removal of any moving objects that appear in only one frame. This step leaves $n = \min(n_s, n_t)$ clusters for each cloud. Before proceeding further, we reorder the clusters in C_t such that cluster s_i corresponds to cluster $t_i \; \forall \; i$.

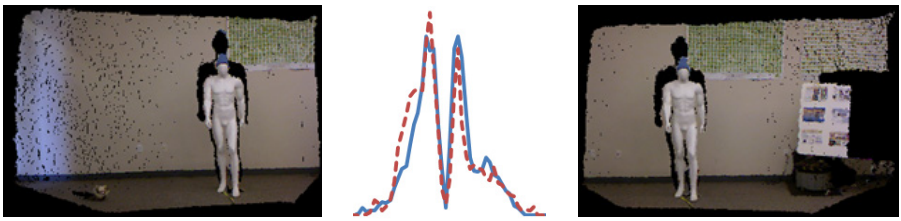


Fig. 3. Local deformations of the yaw component of the Viewpoint Feature Histogram of the mannequin cluster extracted from the two views above. The histogram for the mannequin cluster extracted from the left frame is solid blue; the histogram for the mannequin cluster from the right frame is dashed red. Dynamic Time Warping addresses such local deformations.

3.3 Identifying and Removing Moving Objects

We now calculate the Euclidean distance in world-coordinate space between each pair of clusters for each cloud. Let $d^s_{i,j}$ be the world-coordinate-space Euclidean distance between cluster i and cluster j in the source point cloud, and $d^t_{i,j}$ be the world-coordinate-space distance between the corresponding clusters in the target cloud.

To get a measure of how the position of each cluster has changed from one cloud to the next, we calculate the displacement vector $\Delta = [\delta_1, \delta_2, \ldots, \delta_n]^T$, where δ_i is the displacement of cluster i,

$$\delta_i = \sum_{k=1}^{n} \left| d^s_{i,k} - d^t_{i,k} \right| \tag{1}$$

In essence, δ_i is the sum of how much the distance of cluster i to each other cluster has changed from one cloud to the other cloud.

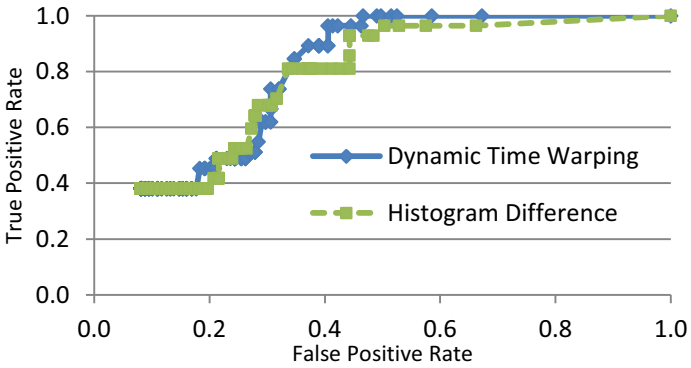


Fig. 4. Object removal threshold ROC curve. A true positive is a removed moving object, and a false positive is a removed static object.

We now iteratively remove the cluster which has the greatest displacement value as long as this value is above a threshold ε. After removing each cluster, we recalculate Δ. In order to find the optimal value of ε, we generated a ROC curve (Fig. 4). For the ROC curve, a true positive is a moving object that was removed from the scene, and a false positive is a static object that was removed from the scene. See Section 4 for details regarding the data used.

We now iteratively remove the cluster which has the greatest displacement value as long as this value is above a threshold ε. After removing each cluster, we recalculate Δ. In order to find the optimal value of ε, we generated a ROC curve (Fig. 4). For the ROC curve, a true positive is a moving object that was removed from the scene, and a false positive is a static object that was removed from the scene. See Section 4 for details regarding the data used.

We ran our algorithm for different thresholds for both the histogram difference and dynamic time warping distance measures, achieving better results with dynamic time warping. In particular, we were able to correctly remove all moving objects with $\varepsilon = 0.7$ meters when using dynamic time warping. We therefore use this value in our experiments (Section 4). In future work, the value of value of ε should be normalized with respect to the number of clusters detected in the scene.

Having removed all of the moving objects from C_s and C_t, we reconstruct each cloud from its remaining clusters as well as the planes that had been removed from it. After this, the clouds are in a standard point cloud format, and any existing registration, alignment, and merging algorithms can be used to concatenate the clouds. The result of this operation is a point cloud model of the environment with no moving objects in the point cloud, even though such objects might have been present in one or both of the original clouds.

4 Experiments

We tested our algorithm on 14 scenes collected around a university building. Each scene consists of two views, and each view has up to 2 moving people. Fig. 5 shows some results, demonstrating several different scenarios our algorithm can deal with.

Fig. 5(a) shows a scene with multiple moving people, as well as significant camera motion between the two frames. As a result of this, we cannot find good correspondences for some objects in the scene, such as the desk to the right of the second view. Note that there is a third person in the scene, sitting at a desk in the background. This person is put into the same cluster as the desk he is sitting at by our algorithm, and since he is not moving he is included in the recreated cloud.

Fig. 5(b) shows scene where a person who is not present in one point cloud moves into view by the time the second point cloud is captured. Our algorithm correctly identifies that there is no equivalent object for the person in the other frame, and removes the person from the point cloud.

Fig. 5(c) shows a scene where a walking person completely changes direction from one frame to the next. Nevertheless, our algorithm matches the person in one frame to the person in the next frame and correctly removes him from both clouds.

Fig. 5. Example results. (a) office scene with three people (two walking, one sitting). The top two frames from columns (a) and (b) were merged together to produce the bottom point cloud. (b) corridor scene where the person just entered the corridor through a doors on the right. (c) a walking person changing direction.

4.1 Quantitative Analysis

Fig. 4 shows the ROC curve obtained by running our algorithm on the dataset, where a true positive is defined as a moving object that is removed from the scene and a false positive is defined as a static object that was removed from the scene. We get better results with the dynamic time warping distance measure than with the simple histogram difference. In particular, for the object removal threshold $\varepsilon = 0.7$, we correctly remove all moving objects while also removing 47% of the static objects. In a

SLAM scenario, we would keep a greater fraction of static objects, since objects that are removed due to occlusion or sensor motion at first would likely be seen again at a later time and thus ultimately kept.

We also evaluate what fraction of the total points that belong to stationary objects we keep in the recreated clouds. For each scene, we calculate this number separately for each of the two frames, and then report their mean and standard deviation. Fig. 6 shows the results. On average, we keep 85% of the static points in a scene.

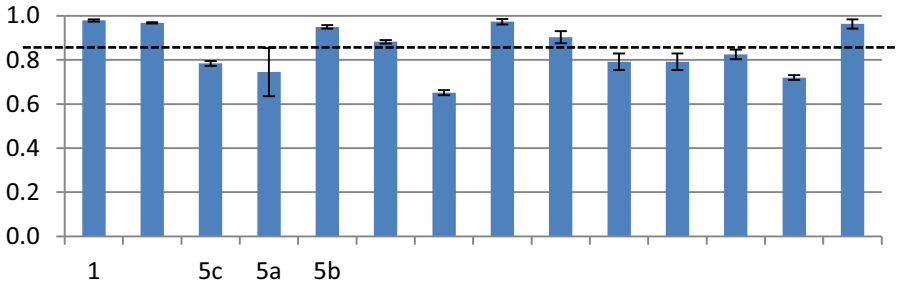


Fig. 6. Fraction of static points kept for each of the 14 scenes on which we evaluated our algorithm. Numbers along the x-axis indicate which figure corresponds to the particular scene, if applicable. The mean is 85% of static points retained.

5 Conclusions and Future Work

We introduce early work on an algorithm for identifying moving objects in point cloud scenes. We can eliminate moving objects from scenes while retaining most of the static objects. Thus, when used as a preprocessing step, our approach can complement existing point cloud-based SLAM algorithms. This will allow existing SLAM approaches to create consistent maps even in the presence of moving objects, making the system applicable in more scenarios.

Future work will include incorporating the approach presented here into an autonomous SLAM system, and making any necessary adjustments necessary to make the system work properly in real-time. Considerations will include tuning plane removal as well as segmentation, and studying the temporal stability of the approach.

References

[1] Henry, P., Krainin, M., Herbst, E., Ren, X., Fox, D.: RGB-D Mapping: Using depth cameras for dense 3D modeling of indoor environments. In: Khatib, O., Kumar, V., Sukhatme, G. (eds.) Experimental Robotics. STAR, vol. 79, pp. 477–491. Springer, Heidelberg (2012)

[2] Newman, P.: SLAM-Loop Closing with Visually Salient Features. In: Proceedings of the 2005 IEEE International Conference on Robotics and Automation (ICRA), pp. 635–642 (2005)

[3] Andreasson, H., Lilienthal, A.J.: 6D scan registration using depth-interpolated local image features. Robotics and Autonomous Systems 58(2), 157–165 (2010)
[4] Du, H., et al.: Interactive 3D modeling of indoor environments with a consumer depth camera. In: Proceedings of the 13th International Conference on Ubiquitous Computing - UbiComp 2011, p. 75 (2011)
[5] May, S., et al.: Three-dimensional mapping with time-of-flight cameras. Journal of Field Robotics 26(11-12), 934–965 (2009)
[6] Wurm, K.M., Hornung, A., Bennewitz, M., Stachniss, C., Burgard, W.: OctoMap: A probabilistic, flexible, and compact 3D map representation for robotic systems. In: Proc. of the ICRA 2010 Workshop on Best Practice in 3D Perception and Modeling for Mobile Manipulation, vol. 2 (2010)
[7] Lemaire, T., Berger, C., Jung, I.-K., Lacroix, S.: Vision-Based SLAM: Stereo and Monocular Approaches. International Journal of Computer Vision 74(3), 343–364 (2007)
[8] Davison, A.J., Reid, I.D., Molton, N.D., Stasse, O.: MonoSLAM: real-time single camera SLAM. IEEE Transactions on Pattern Analysis and Machine Intelligence 29(6), 1052–1067 (2007)
[9] Pfister, H., Zwicker, M., Van Baar, J., Gross, M.: Surfels: Surface elements as rendering primitives. In: Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques, pp. 335–342 (2000)
[10] Newcombe, R., et al.: KinectFusion: Real-time dense surface mapping and tracking. In: 10th IEEE International Symposium on Mixed and Augmented Reality, pp. 127–136 (2011)
[11] Rusu, R.B., Cousins, S.: 3D is here: Point Cloud Library (PCL). In: Proceedings of the 2011 IEEE International Conference on Robotics and Automation, pp. 1–4 (2011)
[12] Rusu, R.B., Bradski, G., Thibaux, R., Hsu, J.: Fast 3D Recognition and Pose Using the Viewpoint Feature Histogram. In: International Conference on Intelligent Robots and Systems, pp. 2155–2162 (2010)
[13] Berndt, D., Clifford, J.: Using dynamic time warping to find patterns in time series. In: Conference on Knowledge Discovery and Data Mining - KDD, vol. 398, pp. 359–370 (1994)
[14] Sakoe, H., Chiba, S.: Dynamic programming algorithm optimization for spoken word recognition. IEEE Transactions on Acoustics, Speech, and Signal Processing 26(1), 43–49 (1978)

High Quality Novel View Synthesis Based on Low Resolution Depth Image and High Resolution Color Image

Jui-Chiu Chiang, Zheng-Feng Liu, and Wen-Nung Lie

Department of Electrical Engineering, National Chung Cheng University
168, University Road, Ming-Hsiung, Chia-Yi, 621, Taiwan, ROC

Abstract. In this paper, a new technique to generate high resolution depth image is proposed. First, a low resolution depth map is obtained by the time-of-flight depth camera. Then a high resolution depth map for a given view is generated by depth warping followed by depth value refinement taking into account the color information at the given view. The edge in the final depth map is then processed by bilateral filtering for edge preserving. With the color image and the corresponding depth image, novel view synthesis can be carried out by depth image based rendering (DIBR). Experimental results show that the depth map generated by the proposed technique is able to ensure novel view images with high quality.

Keywords: novel view synthesis, DIBR, depth camera.

1 Introduction

With the advance in acquisition and display technologies, entertainment ensuring higher perceptual realism is desired and becomes feasible recently. FTV (Free view-point television) [1] is recognized as the next generation of TV. It offers human not only the stereoscopic perception, but also various interactions and possibilities. To provide the audience with arbitrary view on demand, novel views have to be rendered in the display. Novel view synthesis techniques have been developed for many years, where image based rendering (IBR) [2-3], depth image based rendering (DIBR) [4-5] and ray-space method [6-7] are the main technologies to be adopted. In IBR, no depth image is required and the challenge of IBR is to find out the correspondence points in given images pair. Then novel view images can be synthesized by proper interpolation. However, the image quality of the novel view could be limited due to insufficient geometric information. Ray-space is the representation of light rays in three-dimensional space and virtual view rendering for arbitrary positions is performed without any geometry information. However, it usually needs to capture the scene with dense cameras. DIBR attracts lots of attention recently. DIBR needs color images with corresponding depth information to render the novel view-point image. The image quality of novel view based on DIBR could be much improved, compared to IBR methods, under the assumption that the depth image is reliable. Although DIBR has the potential to offer novel views with high quality, several problems

X. Jiang et al. (Eds.): WDIA 2012, LNCS 7854, pp. 59–67, 2013.

during DIBR realization have to be managed, such as hole, disocclusion and occlusion. To cope with these problems, many research works are presented [8-10].

The main functionality supported by FTV is the free navigation by seamless novel view generation, while stereoscopic perception is the main target of 3DTV. 3DTV offers human the stereoscopic experience and became available in the market since 2010. There are several 3D representation formats, including stereo image pair, frame compatible stereo image format and video plus depth. There are two main advantages provided by video plus depth format. First, the depth image is represented as gray level and usually the required bitrate is much less than color video due to the homogeneous property. Second, a stereo image pairs with flexible disparity range can be generated in the receiver side by DIBR synthesis technique.

Both 3DTV and FTV could be realized by DIBR and it is imperative to acquire accurate depth maps for rendering stereo image pair and novel view-point image with satisfactory perception. Generally, the depth image can be captured directly or estimated by stereo matching algorithm [11]. Usually, the estimated depth maps suffer from inaccurate results in the occlusion regions, as well as textureless regions. Although the captured depth image ensures higher accuracy compared to estimated depth image, the resolution is usually too small to be coordinated with the color image directly for DIBR synthesis. There are several works concentrating on high resolution depth image generation under the scenario that a hybrid camera system (including both color camera and depth camera) is available [12-15]. In [12], an initial depth map for the left image is computed by a stereo matching algorithm. Then the ROI (region of interest) in the depth map is refined by warping the depth values obtained from the depth camera to the left image.

In this paper, we propose a new technique to generate high resolution depth map from a low resolution depth image and a high resolution color image. Different from the work in [12] where the depth map is first estimated by stereo matching and the depth information provided by depth camera is used to refine the ROI, the stereo matching is not performed in the proposed scheme and one color image is required in the proposed method. Instead, the color image will be used to refine the depth information obtained by the time-of-flight depth camera.

The rest of this paper is organized as follows. Section 2 introduces the technique to generate the preliminary high resolution depth map using the low-resolution depth map. Section 3 details the method to refine the depth map considering the color information. The experimental results are presented in section 4 and section 5 draws the conclusion.

2 Generation of a Preliminary High-Resolution Depth Maps

In our image capture system, there are one color camera and one time-of-flight depth camera in parallel setting. The captured depth video and color video can be seen as the video plus depth pair for 3DTV application. Note that, the resolution of the depth image is smaller than the color image, and the challenge is to make the resolution of the depth image the same as the color image for rendering purpose. And the same time, how to ensure accurate depth map is also important in providing high-quality novel view image. To facilitate the following procedures, camera calibration has to be employed first for the two cameras to obtain the extrinsic parameters for each camera.

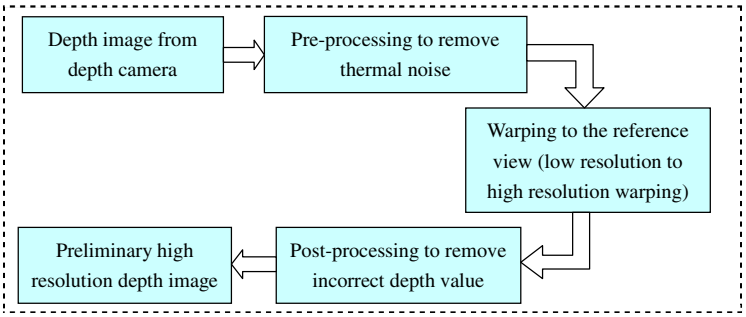


Fig. 1. Block diagram of preliminary high-resolution depth map generation

2.1 Pre-processing of the Depth Map

To realize the novel view synthesis, a color image and the associated depth map are needed. In our camera setting, the view capturing color image has no captured depth image. Thus, the depth image captured in the center view has to be warped into the reference view (i.e., left or right view). Figure 1 presents the procedures to generate high resolution depth map for the view with color image.

Usually, the sensing results of the time-of-flight depth camera are not always reliable. Optical noise, as well as the material of objects in the scene may cause incorrect depth information. To resolve this problem, the captured depth map is pre-processed with a 3×3 median filter. Furthermore, the lens distortion in the depth map has to be removed before warping to the reference view.

2.2 Low Resolution to High Resolution Warping

After removing the noise in the depth map, 3D warping is performed to get the depth map of the reference view. However, the resolutions for the depth camera and color camera are not the same, and the depth map in the reference view will be incomplete when the low resolution depth map is warped directly. It means many pixels in the reference map have no assigned depth value. To overcome this problem, each pixel in the source view will be warped into a block with size 5×5 in the reference view. It means that in the reference view, in addition to the pixel location determined by the warping procedure, the surrounding 24 pixels will have the same. Here, the depth value denotes the pixel intensity of the depth map. Besides, if a pixel in the reference view has more than one warped depth values, the maximum depth value is used to prevent the disocclusion.

2.3 Post-processing by Morphological Operation

After warping, the obtained high resolution depth image may have incorrect values, especially around the edges. To resolve this problem, morphological opening is performed to get the preliminary high resolution depth image.

3 Generation of High Quality High-Resolution Depth Map

The preliminary depth map may have some remaining errors after morphology operation. To enhance the quality of the depth image, the color image at the reference view will be used. Figure 2 illustrates the flowchart of color-image guided depth map refinement.

First, a binary representation of the preliminary depth map $D(x,y)$ is obtained, represented as $D_b(x,y)$ after comparing to a given threshold as following,

$$D_b(x,y) = \begin{cases} 1, & if \quad D(x,y) \geq TH_d \\ 0, & if \quad D(x,y) < TH_d \\ -1, & if \quad D(x,y) = -1 \ (\text{"}hole\text{"}) \end{cases} \tag{1}$$

where $D(x,y) = -1$ denotes the situation that the location (x,y) has no depth value after warping and morphological opening (i.e., it can be seems as "hole"). Here, the depth map is represented in a way that closer objects have higher depth value. The foreground objects have higher value in depth map, and correspond to D_b=1.

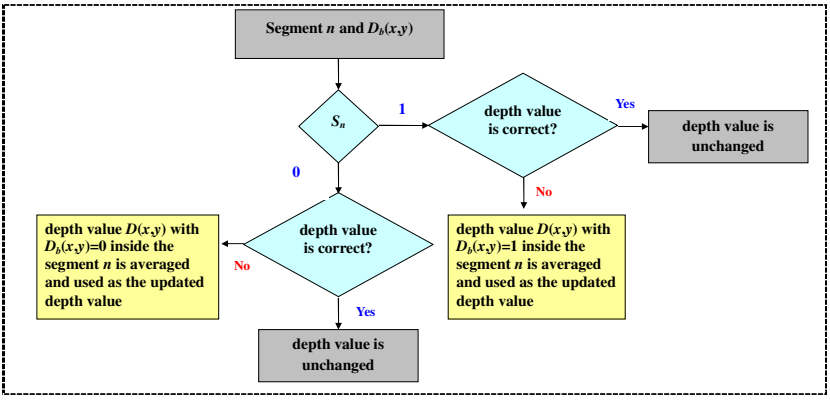


Fig. 2. Flowchart of color-image guided high-resolution depth map refinement

The segmentation results of the color image in the reference view are used to refine the depth map. Mean-shift algorithm [16] is adopted to segment the color image into several groups. The application scenario in this paper is a simple scene with two depth layers; one for the foreground and the other background. Thus, each segment n will be labeled as "foreground" or "background" according to the rule as below,

$$S_n = \begin{cases} 1, & \dfrac{D_n}{C_n} \geq TH \quad \text{(Foreground)} \\ 0, & \dfrac{D_n}{C_n} < TH \quad \text{(Background)} \end{cases} \tag{2}$$

where D_n and C_n denote, respectively, the number of pixel with value D_b=1 inside the segment n and the total pixel number inside the segment n.

After labeling each segment as foreground or background, the correctness of each depth value inside each segment will be verified. If S_n =1, the segment n is recognized as a foreground region and the depth value $D_b(x,y)$ for the pixel inside this segment should be equal to 1. If the depth value $D_b(x,y)$ is not equal to 1, it is treated as an incorrect depth value, and the depth value $D(x,y)$ with $D_b(x,y)$=1 inside the segment n is averaged and used as the updated depth value for this pixel. The verification of depth values correctness in the segment with S_n=0 is similar. There is no more hole in the depth map after this process.

To preserve the object edges and enhance the quality of depth image, a bilateral filter combining domain and range filtering is performed on the depth map. The two filter coefficients are determined by the relative distance Δd and the intensity difference ΔI in the color image, as below:

$$D_{bf}(x,y) = \frac{\sum_{s=-m/2}^{m/2}\sum_{t=-n/2}^{n/2}\left(\omega_{bf} \times D(x+s, y+t)\right)}{\sum_{s=-m/2}^{m/2}\sum_{t=-n/2}^{n/2}\omega_{bf}} \qquad \omega_{bf} = 2^{-(\Delta d + \Delta I)}, \tag{3}$$

where m and n denote the window size of filtering. Note that, to avoid the interference between different color segments, not all the image, but the edge and surrounding pixels are processed by the bilateral filter.

4 Experimental Results

To evaluate the performance of the proposed algorithm, we have carried out the algorithm on the real captured images. The resolution of the captured depth image and color image are 176×144 and 640×480, respectively. Several scenes are captured and experimental results of one scene with different time slot images are presented here due to limited space.

4.1 Depth Map Generation

Figures 3 illustrates the results of preliminary depth map generation, where Figure 3(a) is the color image in the reference view. It shows that many pixels in Figure 3(b) are represented as green due to no assignment during warping process. Figure 3(c) presents a more complete depth map by the proposed warping technique where one pixel in the source view will be mapping to a block in the target view. However, the depth values around the edge seem incorrect and can be resolved by morphological opening operation, as shown in Figure 3(d) & (f).

The refined depth maps after considering the color image information in the same view is illustrated in Figure 4. The segmentation results on the color image are shown in Figure 4(a). Then each segment will be classified as foreground or background and used to judge the correctness of the depth value inside each segment. The pixel represented as red in Figure 4(b) is the incorrect pixel and will be updated, as shown in Figure 4(c). The depth image after bilateral filtering is shown in Figure 4(d), where the edge is more sharped. As highlighted in Figure 4(f), the profile of the edge is clearer than that in Figure 4(e). Figure 5 shows a series of depth map generated by the proposed scheme.

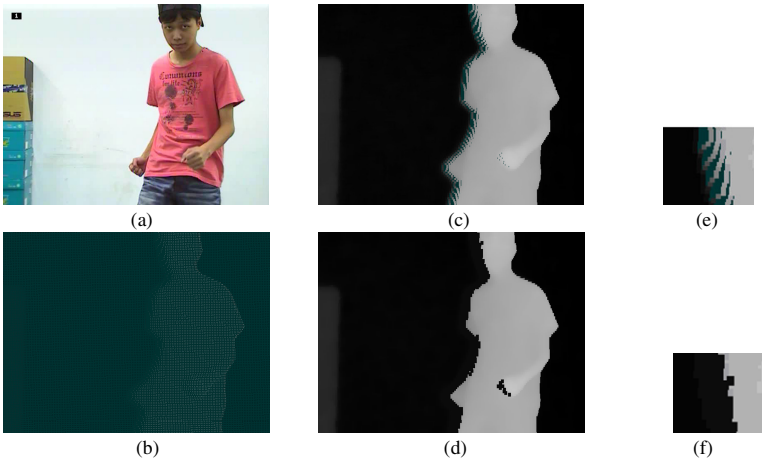


Fig. 3. Illustration of preliminary depth map generation. (a) color image in the left view, (b) the warped depth image in the left view (pixel without intensity assignment is marked as green) (c) the depth map after applying one pixel to 5×5 block warping, (d) the depth image after morphological opening, (e) region before morphological opening, (f) region after morphological opening.

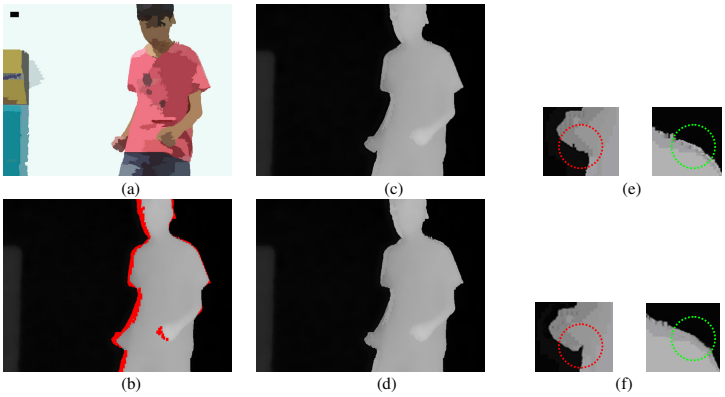


Fig. 4. Illustration of depth map refinement (a) color segmentation results,(b) the high resolution depth image after taking the color information into consideration (the pixel marked in red is labeled as incorrect pixel), (c) the depth image after correction, (d) the depth image after bilateral filtering on the edges, (e) region before bilateral filtering, (f) region after bilateral filtering

4.2 Novel View Synthesis

To evaluate the correctness of the depth image, a novel view synthesis is performed, which can be carried out by one color image and the associated depth map. Typically, the synthesis quality will be better for the scenario with multiple color images and depth maps. Note that, the proposed algorithm can be easily applied to the scenario with two or more color cameras. Then each color image will have a corresponding high resolution depth map for rendering purpose.

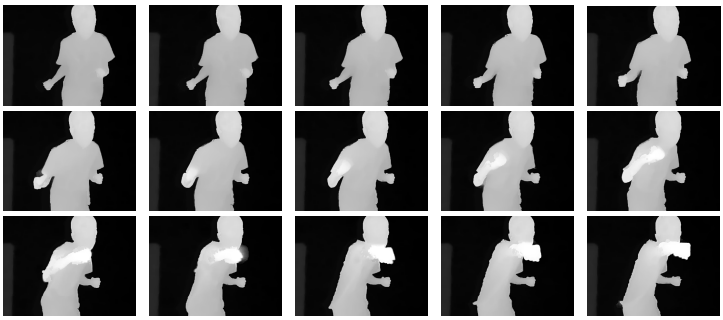


Fig. 5. A series of the generated depth maps

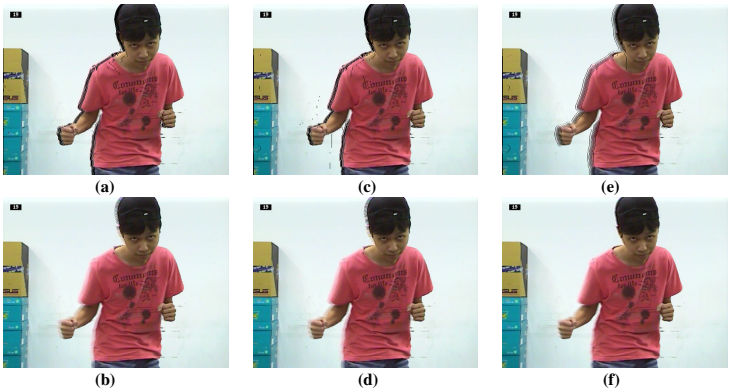


Fig. 6. Virtual view images using (a) depth image with bilateral filtering on whole frame (without hole filling),(b) depth image with bilateral filtering (after hole filling), (c) depth image with bilateral filtering on edges (without hole filling), (d) depth image with bilateral filtering on edges (with hole filling), (e) depth image with bilateral filtering on edge, followed by Gaussian filtering (without hole filling), (f) depth image with bilateral filtering on edge, followed by Gaussian filtering (after hole filling)

If the novel view synthesis relies on only one color image and the corresponding depth map, the difficulty to fill the disocclusion region is higher than the scenario with multiple color images and depth maps. To tackle this problem, the depth map generated by the proposed scheme can be post-processed by Gaussian filtering. In this way, the transition between foreground and background is smoother, and satisfactory image quality is revealed on the synthesized image after simple hole filling. Figure 6 demonstrates the synthesized results using various depth maps.

4.3 Execution Time Analysis

Here, the execution time for each step for generating the final depth map is analyzed, as shown in Table 1. Intel Core 2 Duo Q6600 2.33 GHz, and DDR2 800 4GB are used for the simulation. Table 1 indicates that the color segmentation is the most time-consuming part in the proposed scheme.

Table 1. Execution time for depth map generation

	Procedure	time(sec)
Pre-processing	Depth Warping	0.117
	Color segmentation	0.742
Post-processing	Bilateral filtering on edges	0.352
	Gaussian filtering	0.104
Novel view synthesis by DIBR (including hole filling)		0.144

5 Conclusion

A hybrid camera system consisting of high-resolution color camera and low-resolution depth camera is considered in this paper. The goal is to render a high quality novel view image and an algorithm for accurate depth map generation is proposed. A preliminary depth map is built using the depth image captured by the depth camera and refined after taking the color information into consideration. The color segmentation results will guide whether the depth value is correct and appropriate update is realized accordingly. Experimental results show that a high quality novel view mage can be rendered using the depth map generated by the proposed scheme.

References

[1] ISO/IEC JTC1/SC29/WG11 M8595, FTV-Free Viewpoint Television (2002)
[2] Seiz, S.M., Dyer, C.R.: View Morphing. In: Proc. of SIGGRAPH, pp. 21–30 (1996)
[3] Debevec, P.E., Yu, Y., Borshukov, G.: Efficient View-dependent Image-based Rendering with Projective Texture Mapping. In: Proc. of Eurographics Rendering Workshop, pp. 105–116 (1998)
[4] Fehn, C.: A 3D-TV Approach Using Depth-Image-Based-Rendering (DIBR). In: Proc. of Visualization, Imaging, and Image Processing, pp. 482–487 (2003)
[5] Fehn, C.: Depth-Image-Based Rendering (DIBR), Compression and Transmission for a New Approach on 3D-TV. In: SPIE-IS&T, vol. 5291, pp. 93–104 (January 2004)
[6] Fujii, T., Kimoto, T., Tanimoto, M.: Ray Space Coding for 3D Visual Communication. In: Proc. of Picture Coding Symposium, pp. 447–451 (March 1996)
[7] Naemura, T., Kaneko, M., Harashima, H.: 3-D Visual Data Compression Based on Ray-Space Projection. In: Proc. of SPIE VCIP, pp. 413–424 (1997)
[8] Daribo, I., Tillier, C., Pesquet-Popescu, B.: Distance Dependent Depth Filtering in 3D Warping for 3DTV. In: Proc. of IEEE Int'l Conf. on Multimedia Signal Processing (MMSP), Chania, Crete, Greece, pp. 312–315 (October 2007)

[9] Cheng, C.-M., Lin, S.-J., Lai, S.-H., Yang, J.-C.: Improved Novel View Synthesis from Depth Image with Large Baseline. In: Proc. of IEEE Int'l Conf. on Pattern Recognition (2008)

[10] Mori, Y., Fukushima, N., Yendo, T., Fujii, T., Tanimoto, M.: View Generation with 3D Warping using Depth Information for FTV. Signal Processing Image Communication 24(1), 265–272 (2009)

[11] Sun, J., Zheng, N.N., Shum, H.Y.: Stereo Matching using Belief Propagation. IEEE Trans. of Pattern Analysis and Machine Intelligence 25(5), 787–800 (2003)

[12] Kim, S.-Y., Lee, E.-K., Ho, Y.-S.: Generation of ROI Enhanced Depth Maps Using Stereoscopic Cameras and a Depth Camera. IEEE Trans. on Broadcasting 54(4), 732–740 (2008)

[13] Zhu, J., Wang, L., Yang, R., Davis, J.: Fusion of Time-of-flight Depth and Stereo for High Accuracy Depth Maps. In: Proc. of IEEE Int'l Conf. Computer Vision Pattern Recognition (2008)

[14] Bartczak, B., Koch, R.: Dense Depth Maps from Low Resolution Time-of-Flight Depth and High Resolution Color Views. In: Bebis, G., et al. (eds.) ISVC 2009, Part II. LNCS, vol. 5876, pp. 228–239. Springer, Heidelberg (2009)

[15] Yang, Q., Yang, R., Davis, J., Nistér, D.: Spatial-Depth Super Resolution for Range Images. In: CVPR 2007 (2007)

[16] Cheng, Y.: Mean Shift, Mode Seeking, and Clustering. IEEE Trans. on Pattern Analysis and Machine Intelligence 17, 790–799 (1995)

Color Segmentation Based Depth Image Filtering

Michael Schmeing and Xiaoyi Jiang

Department of Computer Science, University of Münster
Einsteinstraße 62, 48149 Münster, Germany
{m.schmeing,xjiang}@uni-muenster.de

Abstract. We present a novel enhancement method that addresses the problem of corrupted edge information in depth maps. Corrupted depth information manifests itself in zigzag edges instead of straight ones. We extract the depth information from an associated color stream and use this information to enhance the original depth map. Besides the visual results, a quantitative analysis is conducted to prove the capabilities of our approach. For this task, we introduce a new assessment technique which is based on measuring clustering similarity using the Rand index.

1 Introduction

Video-plus-depth is an important 3D scene representation format [1]. It consists of a color stream describing the texture of the scene and an associated depth stream describing for each pixel its distance to the camera. From this representation, arbitrary new views can be generated to enable stereo [1–3], multi view [4] or free viewpoint video [5, 6].

An important presumption for high quality rendering is a high quality depth map. However, there exists at the moment no depth map generation technique that is able to produce a perfect depth map, i.e., a depth map that is free of artifacts, holes, which is temporally stable and has video resolution all together.

Different depth map enhancement methods have evolved to address different aspects of depth map corruption [7–11]. In this paper, we propose a novel enhancement algorithm that takes associated color information into account to enhance the quality of edges in a depth map. We use depth maps generated by the Microsoft Kinect depth camera for our approach, though our algorithm is not restricted to depth maps generated with this camera. The Kinect camera is a structured light depth sensor which suffers from quite poor edge reproduction. Figure 1 shows an example. We use edge information found in the corresponding color stream via a superpixel segmentation and compute a new representative depth map D^r which stores robust edge information corresponding to the color stream. D^r is then used to enhance the source depth map D. A quantitative analysis shows that our method outperforms common depth enhancement algorithms in terms of edge restoration.

The rest of our paper is organized as follows. Section 2 discusses previous work in this field. We describe our algorithm in Section 3. In Section 4 we propose a

X. Jiang et al. (Eds.): WDIA 2012, LNCS 7854, pp. 68–77, 2013.

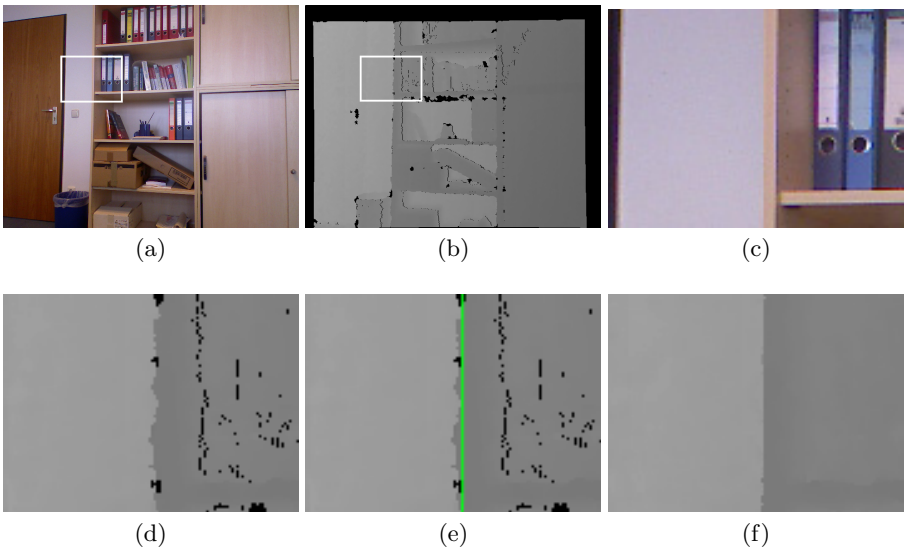


Fig. 1. (a) Example color image. (b) Associated depth map. (c) Magnification of (a). (d) Magnification of (b). (e) The green line marks the edge from the color stream. (f) Result of our approach.

new method to quantitatively assess edge restoration. We present some results in Section 5 and conclude the paper with section 6.

2 Related Work

There exist several methods that enhance depth maps. In [7], depth maps are filtered using a temporal median filter. Holes are filled using a spatial median. The work reported in [8] addresses special issues of the Microsoft Kinect depth camera. Holes that occur due to the offset between color and depth camera are closed by background extraction and other holes by row-wise linear interpolation. No temporal processing is applied.

In [9], the authors port simple filters like Gaussian-weighted hole filling and temporal smoothing as well as edge-preserving denoising on the GPU and achieve very high frame rates of 100fps. The framework is modal and not dependent on a certain depth camera technology. The method looks promising and is said to be applicable to dynamic scenes although no special evaluation was given in this case.

3 Our Method

In our algorithm we address the common problem of corrupted edge information in depth images. Figure 1 shows an example. In the color image, the edge of

the foreground object (the shelf) is a straight line, whereas it is corrupted in the depth stream. The corrupted edges in the depth image do not correspond with the edges of actual objects. When using this depth map for view synthesis (e.g. Depth Image Based Rendering [1]), artifacts will occur. Therefore, it is important to have edges in the depth map that correspond closely to the edges of the objects in the scene.

Our method works for scenes in the *video-plus-depth* format. We assume the depth stream to have the same resolution as the video stream. Depth upsampling [12], which is another field of depth video enhancement, can be applied as a preprocessing step if necessary.

There are two kinds of possible edge defects in a depth map. First, the edge is not straight but rather forms a zigzag line or consists of other visible steps. This can happen through the nature of the sensor (like the Kinect sensor) or through inadequate depth upsampling. The second defect is global misalignment, i.e. the complete edge of the depth map is shifted with respect to the edge in the color image. This defect can arise from insufficient registration between video camera and depth camera.

Let I be a frame of the video sequence and D the corresponding depth map. Our goal is to process D in a way that the edges in D align with the edges (of objects) in I.

As a first step, we perform *normalized convolution* [11] to fill holes in the depth map. A hole pixel x is filled with a weighted sum of the depth values of its non-hole neighboring pixels:

$$D^{nc}(x) = \frac{\sum_{x' \in N_x^*} D(x) g(x, x')}{\sum_{x' \in N_x^*} g(x, x')} \tag{1}$$

where N_x^* is the set of neighboring pixels of x that have a valid depth value and $g(x, x')$ a Gaussian function with parameter σ:

$$g(x, x') = \exp\left(-\frac{||x - x'||^2}{\sigma^2}\right). \tag{2}$$

In the next step, we identify edges in the color image. Instead of finding edges directly with common methods like the Canny operator, we use the implicit edge information given by a segmentation, more precisely, an over-segmentation of the color image. While a normal segmentation divides the image into "meaningful" areas (usually guided by edges), an over-segmentation further divides these areas. Those areas can nevertheless be recovered by combining areas of the over-segmentation. Particularly, the over-segmentation respects the edges of objects in the color image.

With an over-segmentation of the depth map, we can compute representative depth values for each segment, for example by taking the median or the average of all the depth values of pixels in this segment. These representative values are more robust to noise than one single pixel. The representative depth map D^r will be generated by filling each segment with its representative depth value. Since the segmentation respects the color image edges, the edges in the representative

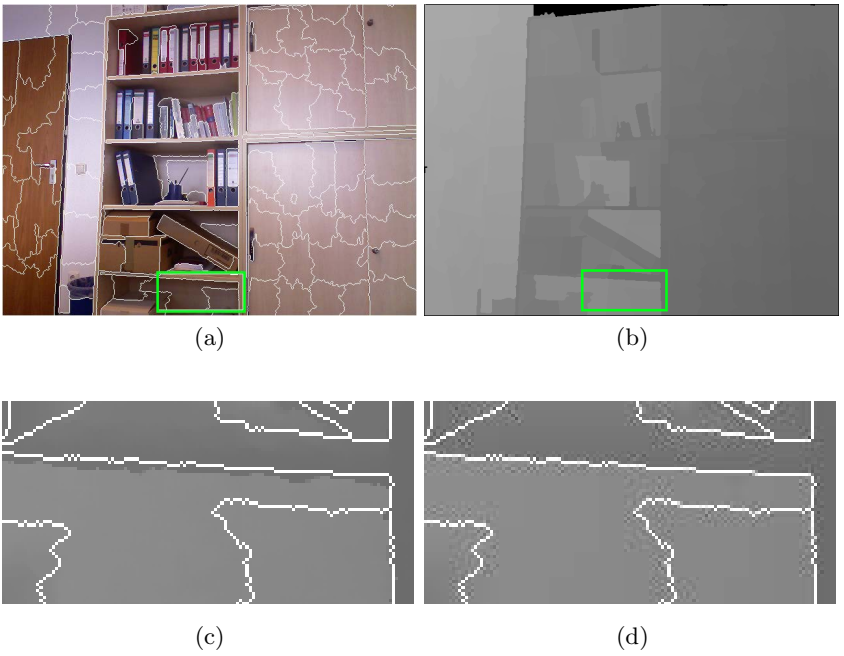


Fig. 2. (a) An example over-segmentation. (b) A representative depth map D^r with marked magnification region. (c) Cutout of original depth with projected segmentation. (d) Cutout of our method with projected segmentation: Depth and color edges align.

depth map will respect these edges, too. Using D^r, we can later discard pixels as corrupted that are too dissimilar to the representative depth value of their segment. See Figure 2(c) for an example: In the upper part of the main light region, the dark depth region overlaps into the light region which means that the depth edge and the color edge (i.e., the segment border) do not correspond. Figure 2(d) shows the corrected edge produced by our algorithm.

We tried different superpixel-segmentation methods including Mean-Shift and the method of [13] but in the end we used a simple watershed segmentation [14] because it delivers sufficient results for our purpose at a very high speed (more than 30fps at 640×480 resolution). We also tried different marker distributions for the watershed segmentation: randomly, on a regular grid, and skewed on a regular grid (which means that the markers of two consecutive rows do not lie in the same column but are slightly shifted). We tested these distributions with the additional constraint that markers are not placed on edges in the color image. The best results where obtained with the regular, skewed grid and no additional constraint.

Figure 2(a) shows an example segmentation. The quality of the segmentation can be degraded by a high amount of image noise, so we first apply a bilateral filter to reduce the noise while simultaneously protect edges in the color image. The filtered color value $I(p)$ of a pixel p is given by:

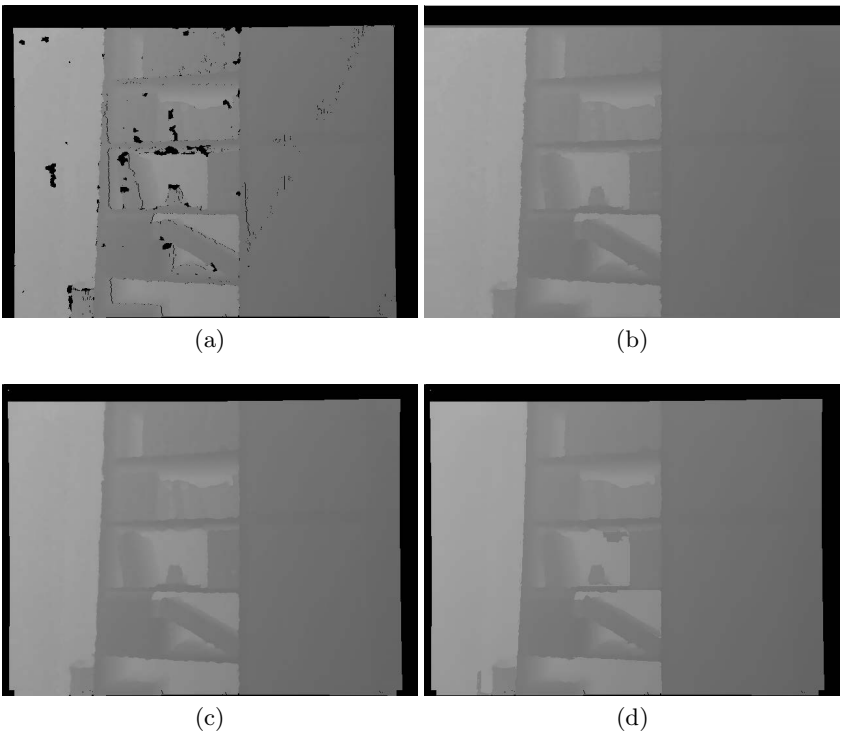


Fig. 3. (a) Example frame of unfiltered depth data generated by the Kinect depth sensor. (b) Frame filtered with method of Berdnikov et al. [8]. (c) Frame filtered with method of Wasza et al. [9]. (d) Our method.

$$I(p) = \frac{\sum_{q \in N} K_s(||p-q||) K_c(||p-q||) I(q)}{\sum_{q \in N} K_s(||p-q||) K_c(||p-q||)} \tag{3}$$

with K_s and K_c being Kernel functions, typically Gaussian distributions.

The obtained over-segmentation is then projected into the depth stream. In an ideal depth map, the edges of the over-segmentation would coincide with the edges in the depth map. Figure 2(c) shows what happens in real world depth maps (taken from a Kinect): Some areas overlap into neighboring segments.

Using a sufficient segment size, though, we can ensure that at least half of the depth pixels in a segment have correct depth (this is clearly the case in Figure 2(c)). We build the representative depth map D^r from this segmentation by computing for each segment the median depth value:

$$D^r(x,y) = \{d_k : (x,y) \in S_k,\ d_k = \underset{(x',y') \in S_k}{\text{median}}\ d(x',y')\}$$

where S_k is a segment in the color image. Figure 2(b) shows an example representative depth map. This depth map corresponds very well with the edges

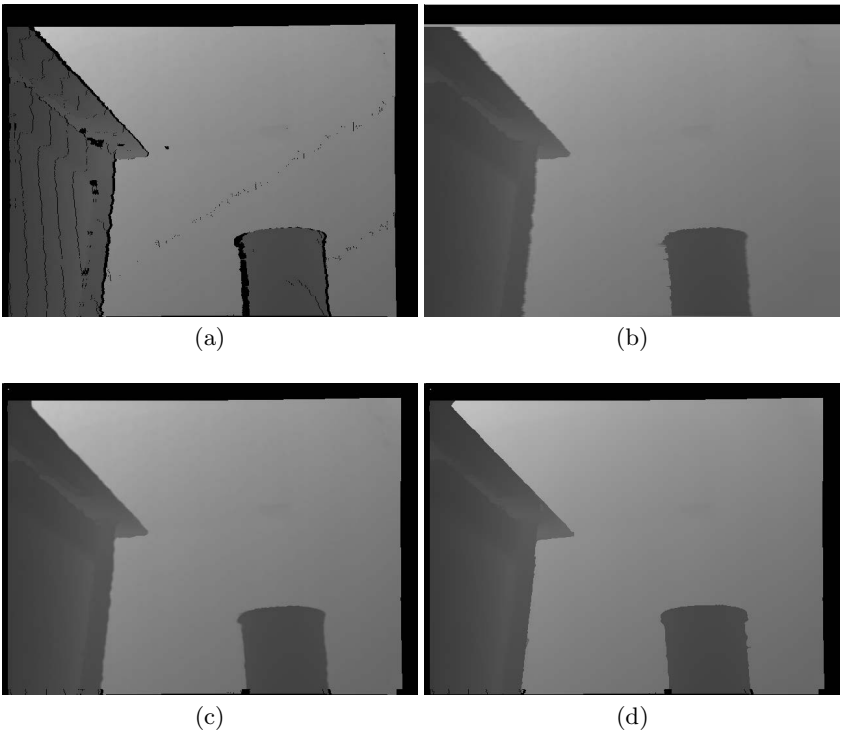


Fig. 4. (a) Example frame of unfiltered depth data generated by the Kinect depth sensor. (b) Frame filtered with method of Brednikov et al. [8]. (c) Frame filtered with method of Wasza et al. [9]. (d) Our method.

in the color image but suffers of course from the fact that it cannot represent smooth depth transitions but rather consists of discrete patches.

The final filtered depth map D^f uses the depth values of D^r only, if D exhibits corrupted depth values. D^f is obtained in the following way:

$$D^f(x,y) = \begin{cases} D^r(x,y) & \text{if } |D(x,y) - D^r(x,y)| > \theta \\ D(x,y) & \text{otherwise} \end{cases} \tag{4}$$

with θ being a threshold.

4 Quantitative Assessment Using Rand Index

It is difficult to obtain quantitative quality results for depth filtering algorithms. This is due to the fact that usually no ground truth depth map is available to compare the filtered depth map with. However, we designed a test method to assess the ability of our algorithm to restore edges.

Recall Figure 1(e) for an example of a corrupted edge in the original depth map. We see that the edge does not correspond very well with the edge in the

Fig. 5. Sample color and depth frame from the *edge test 1* sequence

color stream (green line). Our algorithm, see Figure 1(f), performs way better and we want to quantify this result. To do this, we take a test sequence (color and depth, see Figure 5 for an example) with very simple geometry: It depicts a homogeneous (in terms of depth values) foreground object with a straight edge in front of a homogeneous background. The foreground object also has different color texture than the background.

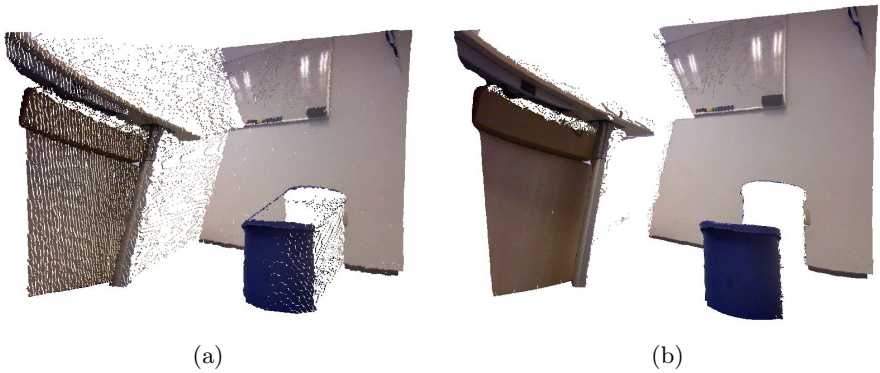
In this situation, we can define two clusterings that divide the scene into foreground and background: C_D is a 2-means clustering of the depth map and C_C is a 2-means clustering of the color stream. If the depth map is aligned with the color stream and does not exhibit cracks or other corruption, then clustering C_D and C_C should be the same.

To determine how similar the clustering C_D and C_C are, we compute the Rand index[15]. The Rand index $\mathbb{R}(\cdot,\cdot) \in [0,\dots 1]$ is a very popular measure to describes how similar two clusterings are. A Rand index of 1 means they are the same whereas 0 means they are completely different. In our situation a high Rand index indicates a very good correlation between the color stream and the (filtered) depth stream.

5 Experimental Results

5.1 Qualitative Results

Figure 3 shows some results of our method compared with other methods. Berdnikov et al. [8] address special issues of the Microsoft Kinect depth camera. Holes that occur due to the offset between color and depth camera are closed by background extraction and other holes by row-wise linear interpolation. The focus of Wasza et al. [9] lies on porting simple filters like Gaussian-weighted hole filling [11] and temporal smoothing as well as edge-preserving denoising on the GPU to achieve very high frame rates. We used in our experiments a window size of 9×9 for hole filling and 10 frames for temporal smoothing. A second example can be found in Figure 4.



(a) (b)

Fig. 6. Problem of intermediate depth values: (a) Depth filtering method of Wasza et al. [9] as a representative of smoothing methods. (b) Our method.

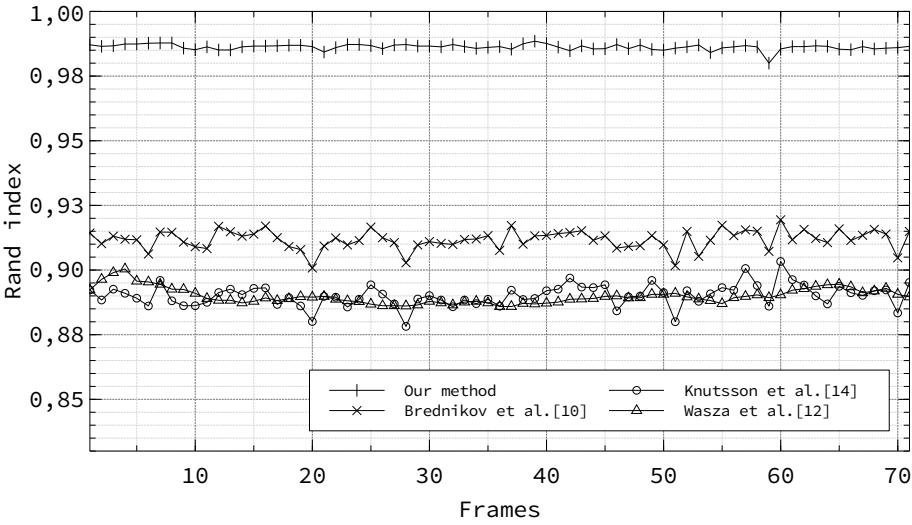


Fig. 7. Rand index values for different depth filtering algorithms. A value near to 1 means a very good correlation with the color stream.

Our method closes all holes and in contrast to other methods, it can restore edges. This behavior can also be seen in Figure 2(c) and (d). In contrast to methods that apply Gaussian or average smoothing, our method does not introduce undesired "intermediate" depth values which can lead to errors when performing 3D reconstruction. These intermediate depth values occur, when smoothing methods interpolate between foreground and background depth values. The interpolated depth values will not correspond with actual physical objects. See Figure 6 for an example.

Table 1. Average Rand Index Values for two different test sequences

Sequence 1	Sequence 2	
Our method	0.9865	0.9778
Berdnikov[8]	0.9118	0.9129
Knutsson[11]	0.8952	0.9120
Wasza[9]	0.8899	0.9121

5.2 Quantitative Results

We measured the capability of our algorithm to restore edges with our proposed method described in Section 4. We recorded two test sequences that met the geometry constraint. Figure 7 shows the Rand index values for all frames of test sequence 1. We can see that our algorithm clearly outperforms all other algorithms. Table 1 shows the average rand values for both test sequences. Again we can see that our method outperforms other methods in terms of edge restoration.

6 Conclusion

We have presented a method to increase the spatial accuracy of depth maps using edge information of the associated color stream. Our method can reliably enhance corrupted edges in the depth stream and outperforms common algorithms. The second contribution is a new quantitative measuring technique to asses the edge reconstruction capabilities of a depth filtering algorithm which is based on measuring clustering similarities using the Rand Index.

Future work aims at the inclusion of inter-frame information to enforce time-consistency and further reduce edge artifacts. Additionally, we want to investigate how a temporally stable color segmentation can help improve the quality of our results.

References

1. Fehn, C., de la Barre, R., Pastoor, R.S.: Interactive 3-DTV-Concepts and Key Technologies. Proceedings of the IEEE 94, 524–538 (2006)
2. Schmeing, M., Jiang, X.: Depth Image Based Rendering: A Faithful Approach for the Disocclusion Problem. In: Proc. 3DTV-Conf.: The True Vision - Capture, Transmission and Display of 3D Video (3DTV-CON), pp. 1–4 (2010)
3. Schmeing, M., Jiang, X.: Time-Consistency of Disocclusion Filling Algorithms in Depth Image Based Rendering. In: Proc. 3DTV Conf.: The True Vision - Capture, Transmission and Display of 3D Video (3DTV-CON), pp. 1–4 (2011)
4. Zitnick, C.L., Kang, S.B., Uyttendaele, M., Winder, S., Szeliski, R.: High-quality Video View Interpolation using a Layered Representation. ACM Transactions on Graphics 23(3), 600–608 (2004)

5. Smolic, A., Mueller, K., Merkle, P., Fehn, C., Kauff, P., Eisert, P., Wiegand, T.: 3D Video and Free Viewpoint Video - Technologies, Applications and MPEG Standards. In: 2006 IEEE International Conference on Multimedia and Expo, pp. 2161–2164 (2006)
6. Zinger, S., Do, L., de With, P.: Free-viewpoint Depth Image Based Rendering. Journal of Visual Communication and Image Representation 21(5-6), 533–541 (2010)
7. Matyunin, S., Vatolin, D., Berdnikov, Y., Smirnov, M.: Temporal Filtering for Depth Maps generated by Kinect Depth Camera. In: Proc. 3DTV Conf.: The True Vision - Capture, Transmission and Display of 3D Video (3DTV-CON), pp. 1–4 (2011)
8. Berdnikov, Y., Vatolin, D.: Real-time Depth Map Occlusion Filling and Scene Background Restoration for Projected-Pattern-based Depth Camera. In: 21st International Conference on Computer Graphics and Vision (GraphiCon 2011), pp. 1–4 (2011)
9. Wasza, J., Bauer, S., Hornegger, J.: Real-time Preprocessing for Dense 3-D Range Imaging on the GPU: Defect Interpolation, Bilateral Temporal Averaging and Guided Filtering. In: IEEE International Conference on Computer Vision Workshops (ICCV Workshops), pp. 1221–1227 (2011)
10. Min, D., Lu, J., Do, M.: Depth Video Enhancement Based on Weighted Mode Filtering. IEEE Transactions on Image Processing 21(3), 1176–1190 (2012)
11. Knutsson, H., Westin, C.F.: Normalized and Differential Convolution. In: 1993 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 515–523 (1993)
12. Diebel, J., Thrun, S.: An Application of Markov Random Fields to Range Sensing. In: Advances in Neural Information Processing Systems 18, pp. 291–298. MIT Press, Cambridge (2006)
13. Liu, M.Y., Tuzel, O., Ramalingam, S., Chellappa, R.: Entropy Rate Superpixel Segmentation. In: 2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2097–2104 (2011)
14. Beucher, S., Lantuejoul, C.: Use of Watersheds in Contour Detection. In: International Workshop on Image Processing: Real-time Edge and Motion Detection/Estimation, Rennes, France (1979)
15. Rand, W.M.: Objective Criteria for the Evaluation of Clustering Methods. Journal of the American Statistical Association 66(336), 846–850 (1971)

Stereoscopic Image Inpainting Considering the Consistency of Texture Similarity

Ayako Abe and Ikuko Shimizu

Tokyo University of Agriculture and Technology,
2-24-16 Nakacho, Koganei-City, Tokyo, Japan
{aabe,ikuko}@cc.tuat.ac.jp

Abstract. Recently, as 3D display devices become popular, automatic editing methods for stereoscopic image become important. In this paper, we propose an inpainting method for a stereoscopic image considering the texture similarity. By extending the conventional image inpainting method for an image considering the texture similarity to for a stereoscopic image by considering the consistency of two images, natural results are obtained by our method. As in the conventional methods for a stereoscopic image inpainting, first the depth maps are recovered and then the color images are inpainted based on the depth maps in our method. Consistency of the depth maps is guaranteed in our method which are important for the inpainting of color image. For color image inpainting, similar textures are searched considering not only the depth layer but also the depth values of respective pixels. Experimental results demonstrate the efficiency of our method.

Keywords: stereoscopic image, inpainting, texture consistency, depth, disparity, 3D warping.

1 Introduction

There has recently been a high demand for the development of techniques for the automatic generation and efficiency editing of stereoscopic images, because 3D displays are becoming increasingly popular. They allow users to feel the depth of scene on a screen by showing different images to the left and right eye. A pair of images shown to the left and right eyes is called a 'stereoscopic image'.

Image inpainting is one of the most important among the many existing editing techniques. It reconstructs an input image that contains "hole(s)", so that it looks natural. In this paper, we denote an inpainting method for an (ordinary) image as the '2D inpainting method' and an inpainting method for a stereoscopic image as the '3D inpainting method'.

There have been a lot of 2D inpainting methods proposed in the literature. There are basically two categories for 2D inpainting methods: one is to inpaint from around the hole(s) while taking the color continuity into account [1–3], and the other is to inpaint using the texture(In this paper, 'texture' represents the intensity pattern.) of the other parts in the same image [4–8]. The latter has recently been considered to be better because it can be used for an image with a big hole.

X. Jiang et al. (Eds.): WDIA 2012, LNCS 7854, pp. 78–86, 2013.

For 3D inpainting, even if we apply the best 2D inpainting methods to the respective images, the results will be unnatural because there seems to be inconsistent intensity between two images consisting of a stereoscopic image and the unnatural depth in the hole. Therefore, 3D inpainting methods have been proposed [9–11]. In these methods, not only color images but also the depth maps of two images are inpainted. In Wang's method [9], both color images and depth maps are iteratively inpainted. However, the unnatural artifacts occur because this method does not ensure a convergence of the iteration. Furthermore, it allows for corresponding pixels to have slightly different intensity values from each other. One method proposed by Hervieu [10] ensures a convergence and the color consistency by using a two steps algorithm. In the first step, the depth maps are inpainted, and in the second step, the color images are inpainted under a constraint in which the corresponding pixels have the same intensity value using the depth maps. However, this method can be applied only to images whose depth maps are relatively smooth. Then, Hervieu [11] proposed another method that is applicable to an image whose depth maps have various values by introducing the assumption that the depth maps consist of some depth planes. However, the depth map inpainting in this method does not guarantee consistency between corresponding pixels. This leads to an unnatural color image inpainting because the depth maps are important for ensuring the color image consistency. In addition, this method extended the 2D inpainting method [7] in color image inpainting, but it has a drawback in that it tends to induce discontinuous textures when the image contains complex ones. So, similar pixels are searched for without taking the depth into consideration, and if two or more objects are contained in one depth layer, the inpainted pixels will be unnatural. To prevent this, the user may tune the parameter, but it is difficult for users to do this themselves.

In this paper, a 3D inpainting method is proposed that takes the texture similarity into consideration to produce a consistent inpainting between two images and natural inpainting when the input image has a complex texture. Our method guarantees the consistency of color images and depth maps between two images. The 2D inpainting method [8] is extended to a 3D one that is applicable to an image with a complex texture and does not induce a discontinuous texture by taking the texture similarity into account to obtain natural results for an image with a complex texture. In this process, similar pixels are searched for in the same depth layer while taking the depth of the pixel into consideration. By using our method, similar textures are obtained that are supposed to belong to the same object without needing any tuning parameters.

2 Overview of Proposed Method

An overview our 3D inpainting method is shown in Fig.1. A two step algorithm is used in our method as in the conventional methods [10, 11]; depth maps are inpainted in the first step, and the color images are inpainted to ensure convergence in the second step. Our method guarantees there is consistency between the corresponding pixels in both the color images and the depth maps of two images that consist of a stereoscopic image.

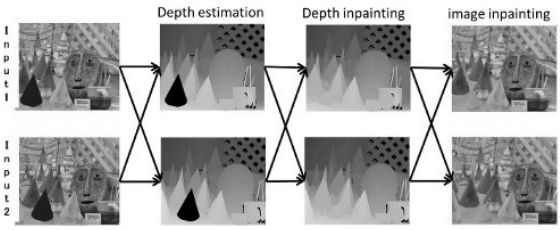


Fig. 1. Overview of proposed method

3 Estimation of Depth Map

First, we estimate the depth map of each image consisting of a stereoscopic image. There are many methods for the estimation of depth maps. For example, the graphcut algorithm [12] is applicable.

Two depth maps for the respective images are estimated by establishing a correspondence between the two images for each pixel of each image. Note that there are some holes in the depth map that correspond to the holes in the image in this step.

4 Inpainting of Depth Map

The holes in the depth maps are inpainted by using the following two steps: the initialization by 3D warping described in Sec.4.1 and inpainting by extension of the depth layer described in Sec.4.2.

4.1 Initialization of Depth Map

We denote the pixels in the left and right images as (x^l, y) and (x^r, y) and the depth maps of the respective images as D_l and D_r. The relation between the corresponding pixels of the depth maps are as follows:

$$D_l(x^l, y) = D_r(x^l - D_l(x^l, y), y), \quad D_r(x^r, y) = D_l(x^r + D_r(x^r, y), y). \quad (1)$$

If pixel (x^l, y) in the left image is in the hole and the corresponding pixel in the right image is not, $D_l(x^l, y)$ is obtained by $D_r(x^l - D_l(x^l, y), y)$. This transformation is called 3D warping. If $D_r(x^r, y)$ is in the hole and the corresponding pixel is not, $D_r(x^r, y)$ is obtained by using 3D warping in the same way. When multiple pixels are warped into one pixel, the largest depth is selected since it corresponds to the closest object to the camera.

However, sometimes the initialization by using 3D warping fails because the object closest to the camera is occluded by the object in the hole and a smaller depth than the true depth is recovered (Fig.2). The 3D warping property [13], in which the width of a hole is the same as the difference in the depths between

the pixels in both sides of the hole, is used to eliminate such a false depth value. The pixel which does not satisfy this property is supposed to have a false depth value and the depth value is eliminated. For the depth map of the image on the left, the false pixel is searched for from the left side of the hole (Fig.3(a)). On the other hand, for the depth map of the image on the right, the right side of the hole is searched for (Fig.3(b)).

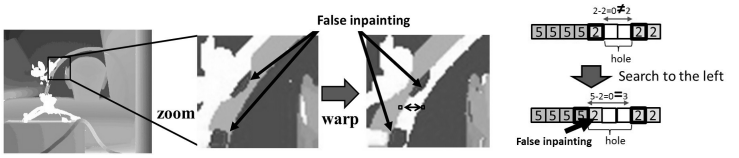


Fig. 2. Error in depth map

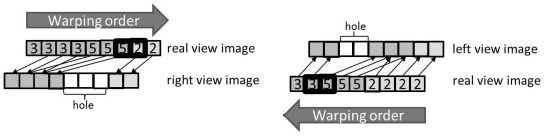


Fig. 3. (a) 3D warping of image on right. (b) 3D warping of image on left. The number represents the depth for each pixel.

4.2 Inpainted by Extension of Depth Layer

Next, the rest of the hole in the depth map is inpainted. We assume that the depth map consists of multiple depth planes, called depth layers [11]. The hole of the depth map is inpainted by extending its neighboring depth layer, as shown in [11].

However, the recovered depth values of each pixel do not guarantee to satisfy Eqn. (1). Therefore, our method iteratively estimates the depth of the hole until all the pixels satisfy Eqn.(1). Concretely, the depth that does not satisfy Eqn. (1 is eliminated and the neighboring depth layer is re-extended to the hole. These processes are repeated. The hole of the depth will shrink when these processes are repeated and the depth of the hole will converge to satisfy Eqn. (1).

5 Inpainting of Color Image

After completing inpainting the depth maps, the color images are inpainted by using the following two steps: the initialization by using 3D warping described in Sec.5.1 and the inpainting by minimizing the energy function while taking the texture similarity describe in Sec.5.2 into account.

5.1 Initialization of Color Image

We denote the intensities of the respective images as I_l and I_r. The relation between the corresponding pixels of the color images are as follows:

$$I_l(x^l, y) = I_r(x^l - D_l(x^l, y), y), \quad I_r(x^r, y) = I_l(x^r + D_r(x^r, y), y). \tag{2}$$

As for the depth map, 3D warping is used to initialize the hole of the color image by using Eqn. (2). In this step, the pixels in the hole that are observed in one of the stereoscopic images are recovered.

5.2 Inpainting of Color Image Taking Consistency of Corresponding Pixels' Intensity and Texture Similarity into Consideration

Next, the rest of the hole is inpainted by minimizing the energy function while taking the texture similarity into consideration. Among the many 2D inpainting methods currently proposed, a method by Kawai [8] is known to have the necessary flexibility to texture a pattern and produce natural results by taking the similarity in texture into consideration even when not exactly the same texture in the non-missing region as in the hole exists. Therefore, we extend this method to 3D inpainting.

Extension to 3D inpainting is done by adding some constraints in which the corresponding pixels between the left and right images have the same intensities. The energy function for 3D inpainting is as follows:

$$E = E_l + E_r, \quad \text{s.t. } g = I_l(x_i^l, y_i) - I_r(x_i^r, y_i) = 0, \tag{3}$$

where E_l and E_r are the energies in left and right images, respectively, and they are the same as the energy functions defined in 2D inpainting [8], and (x_i^l, y_i) and $(x_i^r, y_i) = (x_i^l + D_l(x_i^l, y_i), y_i)$ are the i-th corresponding pixel pair.

The minimization of E is done by using a greedy algorithm as follows: (i) update the similar texture patterns for respective pixels $(x_i^r, y_i) and (x_i^l, y_i)$, and then (ii) update the intensity values in the hole. These two steps are repeated until convergence occurs. In addition, we create an image pyramid and minimize the energy from coarse to fine.

In Step (i), similar texture patterns are searched for in only the depth layer that belong to each pixel in the hole. In this step, our method restricts the search region of a similar pixel to the same depth layer while taking the depth value of the pixel into account. By doing this, our method prevents searches for a different object in the same layer. By searching for similar pixels for only the pixels belonging to the same depth layer, unnatural inpainting is supposed to prevented. In addition, the computational time will be reduced by restricting the search region.

In Step (ii), energy E, which is the sum of both images' energy, is minimized. The relation between the E can be rewritten using energy elements $E_l(\boldsymbol{x}_i^l)$, and $the E_r(\boldsymbol{x}_i^r)$ of the respective images' respective pixels is as follows:

$$E = \sum_{i=1}^{N_\Omega} (E_l(\boldsymbol{x}_i^l) + E_r(\boldsymbol{x}_i^r)) + C, \tag{4}$$

where $\boldsymbol{x}_i^l = (x_i^l, y_i)$, $\boldsymbol{x}_i^r = (x_i^r, y_i) = (x_i^l - d, y_i)$, N_Ω is the number of pixels included in the hole, Ω' is the region including hole Ω and the region adjacent to Ω, and C is the energy in $\overline{\Omega} \cap \Omega'$. Note that C is a constant value in this step, because a similar pattern is fixed. The Lagrange multiplier are used to minimize E in Eqn. (4). We define $J = E + \lambda g$ using constraint $g = 0$ in Eqn. (3) and the variables are $I_l(x_i^l, y_i), I_r(x_i^r, y_i)$. Then, the minimization of E is equivalent to these partial differential equations as follows:

$$\frac{\partial J}{\partial I_l(x_i^l, y_i)} = \frac{\partial E}{\partial I_l(x_i^l, y_i)} + \lambda = 0, \tag{5}$$

$$\frac{\partial J}{\partial I_r(x_i^r, y_i)} = \frac{\partial E}{\partial I_r(x_i^r, y_i)} - \lambda = 0, \tag{6}$$

$$I_l(x_i^l, y_i) - I_r(x_i^r, y_i) = 0. \tag{7}$$

$I_l(x_i^l, y_i), I_r(x_i^r, y_i)$ is analytically obtained from Eqn. (5), (6), (7) by eliminating λ from Eqns.(5) and (6):

$$I_l(x_i^l, y_i) = I_r(x_i^r, y_i) = \frac{\sum_{\boldsymbol{p} \in W} (w_l(\boldsymbol{x}_i^l + \boldsymbol{p}) \alpha(\boldsymbol{x}_i^l + \boldsymbol{p}) f(\boldsymbol{x}_i^l + \boldsymbol{p}) + w_r(\boldsymbol{x}_i^r + \boldsymbol{p}) f(\boldsymbol{x}_i^r + \boldsymbol{p})) I_l(f(\boldsymbol{x}_i^l + \boldsymbol{p}) - \boldsymbol{p})}{\sum_{\boldsymbol{p} \in W} w_l(\boldsymbol{x}_i^l + \boldsymbol{p}) + w_r(\boldsymbol{x}_i^r + \boldsymbol{p})}, \tag{8}$$

where w_r and w_l are the weights for each pixel of the respective images, $f(\boldsymbol{x})$ is a similarity pattern of $\boldsymbol{x}$, and α is a correction parameter, which allows for the variation in intensity defined in [8].

6 Experiment Results

In the experimental results in this section, we used the images and depth maps from the Middleburry stereo datasets (`http://vision.middlebury.edu/stereo/data/`).

6.1 Initialization of Depth Map

As mentioned in 4.1, this initialization process consists of two steps: (i) 3D warping and (ii) elimination of the false inpainting. In this section, the depth map created by using our method is compared with the depth maps when using a method without these steps to show the effectiveness of our initialization of the depth map.

A depth map without the initialization of the depth map from Fig.4(c) and its inpainted depth map is shown in Fig.4(f). The depth map by using only 3D warping without elimination of the false inpainting in Fig.4(e) and its inpainted depth map are shown in Fig.4(g). The depth map initialized by our method in Fig.4(e) and its inpainted depth map are shown in Fig.4(h). The effectiveness of our depth initialization is proven from these results, especially in the region around the round-arched object.

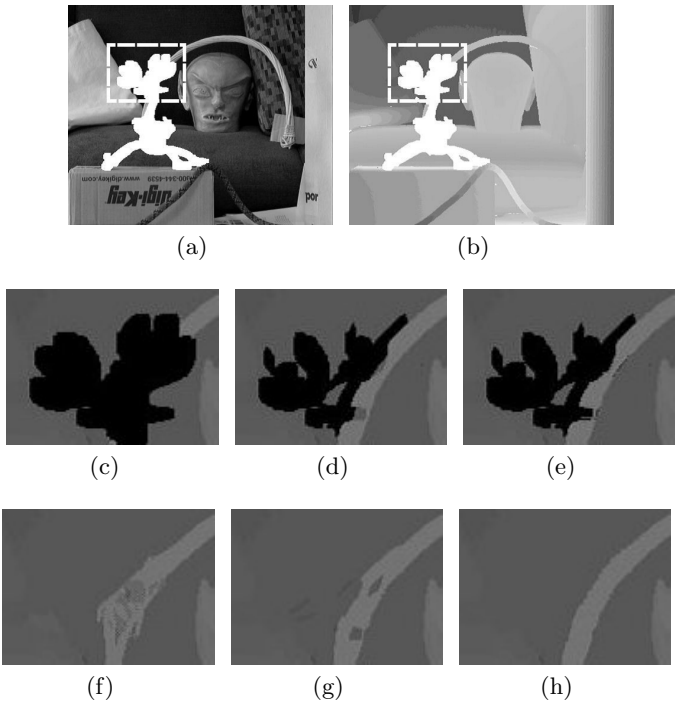


Fig. 4. Results of initialization of depth map: (a)(b)input image and corresponding depth map (white region is missing.) (c) (d) shows the results in the box with dash line (c) without initialization, (d) only by using 3D warping, (e) without using a method, and (f), (g), and (h) are the respective results by our method.

6.2 Consistency of Corresponding Pixels' Depth Values

In our method, the depth inpainting is repeated until the corresponding pixels have the same depth values, as described in Sec.4.2. The depth map without a consistency check of the corresponding pixels is shown in Fig.5(c). In contrast, Fig.5(d) shows the results by using our method. Better results are obtained by using our method.

6.3 Comparison with Our Method and the Conventional Method

The results when using our method and those when using the conventional method are shown in Fig.6. In Fig.6(c), the bottom of a red cone is not natural. In contrast, the results by using our method shown in Fig.6(d), looks natural. One of the reasons for this is that our method initializes the color images by using 3D warping based on the consistent depth maps.

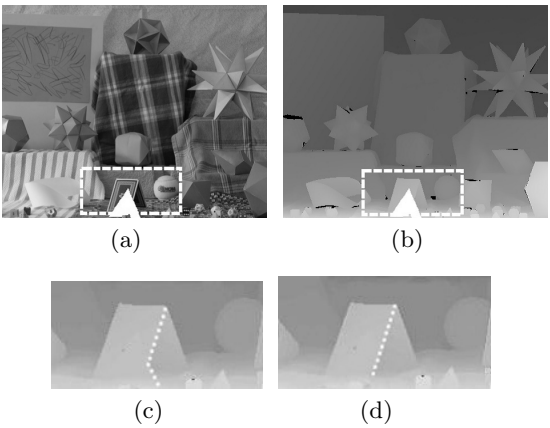


Fig. 5. Results when taking consistency of corresponding pixels' depths into consideration (a)(b)input image and corresponding depth map (white region is missing.) (c)without a consistency check of the corresponding pixels (d)by using our method. White dash line shows the border of the depth value.

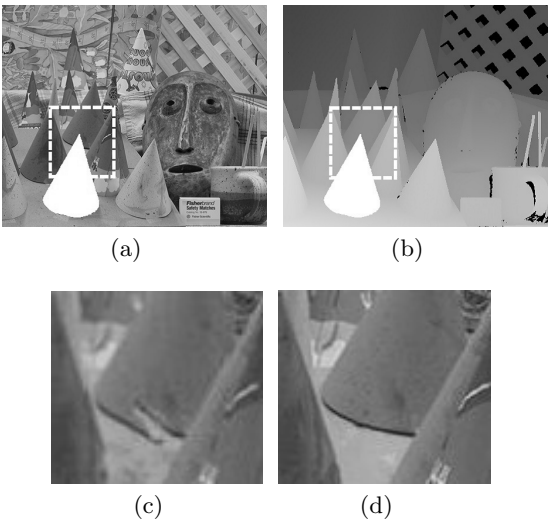


Fig. 6. Comparison of 3D inpainting: (a)(b)input image and corresponding depth map (white region is missing.) (c) conventional method [11] and (d) proposed method.

7 Conclusion

We proposed an inpainting method for a stereoscopic image that takes the texture similarity based on the consistency of the corresponding pixels between two images consisting of a stereoscopic image into consideration. Just like for conventional methods for stereoscopic image inpainting, our method consists of two

steps to guarantee convergence: the first step involves inpainting the depth maps and the second step the inpainting of the color images based on the depth maps.

In the depth map inpainting, our method guarantees the consistency of the depth maps, which plays an important role in the second step. In the color image inpainting, the conventional inpainting method [8], which causes discontinuous and unclear textures even for an input image with a complex texture, was extended to a 3D inpainting method that takes the consistency of corresponding pixels between two images into consideration. In addition, similar textures are searched for by taking not only the depth layer but also the depth values of the respective pixels into consideration.

We confirmed that more natural results for a stereoscopic image were obtained by using our method than by the conventional method. To extend our method for stereoscopic video by taking into account that the depth varies smoothly along a timeline will be one of our future works.

References

1. Masnou, S., Morel, J.-M.: Level lines based disocclusion. In: Proc. of ICIP, vol. 3, pp. 259–263 (1998)
2. Bertalmio, M., Sapiro, G., et al.: Image Inpainting. In: Proc. of SIGGRAPH, pp. 417–424 (2000)
3. Chan, T.F., Shen, J.: Non-Texture Inpainting by Curvature-Driven Diffusions (CDD). Journal of Visual Communication and Image Representation 12, 436–449 (2001)
4. Efros, A.A., Leung, T.K.: Texture synthesis by non-parametric sampling. In: Proc. of ICCV, vol. 2, pp. 1033–1038 (1999)
5. Komodakis, N.: Image Completion Using Global Optimization. In: Proc. of CVPR, vol. 1, pp. 442–452 (2006)
6. Wexler, Y., Shechtman, E., et al.: Space-Time Completion of Video. IEEE Trans. on PAMI 29(3), 463–476 (2007)
7. Criminisi, A., Perez, P., et al.: Region filling and object removal by exemplar-based inpainting. IEEE Trans. on IP 13(9), 1200–1212 (2004)
8. Kawai, N., Sato, T., et al.: Image Inpainting Considering Brightness Change and Spatial Locality of Textures. In: Proc. of VISAPP, vol. 1, pp. 66–73 (2008)
9. Wang, L., Jin, H., et al.: Stereoscopic inpainting: Joint color and depth completion from stereo images. In: Proc. of CVPR, pp. 1–8 (2008)
10. Hervieu, A., Papadakis, N., et al.: Stereoscopic Image Inpainting: Distinct Depth Maps and Images Inpainting. In: Proc. of ICPR, pp. 4101–4104 (2010)
11. Hervieu, A., Papadakis, N., et al.: Stereoscopic image inpainting using scene geometry. In: Proc. of ICME, pp. 1–6 (2011)
12. Boykov, Y., Kolmogorov, V.: An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. IEEE Trans. on PAMI 26(9), 1124–1137 (2004)
13. Horng, Y.R., et al.: Stereoscopic images generation with directional Gaussian filter. In: Proc. of ISCAS, pp. 2650–2653 (2010)

A Dynamic MRF Model for Foreground Detection on Range Data Sequences of Rotating Multi-beam Lidar

Csaba Benedek[1], Dömötör Molnár[1,2], and Tamás Szirányi[1,2,⋆]

[1] Distributed Events Analysis Research Laboratory
Institute for Computer Science and Control, Hungarian Academy of Sciences
Kende utca 13-17, H-1111 Budapest, Hungary
[2] Department of Information Technology, Péter Pázmány Catholic University
Práter utca 50/A, H-1083 Budapest, Hungary
firstname.lastname@sztaki.mta.hu

Abstract. In this paper, we propose a probabilistic approach for foreground segmentation in 360°-view-angle range data sequences, recorded by a rotating multi-beam Lidar sensor, which monitors the scene from a fixed position. To ensure real-time operation, we project the irregular point cloud obtained by the Lidar, to a cylinder surface yielding a depth image on a regular lattice, and perform the segmentation in the 2D image domain. Spurious effects resulted by quantification error of the discretized view angle, non-linear position corrections of sensor calibration, and background flickering, in particularly due to motion of vegetation, are significantly decreased by a dynamic MRF model, which describes the background and foreground classes by both spatial and temporal features. Evaluation is performed on real Lidar sequences concerning both video surveillance and traffic monitoring scenarios.

Keywords: rotating multi-beam Lidar, MRF, motion segmentation.

1 Introduction

Foreground detection and segmentation are a key issues in automatic visual surveillance. Foreground areas usually contain the regions of interest, moreover, an accurate object-silhouette mask can directly provide useful information for, among others, people or vehicle detection, tracking or activity analysis.

Range image sequences offer significant advantages versus conventional video flows for scene segmentation, since geometrical information is directly available [1], which can provide more reliable features than intensity, color or texture values [2,3]. Using Time-of-Light (ToF) cameras [1] or scanning Lidar sensors [4]

⋆ This work was partially funded by the i4D Project of MTA SZTAKI and by the Hungarian Research Fund (Grants OTKA #76159 and #101598). C. Benedek was also supported by the János Bolyai Research Scholarship of the Hungarian Academy of Sciences.

X. Jiang et al. (Eds.): WDIA 2012, LNCS 7854, pp. 87–96, 2013.

enable recording range images independently of the outside illumination conditions and we can also avoid artifacts of stereo vision techniques. From the point of view of data analysis, ToF cameras record depth image sequences over a regular 2D pixel lattice, where established image processing approaches, such as Markov Random Fields (MRFs) can be adopted for smooth and observation consistent segmentation [3]. However, such cameras have a limited Field of View (FoV), which can be a drawback for surveillance and monitoring applications.

Rotating multi-beam Lidar systems (RMB-Lidar) provide a 360° FoV of the scene, with a vertical resolution equal to the number of the sensors, while the horizontal angle resolution depends on the speed of rotation (see Fig. 1). For efficient data processing, the 3-D RMB-Lidar points are often projected onto a cylinder shaped range image [4,5]. However, this mapping is usually ambiguous: On one hand, several laser beams with slight orientation differences are assigned to the same pixel, although they may return from different surfaces. As a consequence, a given pixel of the range image may represent different background objects at the consecutive time steps. This ambiguity can be moderately handled by applying multi-modal distributions in each pixel for the observed background-range values [4], but the errors quickly aggregate in case of dense background motion, which can be caused e.g. by moving vegetation. On the other hand, due to physical considerations, the raw data of distance, pitch and angle provided by the RMB-Lidar sensor must undergo a strongly non-linear calibration step to obtain the Euclidean point coordinates [6], therefore, the density of the points mapped to the regular lattice of the cylinder surface may be inhomogeneous. To avoid the above artifacts of background modeling, [5] has directly extracted the foreground objects from the range image by mean-shift segmentation and blob detection. However, we have experienced that if the scene has simultaneously several moving and static objects in a wide distance range, the moving pedestrians are often merged into the same blob with neighboring scene elements.

Instead of projecting the points to a range image, another way is to solve the foreground detection problem in the spatial 3D domain. However, 3D object level techniques principally aim to extract the bounding boxes of the pedestrians [7], instead of labeling each foreground point of the input cloud, which may be necessary for activity recognition by e.g. skeleton fitting to the silhouettes. MRF techniques based on 3D spatial point neighborhoods are frequently applied in remote sensing [8], however the accuracy is low in case of small neighborhoods, otherwise the computational complexity rapidly increases.

In this paper, we propose a hybrid approach for dense foreground-background point labeling in a point cloud obtained by a RMB-Lidar system, which monitors the scene from a fixed position. Our method solves the computationally critical spatial filtering steps in the 2D range image domain by an MRF model, however, ambiguities of discretization are handled by joint consideration of true 3D positions and back projection of 2D labels. By developing a spatial foreground model, we significantly decrease the spurious effects of irrelevant background motion, which principally caused by moving tree crowns and bushes. For quantitative evaluation, we have developed a 3D point cloud Ground Truth (GT)

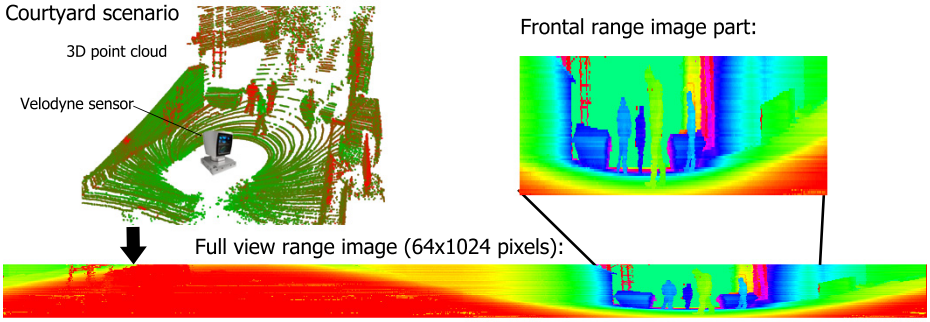


Fig. 1. Point cloud recording and range image formation with a Velodyne HDL 64E RMB-Lidar sensor

annotation tool, and compared the detection results of the proposed model to three reference methods.

2 Problem Formulation and Data Mapping

Assume that the RMB-Lidar system contains R vertically aligned sensors, and rotates around a fixed axis with a possibly varying speed[1]. The output of the Lidar within a time frame t is a *point cloud* of $l^t = R \cdot c^t$ points: $\mathcal{L}^t = \{p_1^t, \ldots, p_{l^t}^t\}$. Here c^t is the number of point *columns* obtained at t, where a given column contains R concurrent measurements of the R sensors, thus c^t depends on the rotation speed. Each point, $p \in \mathcal{L}^t$, is associated to sensor distance $d(p) \in [0, D_{\max}]$, pitch index $\hat{\vartheta}(p) \in \{1, \ldots, R\}$ and yaw angle $\varphi(p) \in [0, 360^\circ]$ parameters. $d(p)$ and $\hat{\vartheta}(p)$ are directly obtained from the Lidar's data flow, by taking the measured distance and sensor index values corresponding to p. Yaw angle $\varphi(p)$ is calculated from the Euclidean coordinates of p projected to the ground plane, since the R sensors have different horizontal view angles, and the angle correction of calibration may also be significant [6].

The goal of the proposed method is at a given time frame t to assign each point $p \in \mathcal{L}^t$ to a label $\omega(p) \in \{\text{fg}, \text{bg}\}$ corresponding to the moving object (i.e. foreground, fg) or background classes (bg), respectively.

For efficient data manipulation, we also introduce a range image mapping of the obtained 3D data. We project the point cloud to a cylinder, whose central basis point is the ground position of the RMB-Lidar and the axis is prependicular to the ground plane. Note that slightly differently from [5], this mapping is also efficiently suited to configurations, where the Lidar axis is tilted do increase the vertical Field of View. Then we stretch a $S_H \times S_W$ sized 2D pixel lattice S on the cylinder surface, whose height S_H is equal to the R sensor number, and

[1] The speed of rotation can often be controlled by software, but even in case of constant control signal, we must expect minor fluctuations in the measured angle-velocity, which may result in different number of points for different 360° scans in time.

the width S_W determines the fineness of discretization of the yaw angle. Let us denote by s a given pixel of S, with $[y_s, x_s]$ coordinates. Finally, we define the $\mathcal{P} : \mathcal{L}^t \to S$ point mapping operator, so that y_s is equal to the pitch index of the point and x_s is set by dividing the $[0, 360°]$ domain of the yaw angle into S_W bins:

$$s \stackrel{\text{def}}{=} \mathcal{P}(p) \text{ iff } y_s = \hat{\vartheta}(p),\ x_s = \text{round}\left(\varphi(p) \cdot \frac{S_W}{360°}\right) \tag{1}$$

3 Background Model

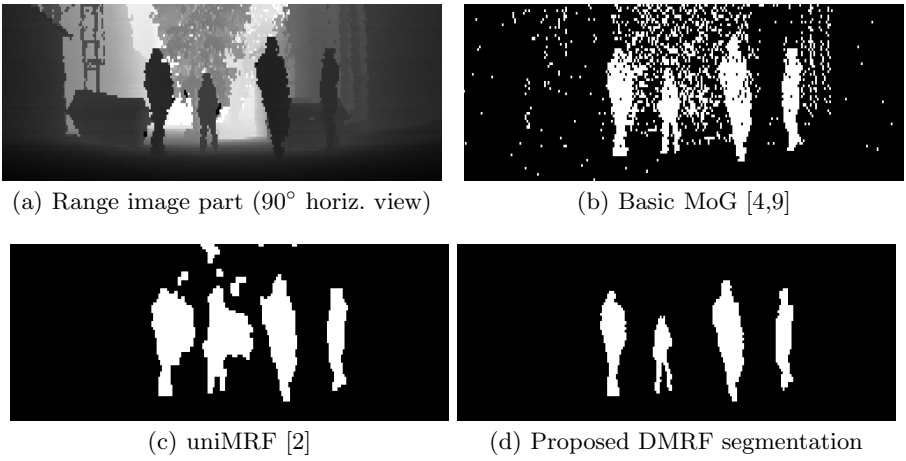
The background modeling step assigns a fitness term $f_{\text{bg}}(p)$ to each $p \in \mathcal{L}^t$ point of the cloud, which evaluates the hypothesis that p belongs to the background. The process starts with a cylinder mapping of the points based on (1), where we use a $R \times S_W^{\text{bg}}$ pixel lattice S^{bg} (R is the sensor number). Similarly to [4], for each s cell of S^{bg}, we maintain a Mixture of Gaussians (MoG) approximation of the $d(p)$ distance histogram of p points being projected to s. Following the approach of [9], we use a fixed K number of components (here $K = 5$) with weight w_s^i, mean μ_s^i and standard deviation σ_s^i parameters, $i = 1 \dots K$. Then we sort the weights in decreasing order, and determine the minimal k_s integer which satisfies $\sum_{i=1}^{k_s} w_s^i > T_{\text{bg}}$(we used here $T_{\text{bg}} = 0.89$). We consider the components with the k_s largest weights as the background components. Thereafter, denoting by $\eta()$ a Gaussian density function, and by $\mathcal{P}^{\text{bg}}$ the projection transform onto S^{bg}, the $f_{\text{bg}}(p)$ background evidence term is obtained as:

$$f_{\text{bg}}(p) = \sum_{i=1}^{k_s} w_s^i \cdot \eta\left(d(p), \mu_s^i, \sigma_s^i\right),\ \text{where } s = \mathcal{P}^{\text{bg}}(p). \tag{2}$$

The Gaussian mixture parameters are set and updated based on [9], while we used $S_W^{\text{bg}} = 2000$ angle resolution, which provided the most efficient detection rates in our experiments. By thresholding $f_{\text{bg}}(p)$, we can get a dense foreground/background labeling of the point cloud [4,9] (referred later as *Basic MoG* method), but as shown in Fig. 5(a),(c), this classification is notably noisy in scenarios recorded in large outdoor scenes.

4 DMRF Approach on Foreground Segmentation

In this section, we propose a Dynamic Markov Random Field (DMRF) model to obtain smooth, noiseless and observation consistent segmentation of the point cloud sequence. Markov Random Fields (MRFs) [10] are widely used for image segmentation or image restoration task since the early eighties, due to their ability to simultaneously embed a data model, reflecting the knowledge on the image, and some prior constraint about the solution given by some expertise of the problem. MRFs model the searched image (restored or segmented image) as a realization of a random field. The solution is then obtained by maximizing the



(a) Range image part (90° horiz. view) (b) Basic MoG [4,9]

(c) uniMRF [2] (d) Proposed DMRF segmentation

Fig. 2. Foreground segmentation in a range image part with three different methods

density of this field. Prior information embedded in such models usually concern spatial smoothness of the solution.

Since MRF optimization is computationally intensive [10], we define our DMRF model in the range image space, and 2D image segmentation is followed by a point classification step to handle ambiguities of the mapping. As defined by (1) in Sec. 2, we use a $\mathcal{P}$ cylinder projection transform to obtain the range image, with a $S_W = \hat{c} < S_W^{\text{bg}}$ grid with, where $\hat{c}$ denotes the expected number of point columns of the point sequence in a time frame. By assuming that the rotation speed is slightly fluctuating, this selected resolution provides a dense range image, where the average number of points projected to a given pixel is around 1. Let us denote by $P_s \subset \mathcal{L}^t$ the set of points projected to pixel s. For a given direction, foreground points are expected being closer to the sensor than the estimated mean background range value. Thus, for each pixel s we select the closest projected point $p_s^t = \text{argmin}_{p \in P_s} d(p)$, and assign to pixel s of the range image the $d_s^t = d(p_s^t)$ distance value. For 'undefined' pixels ($P_s = \emptyset$), we interpolate the distance from the neighborhood. For spatial filtering, we use an eight-neighborhood system in S, and denote by $N_s \subset S$ the neighbors of pixel s.

Our proposed DMRF segmentation model uses the following definitions:

- S - set of pixels of the range image lattice
- $X^t = \{d_s^t | \forall s \in S\}$ - set of image data at time t (d_s^t is the range value assigned to pixel s).
- $Q = \{\text{fg}, \text{bg}\}$ - class labels
- $\Omega^t = \{\omega_s^t | \forall s \in S\}$ - global labeling of the range image at time t ($\omega_s \in Q$ is the label of pixel s at time t)

We aim to find the labeling which maximizes a $P(\Omega^t | X^t, \Omega^{t-1}) \propto P(X^t | \Omega^t) \cdot P(\Omega^t | \Omega^{t-1})$ probability, so it will minimize the energy:

$$E = -\log P(\Omega^t | X^t, \Omega^{t-1}) = -\log P(X^t | \Omega^t) - \log P(\Omega^t | \Omega^{t-1}) + \text{const} \quad (3)$$

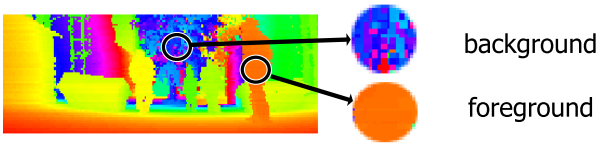


Fig. 3. Demonstrating the different local range value distributions in the neighborhood of a given foreground and background pixel, respectively

where the $-\log P(X^t|\Omega^t)$ part is called the data energy and $-\log P(\Omega^t|\Omega^{t-1})$ is the prior energy.

Next, we assign to each $s \in S$ foreground and background data energy (i.e. negative fitness) terms, which describe the class memberships based on the observed $d(s)$ values. The background energies are directly derived from the parametric MoG probabilities using (2):

$$\varepsilon^t_{\mathrm{bg}}(s) = -\log\left(f_{\mathrm{bg}}(p^t_s)\right).$$

For description of the foreground, using a constant $\varepsilon_{\mathrm{fg}}$ could be a straightforward choice [2] (we call this approach *uniMRF*), but this uniform model results in several false alarms due to background motion and quantization artifacts. Instead of temporal statistics, we use spatial distance similarity information to overcome this problem by using the following assumption: whenever s is a foreground pixel, we should find foreground pixels with similar range values in the neighborhood (Fig. 3). For this reason, we use a non-parametric kernel density model for the foreground class:

$$\varepsilon^t_{\mathrm{fg}}(s) = \sum_{r\in N_s} \zeta(\varepsilon^t_{\mathrm{bg}}(r), \tau_{\mathrm{fg}}, m_\star) \cdot k\left(\frac{d^t_s - d^t_r}{h}\right),$$

where h is the kernel bandwidth and $\zeta : \mathbb{R} \to [0,1]$ is a sigmoid function:

$$\zeta(x,\tau,m) = \frac{1}{1+\exp(-m\cdot(x-\tau))}.$$

We use here a uniform kernel: $k(x) = \mathbf{1}\{|x| \le 1\}$, where $\mathbf{1}\{.\} \in \{0,1\}$ is the binary indicator function of a given event.

To formally define the range image segmentation task, to each pixel $s \in S$, we assign a $\omega^t_s \in \{\mathrm{fg}, \mathrm{bg}\}$ class label so that we aim to minimize the following form of the (3) energy function:

$$E = \sum_{s\in S} V_D(d^t_s|\omega^t_s) + \sum_{s\in S} \underbrace{\sum_{r\in N_s} \alpha\cdot\mathbf{1}\{\omega^t_s \neq \omega^{t-1}_r\}}_{\xi^t_s} + \sum_{s\in S} \underbrace{\sum_{r\in N_s} \beta\cdot\mathbf{1}\{\omega^t_s \neq \omega^t_r\}}_{\chi^t_s}, \quad (4)$$

where $V_D(d^t_s|\omega^t_s)$ denotes the data term, while ξ^t_s and χ^t_s are the temporal and spatial smoothness terms, respectively, with $\alpha > 0$ and $\beta > 0$ constants. Let us

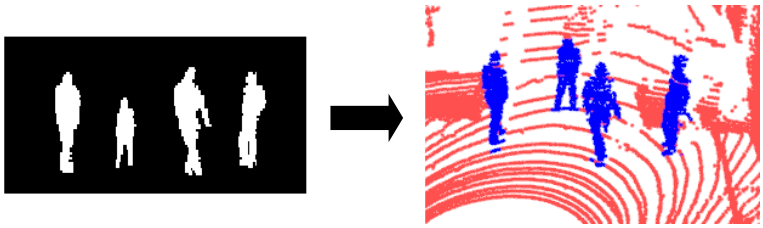


Fig. 4. Backprojection of the range image labels to the point cloud

observe, that although the model is dynamic due to dependencies between different time frames (see the ξ_s^t term), to enable real time operation, we develop a causal system, i.e. labels from the past are not updated based on labels from the future.

The data terms are derived from the data energies by sigmoid mapping:

$$V_D(d_s^t|\omega_s^t = \text{bg}) = \zeta(\varepsilon_{\text{bg}}^t(s), \tau_{\text{bg}}, m_{\text{bg}})$$

$$V_D(d_s^t|\omega_s^t = \text{fg}) = \begin{cases} 1 & \text{if } \; d_s^t > \max_{\{i=1\ldots k_s\}} \mu_s^{i,t} + \epsilon \\ \zeta(\varepsilon_{\text{fg}}^t(s), \tau_{\text{fg}}, m_{\text{fg}}) & \text{otherwise.} \end{cases}$$

The sigmoid parameters τ_{fg}, τ_{bg}, m_{fg}, m_{bg} and $m_\star$ can be estimated by Maximum Likelihood strategies based on a few manually annotated training images. As for the smoothing factors, we use $\alpha = 0.2$ and $\beta = 1.0$ (i.e. the spatial constraint is much stronger), while the kernel bandwidth is set to $h = 30$cm. The MRF energy (4) is minimized via the fast graph-cut based optimization algorithm [10].

The result of the DMRF optimization is a binary foreground mask on the discrete S lattice. As shown in Fig. 4, the final step of the method is the classification of the points of the original $\mathcal{L}$ cloud, considering that the projection may be ambiguous, i.e. multiple points with different true class labels can be projected to the same pixel of the segmented range image. With denoting by $s = \mathcal{P}(p)$ for time frame t:

- $\omega(p) = \text{fg}$, iff one of the following two conditions holds:
  - $\omega_s^t = \text{fg}$ and $d(p) < d_s^t + 2 \cdot h$ (a)
  - $\omega_s^t = \text{bg}$ and $\exists r \in N_r : \{\omega_r^t = \text{fg}, |d_r^t - d(p)| < h\}$ (b)
- $\omega(p) = \text{bg}$: otherwise.

The above constraints eliminate several (a) false positive and (b) false negative foreground points, projected to pixels of the range image near the object edges.

5 Evaluation

We have tested our method in real Lidar sequences concerning both video surveillance (*Courtyard*) and traffic monitoring (*Traffic*) scenarios (see Fig. 5). The data flows have been recorded by a Velodyne HDL 64E S2 camera, which operates with $R = 64$ vertically aligned beams. The *Courtyard* sequence contains

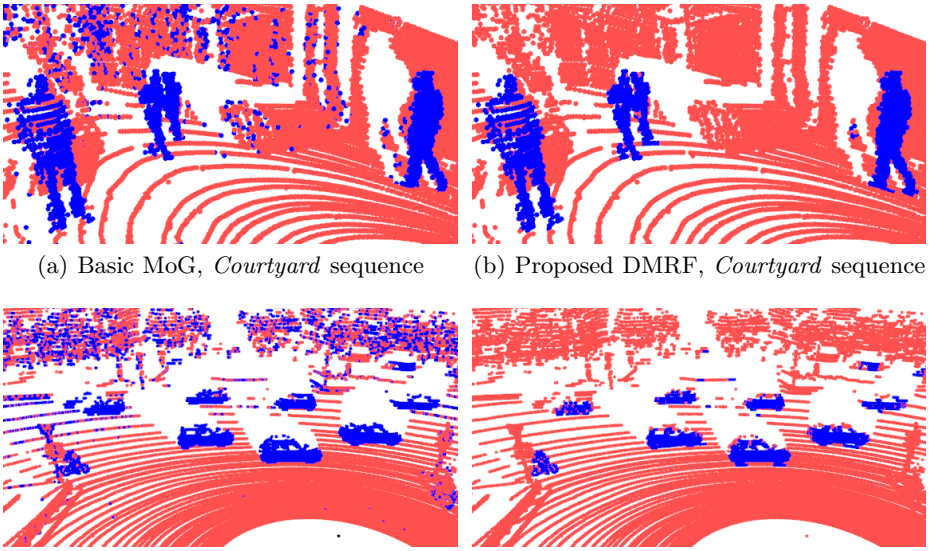
Table 1. Numerical evaluation on the *Courtyard* and *Traffic* sequences: detection accuracy (F-rate in %) and processing speed (fps, measured in a desktop computer)

Aspect	Sequence	Seq. prop.	Bas. MoG	uniMRF	3D-MRF	**DMRF**
Detection rate (F-mes in %)	*Courtyard*	4 obj/fr.	55.7	81.0	88.1	95.1
	Traffic	20 obj/fr.	70.4	68.3	76.2	74.0
Proc. speed (fr per sec)	*Courtyard*	65K pts/fr.	120 fps	18 fps	7 fps	16 fps
	Traffic	260K pts/fr.	120 fps	18 fps	2 fps	16 fps

2500 frames with four people walking in a $25m^2$ area in 1-5m distances from the Lidar, with crossing trajectories. The rotation speed was set to 20Hz. In the background, heavy motion of the vegetations make the accurate classification challenging. The *Traffic* sequence was recorded with 5Hz from the top of a car waiting at a traffic light in a crowded crossroad. The adaptive background model was automatically built up within a few seconds, then 160 time frames were available for traffic flow analysis. We have compared our DMRF model to three reference solutions:

1. *Basic MoG*, introduced in Sec. 3, which is based on [4] with using on-line K-means parameter update [9].
2. *uniMRF*, introduced in Sec. 4, which partially adopts the uniform foreground model of [2] for range image segmentation in the DMRF framework.
3. *3D-MRF*, which implements a MRF model in 3D, similarly to [8]. We define here point neighborhoods in the original $\mathcal{L}^t$ clouds based on Euclidean distance, and use the background fitness values of (2) in the data model. The graph-cut algorithm [10] is adopted again for MRF energy optimization.

Qualitative results on two sample frames are shown in Fig. 5. For Ground Truth (GT) generation, we have developed a 3D point cloud annotation tool, which enables labeling the scene regions manually as foreground or background. Next, we manually annotated 700 relevant frames of the *Courtyard* and 50 frames of the *Traffic* sequence. For quantitative evaluation metric, we have chosen the point level F-rate of foreground detection [3], which can be calculated as the harmonic mean of precision and recall. We have also measured the processing speed in frames per seconds (fps). The numerical performance analysis is given in Table 1. The results confirm that the proposed model surpasses the *Basic MoG* and *uniMRF* techniques in F-rate for both scenes, and the differences are especially notable at the *Courtyard*. Compared to the *3D-MRF* method, our model provides similar detection accuracy, but the *proposed DMRF* method is significantly quicker. Observe that differently from 3D-MRF, our range image based technique is less influenced by the size of the point cloud. In the *Traffic* sequence, which contains around 260000 points within a time frame, we measured 2fps processing speed with 3D-MRF and 16fps with the proposed DMRF model.



(a) Basic MoG, *Courtyard* sequence (b) Proposed DMRF, *Courtyard* sequence

(c) Basic MoG, *Traffic* sequence (d) Proposed DMRF, *Traffic* sequence

Fig. 5. Point cloud classification result on sample frames with the *Basic MoG* and the proposed DMRF model: foreground points are displayed in blue (dark in gray print)

6 Conclusions

We have proposed a Dynamic MRF model for foreground segmentation in point clouds obtained by a rotating multi-beam Lidar system. We have introduced an efficient spatial foreground filter to decrease artifacts of angle quantization and background motion. The model has been quantitatively validated based on Ground Truth data, and the advantages of the proposed solution versus three reference methods have been demonstrated. The authors thank Miklós Homolya for help in MRF code integration [10].

References

1. Schiller, I., Koch, R.: Improved video segmentation by adaptive combination of depth keying and Mixture-of-Gaussians. In: Heyden, A., Kahl, F. (eds.) SCIA 2011. LNCS, vol. 6688, pp. 59–68. Springer, Heidelberg (2011)
2. Wang, Y., Loe, K.F., Wu, J.K.: A dynamic conditional random field model for foreground and shadow segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence 28(2), 279–289 (2006)
3. Benedek, C., Szirányi, T.: Bayesian foreground and shadow detection in uncertain frame rate surveillance videos. IEEE Transactions on Image Processing 17(4), 608–621 (2008)
4. Kästner, R., Engelhard, N., Triebel, R., Siegwart, R.: A Bayesian approach to learning 3D representations of dynamic environments. In: Khatib, O., Kumar, V., Sukhatme, G. (eds.) Experimental Robotics. STAR, vol. 79, pp. 461–475. Springer, Heidelberg (2012)

5. Kalyan, B., Lee, K.W., Wijesoma, W.S., Moratuwage, D., Patrikalakis, N.M.: A random finite set based detection and tracking using 3D LIDAR in dynamic environments. In: IEEE International Conference on Systems, Man, and Cybernetics (SMC), Istanbul, Turkey, pp. 2288–2292. IEEE (2010)
6. Muhammad, N., Lacroix, S.: Calibration of a rotating multi-beam Lidar. In: International Conference on Intelligent Robots and Systems (IROS), Taipei, Taiwan, pp. 5648–5653. IEEE (2010)
7. Spinello, L., Luber, M., Arras, K.: Tracking people in 3D using a bottom-up top-down detector. In: IEEE International Conference on Robotics and Automation (ICRA), Shanghai, China, pp. 1304–1310 (2011)
8. Lafarge, F., Mallet, C.: Creating large-scale city models from 3D-point clouds: A robust approach with hybrid representation. Int. J. of Computer Vision (2012)
9. Stauffer, C., Grimson, W.E.L.: Learning patterns of activity using real-time tracking. IEEE Transactions on Pattern Analysis and Machine Intelligence 22, 747–757 (2000)
10. Boykov, Y., Kolmogorov, V.: An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. IEEE Transactions on Pattern Analysis and Machine Intelligence 26(9), 1124–1137 (2004)

Posture Analysis and Range of Movement Estimation Using Depth Maps

Miguel Reyes[1,2], Albert Clapés[1,2], Sergio Escalera[1,2],
José Ramírez[3], and Juan R. Revilla[3]

[1] Dept. Matemàtica Aplicada i Anàlisi, UB,
Gran Via de les Corts Catalanes 585, 08007, Barcelona
[2] Computer Vision Center, Campus UAB, Edifici O, 08193, Bellaterra, Barcelona
[3] Instituto de Fisioterapia Global Mezières, Guillem Tell 27, 08006, Barcelona

Abstract. World Health Organization estimates that 80% of the world population is affected of back pain during his life. Current practices to analyze back problems are expensive, subjective, and invasive. In this work, we propose a novel tool for posture and range of movement estimation based on the analysis of 3D information from depth maps. Given a set of keypoints defined by the user, RGB and depth data are aligned, depth surface is reconstructed, keypoints are matching using a novel point-to-point fitting procedure, and accurate measurements about posture, spinal curvature, and range of movement are computed. The system shows high precision and reliable measurements, being useful for posture reeducation purposes to prevent musculoskeletal disorders, such as back pain, as well as tracking the posture evolution of patients in rehabilitation treatments.

Keywords: Depth maps, Physiotherapy, Posture Analysis, Range of Movement Estimation, Rehabilitation, Statistical Pattern Recognition.

1 Introduction

World Health Organization has categorized disorders of the musculoskeletal system as the main cause for absence from occupational work and one of the most important causes of disability in elders in the form of rheumatoid arthritis or osteoporosis. It is estimated that 80% of world population will suffer from musculoskeletal disorders during their life.

The body posture evaluation of a subject manifests, in different degrees, his level of physic-anatomical health given the behavior of bone structures, and especially of the dorsal spine. For instance, common musculoskeletal dysfunctions or disorders (MSDs) such as scoliosis, kyphosis, lordosis, arthropathy, or spinal pain show some of their symptoms through body posture. This requires the use of reliable, noninvasive, automatic, and easy to use tools for supporting diagnostic. However, given the articulated nature of the human body, the development of this kind of systems is still an open issue.

The solution more frequently applied to measure body posture consists of the synchronization of multiple cameras, applying stereo vision techniques [3,5]. This kind of systems use to be expensive and invasive. Moreover, it uses to require specific and restricted illumination conditions. The main alternative is accelerometers. These systems also use to be expensive, invasive, and inaccurate because of the spatial measurements

X. Jiang et al. (Eds.): WDIA 2012, LNCS 7854, pp. 97–105, 2013.

of multi-axial articulations. Most of these systems only treat specific areas of the body with little configurability, which implies that therapists cannot use their own methods of analysis. A recent alternative is the use of the depth maps provided by the Microsoft Kinect device [1]. The Kinect camera uses a structured light technique to generate real-time depth maps containing discrete range measurements of the physical scene [2].

In this work, we present a novel semi-automatic system that uses RGB-Depth information to elaborate a clinical postural analysis through the examination of anthropometric values. Given a set of keypoints defined by the user, our proposed method performs the following steps: a) RGB and depth data are aligned, b) noise is removed and depth surface is reconstructed, c) user keypoints and predefined protocols are matched using a novel point-to-point fitting procedure, d) static measurements about posture and spinal curvature are accurately computed, and d) dynamic range of movement is robustly estimated. Compared to standard alternatives and supported by clinical specialists, the system shows high precision and reliable measurements to be include in the clinical routine.

The paper is organized as follows: Section 2 present the system for posture analysis and range of movement estimation. Section 3 presents the validation of the proposal, and finally, Section 4 concludes the paper.

2 Posture Analysis System

We designed a full functional system devoted to help in the posture reeducation task with the aim of preventing and correcting musculoskeletal disorders. The system is composed by three main functionalities: a) static posture analysis (SPA), b) spine curvature analysis (SCA), and c) range of movement analysis (RMA). The architecture of the system is shown in Figure 1. First, a pre-processing step to remove noise and reconstruct surfaces is performed. Next, we describe each of these stages.

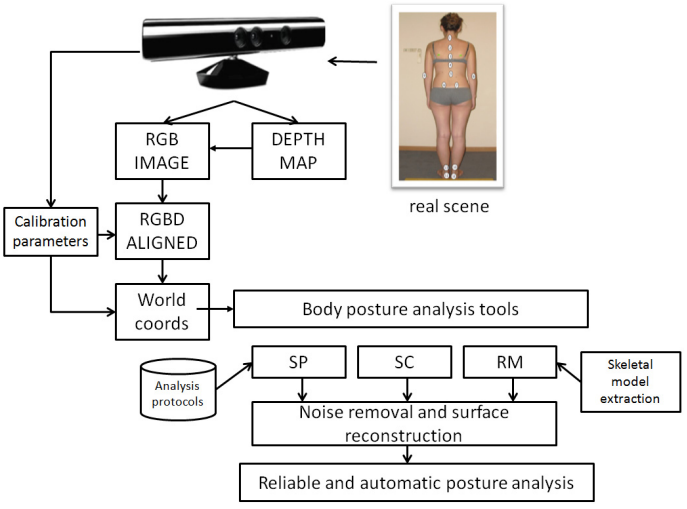


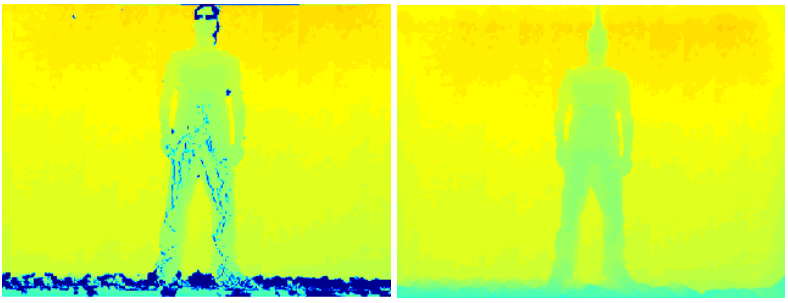
Fig. 1. Posture analysis system

2.1 Noise Removal and Surface Reconstruction

After aligning RGB and depth data [7,11], and even though the used depth information is compelling it is still inherently noisy. Depth measurements often fluctuate and depth maps contain numerous holes where no readings are obtained. In order to obtain a valid and accurate depth map, we perform a depth preprocessing step to eliminate erroneous information caused by noise and to reconstruct surfaces not well defined. We perform the following methodology:

Noise removal: For each point we compute the mean distance from it to all its neighbors. By assuming that the resulted distribution is Gaussian with a mean and a standard deviation, all points whose mean distances are outside an interval defined by the global distances mean and standard deviation are considered as outliers.

Surface reconstruction: We use a resampling algorithm [10], which attempts to recreate the missing parts of the surface by higher order polynomial interpolation between the surrounding data points. By performing resampling, these small errors can be corrected. Figure 2 shows an example of this process [1].



(a) (b)

Fig. 2. (a) Original depth map. (b) Filtered and resampled.

Once the system is calibrated, data is aligned, and depth maps are filtered, the user can access to the three posture facilities.

2.2 Static Posture Analysis (SPA)

This module computes and associates a set of three-dimensional angles and distances to keypoints defined by the user. These keypoints correspond to manual interactions of the user with the RGB data displayed in the screen (which internally is aligned with the corresponding depth data). The module also allows the therapist the possibility of designing a protocol of analysis. That is, a predefined set of angular-distance measurements among a set of body keypoints, all of them defined and saved by the user for

[1] We experimentally found that our approach for noise removal and background reconstruction obtained better results than standard approaches based on accumulating temporal images (e.g. 30 frames of a stationary subject) for noise reduction and hole filling.

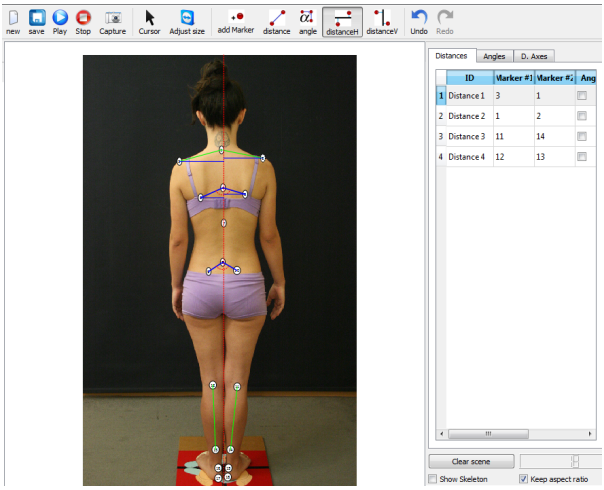


Fig. 3. Static posture analysis example

posterior automatic matching. Figure 3 shows an example of a predefined protocol (the set of manual annotated keypoints together with the list of distance and angle relations to be computed).

In order to obtain an intelligent and automatic estimation of posture measurements, we define a correspondence procedure among manually placed virtual markers and protocol markers. We formulate markers matching as an optimization problem. Suppose a protocol analysis (template) T composed by N markers, $T = \{T_1, T_2, ..., T_N\}$, $T_i = (x_i, y_i, z_i)$, and the current analysis C composed by the same number of markers, $C = \{C_1, C_2, ..., C_N\}$ (predefined template and current set of keypoints defined by the user, respectively). Our goal is to make a one-to-one correspondence so that we minimize the sum of least square distances among assignments as follows:

$$\text{argmin}_{C'} \sum_{i=1}^{N} \|C'_i - T_i\|^2, \tag{1}$$

where C' is evaluated as each of the possible permutations of the elements of C. For this task, first, we perform a soft pre-alignment between C and T using Iterative Closest Point (ICP) [8], and then, we propose a sub-optimal approximation to the least-squares minimization problem. ICP is based on the application of rigid transformations (translation and rotation) in order to align both sequences C and T. This attempts to minimize the error of alignment $E(.)$ between the two marker sequences as follows:

$$E(\mathcal{R}, \mathcal{T}) = \sum_{i=1}^{N} \sum_{j=1}^{N} w_{i,j} \|T_i - \mathcal{R}(C_j) - \mathcal{T}\|^2, \tag{2}$$

where $\mathcal{R}$ and $\mathcal{T}$ are the rotation and translation 3D vectors, respectively. It is assigned 1 to $w_{i,j}$ if the i-th point of T described the same point in space as the j-th point of

C. Otherwise $w_{i,j} = 0$. Two things have to be calculated: First, the corresponding points, and second, the transformation $(\mathcal{R}, \mathcal{T})$ that minimizes $E(\mathcal{R}, \mathcal{T})$ on the base of the corresponding points. For this task, we apply Singular Value decomposition (SVD). At the end of the optimization, the new projection of the elements of C is considered for final correspondence. Then, Eq. 1 is approximated as follows: Given the symmetric matrix of distances M of size $N \times N$ which codifies the set of $N \cdot (N-1)/2$ possible distances among all assignments between the elements of C and T, we set a distance threshold θ_M to define the adjacency matrix A:

$$A(i,j) = \begin{cases} 1 & \text{if} \quad M(i,j) < \theta_M \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

Then, instead of looking for the set of $N!$ possible assignments of elements of C and T that minimizes Eq. 1, only the possible assignments (C_i, T_j) that satisfies $A(i,j) = 1$ are considered, dramatically reducing the complexity of the correspondence procedure[2].

2.3 Spine Curvature Analysis (SCA)

The objective of this task is to evaluate sagittal spine curvatures (curves of the spine projected on the sagittal plane) by noninvasive graphic estimations in kyphotic and lordotic patients. Kyphosis and lordosis are, respectively, conditions of over-curvature of the thoracic spine (upper back) and the lumbar spine (lower back). The methodology proposed by Leroux et al [4] offers a three-dimensional analysis valid for clinical examinations of those conditions. In order to perform this analysis we proceed as follows. First, the therapist places the markers on the spine. Then, a few markers are selected and the 3D curve that represents the spine is reconstructed by linear interpolation (Figure 4(c)). Finally, the anthropometric kyphosis K_a and lordosis L_a are obtained.

The geometric model to compute K_a is represented in the Figure 4(a). F divides the curve representing the thoracic spine in two asymmetric arcs with different radius. Note that the F component begins at the farthest marker (apex, corresponding to T5) and it ends at the intersection with the T2-to-T12 line. h_1 and h_2 are the distances from manual annotation of T2 to the intersection and the distance from the intersection to annotated landmark T12 (as shown in Figure 4), respectively. Then, the summation of two angles, φ_1 and φ_2, represents the kyphosis curve value, where:

$$\begin{aligned} \varphi_1 &= 180 - 2 \cdot \arctan\left(\tfrac{h_1}{F}\right), \\ \varphi_2 &= 180 - 2 \cdot \arctan\left(\tfrac{h_2}{F}\right). \end{aligned} \tag{4}$$

L_a is calculated in a similar way, though the therapist should note the markers in the lumbar spine region. The capacity analysis of the spine is reinforced by a three-dimensional environment for a thorough examination by the therapist (Figure 4(d)). An example of spine interaction and computation are shown in Figure 4(a) and (b), respectively.

[2] We experimentally found that high values of θ_M obtain optimal results and reduces the computational cost in comparison to other approaches, such as Shape Context [6].

2.4 Range of Movement Analysis (RMA)

In order to complement the posture analysis procedure, we compute the range of movement of different body articulations. This means, for a particular articulation (joint), we detect and track its movement, and then we compute which is the range of angles that this joint performs for a particular period of time. For this purpose, we perform user detection using the Random Forest approach with depth features of Shotton et al [9] and compute the skeletal model. This process is performed computing random offsets of depth features as follows:

$$f_\theta(D, \mathbf{x}) = \mathbf{D}_{\left(\mathbf{x}+\frac{\mathbf{u}}{\mathbf{D_x}}\right)} - \mathbf{D}_{\left(\mathbf{x}+\frac{\mathbf{v}}{\mathbf{D_x}}\right)}, \tag{5}$$

where $\theta = (\mathbf{u}, \mathbf{v})$, and $\mathbf{u}, \mathbf{v} \in \mathbb{R}^{\mathbf{2}}$ is a pair of offsets, depth invariant. Thus, each θ determines two new pixels relative to $\mathbf{x}$, the depth difference of which accounts for the value of $f_\theta(D, \mathbf{x})$. Using this set of random depth features, Random Forest is trained for a set of trees, where each tree consists of split and leaf nodes (the root is also a split node). Finally, we obtain a final pixel probability of body part membership l_i as follows:

$$P\left(l_i|D, \mathbf{x}\right) = \frac{1}{\tau}\sum_{j=1}^{\tau} P_j\left(l_i|D, \mathbf{x}\right), \tag{6}$$

where $P\left(l_i|D, \mathbf{x}\right)$ is the PDF stored at the leaf, reached by the pixel for classification $(D, \mathbf{x})$ and traced through the tree j, $j \in \tau$. Computing the intersection borders among mean shift clusters estimated after Random Forest procedure, we obtain a three-dimensional skeletal model composed by nineteen joints. The physician then selects joint articulations and automatically obtains their maximum opening and minimum closing values measured in degrees for a certain period of time (Figure 4(e)).

3 Results

3.1 Software Details

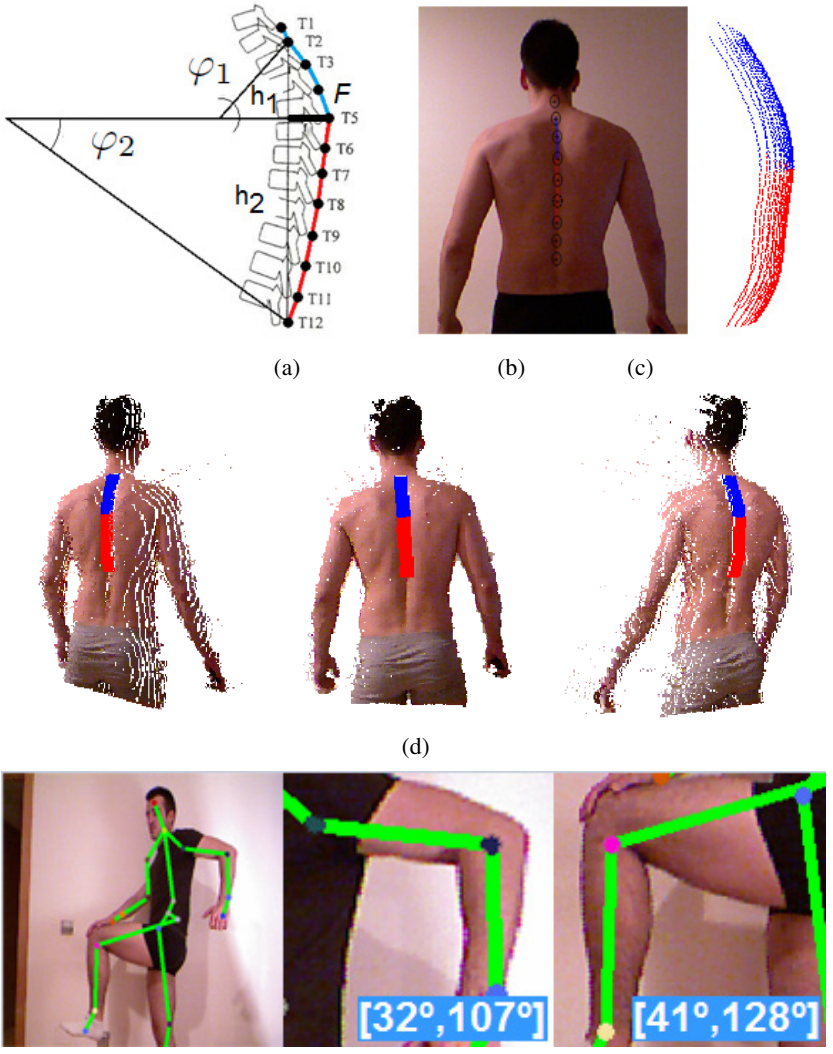
The video data uses a 8 bits VGA resolution at 30Hz, and we capture frames at 640×480 pixels, like the infrared camera. Regarding the implementation we used the Kinect SDK

Table 1. Pose and range of movement precision

Distance subject-device (m)	1,3	1,9	2,2
AAV (∘ movement)	2,2	3,8	5,2
AAV (mm)	0,98	1,42	2,1
AAV (∘ angles)	0,51	1,04	1,24
AAV (%)	0,46	0,77	1,3
Standard Error (%)	1,01	1,18	1,71

Table 2. Validation of spinal analysis

	Khyposis range	Lordosis range
AAV (∘)	5	6



(a) (b) (c)

(d)

(e)

Fig. 4. (a) Geometric model to obtain anthropometric kyphosis, and lordosis value. (b) Sample of analysis. (c) Automatically reconstructed 3D spinal cloud. (d) Three-dimensional examination environment. (e) Skeletal model and example of selected articulations with computed dynamic range of movement.

framework. We also used the PCL-Library to treat cloud points, and to support a free and three-dimensional visualization we used the VTK library. The user interface has been developed in multi-platform Nokia Qt technology.

3.2 Data and Validation

In order to measure the precision of the proposed methodology in the different modules of the system, a battery of 500 simple tests has been labeled by three different observers, with an inter observer correlation superior to 99% for all planes (X, Y, Z). Each test contains a set of angles and distances in order to simulate an analysis protocol for the study of posture, placing twelve infrared led markers on the body of the subject. A total of 20 subjects participated in the validation of the method. In order to perform automatic validation of the tests, infrared markers are detected by means of thresholding a HSV infrared-filtered image.

Results for different distance of the device to the scene are shown in Table 1. AAV and '$\circ$' correspond to the average absolute value and degree, respectively. This analysis validates the accuracy of the SPA and RMA in millimeters and degrees, respectively. Note the high precision in both tests. In addition, in order to validate the curvature analysis of the spine (SCA), we used a group of 10 patients and performed the Leroux protocol [4], placing nine markers over the spine. The relationship between lateral radiographic and anthropometric measures was assessed with the mean difference. It has used Cobb technique on the lateral radiograph in order to obtain the coefficients of kyphosis and lordosis. The results of the SPA validation are shown in Table 2.4. Moreover, after discussing with specialists in physiotherapy they agreed that the accuracy of the results is more than sufficient for diagnostic purposes.

4 Conclusion

We presented a system for semi-automatic posture analysis and range of movement estimation using depth maps. The aim of the system is to assist in the posture reeducation task to prevent and treat musculoskeletal disorders. Given a set of keypoints defined by the user, RGB and depth data are aligned, depth surface is reconstructed, keypoints are matching using a novel point-to-point fitting procedure, and accurate measurements about posture, spinal curvature, and range of movement are obtained. The system showed high precision in terms of distance, degree, and range of movement estimation. Supported by clinical specialists, the system shows high precision and reliable measurements to be include in the clinical routine.

Acknowledgements. This work is partly supported by projects TIN2009-14404-C02, IMSERSO-Ministerio de Sanidad 2011 Ref. MEDIMINDER and RECERCAIXA 2011 Ref. REMEDI.

References

1. Microsoft Corporation. Kinect for windows sdk beta programming guide beta 1 draft version 1.1 (2012)
2. Freedman, B., Shpunt, A., Machline, M., Arieli, Y.: 2008 - depth mapping using projected patterns.pdf, p. 12 (2010)
3. Koontz, A.M., Lin, Y.-S., Kankipati, P., Boninger, M.L., Cooper, R.A.: Development of custom measurement system for biomechanical evaluation of independent wheelchair transfers, vol. 48, pp. 1015–1028 (2011)
4. Leroux, M.A., Zabjek, K.: A noninvasive anthropometric technique for measuring kyphosis and lordosis: application for scoliosis, vol. 25, pp. 1689–1694 (2000)
5. Mikic, I., Trivedi, M., Hunter, E., Cosman, P.: Articulated Body Posture Estimation from Multi-Camera Voxel Data. In: CVPR, pp. 455–460 (2001)
6. Mori, G., Belongie, S., Malik, J.: Efficient shape matching using shape contexts. TPAMI 27, 1832–1837 (2005)
7. Potmesil, M., Chakravarty, I.: A lens and aperture camera model for synthetic image generation, vol. 15, pp. 297–305. ACM, New York (1981)
8. Estépar, R.S.J., Brun, A., Westin, C.-F.: Robust generalized total least squares iterative closest point registration. In: Barillot, C., Haynor, D.R., Hellier, P. (eds.) MICCAI 2004. LNCS, vol. 3216, pp. 234–241. Springer, Heidelberg (2004)
9. Shotton, J., Fitzgibbon, A.W., Cook, M., Sharp, T.: Real-time human pose recognition in parts from single depth images. In: CVPR, pp. 1297–1304 (2011)
10. Yao, J., Ruggeri, M.R., Taddei, P.: Automatic scan registration using 3d linear and planar features, vol. 1, pp. 22:1–22:18. Springer, Secaucus
11. Zhang, Z.: A flexible new technique for camera calibration. TPAMI 22(11), 1330–1334 (2000)

Fast 3D Keypoints Detector and Descriptor for View-Based 3D Objects Recognition

Ayet Shaiek and Fabien Moutarde

Robotics laboratory (CAOR) Mines ParisTech 60 Bd St Michel, F-75006 Paris, France

Abstract. In this paper, we propose a new 3D object recognition method that employs a set of 3D keypoints extracted from point cloud representation of 3D views. The method makes use of the 2D organization of range data produced by 3D sensor. Our novel 3D interest points approach relies on surface type classification and combines the Shape Index (SI) - curvedness(C) map with the Gaussian (H) - Mean (K) map. For each extracted keypoint, a local description using the point and its neighbors is computed by joining the Shape Index histogram and the normalized histogram of angles between normals. This new proposed descriptor IndSHOT stems from the descriptor CSHOT (Color Signature of Histograms of OrienTations) which is based on the definition of a local, robust and invariant Reference Frame RF. This surface patch descriptor is used to find the correspondences between query-model view pairs in effective and robust way. Experimental results on Kinect based datasets are presented to validate the proposed approach in view based 3D object recognition.

Keywords: Depth Image, 3D Keypoints detector, Mean Curvature, Gaussian Curvature, Shape Index, HK Map, SC Map, SHOT Descriptor, IndSHOT.

1 Introduction

There has been strong research interest in 3D object recognition over the last decade, due to the promising reliability of the new 3D acquisition techniques. 3D recognition, however, conveys several issues related to the amount of information, class variability, partial information, as well as scales and viewpoints differences are encountered. As previous works in the 2D case have shown, local methods perform better than global features to partially overcome those problems. Global features need the complete, isolated shape for their extraction. Examples of global 3D features are volumetric part-based descriptions [1]. These methods are less successful when dealing with partial shape and intra-class variations while remaining partially robust to noise, clutter and inter-class variations. The field of 2D Point-of-interest (POI) feature has been the source of inspiration for the 3D interest-points detectors. For example, the Harris detector has been extended to three dimensions, first in [2] with two spatial dimensions plus the time dimension, then in [3] which discusses variants of the Harris measure and recently in [4] where a 3D-SURF adaptation is proposed. The 3D shape of a given object can be described by a set of local features extracted from patches around salient interest points. Regarding efficient 3D descriptors, the SHOT

X. Jiang et al. (Eds.): WDIA 2012, LNCS 7854, pp. 106–115, 2013.

descriptor [5] achieves both state-of-the-art robustness and descriptiveness. Results demonstrate the higher descriptiveness embedded in SHOT with respect to Spin Images [6], Exponential Mapping (EM) and Point Signatures (PS). Given the local RF, an isotropic spherical grid is defined to encode spatially well localized information. For each sector of the grid a histogram of normals is defined and the overall descriptor SHOT results from the juxtaposition of these histograms.

Our proposed new method aims to detect salient keypoints that are repeatable under moderate viewpoint variations. We propose to use a measure of curvature in the line of Chen and Bhanu's work [7] and construct a patch labeling to classify different surface shapes [7, 8] using both mean-Gaussian curvatures (HK) and shape index-curvedness (SC) couples. Thus, we select keypoints according to their local surface saliency. Furthermore, we suggest a novel descriptor, dubbed IndSHOT, that emphasizes the shape description by merging the SHOT descriptor with the Shape Index histogram. The complete recognition system with detection, description and matching phases is introduced in section 2. The proposed method is then evaluated in section 3.

2 Methodology

2.1 Resampling of the 3D Points Cloud

As we address a recognition scenario wherein only 2.5 views are matched, we deal with some views of the models from specific viewpoints. In the work presented here, we exploit the lattice structure provided by the range image. First, we search the coordinates of the maximum and minimum points at x-axis and y-axis in the sample, and build a bounding box based on the two limit points. Using the (i, j) coordinates of each point in this box, we smooth the initial 3D point cloud by resampling down to *1/span* of its original point density in order to avoid noise perturbation. The smoothing process generates new points corresponding to the average of points belonging to a rectangular region with a span in the x and y direction. Then, we construct a mesh using the new vertices. The x and y spans are proportional to the density of points and to a fraction r_1 of the bounding box dimensions, so as to make our method robust to different spatial samplings and to scaling. In our approach, neighbour points are given by a spherical region around the point, with a support radius R proportional to a fraction r_2 of bounding box diagonal. In practice, we adjust a local polynomial surface to the selected neighborhood. CGAL[1] library is used for curvature computation. An advantage of subdividing the point cloud in local regions is to avoid mutual impact between them.

2.2 Keypoint Detectors

The aim of this step is to pick out a repeatable and salient set of 3D points. Principal curvatures correspond to the eigenvalues of the Hessian matrix and are invariant under rotation and translation. Hence, we propose to use local curvatures which can be calculated either directly from first and second derivatives, or indirectly as the rate of change of normal orientations in a local context region. The usual pair of Gaussian

[1] http://www.cgal.org/

curvature K and mean curvature H only provides a poor representation, since the values are strongly correlated. Instead, we use them in composed form with curvature based quantities. In the following, we first introduce state-of-the-art detector methods based on shape index, HK and SC classification; then we present the principle of our new detector.

Shape Index. This detector type was proposed in [7], and uses the shape index (*SIp*) for feature point extraction. It is a quantitative measure of the surface shape at a point p, and is defined by (1),

$$SI_p = \frac{1}{2} - \frac{1}{\pi} \times \text{arctg}\left(\frac{k_p^1 + k_p^2}{k_p^1 - k_p^2}\right) \tag{1}$$

where k^1_p and k^2_p are maximum and minimum principal curvatures, respectively and *arctg* = *arctangent*. With this definition, all shapes are mapped into the interval [0, 1] where every distinct surface shape corresponds to a unique value of SI (except for planar surfaces, which will be mapped to the value 0.5, together with saddle shapes). Larger shape index values represent convex surfaces and smaller shape index values represent concave surfaces. The main advantage of this measure is the invariance to orientation and scale. A point is marked as a feature point if its shape index SI_p satisfies (2) within point neighbors,

$$\begin{cases} SI_p = max(SI_k)\,; k \in neighbors \;\; and \;\; SI_p \geq (1+\alpha) \times \mu \\ \qquad\qquad or \\ SI_p = min(SI_k)\,; k \in neighbors \;\; and \;\; SI_p \leq (1-\beta) \times \mu \end{cases} \tag{2}$$

where μ is the mean of shape index over the SI point neighbors values and $0 \leq \alpha, \beta \leq 1$. In above expression (2), parameter α and β control the selection of feature points. We denote this detector by SID.

HK and SC Classification. The idea here is to build shape classification space using the pair mean-Gaussian curvatures (HK) or the pair shape index-curvedness (SC). Typically, for HK classification, we use the type function Tp used in LSP descriptor [7] that associates to each couple of H and K values a unique type value (4),

$$\mathrm{T_p} = 1 + 3\left(1 + \mathrm{sgn}_{\varepsilon_H}(H)\right) + \left(1 - \mathrm{sgn}_{\varepsilon_K}(K)\right);\ \mathrm{sgn}_{\varepsilon_X}(X) \begin{cases} +1 & if\ X > \varepsilon_X, \\ 0 & if\ |X| \leq \varepsilon_X, \\ -1 & if\ X < \varepsilon_X \end{cases} \tag{3}$$

where ε_H and ε_K are two thresholds over the H and K. Nine region types are defined.

In the shape index-curvedness (SC) space, S defines the shape type and C defines the degree of curvature and is the square-root of the deviation from flatness. Similarly to HK representation, the continuous graduation of S subdivides surface shapes into 9 types. Planar surfaces are classified using the C value. We define a type function $\mathrm{S_p}$ (5) that associates a unique type value to each couple of SI and C values (i.e values between 0.8125 and 0.9375 correspond to dome and $\mathrm{S_p} = 7$),

$$\begin{cases} \mathrm{S_p} = 0 \text{ if } \mathrm{C} \leq \varepsilon_\mathrm{C} \\ \text{else} \\ \mathrm{S_p} \in [1,8\,]\ ;\ \mathrm{SI} \in [\![0,1]\!]\,. \end{cases} \tag{4}$$

For both classifications, salient regions are selected as those of one of the 5 following types: dome, trough, spherical, saddle rut and saddle ridge regions. More details are given in [9, 10].

Combination of Criteria. Theoretically, the two classifications HK and SC should provide the same result; therefore we suggest combining the two criteria to increase reliability. In fact, our result will be validated with two measures of keypoints detection. After labeling points with a pair of value (T_p, S_p), points with salient type pair are selected, in other words, if the two labels correspond to the same of the 5 salient region types previously mentioned. Moreover, in this paradigm, plane surfaces aren't taken in account. So, we chose to select, in addition to those 5 surface types, planar regions. We note this detector « SC_HK ». Then, points with the same pair value are grouped using the connected component labeling. Connectivity is carried out by checking the 8-connectivity of each point. Finally, the centers of the connected components are selected as keypoints. We also propose further combination by ranking the selected keypoints according to their curvedness value. The point with the maximum value of curvedness over the selected keypoints is chosen to represent each connected component. In the case of planar regions, a big number of points are chosen and are not all really representative of the saliency. In order to have a good distribution of interest points in the object surface, the proposed idea here is to cluster preselected points according to their relative distance and we threshold the distance between final keypoints (as a fraction of the bounding box's diagonal). We call the detector combining the two criteria SC_HK_connex.

2.3 Keypoint Descriptors

After keypoints detection step, a 3D descriptor is computed around each selected interest point. In the case of range data, the dominant orientation at a point is the direction of the surface normal at that point. Histogram-based methods are typically based on the feature point normals. For example, Local Surface Patches [7] computes histograms of normals and shape indexes of the points belonging to the keypoint support. The recently proposed SHOT descriptor achieves computational efficiency, descriptive power and robustness by defining 3D repeatable local Reference Frame (RF). We briefly summarize here the structure of the SHOT descriptor. The reader is referred to [5] for details on the descriptor. The introduction of geometric information concerning the location of the points within the support is performed by first calculating a set of local histograms of normals over the 3D volumes defined by a 3D grid superimposed on the support and then grouping together all local histograms to form the final descriptor. The normal estimation is based on the Eigenvalue Decomposition of a novel scatter matrix defined by a weighted linear combination of neighbour point distances to the feature point, lying within the spherical support. The eigenvectors of this matrix define repeatable, orthogonal directions in presence of noise and clutter. Furthermore, the CSHOT descriptor [11] is proposed as an amelioration of the SHOT descriptor and makes profits from the 3D data enriched with texture. The process of combination succeeds to form more robust and descriptive signature.

Inspired by these state-of-the-art descriptors, we compute the histograms of shape index values and angle values between the reference surface normals at the feature point

and the neighbour's ones and join the two histograms similarly to the design of CSHOT descriptor. First of all, we accumulate point counts into bins according to a cosine function of the angle between the normal at each point within the corresponding part of the grid and the normal at the feature point. For each of the local histograms, a coarser binning is created for directions close to the reference normal direction and a finer one for orthogonal directions. In this way, small differences in orthogonal directions to the normal, which are the most informative ones, cause a point to be accumulated in different bins. Secondly, shape index values of the feature point and those of its neighbours relying in the spherical support are grouped into bins. Finally, we merge the shape index values and the cosine values into one descriptor that we call IndSHOT. We perform the same process as in the CSHOT to juxtapose the two histograms, where index shape histogram replaces the color histogram (shown in *figure 1*). In addition, the mean and standard deviation of shape index of the neighbors around the feature point are computed. The final descriptors, composed of (model ID, index shape + cosines histograms, surface type, the 3D coordinates of keypoint, mean and standard deviation of shape index), are saved to be used in the matching process.

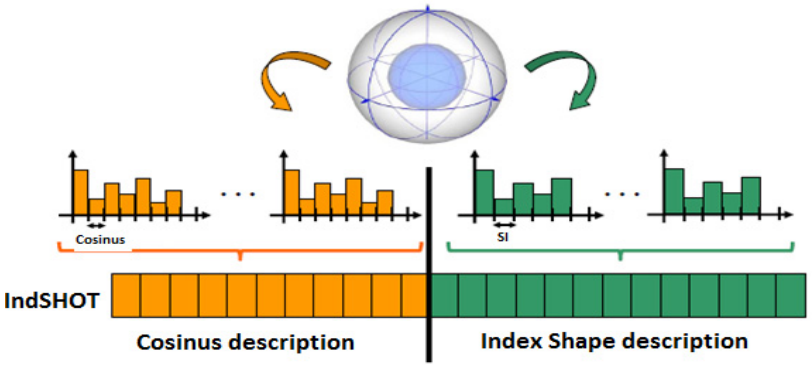


Fig. 1. IndSHOT representation

2.4 Matching and Recognition

We validate the proposed detector and descriptor using a view matching approach. Here, we focus on solving the surface matching problem based on local features, by point-to-point correspondences obtained by matching local invariant descriptors of feature points. Given a test object, we compute a measure of similarity between descriptors extracted on the test view and those of the models in database. The information (model ID, histogram, surface type, the centroïd, mean and standard deviation of SI) are used for matching process. Hence, for each histogram from test view, we find the best matching histogram from database view using the Euclidian distance. To speed up the comparison process, we use a KD-tree structure. Two keypoints are matched according to their histogram distance and their types of surface. For a test object, a set of nearest neighbors is returned after histogram matching. In the case of multiple correspondences, the potential corresponding pairs are filtered based on the geometric constraint: Euclidean distance between features coordinates of the two matched surface patches. The closest couple of features in term of coordinates

distance is the more likely to form a consistent correspondence. A system of incremental votes for each class gives the final matched class.

3 Experimental Results

3.1 Data and Parameters

We performed our experiments on two real range data sets. The first one is our own dataset (Lab-Dataset) captured with the Kinect sensor and composed of 20 objects (Ex. prism, ball, fan, trash can, etc) with 3 to 10 different angle views per object (figure 3). The second data set is the public RGB-D Object Dataset[2] (figure 2). There are 51 common household object categories. In our experimentation, we use 46 objects with 25 views per object for only one object per category, which constitute a dataset of 1150 views. The list of the following objects are labelled from 1 to 46 respectively: apple_1, ball_1, banana_1, bell_peper_1, binder_1, calculator_1, camera_1, cap_1, cell_phone_1, cereal_box_3, coffee_mug_1, comb_1, flashlight_1, food_bag_1, food_box_1, food_can_1, food_cup_1, garlic_1, greens_1, hand_towel_1, instant_noodles_1, keyboard_1, Kleenex_1, lemon_1, lightbulb_1, lime_1, marker_1, mushroom_1, notebook_1, onion_1, orange_1, peach_1, pear_1, pitcher_1, plate_1, potato_1, rubber_eraser_1, scissors_1, shampoo_1, soda_can_1, sponge_1, stapler_1, tomato_1, toothbrush_1 and watter_bottle_1. The numbers of feature points detected from these range images vary from 4 to 250, depending on the viewpoint and the complexity of input shape. In our experimentations, we have tried several values of our parameters and here we give the values achieving the best performance: $r_1 = 2$, $r_2 = 0.04$, $span = 5$, $\alpha = 0.05$, $\beta = 0.05$, $\varepsilon_H = 0.009$, $\varepsilon_K = 0.0001$, $\varepsilon_C = 0.01$.

Fig. 2. Examples of objects from the RGB-D Object Dataset[2]

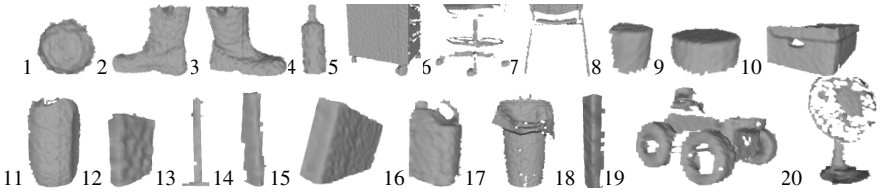


Fig. 3. The 20 objects of the lab-Dataset

[2] http://www.cs.washington.edu/rgbd-dataset/

3.2 Keypoint Stability

To evaluate detector performance, we illustrate a visual comparison of keypoint positions detected with SC_HK, SC_HK_connex, and SID detectors as shown on figure 4. It reveals that the final selected points are quite well localized. The combining process allows a better feature point filtering than SC or HK alone, as false detected points in both are eliminated, and points with correct surface type remain. Figure 5 illustrates the relative stability of keypoint's positions detected with SC_HK_connex detector when varying viewpoints for the same object. Clearly, we recover almost same keypoint positions in the different views. For a quantitative analysis showing the superior repeatability of our keypoints, we refer the reader to our previous publication [12].

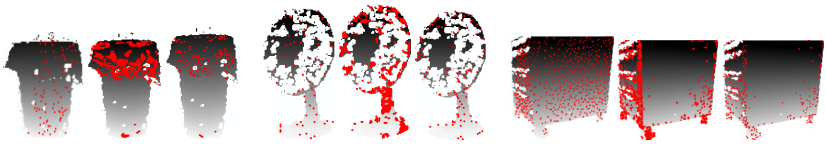


Fig. 4. Detected keypoints on trash can, fan and storage cupboard models with: SID in first column, SC_HK in second column and SC_HK_connex in third column

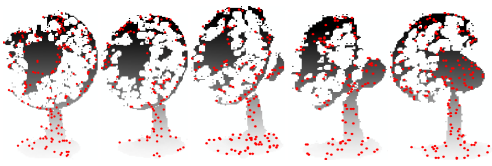


Fig. 5. Detected keypoint on fan model with SC_HK_connex, in view angle variation

3.3 Matching Result

The test protocol for object recognition from different angle views is the following: for the RGBD dataset, we select one test view from the N total number of views in the dataset, and the N-1 views are used as the training set; this process is repeated for the N views of the whole database. For the Lab-Dataset, we select one to four random views per object as the query and use the remaining views for training. We carry out three experiments using the three descriptors SHOT, CSHOT and IndSHOT. The same evaluation is done for the two detectors SID and SC_HK_connex. The overall recognition rates, which correspond to the mean recognition rate over the objects, are given in table 1 for respectively our Lab-Dataset and the RGBD dataset. In figure 6, the cross recognition rates between models are displayed in the confusion matrix. Gray level determines the rate of the recognition. Black is for high and white is for low recognition rate. The overall recognition rate is quite promising for our SC_HK_connex method in comparison to the SID results, with 91.12% on the RGBD dataset. This rate is achieved using the new proposed descriptor IndSHOT, which suggests that it is more descriptive than the CSHOT and SHOT versions. The recognition rate in the Lab-Dataset is about 82%. The reason behind this lower result is the high similarity between object shapes included in this dataset (two boots objects, parallelepipedic shapes, cylindrical shapes, etc). In another hand, the recognition rate

varies according to the view angle chosen for the query. In fact, higher rates are achieved when the query's view angle is given between two view angles in the training base.

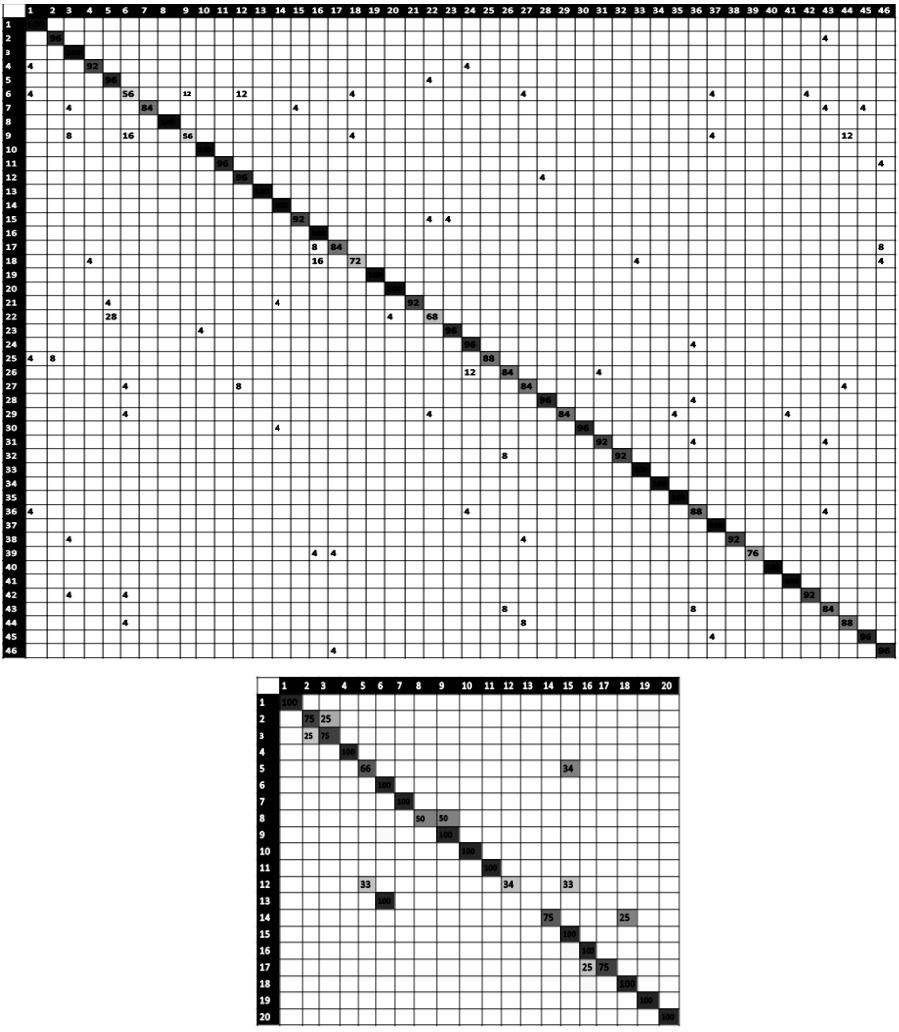


Fig. 6. Confusion matrix for the result of SC_HK_connex method on RGB-D object dataset (Top) and on the Lab-Dataset (Bottom)

The conjunction of the SC_HK_connex detector with the IndSHOT descriptor seems to provide more pertinent description of the local surface typology. It should also be noted that the overall computation time for our recognition process (detection+ description+ matching features) is quite low (~0.7s), which is a great advantage when dealing with real time application.

Table 1. Recognition rates for our Lab-Dataset (on left) and RGB-D object dataset (on right)

	IndSHOT	SHOT	CSHOT
SC_HK	**82.5%**	67,5%	65%

	IndSHOT	SHOT	CSHOT
SID	89.06%	70,75%	77.77%
SC_HK	**91.12%**	75,28%	82.14%

4 Conclusions and Perspectives

We have presented two main complementary contributions: 1/ an original 3D keypoint detector, SC_HK_connex, based on the idea of combining criteria; 2/ a new 3D keypoint descriptor, IndSHOT, based solely on shape characteristics.

The proposed detector combines SC (shape curvedness) and HK criteria with the principle of connected components. It was already shown in our previous work that the selected 3D keypoints are more repeatable than for alternative detectors, and this is confirmed here by the good inter-view matching reached in our experiments. The proposed IndSHOT descriptor encodes the occurrence frequency of shape index values vs. the cosine of the angle between the normal of reference feature point and that of its neighbours. It seems to be significantly more descriptive than original SHOT and CSHOT from which we have crafted it.

Finally, our new combination of SC_HK_connex detector + IndSHOT descriptor is evaluated in challenging 3D object recognition scenarios characterized by the presence of viewpoint variations and a few number of views on real-world depth data. The outcome is very promising results, with 92% correct recognition on 46 objects from a public dataset, and 82% on our own Lab-Dataset containing 20 "everyday" objects, some of which are rather similar one to another.

For the moment, measures of curvatures in our process are calculated at a constant scale level, so the feature's scale is still ambiguous. To overcome this fact, we plan, as a future work, to search for features at different scale levels.

References

[1] Medioni, G.G., François, A.R.J.: 3-D structures for generic object recognition. In: International Conference on Pattern Recognition, pp. 30–37 (2000)

[2] Scovanner, P., Ali, S., Shah, M.: A 3-dimensional SIFT descriptor and its application to action recognition. ACM Multimedia, 357–360 (2007)

[3] Vikstén, F., Nordberg, K., Kalms, M.: Point-of-Interest Detection for Range Data. In: ICPR, pp. 1–4 (2008)

[4] Knopp, J., Prasad, M., Willems, G., Timofte, R., Van Gool, L.: Hough Transform and 3D SURF for robust three dimensional classification. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part VI. LNCS, vol. 6316, pp. 589–602. Springer, Heidelberg (2010)

[5] Tombari, F., Salti, S., Di Stefano, L.: Unique Signatures of Histograms for Local Surface Description. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part III. LNCS, vol. 6313, pp. 356–369. Springer, Heidelberg (2010)

[6] Johnson, A.E., Hebert, M.: Using spin images for efficient object recognition in cluttered 3d scenes. IEEE PAMI 21, 433–449 (1999)

[7] Chen, H., Bhanu, B.: 3D free-form object recognition in range images using local surface patches. Pattern Recognition Letters 28(10), 1252–1262 (2007)
[8] Akagunduz, E., Eskizara, O., Ulusoy, I.: Scale-space approach for the comparison of HK and SC curvature descriptions as applied to object recognition. In: ICIP, pp. 413–416 (2009)
[9] Cantzler, H., Fisher, R.B.: Comparison of HK and SC curvature description methods. In: Conference on 3D Digital Imaging and Modeling, pp. 285–291 (2001)
[10] Koenderink, J., Doorn, A.J.: Surface shape and curvature scale. Image Vis. Comput. 10(8), 557–565 (1992)
[11] Tombari, F., Salti, S., Di Stefano, L.: A combined texture-shape descriptor for enhanced 3D feature matching. In: ICIP, pp. 11–14 (2011)
[12] Shaiek, A., Moutarde, F.: Détecteurs de points d'intérêt 3D basés sur la courbure. In: COmpression et REprésentation des Signaux Audiovisuels, CORESA (2012)

Incremental Dense Reconstruction from Sparse 3D Points with an Integrated Level-of-Detail Concept

Jan Roters and Xiaoyi Jiang

Department of Computer Science, University of Münster,
Einsteinstraße 62, 48149 Münster, Germany
{jan.roters,xjiang}@uni-muenster.de

Abstract. For decades scene reconstruction from multiple images is a topic in computer vision and photogrammetry communities. Typical applications require very precise reconstructions and are not bound to a limited computation time. Techniques for these applications are based on complete sets of images to compute the scene geometry. They require a huge amount of resources and computation time before delivering results for visualization or further processing.

In the application of disaster management these approaches are not an option since the reconstructed data has to be available as soon as possible. Especially, when it comes to *Miniature Unmanned Aerial Vehicles (MUAVs)* sending aerial images to a ground station wirelessly while flying, operators can use the 3D data to explore the virtual world and to control the MUAVs.

In this paper an incremental approach for dense reconstructions from sparse datasets is presented. Instead of focussing on complete datasets and delivering results at the end of the computation process, our incremental approach delivers reasonable results while computing, for instance, to quickly visualize the virtual world or to create obstacle maps.

1 Introduction

Scene reconstruction from multiple images is still a hot topic in the computer vision and photogrammetry community. Current approaches are focussing on accuracy while requiring a lot of computation resources and computation time and delivering results at the end of the computation process.

Since the 3D information is only available after complete computation there are some use cases which are not suitable to use these approaches, e. g. disaster management with a swarm of *Miniature Unmanned Aerial Vehicles (MUAVs)* delivering still images over the air while flying. A quick visualization would help operators to get a better view of the scene and to control the MUAVs (see Fig. 1). For that purpose, one can show reconstructed 3D points directly [1] or build up a 3D mesh from these points [2]. Another example is the creation of obstacle maps for autonomous flights of those MUAVs. Therefore, the denser the map of 3D points, the better obstacles are known and collisions can be prevented.

A taxonomy of 3D reconstruction is given in Table 1. In the following we discuss the different cases with focus on the incremental cases.

A lot of research has been done on sparse 3D reconstruction from wide-baseline image data. Whereas there are some non-incremental algorithms to compute the sparse

X. Jiang et al. (Eds.): WDIA 2012, LNCS 7854, pp. 116–125, 2013.

Fig. 1. Operators controlling MUAVs in a simulated environment and exploring the scene at a multi-touch wall

Table 1. Different cases of reconstruction types pointing out which cases have been extensively studied in the literature

	incremental	non-incremental
narrow-baseline	yes	yes
wide-baseline	yes	yes

(a) sparse reconstruction

	incremental	non-incremental
narrow-baseline	yes	yes
wide-baseline	no	yes

(b) dense reconstruction

scene geometry and the camera positions at once [3], a lot of methods compute this data incrementally one or a few images after another [4].

For dense reconstruction of wide-baseline images current methods focus on computing very accurate 3D information but at the cost of long computation times [5]. At the downside the current dense reconstruction approaches are not designed to work incrementally and thus, they are not suitable for all applications, e. g. the previously mentioned disaster management.

However, incremental dense reconstruction for videos or narrow-baseline image data, respectively, has been covered in literature, even for live video streams [6]. Since these methods are focussing on small baseline image data they are not suitable for wide baseline dense reconstructions.

In this paper we present an approach to incremental wide baseline dense reconstruction from sparse 3D dataset computed from multiple still images. The main contributions are (1) the reconstruction of reasonable points to get a quick denser overview of the scene within a few seconds, (2) to get incremental supply of denser 3D data while the data is further refined incrementally in the background and furthermore, (3) the approach we present integrates a level-of-detail concept.

Our incremental dense reconstruction approach is presented in Section 2. In Section 3 we show some experiments and results to evaluate our algorithm. On that account we use a ground truth dataset to get a reliable quality measure. We conclude this work in Section 4.

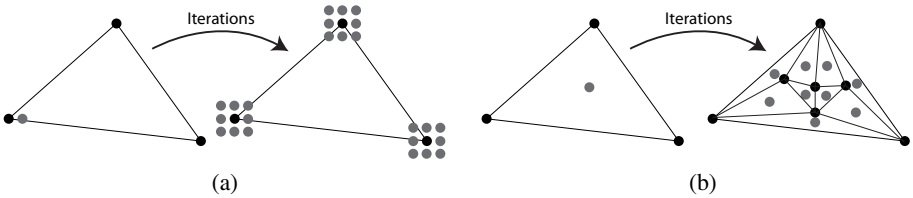


Fig. 2. Comparison of dense reconstruction methods. (a) Traditional approaches reconstruct points in the neighborhood of already known points. (b) Our incremental approach firstly reconstructs the midpoints of given triangles.

2 Incremental Dense Reconstruction

For traditional approaches it does not matter in which order the 3D points of a scene are computed as long as the final reconstruction is correct. In general, these approaches compute dense reconstructions by searching point matches in the neighborhood around already known scene points [5]. This technique has the advantage to get a consistency between neighbored matches since they are very close to each other. Furthermore, this consistency measure also detects larger discontinuities in the depth data, e. g. at the borders of objects. As a downside, these algorithms are not designed to work incrementally. Even if they may be adapted to work incrementally delivering results throughout the computation the reconstructed information will not be reasonable due to the low visual entropy. There is a very high density around the previously known points but the major areas between these points would not contain any information up to a later computation progress.

Especially in disaster management with flying MUAVs the operator does not gain much more information from a bunch of neighbored points as they may appear as one point in the virtual world which the operator explores and in which he has to control the MUAVs.

In our approach we focus on creating information in those major areas between known points instead of only in their neighborhood. To get a reasonable incremental result we reconstruct the midpoints of each triangle in a given 2D triangulation of the already known 2D points, at the beginning the sparse feature matches. These midpoints have the maximum distance to the points building up the triangles. A reconstructed point therefore gives at once more information than only neighbored points (see Fig. 2). Furthermore, instead of computing the dense reconstruction from all images at once using a lot of computation time before returning a result, our approach computes the dense reconstruction incrementally from two images at a time.

On the one hand the results can be visualized very quickly with good visual entropy and can be used for further computations, e.g. incremental mesh computation. On the other hand we have to deal with the problem of matching feature points without relying on the consistency of neighbored points. These matchings are much more computationally expensive due to a wider search range, even with guided matching along the epipolar line. For that purpose, we will introduce a method to limit the search range and therefore, decrease the computation time.

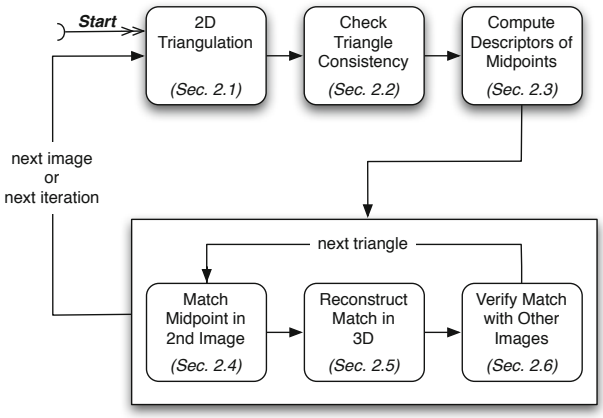


Fig. 3. Outline of the computation steps of our algorithm

In Figure 3 the outline of our algorithm is shown. It consists of 6 steps which have to be done for each image or each iteration. Due to the structure of our algorithm it is possible to change the quality of parts of the reconstruction very dynamically. For instance, it is possible to compute the reconstructions of some images toward a specified level of detail by computing more iterations for one image or a few images, respectively. Furthermore, it is possible to get all images toward the same level of detail, e. g. when image 1 has been processed for 3 iterations and image 2 has been processed for 1 iteration it is possible to compute the 2 remaining iterations of image 2 to get image 1 and 2 toward the same level of detail. The total number of iterations required to compute the complete dense reconstruction of a set of images depends on the requested density, i. e. the best level of detail.

At the beginning of the whole process and the beginning of each iteration a 2D triangulation (see Sec. 2.1) has to be computed from the known 2D feature points in one image that have corresponding matches in another image. For each triangle of the triangulation we do a consistency check (see Sec. 2.2), i. e. it is checked if a triangle is good enough to try the reconstruction of its midpoint. To match the midpoints in a further step we have to compute their descriptors (see Sec. 2.3). A successfully matched midpoint (see Sec. 2.4) can then be reconstructed to get the 3D location (see Sec. 2.5). Finally, a verification step (see Sec. 2.6) using other images ensures that the match is correct.

2.1 2D Triangulation

A triangulation method of the sparse 2D points is used to determine reasonable points to reconstruct. The reconstruction is done for the midpoint of each triangle since it is the point that has the maximum distance to the triangle points and thus, that is the one with maximum visual entropy for our purpose.

The triangulation is computed from the set of known 2D feature points of one image that have corresponding matches in another image. At the beginning the set is equal to

the sparse matched feature points. After each iteration the successfully reconstructed midpoints are added to this set.

Due to the fact that our algorithm is optimized for GPU computation and the structure of our algorithm is focussing on short computation time we cannot currently use triangulation algorithms that may insert additional points to compute a good triangulation.

One of the best triangulation algorithms without the requirement to insert additional points is the Delaunay triangulation due to its ability to maximize the minimum angle. It guarantees that the minimum angle is at least as large as in any other triangulation method. In the next section (see Sec. 2.2) we will use a filter rule to reject triangles with interior angles that are smaller than a specified threshold.

2.2 Triangle Consistency

Traditional non-incremental approaches search the features within a small area around a previously known sparse feature point. For these approaches it is more likely to find the correct match requiring less computation time [5].

In our approach the feature matching is the most computation time-consuming part due to missing constraints that limit the search region. The only valid constraint is the epipolar constraint. Although the search for matches has to be done only on the epipolar line, it would be a very computation time-consuming task.

To save computation time, we have to bound the search region as much as possible. We propose to search for feature matches only in the corresponding triangle in the second image which is given by the previously known feature point matches of the first image. This boundary combined with the epipolar constraint significantly limits the search region.

This type of boundary has advantages and disadvantages. On the one hand, it reduces the number of points to match and thus, reduces the computation time. Furthermore, as long as the correct point match is inside the triangle the uncertainties of the point matching are reduced since there may be a better match on the remaining region of the epipolar line. On the other hand, the point may be outside the triangle and thus it cannot be matched correctly using this triangle constraint. This problem can be handled in the same way other algorithms handle occluded points or points which lie outside the normal image boundary, e. g. a thresholding operation in the matching quality (see Sec. 2.4).

There is no guarantee that the correct feature point is really within such a triangle. In fact, some of the triangles are more likely to contain the correct point and others are more unlikely. We propose to filter out the triangles in the latter case. On that account we use the following four filter rules.

1. **Level of detail**
   Our level-of-detail concept is covered by this filter rule. A triangle which has a smaller surface area than the specified threshold will be marked to be completed. This threshold is given by the best level of detail that is requested.
2. **Interior angle limitation**
   A triangle that contains an interior angle less than 10 degrees will be rejected. This rule is applied to both triangles, the one in the first image and the corresponding

one in the second image which is deformed by another perspective, i. e. another camera position.

3. **Surface area ratio**
   The ratio between the surface area of the triangle in one image to the surface area of the corresponding triangle in the other image has to be between $\frac{2}{3}$ and $\frac{3}{2}$, i. e. if the surface area of one triangle is larger than 1.5 times the surface area of the other triangle it will be rejected.
4. **Orientation test of the triangle points**
   A triangle is rejected if the rotation of the triangle points in the one image do make a left turn and in the other a right turn or vice versa, i. e. it is rejected if the sign of the signed areas of the triangles differ between the first image and the second image.

If a triangle is rejected the corresponding triangle in the other image is rejected as well, since they are linked through the feature matches. The given thresholds have been determined by experiments.

There are two ways to handle rejected triangles. Firstly, the search region of a rejected triangle is extended, for instance to the whole epipolar line. Secondly, a rejected triangle will not be handled further. Since our goal is to compute the first denser reconstruction as fast as possible we have chosen the second way. Triangles that have been marked to be completed are not handled further since they already reached the finest level of detail.

2.3 Computation of Feature Descriptors

To match feature points we are in need for a reliable feature descriptor, especially due to the task of wide baseline matching. A further requirement for the feature descriptor is the fast computation since we have to compute a lot of descriptors. For each triangle one descriptor in the first image and up to a few thousand descriptors in the other image have to be computed.

One popular feature descriptor for dense feature matching is DAISY [7] which has been improved by Wan et. al. [8]. Although this descriptor shows good performance in these early works we could not achieve a good matching rate in our case.

Instead of using a dense feature descriptor our approach uses Fast Retinal Keypoints (FREAK) [9] which has been designed for sparse features but also shows very good performance for dense matching. Furthermore, its computation process is simple and could be implemented very efficiently on the GPU.

2.4 Point Matching

Given the feature descriptor of the midpoint of a triangle we have to find the correct match in the bounded search region given by the intersection of the epipolar line and the corresponding triangle in the second image. Therefore, we have to compute a descriptor for every pixel on this line segment and compare it to the descriptor of the midpoint.

If the given descriptor matches one of the descriptors in the search region with a certain quality and if there is not any other descriptor in that region matching with a similar quality we will mark the corresponding pixel location as the correct match. If none of the features in the search region matched the midpoint with a certain quality we would reject the triangle.

To decide if a feature matches with a certain quality we check whether the quality q is smaller than a given threshold $t_{\text{best}} = 80$, so $q < t_{\text{best}}$. To accept this match every other potential descriptor i in the search region should match with a worse quality $q_i > 1.2 \cdot q$.

2.5 3D Triangulation

To triangulate the matched points and therefore retrieve the reconstructed 3D point a lot of methods are available [10]. These methods are almost all based on non-linear refinements either in the 3D space or in the image space.

In our approach we use an algorithm which is a good trade off between computation expense, accuracy, and simplicity. The normalized direct linear transform for two-view reconstructions [10] can be implemented efficiently on the GPU while delivering good results.

Similar to other reconstruction algorithms the angle of the two rays between the 3D point and the camera centers should be at least 2° to prevent numerically unstable points.

2.6 Point Verification

To verify a reconstructed point we project it onto at most two other image planes which also contain the three matched points of the triangle and search for the feature point in a small region around the projection.

A feature point is rejected if none of the images is confirming the feature point at the projected position. The verification step is very important since there a two different cases outliers could occur: (1) The correct match is inside the triangle but another match has been found. Thus, the false match has to be rejected by this verification step. (2) The correct match is occluded or outside the triangle so that it is impossible to find the correct match inside the triangle. Thus, a false match inside the triangle may be detected which has to be rejected by this verification.

A rejected point will not occur in the final reconstruction based on this image combination but the point or a point close to it is likely to be reconstructed in another image combination.

3 Experiments and Results

To evaluate our approach we have generated a ground truth dataset from the City of Sights model [11] containing seven images (see Fig. 4). We have done that by rendering the scene twice. Firstly, the model has been rendered photorealistically. Secondly, for each photorealistic rendering we extracted a depth image which represents the ground truth data. Furthermore, we stored the precise camera position and orientation

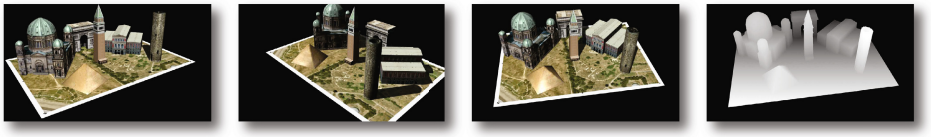
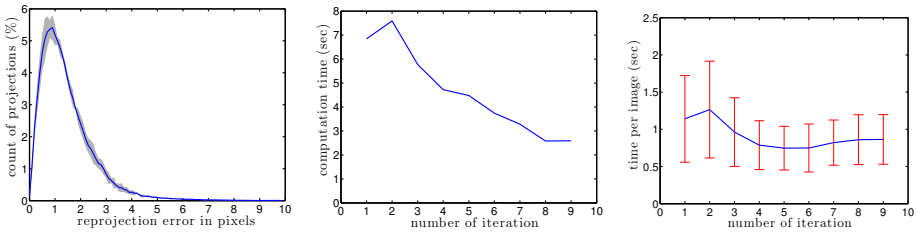


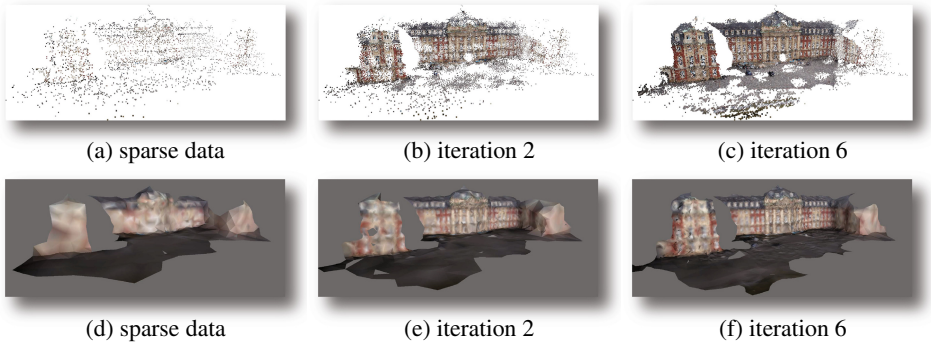
Fig. 4. Three example photorealistic renderings of the ground truth dataset. The image at the right shows a depth image corresponding to the third image.



(a) (b) (c)

Fig. 5. Evaluation: (a) Relative histogram of reprojection errors with additional standard deviation (gray area), (b) total computation time per iteration and (c) mean computation time for each image and each iteration from the ground truth dataset

Fig. 6. Example aerial images of a testing sequence with 7 images showing the front of the castle of Münster



(a) sparse data (b) iteration 2 (c) iteration 6

(d) sparse data (e) iteration 2 (f) iteration 6

Fig. 7. (a) - (c) Incremental dense reconstruction, (d) - (f) mesh reconstruction from the incremental dense reconstructions

for each image. Since the created images are synthetic we also know the intrinsic camera calibration.

For the evaluation the black areas around the scene are ignored since there is neither correct image data nor depth data. We measure the accuracy, namely the reprojection error, i.e. the Euclidean distance between the reprojected point and correct point which is the projection of the ground truth 3D point onto the same image plane.

In comparison to the ground truth data we reach a mean accuracy of 1.499 pixels and a median accuracy of 1.220 pixels. The histogram of reprojection errors related to ground truth is shown in Figure 5(a). The standard deviation is presented by the gray area and describes the deviation of single iterations.

Further results are given by a second scene which consists of 7 real aerial images (see Fig. 6). The quality of our incremental dense reconstruction approach is shown in Figure 7(a) - (c). There, the quality of different iterations are presented. Some areas cannot be reconstructed more densely due to the triangle filters. In Figure 7(d) - (f) the process of mesh reconstruction is shown on the incrementally dense reconstructed point clouds. The result shown there are created with a further development of [2].

Besides the accuracy we measured the computation time for each iteration. The computation has been done on a computer with an Intel Core i7 930 processor with 2.8 GHz, 12 GB of RAM and a nVidia GTX 470 graphics device.

The first incremental update of the 3D points is delivered in less than 4 seconds. For the ground truth dataset the computation time for all images and one iteration is decreasing over time (see Fig. 5(b)). Whereas in iteration 2 the number of triangles has increased the number of triangles in each iteration afterwards is decreased mainly due to the level-of-detail triangle filter, i.e. more and more triangles have reached the best level of detail. Furthermore, some images have reached the best level of detail and do not need further computations. In Figure 5(c) one can see a similar result for single images, except the difference at the end of the reconstruction process. At the last iterations there is only one image left which did not have reached the best level of detail.

4 Conclusion

In this paper we have presented an approach to computing a dense reconstruction incrementally from wide baseline images and previously known sparse geometry. There are several applications for which our approach is applicable, especially, computation of obstacle maps and quick 3D visualization of the captured scene. While other algorithms require a lot of time to present the first result, our approach retrieves first results within a few seconds.

Furthermore, our approach reconstructs the points in a reasonable order. Instead of reconstructing neighbored points of already known scene points, the points which have the maximum distance to its neighbors are reconstructed. Thus, the major empty areas of the 3D scene gets filled earlier with information.

The feature descriptor is the most critical part in our approach since it decides whether the midpoint of a triangle could be matched correctly or not. Thus, we will concentrate on improving the used feature descriptor or on designing a new feature descriptor with better matching performance and lower computation expense.

One of the problems with this approach concerns the borders of objects which are unlikely to be reconstructed in general. Especially, with very wide baseline and thin objects the borders of these objects are not be reconstructed very well. On that account we will study a hybrid approach combining the proposed method and another method for reconstructing the scene points near the borders.

Acknowledgements. This work was developed in the project AVIGLE funded by the State of North Rhine Westphalia (NRW), Germany, and the European Union, European Regional Development Fund "Europe - Investing in your future". AVIGLE is conducted in cooperation with several industrial and academic partners. We thank all project partners for their work and contributions to the project. Furthermore, we thank Cenalo GmbH for their image acquisition.

References

1. Roters, J., Steinicke, F., Hinrichs, K.H.: Quasi-real-time 3d reconstruction from low-altitude aerial images. In: Zlatanova, S., Ledoux, H., Fendel, E., Rumor, M. (eds.) Proc. of the 28th Urban Data Management Symposium, pp. 231–241 (2011)
2. Vierjahn, T., Lorenz, G., Mostafawy, S., Hinrichs, K.H.: Growing cell structures learning a progressive mesh during surface reconstruction - a top-down approach. In: Andujar, C., Puppo, E. (eds.) EG 2012 - Short Papers, pp. 29–32 (2012)
3. Crandall, D., Owens, A., Snavely, N., Huttenlocher, D.P.: Discrete-continuous optimization for large-scale structure from motion. In: Proc. of IEEE Conf. on Computer Vision and Pattern Recognition, pp. 3001–3008 (2011)
4. Gherardi, R., Farenzena, M., Fusiello, A.: Improving the efficiency of hierarchical structure-and-motion. In: Proc. of IEEE Conf. on Computer Vision and Pattern Recognition, pp. 1594–1600 (2010)
5. Furukawa, Y., Ponce, J.: Accurate, dense, and robust multi-view stereopsis. IEEE Trans. on Pattern Analysis and Machine Intelligence 32(8), 1362–1376 (2010)
6. Newcombe, R.A., Davison, A.J.: Live dense reconstruction with a single moving camera. In: Proc. of IEEE Conf. on Computer Vision and Pattern Recognition, pp. 1498–1505 (2010)
7. Tola, E., Lepetit, V., Fua, P.: Daisy: An efficient dense descriptor applied to wide baseline stereo. IEEE Trans. on Pattern Analysis and Machine Intelligence 32(5), 815–830 (2010)
8. Wan, Y., Miao, Z., Tang, Z., Wan, L., Wang, Z.: An efficient wide-baseline dense matching descriptor. IEICE Transactions 95-D(7), 2021–2024 (2012)
9. Alahi, A., Ortiz, R., Vandergheynst, P.: FREAK: Fast Retina Keypoint. In: Proc. of IEEE Conf. on Computer Vision and Pattern Recognition, pp. 510–517 (2012)
10. Hartley, R.I., Zisserman, A.: Multiple View Geometry in Computer Vision, 2nd edn. Cambridge University Press (2004)
11. Gruber, L., Gauglitz, S., Ventura, J., Zollmann, S., Huber, M., Schlegel, M., Klinker, G., Schmalstieg, D., Höllerer, T.: The city of sights: Design, construction, and measurement of an augmented reality stage set. In: Proc. of the 9th IEEE International Symposium on Mixed and Augmented Reality, pp. 157–163 (2010)

Probability-Based Dynamic Time Warping for Gesture Recognition on RGB-D Data

Miguel Ángel Bautista[1,2], Antonio Hernández-Vela[1,2], Victor Ponce[1,2], Xavier Perez-Sala[2,3], Xavier Baró[2,4], Oriol Pujol[1,2], Cecilio Angulo[3], and Sergio Escalera[1,2]

[1] Dept. Matemàtica Aplicada i Anàlisi, Universitat de Barcelona, Gran Via 585, 08007 Barcelona, Spain
[2] Centre de Visió per Computador, Campus UAB, Edifici O, 08193 Bellaterra, Barcelona, Spain
[3] CETpD-UPC, Universitat Politècnica de Catalunya Neàpolis, Rambla de l'Exposició, 59-69, 08800 Vilanova i la Geltrú, Spain
[4] EIMT, Universitat Oberta de Catalunya, Rambla del Poblenou 156, 08018, Barcelona, Spain

Abstract. *Dynamic Time Warping* (DTW) is commonly used in gesture recognition tasks in order to tackle the temporal length variability of gestures. In the DTW framework, a set of gesture patterns are compared one by one to a maybe infinite test sequence, and a query gesture category is recognized if a warping cost below a certain threshold is found within the test sequence. Nevertheless, either taking one single sample per gesture category or a set of isolated samples may not encode the variability of such gesture category. In this paper, a probability-based DTW for gesture recognition is proposed. Different samples of the same gesture pattern obtained from RGB-Depth data are used to build a Gaussian-based probabilistic model of the gesture. Finally, the cost of DTW has been adapted accordingly to the new model. The proposed approach is tested in a challenging scenario, showing better performance of the probability-based DTW in comparison to state-of-the-art approaches for gesture recognition on RGB-D data.

Keywords: Depth maps, Gesture Recognition, Dynamic Time Warping, Statistical Pattern Recognition.

1 Introduction

Nowadays, human gesture recognition is one of the most challenging tasks in computer vision. Current methodologies have shown preliminary results on very simple scenarios, but they are still far from human performance. Due to the large number of potential applications involving human gesture recognition in fields like surveillance [8], sign language recognition [10], or in clinical assistance [9] among others, there is a large and active research community devoted to deal with this problem.

The release of the Microsoft Kinect^{TM} sensor in late 2010 has allowed an easy and inexpensive access to synchronized range imaging with standard video data. This data combines both sources into what is commonly named RGB-D images (RGB plus Depth). This very welcomed addition by the computer vision community has reduced the burden of the first steps in many pipelines devoted to image or object segmentation and opened new questions such as how these data can be effectively fused and

X. Jiang et al. (Eds.): WDIA 2012, LNCS 7854, pp. 126–135, 2013.

described. This depth information has been particularly exploited for human body segmentation and tracking. Shotton et. al [11] presented one of the greatest advances in the extraction of the human body pose using RGB-D, which is provided as part of the Kinect^{TM} human recognition framework. The extraction of body pose information opens the door to one of the most challenging problems nowadays, i.e. human gesture recognition. This fact has enabled researchers to apply new techniques to obtain more discriminative features. As a consequence, new methodologies on gesture recognition can improve their performance by using RGB-D data.

From a learning point of view, the problem of human gesture recognition is an example of sequential learning. The main problem in this scenario comes from the fact that data sequences may have different temporal duration and even be composed of intrinsically a different set of component elements. There are two main approaches for this problem: On the one hand, methods such as Hidden Markov Models (HMM) or Conditional Random Fields (CRF) are commonly used to tackle the problem from a probabilistic point of view [10], especially for classification purposes. Furthermore, methods based on key poses for gesture recognition have been proposed [6]. On the other hand, dynamic programming inspired algorithms can be used for both alignment and clustering of temporal series [5]. One of the most common dynamic programming methods used for gesture recognition is Dynamic Time Warping (DTW) [3,4].

However, the application of such methods to gesture recognition in complex scenarios becomes a hard task due to the high variability of environmental conditions. Common problems are: the wide range of human pose configurations, influence of background, continuity of human movements, spontaneity of humans actions, speed, appearance of unexpected objects, illumination changes, partial occlusions, or different points of view, just to mention a few. These effects can cause dramatic changes in the description of a certain gesture, generating a great intra-class variability. In this sense, since usual DTW is applied to compare a sequence and a single pattern, it fails when such variability is taken into account. We propose a probability-based extension of DTW method, able to perform an alignment between a sequence and a set of N pattern samples from the same gesture category. The variance caused by environmental factors is modelled using a Gaussian Mixture Model (GMM) [7]. Consequently, the distance metric used in the DTW framework is redefined in order to provide a probability-based measure. Results on a public and challenging computer vision dataset show a better performance of the proposed probability-based DTW in comparison to standard approaches.

The remaining of this paper is organized as follows: Section 2 presents the probability-based DTW method for gesture recognition, Section 4 presents the results and, finally, Section 5 concludes the paper.

2 Standard DTW for Begin-End Gesture Recognition

In this section we first describe the original DTW and its common extension to detect a certain pattern sequence given a continuous and maybe infinite data stream. Then, we extend the DTW in order to align several patterns, taking into account the variance of the training sequence by means of a Gaussian mixture model.

2.1 Dynamic Time Warping

The original DTW algorithm was defined to match temporal distortions between two models, finding an alignment/warping path between the two time series $Q = \{q_1, .., q_n\}$ and $C = \{c_1, .., c_m\}$. In order to align these two sequences, a $M_{m \times n}$ matrix is designed, where the position (i, j) of the matrix contains the alignment cost between c_i and q_j. Then, a warping path of length τ is defined as a set of contiguous matrix elements, defining a mapping between C and Q: $W = \{w_1, .., w_\tau\}$, where w_i indexes a position in the cost matrix. This warping path is typically subjected to several constraints:

Boundary conditions: $w_1 = (1, 1)$ and $w_\tau = (m, n)$.

Continuity and monotonicity: Given $w_{\tau'-1} = (a', b')$, then $w_{\tau'} = (a, b)$, $a - a' \leq 1$ and $b - b' \leq 1$, this condition forces the points in W to be monotonically spaced in time.

We are generally interested in the final warping path that, satisfying these conditions, minimizes the warping cost:

$$DTW(M) = \min_{w} \{M(w_\tau)\}, \tag{1}$$

where τ compensates the different lengths of the warping paths. This path can be found very efficiently using dynamic programming. The cost at a certain position $M(i, j)$ can be found as the composition of the Euclidean distance $d(i, j)$ between the feature vectors of the sequences c_i and q_j and the minimum cost of the adjacent elements of the cost matrix up to that point, i.e.:

$$M(i, j) = d(i, j) + \min\{M(i - 1, j - 1), M(i - 1, j), M(i, j - 1)\}. \tag{2}$$

Given the streaming nature of our problem, the input vector Q has no definite length and may contain several occurrences of the gesture pattern C. At that point the system considers that there is correspondence between the current block k in Q and a gesture if satisfying the following condition, $M(m, k) < \mu, k \in [1, .., \infty]$ for a given cost threshold μ.

This threshold value is estimated in advance using leave-one-out cross-validation strategy. This involves using a single observation from the original sample as the validation data, and the remaining observations as the training data. This is repeated such that each observation in the sample is used once as the validation data. At each iteration, we evaluate the similarity value between the candidate and the rest of the training set. Finally we choose the threshold value which is associated with the largest number of hits.

Once detected a possible end of pattern of gesture, the working path W can be found through backtracking of the minimum path from $M(m, k)$ to $M(0, z)$, being z the instant of time in Q where the gesture begins. Note that $d(i, j)$ is the cost function which measures the difference among our descriptors V_i and V_j.

An example of a begin-end gesture recognition together with the warping path estimation is shown in Figure 2.

3 Handling Variance with Probability-Based DTW

Consider a training set of N sequences $\{S_1, S_2, \ldots, S_N\}$, where each S_g represents a sample of the same gesture class. Then, each sequence S_g composed by a set of feature vectors at each time t, $S_g = \{s_1^g, \ldots, s_{L_g}^g\}$ for a certain gesture category, where L_g is the length in frames of sequence S_g. Let us assume that sequences are ordered according to their length, so that $L_{g-1} \leq L_g \leq L_{g+1}, \forall g \in [2, .., N-1]$, the median length sequence is $\bar{S} = S_{\lceil \frac{N}{2} \rceil}$. This sequence $\bar{S}$ is used as a reference, and the rest of sequences are aligned with it using the classical Dynamic Time Warping with Euclidean distance [3], in order to avoid the temporal deformations of different samples from the same gesture category. Therefore, after the alignment process, all sequences have length $L_{\lceil \frac{N}{2} \rceil}$. We define the set of warped sequences as $\tilde{S} = \{\tilde{S}_1, \tilde{S}_2, \ldots, \tilde{S}_N\}$. Once all samples are aligned, the features vectors corresponding to each sequence element at a certain time t $\tilde{s}_t$ are modelled by means of an $G-$component Gaussian Mixture Model (GMM) $\lambda_t = \{\alpha_k, \mu_k, \Sigma_k\}, \quad k = 1, \ldots, G$, α is the mixing value and μ and Σ are the parameters of each of the G Gaussian models in the mixture. The underlying reason of choosing a GMM instead of a single Gaussian follows from the definition of the problem, where an arbitrarily large number of samples $\{S_1, S_2, \ldots, S_N\}$ is available. In this sense, in order to accurately model the feature vectors a GMM seems a more powerful way to model the variability than a single Gaussian. As a result, each one of the GMMs that model each component of a gesture pattern $\tilde{s}_t$ is defined as follows:

$$p(\tilde{s}_t) = \sum_{k=1}^{G} \alpha_k \cdot e^{-\frac{1}{2}(x-\mu_k)^T \cdot \Sigma_k^{-1} \cdot (x-\mu_k)}. \tag{3}$$

The resulting model is composed by the set of GMMs that model each one of the component elements among all warped sequences of a certain gesture class. An example of the process is shown in Figure 1.

3.1 Distance Measures

In the classical DTW, a pattern and a sequence are aligned using a distance metric, such as the Euclidean distance. Since our gesture pattern is modelled by means of probabilistic models, if we want to use the principles of DTW, the distance needs to be redefined. In this paper we consider a soft-distance based on the probability of a point belonging to each one of the G components in the GMM, i.e., the posterior probability of x is obtained according to (3). In addition, since $\sum_{1}^{k} \alpha_k = 1$, we can compute the probability of an element $q \in Q$ belonging to the whole GMM λ as the following:

$$P(q, \lambda) = \sum_{k=1}^{M} \alpha_k \cdot P(q)_k, \tag{4}$$

$$P(q)_k = e^{-\frac{1}{2}(q-\mu_k)^T \cdot \Sigma_k^{-1} \cdot (q-\mu_k)}, \tag{5}$$

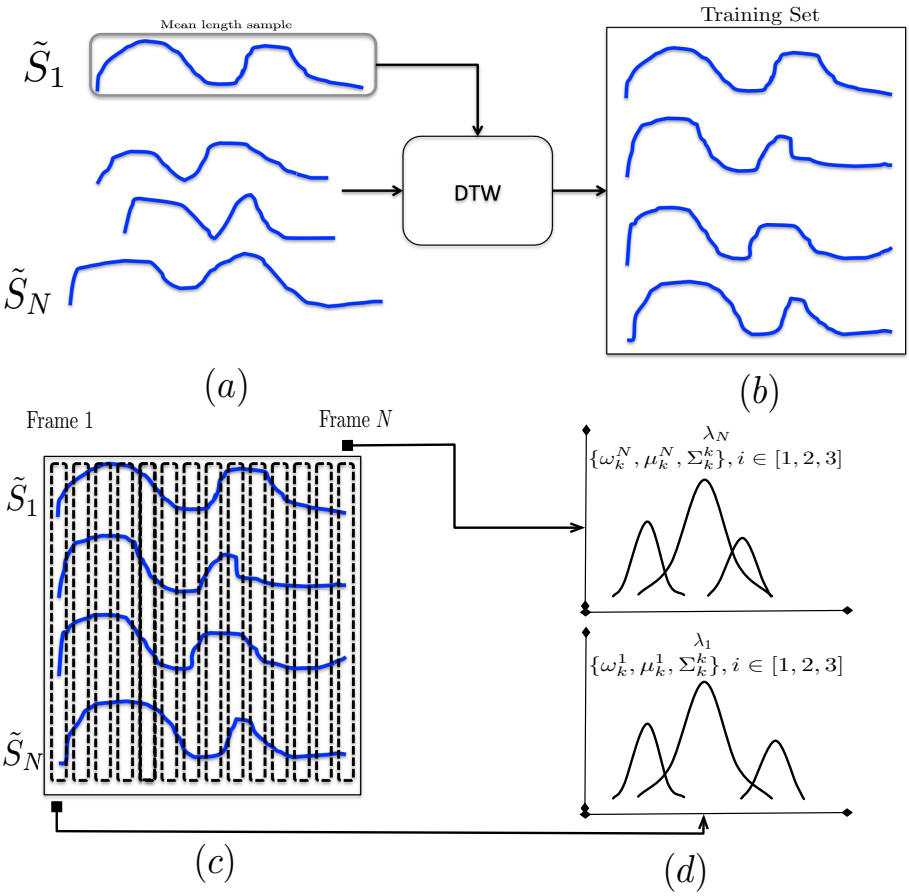


Fig. 1. (a) Different sample sequences of a certain gesture category and the mean length sample. (b) Alignment of all samples with the mean length sample by means of Euclidean DTW. (c) Warped sequences set $\tilde{S}$ from which each set of t-th elements among all sequences are modelled. (d) Gaussian Mixture Model learning with 3 components.

which is the sum of the weighted probability of each component. An additional step is required since the standard DTW algorithm is conceived for distances instead of similarity measures. In this sense, we use a soft-distance based measure of the probability, which is defined as:

$$D(q, \lambda) = e^{-P(q,\lambda)}. \tag{6}$$

In conclusion, possible temporal deformations of the gesture category are taken into account by aligning the set of *N* gesture sample sequences. In addition, modelling with a GMM each of the elements which compose the resulting warped sequences, we obtain a methodology for gesture detection that is able to deal with multiple deformations in data. The algorithm that summarizes the use of the probability-based DTW to detect start-end of gesture categories is shown in Table 1. Figure 4 illustrates the application of the algorithm in a toy problem.

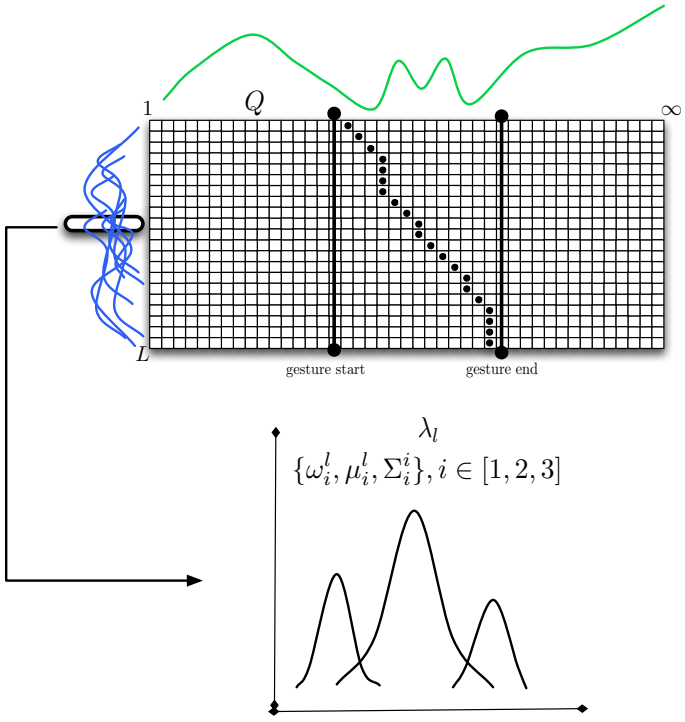


Fig. 2. Begin-end of gesture recognition of a gesture pattern in an infinite sequence Q using the probability-based DTW. Note that different samples of the same gesture category are modelled with a GMM and this model is used to provide a probability-based distance. In this sense, each cell of M will contain the accumulative D distance.

Table 1. Probability-based DTW algorithm

Input: A gesture model $C = \{c_1, .., c_m\}$ with corresponding GMM models $\lambda = \{\lambda_1, .., \lambda_m\}$, its similarity threshold value μ, and the testing sequence $Q = \{q_1, .., q_v\}$. Cost matrix $M_{m \times v}$ is defined, where $\mathcal{N}(x), x = (i, t)$ is the set of three upper-left neighbor locations of x in M.
Output: Working path W of the detected gesture, if any.
// Initialization
for $i = 1 : m$ **do**
  for $j = 1 : \infty$ **do**
    $M(i, j) = v$
end **end**
for $j = 1 : v$ **do**
  $M(0, j) = 0$
end
for $t = 0 : v$ **do**
  for $i = 1 : m$ **do**
    $x = (i, t)$
    $M(x) = D(q_t, \lambda_i) + \min_{x' \in \mathcal{N}(x)} M(x')$
  end
  if $M(m, t) < \epsilon$ **then**
    $W = \{\text{argmin}_{x' \in \mathcal{N}(x)} M(x')\}$
    return
  end
end

4 Experiments

In order to present the experiments, we discuss the data, methods and evaluation measurements.

4.1 Data

The data source used is the ChaLearn [2][1] data set provided from the CVPR2012 Workshop challenge on Gesture Recognition. The data set consists of 50,000 gestures each one portraying a single user in front of a fixed camera. The images are captured by the Kinect^{TM} device providing both RGB and depth images. The data used (a subset of the whole) are 20 development batches with a manually tagged gesture segmentation. Each batch includes 100 recorded gestures, grouped in sequences of 1 to 5 gestures performed by the same user. For each sequence the actor performs a resting gesture between each gesture of the gestures to classify. For this data set, we performed background subtraction based on depth maps, and we defined a 10×10 grid approach to extract HOG+HOF feature descriptors per cell, which are finally concatenated in a full

[1] `http://gesture.chalearn.org/data/data-examples`

(a) (b)

Fig. 3. Sample a) depth and b) RGB image for the ChaLearn database

image (posture) descriptor. In this data set we will test the recognition of the resting gesture pattern, using 100 samples of the pattern in a ten-fold validation procedure. An example of the ChaLearn dataset is shown in Figure 3.

4.2 Methods and Evaluation

We compare the usual DTW and Hidden Markov Model (HMM) algorithms with our probability-based DTW approach using the proposed distance D shown in (6). The evaluation measurements are the accuracy of the recognition and the overlapping for the resting gesture (in percentage). We consider that a gesture is correctly detected if the overlapping in the resting gesture sub-sequence is grater than 60% (a standard overlapping value). The cost-threshold for all experiments was obtained by cross-validation on training data, using a 5-fold cross-validation, and the confidence interval was computed with a two-tailed t-test. Each GMM in the probability-based DTW was fit with $k = 4$ components, this value was obtained using a 2-fold cross-validation procedure on training data. For HMM, it was trained using the Baum-Welch algorithm, and 3 states were experimentally set for the resting gesture, using a vocabulary of 60 symbols computed using K-means over the training data features. Final recognition is performed with temporal sliding windows of different wide sizes, based on the training samples length variability.

Table 2 shows the results of HMM and the classical DTW algorithm, in comparison to our proposal on the ChaLearn dataset. We can see how the proposed probability-based DTW outperforms the usual DTW and HMM algorithms in both experiments. Moreover, confidence intervals of DTW and HMM do not intersect with the probability-based DTW in any case. From this results we can observe how performing dynamic programming increases the generalization capability of the HMM approach, as well as a model defined by a set of GMMs outperforms the classical DTW [3] on RGB-Depth data without increasing the computational complexity of the method.

Fig. 4. Examples of resting gesture detection on the Chalearn dataset using the probability-based DTW approach. The line below each pair of depth and RGB images represents the detection of a resting gesture.

Table 2. Overlapping and Accuracy results of different gesture recognition approaches

	Overlap.	Acc.
Probability-based DTW	**39.08± 2.11**	**67.81±2.39**
Euclidean DTW	30.03±3.02	60.43± 3.21
HMM	28.51±4.32	53.28±5.19

5 Conclusions and Future Work

In this paper, we proposed a probability-based DTW for gesture recognition on RGB-D data, where the pattern model is learned from several samples of the same gesture category. Different sequences were used to build a Gaussian-based probabilistic model of the gesture whose possible deformations are implicitly encoded. In addition, a soft-distance based on the posterior probability of the GMM was defined. The novel approach has been successfully applied on a public RGB-D gestures dataset, being able to deal with multiple deformations in data, and showing performance improvements compared to the classical DTW and HMM approaches. In particular, the proposed method

benefits from both the generalization capability from the probabilistic framework, when several observations of the training data are available, and the temporal warping capability from dynamic programming.

Future work lines include, between others, the inclusion of samples with different points of view of the same gesture class, the analysis of state-of-the-art one-class classifiers in order to obtain a performance improvement, and the definition of powerful descriptors to obtain gesture-discriminative features.

Acknowledgements. This work is partly supported by projects IMSERSO-Ministerio de Sanidad 2011 Ref. MEDIMINDER and RECERCAIXA 2011 Ref. REMEDI, and SUR-DEC of the Generalitat de Catalunya and FSE. The work of Antonio is supported by an FPU fellowship from the Ministerio de Educacion of Spain.

References

1. Mikolajczyk, K., Tuytelaars, T., Schmid, C., Zisserman, A., Matas, J., Schaffalitzky, F., Kadir, T., Van Gool, L.: A comparison of affine region detectors. International Journal of Computer Vision 65(1/2), 43–72 (2005)
2. ChaLearn Gesture Dataset (CGD 2011), ChaLearn, California, Copyright (c) ChaLearn - 2011 (2011)
3. Sakoe, H., Chiba, S.: Dynamic programming algorithm optimization for spoken word recognition. IEEE Transactions on Acoustics, Speech and Signal Processing 26(1), 43–49 (1978)
4. Reyes, M., Dominguez, G., Escalera, S.: Feature weighting in dynamic time warping for gesture recognition in depth data. In: 2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops), pp. 1182–1188 (2011)
5. Zhou, F., la Torre, F.D., Hodgins, J.K.: Hierarchical aligned cluster analysis for temporal clustering of human motion. IEEE Transaction on Pattern Analysis and Machine Intelligence 35(3), 582–596 (2010)
6. Lv, F., Nevatia, R.: Single View Human Action Recognition using Key Pose Matching and Viterbi Path Searching. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2007, pp. 1–8, 17–22 (2007)
7. Svensen, M., Bishop, C.M.: Robust bayesian mixture modelling. In: Proceedings of European Symposium on Artificial Neural Networks, vol. 64, pp. 235–252 (2005)
8. Hampapur, A., Brown, L., Connell, J., Ekin, A., Haas, N., Lu, M., Merkl, H., Pankanti, S.: Smart video surveillance: exploring the concept of multiscale spatiotemporal tracking. IEEE Signal Processing Magazine 22(2), 38–51 (2005)
9. Pentland, A.: Socially aware computation and communication. Computer 38, 33–40 (2005)
10. Starner, T., Pentland, A.: Real-time American Sign Language recognition from video using hidden Markov models. In: Proceedings of the International Symposium on Computer Vision, pp. 265–270 (1995)
11. Shotton, J., Sharp, T., Kipman, A., Fitzgibbon, A., Finocchio, M., Blake, A., Cook, M., Moore, R.: Real-time human pose recognition in parts from single depth images. Communications of the ACM 56, 116–124 (2013)

Towards an Augmented Reality System for Violin Learning Support

Hiroyuki Shiino, François de Sorbier, and Hideo Saito

Graduate School of Science and Technology, Keio University, Yokohama, Japan
{shiino,fdesorbi,saito}@hvrl.ics.keio.ac.jp

Abstract. The violin is one of the most beautiful but also one of the most difficult musical instruments for a beginner. This paper presents an on-going work about a new augmented reality system for training how to play violin. We propose to help the players by virtually guiding the movement of the bow and the correct position of their fingers for pressing the strings. Our system also recognizes the musical note played and the correctness of its pitch. The main benefit of our system is that it does not require any specific marker since our real-time solution is based on a depth camera.

Keywords: Augmented Reality, Marker-Less, Violin Pedagogy, Depth Camera.

1 Introduction

Learning how to play violin is very difficult for a novice player. Unlike the guitar, the violin has no frets or marks to help the finger placement. Violinists also have to maintain a good body posture for the bowing movement. Some studies state that a player needs approximately 700 hours to master the basics of violin bowing [1].

Some methods have been introduced to help this learning process. MusicJacket [2] is a wearable system with a vibrotactile feedback that guides the player's movements. However, we consider that wearing such specific device limits the ease of the players since they will not practice under normal conditions. Moreover, this approach does not support the fingering teaching.

Augmented reality technology has the benefit to be non-intrusive and has consequently been applied to musical instrument learning. Motokawa and Saito [3] proposed a guitar support system that displays a computer-generated model of a hand. It helps the player for finger placement and overlays lines where to press the strings. However this kind of approach is using markers [4] added onto the instrument which makes it not robust to occlusions. The limits of markers can be overcome by using feature point detectors such as SIFT [5]. Although, the surface of the violin is very reflective and uniform which will provide a small number of unstable features that is not adapted for our system. Moreover, feature point detectors are often not robust to illumination changes.

In this on-going research, we proposed a marker-free system using augmented reality for violin pedagogy. It teaches the player where to correctly press the strings

X. Jiang et al. (Eds.): WDIA 2012, LNCS 7854, pp. 136–145, 2013.

on the fingerboard and how to perform the bowing movement by displaying virtual information on a screen. At the same time, our system analyses the musical note played and the correctness of its pitch (frequency of the sound). In this paper, we are aiming at presenting a technical description of our system and not yet focusing on the benefits of its pedagogic side.

We removed the constraint of markers and detectors by including a depth camera that capture the depth information from a scene in real-time. We take advantage of the classic Iterative Closest Point (ICP) algorithm [6] for estimating the pose of the violin based on a pre-reconstructed 3-D model. We also use the human body tracking capability of the depth camera for teaching novice player how to correctly manipulate the bow.

The remainder of the paper is structured as follows: Section 2 briefly gives an overview of our system. The detection and segmentation of the violin is presented in Section 3. In Section 4, we explain how we perform the real time tracking of the violin by taking advantage of several models stored in a database. Our approach for analysing and advising the bow movements is presented in Section 5. Section 6 details how we display the virtual information. Finally, in Section 7 and 8, we present quantitatively our results and our future extensions. Please note that we have not yet perform any user based studies, which will be organized in the future.

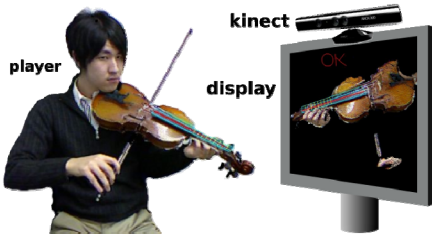


Fig. 1. The violinist is captured by a depth camera located over the screen that is a suitable position for both tracking and feedback processing. Virtual advice is displayed on a screen.

2 Overview of Our Learning System

Our system is based on Kinect[1], a depth camera that captures in real-time a colour image and its corresponding depth information. The depth information can easily be converted into a 3-D point cloud using internal parameters of the camera. The violin and the player are extracted from both of these images and analysed for estimating their pose in the 3-D space. Our learning approach is made of two parts: the first one focuses on the finger position, while the second tries to improve the bowing technique of the player. Virtual information for both approaches is displayed on a screen.

In the first case, our system displays the captured violin from a constant viewpoint (even if the player moves, the violin is presented from always the same viewpoint on

[1] www.kinectforwindows.org

the screen) with the virtual frets and emphasized strings. In the second case, we display a full view of the player with a virtual skeleton overlaid and containing specific tags located on the bowing arm bones. Meanwhile, a microphone captures the notes played by the violinist that are analysed for further virtual advice. An overview of our system is presented in Fig. 1.

3 Detection and Extraction of the Violin

To be able to display the virtual guides onto the violin, we need to detect and extract it from each input video frame. Based on this segmentation, we aim at computing the pose of the violin in the 3-D coordinate system, the same for which the virtual data have been defined. Our main constraint is to perform all those stages without the help of markers that give poor results with this setup.

3.1 Colour-Based Segmentation of the Violin

A common approach for detecting objects [7] like a violin is to segment the image based on the colour. Usually, a violin is characterized by its almost uniform brown surface. We base our segmentation on this fact. The segmentation is performed into the HSV colour space for avoiding miss-detection when the lighting condition slightly changes. The result presented in Fig. 2.a shows the limitations of this approach. This simple segmentation does not extract the fingerboard and other dark areas of the violin. Also, several parts of the surface are missing because of the specular reflections.

However, adding these dark and bright colours as candidates for the detection will result in a very noisy segmentation. For all those reasons, we extended the colour segmentation to the 3-D space by using the depth data.

(a) **(b)**

Fig. 2. (a) presents the rough segmentation of the violin based only on the colour information. (b) is the result of our segmentation using also the depth information.

3.2 Extension of the Segmentation with the Depth Information

To obtain the missing parts of the violin, we propose to also consider the depth data. Our idea is to base our segmentation on the general dimensions of a violin (about 600x155x30mm). Using the depth data from Kinect, we can get the 3-D coordinates corresponding to each of the pixels extracted during the colour segmentation. A volume with the same dimensions as a typical violin is then aligned with those points. The goal of this extended segmentation is to keep all the 3-D points inside of this volume as a result of our segmentation.

The main problem is to find the correct alignment of the volume with the segmented 3-D points. Firstly, we propose to compute the principal orientation describing the surface. It can also be represented as the plane that fits the maximum number of 3-D points. Second, we randomly select 3 points from which a planar equation is deduced and estimate the number of 3-D points resulting as inliers. This process is iteratively conducted as a RANSAC process until we found the maximum number of inliers.

We use the resulting plane equation for orientating the 3-D segmentation volume. The position of the volume is defined by computing the centroid of the points from the colour based segmentation. The improvement of the segmentation based on the depth information is presented in Fig. 2.b. Several other results are depicted in Fig. 3.

Fig. 3. Results of the colour plus depth based segmentation. Even specular and occluded parts are correctly segmented.

4 Tracking of the Violin

Displaying the virtual advice requires knowing where the violin is located according to a basis in which we defined the virtual objects, such as the frets. We then propose a method to robustly compute this rigid transformation in real time.

4.1 Tracking of the Violin

For the tracking, we estimate the pose that transforms the extracted violin to a pre-computed 3-D model of this violin. This transformation is computed using the ICP [6]

algorithm. This algorithm iteratively searches for each point of an input point cloud the closest point in a target point cloud. The rigid transformation (rotation and translation) is then computed with least-squares methods like described in [8, 9]. Our experience of the ICP algorithm suggests that a 3-D model defined with too many points will lead to a high computational time. Conversely, a 3-D model described with not enough points will decrease the accuracy of the pose estimation. We decided to use multiple models stored into a database instead of a single one detailed for optimizing the effectiveness of the pose estimation during the tracking phase.

During the tracking stage, we compute the plane equation from the current view of the violin. This information is compared with each of the models from the database and the closest solution is retained based on the difference of orientation. We then apply the ICP algorithm to evaluate the rigid transformation. The comparison with a specific model has also the advantage to provide a good initial guess of the transformation and, consequently, a faster convergence. Thanks to this approach, we are able to get the result of our tracking in real time.

4.2 Creation of the Database

As explained in the previous section, we are using a database to store the models used for the computation of the rigid transformation. During an offline stage, we capture the violin from multiple viewpoints. We tried mainly to capture the front face of the violin since it is the side that will be the mostly observed during the tracking. As explained in previous sections, we extract the violin and compute the plane equation corresponding to its surface.

Based on this information, we search in the database for a similar view by comparing the plane equations. If there is no result (difference of five degrees) then the candidate is added to the database. We follow this process to store up to 25 models, which is enough for capturing most of the surface and sides of the violin.

This offline stage is also used to create a detailed model of the violin by using the models from the database. It is used to define manually the position of the fret and other virtual data on the violin. The transformation between this full model and the other models is also stored in the database to be able to easily display the virtual advices on the screen.

5 Further Analysis for Bowing Advices

Tracking the violin is necessary for advising the player about the position of the finger, but hardly helps about the bowing technique. We then propose to track the position of the bowing arm and to analyse the sound produced by the movement of the bow on the string for advising about the position of the bow.

5.1 Tracking of the User

Playing violin requires a prefect movement of the bow that is difficult skill difficult to acquire. We propose to advise the novice violinists about the movements of their bow by comparing their gesture with the one from an accomplished player.

Our approach uses the skeleton tracking [10] included in *OpenNI*[2] to capture the movements from both the novice and the experimented players. It detects and tracks in real-time the different parts of the body and deduces from it a skeleton (joints and bones) defined in the 3-D space. The "skilled 3-D skeleton" movements are captured beforehand and replayed during the learning stage. However, the skeletons may not directly match since the novice and the skilled players probably do not have the same body morphology. Our solution is to align the skilled skeleton to the novice's one by orienting and scaling the axis of shoulders. In that case, the shoulder of the bowing arm will correspond (position and orientation) for both of the skeletons. Finally, we scale the shoulder-elbow and elbow-hand bones from the skilled skeleton to match the size of the novice bones.

5.2 Sound Analysis

By visualizing the virtual frets and strings, the player can understand where to press to play the violin. However, it remains difficult to know if the bow was correctly placed on the string. Considering that the player correctly presses the string based on the virtual fret, our idea is to analyse the sound produced to modify the position of the bow. We use a spectrum analyser[3] based on a wavelet transformation to analyse the violin's sound and to evaluate the accuracy of the pitch in cent unit (1 cent unit represents a change of tone). By asking the user to play a given note and showing where to press the strings, we can compare the sound obtained with the expected one. The result of this comparison is then used to modify the position of the bow.

6 Augmented Reality Based Learning Support

6.1 Bowing Support

The players start this learning stage by selecting the string on which they want to practice. Then they need to follow the movements of the skilled violinist that we previously recorded. The parts of the bowing arm (shoulder, elbow and hand) have been emphasized with big dots. The dots from the skilled movement are coloured in red while the dots from the novice player are in white. Fig. 4 shows a view of our bowing support system.

We compare the position of the player's hand and elbow with the one from the skilled skeleton in the 3-D. If the distance is correct then an "OK" mark is displayed. Otherwise a "NG" mark is displayed. Shoulders are not considered since they are supposed to be at the same position for both skeletons. By persevering at maintaining the "OK" position, the player might be able to improve his skills when using the bow.

[2] http://www.openni.org
[3] http://www.fmod.org/

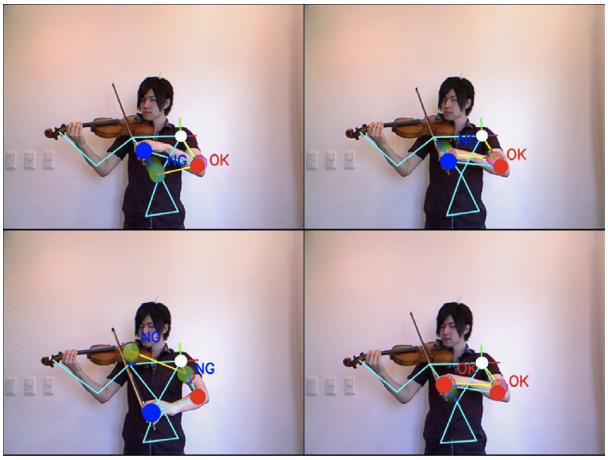


Fig. 4. The bowing support emphasized the elbow and the hang of the bowing arm. If the position differs from the pre-recorded movement then a message is displayed. The white point represents the align shoulder point between both skeletons. Greens points are the skeleton the player should follow, while the red and blue dots define the current position of the elbow and hand of the player.

6.2 Displaying the Frets

Our proposed system can teach where to place the finger on the neck of the violin by adding virtual frets and emphasizing the strings. The string and the fret that the violinist needs to press are displayed using respectively a red line and a red dot. Fig. 5 presents the virtual information overlaid onto the violin.

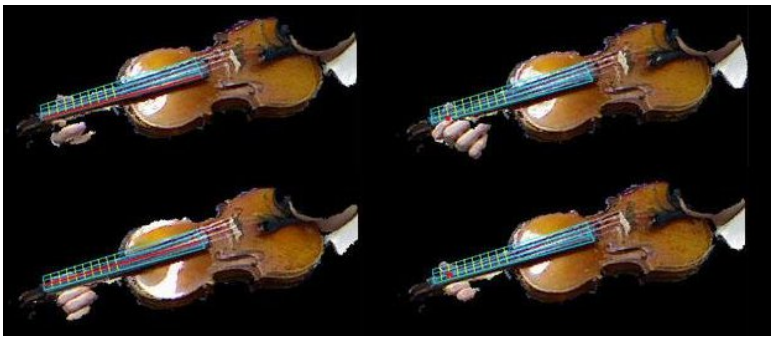


Fig. 5. Left side: The string that the player needs to press is in red. Right side: The fret that has to be pressed is marked with a red dot.

We decided to display the violin on the screen always from the same viewpoint. So even if the player moves the violin, it remains stable on the screen. This should allow the user to easily find the useful information on the screen since the virtual frets and strings will always be located at the same position.

6.3 Virtual Advice Based on Sound Analysis

If we consider that the player is correctly pressing the strings based on information provided by the virtual fingerboard then the goal of the sound analysis is to give advices about the position of the bow. Examples of our learning stage using the pitch's accuracy of the note played is depicted in Fig. 6. If the player applies the bow at the correct position then an "OK" mark is displayed. Otherwise, if the difference of pitch is too low or too high, then the corresponding "Low" or "High" marks are displayed. In this latest case, a green arrow is also shown to indicate to the player the direction where the bow has to be moved to get the correct pitch.

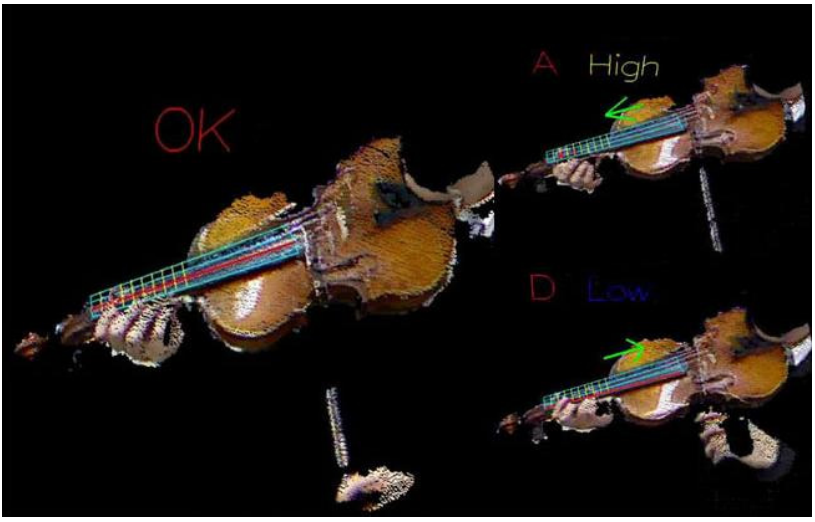


Fig. 6. Information is displayed to advise the position of the bow on the strings depending on the correctness of the pitch

7 Results

Experiments were performed on an "Intel Core2 DUO 2.80GHz" PC. We measured an average computational time of 21ms (~45 frames per second) that is suitable for a real-time rendering. For this experiment, we first evaluated the accuracy of our tracking approach based on ICP. We compared it with the AR-Toolkit marker tracking while trying to avoid occlusions of the markers. We added four markers on the body of the violin and pre-computed the sub-models based with it. During the online phase, we compute the rigid transformation between the first sub-model and the segmented violin using our approach and using the marker-based approach. Considering the marker-based transformation as the ground truth, we had the results presented in Table 1. Even if our results seem a little bit less accurate, our approach has still the benefit to be robust against occlusions.

Table 1. Evaluation of our tracking compared to the ground truth. It shows the rigid transformation matrix decomposed in three rotations and one translation.

	Rx(deg)	Ry(deg)	Rz(deg)	T(mm)
Minimum error	0.12	0.25	0.20	0.22
Maximum error	13.29	8.27	7.89	32.1
Average error	3.07	2.69	2.78	7.20

We also evaluated the accuracy of the virtual frets' position. Each fret has a corresponding pitch, so by pressing the strings we expect to obtain a similar pitch. For this experiment, we measured the correctness of the pitch when a skilled player (to ensure a correct manipulation of the bow) was using the virtual frets. Table 2 presents the results of this experiment for each fret where a difference of pitch closes to zero means that the accuracy is good (knowing that a value of 100 means that the tone is changed). These results show that the position of the frets is almost correct since the tone of the played note will be the expected one.

Table 2. Difference of pitch (in cent unit)

Fret number	1	2	3	4	5	6	7	8	9	Average
Difference of pitch	11.1	14.1	12.0	12.4	13.4	15.8	12.8	13.9	19.2	13.8

8 Conclusions

We have presented the technical part of our on-going work on a marker-free augmented reality system for assisting the novice violinists during their learning. Thanks to a depth camera, we are able to advise the player on his fingering and bowing techniques by displaying virtual information on a screen.

Our next step is to perform a user based study with novice and skilled players to confirm our choices or improve it. We are also working on a see-through HMD version of our system for a better view of the virtual information directly on the violin. Finally, we are thinking about applying our system on different musical instruments, like the Japanese shamisen, that also does not have frets.

References

1. Konczak, J., van der Velden, H., Jaeger, L.: Learning to play the violin: motor control by freezing, not freeing degrees of freedom by freezing. Journal of Motor Behavior 41(3), 243–252 (2009)

2. van der Linden, J., Schoonderwaldt, E., Bird, J., Johnson, R.: MusicJacket - Combining motion capture and vibrotactile feedback to teach violin bowing. IEEE Transactions on Instrumentation and Measurements, Special Issue on Haptic, Audio and Visual Environments for Games (2009)
3. Motokawa, Y., Saito, H.: Support system for guitar playing using augmented reality display. In: Proceedings of the 5th IEEE and ACM International Symposium on Mixed and Augmented Reality, pp. 243–244 (2006)
4. Kato, H., Billinghurst, M.: Marker tracking and HMD calibration for a video-based augmented reality conferencing system. In: Proceedings of the 2nd International Workshop on Augmented Reality, pp. 85–94 (1999)
5. Lowe, D.: Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision 60(2), 91–110 (2004)
6. Zhang, Z.: Iterative point matching for registration of freeform curves and surfaces. International Journal of Computer Vision 13(2), 119–152 (1994)
7. Cheng, H.D., Jiang, X.H., Sun, Y., Wang, J.: Color image segmentation: advances and prospects. Pattern Recognition 34(12), 2259–2281 (2001)
8. Arun, K., Huang, T., Blostein, S.: Least-squares fitting of two 3-D point sets. IEEE Transactions on Pattern Analysis and Machine Intelligence 5, 698–700 (1987)
9. Umeyama, S.: Least-squares estimation of transformation parameters between two points pattern. IEEE Transactions on Pattern Analysis and Machine Intelligence 13(4), 376–380 (1991)
10. Shotton, J., Fitzgibbon, A., Cook, M., Sharp, T., Finocchio, M.: Real-time human pose recognition in parts from single depth images. Communications of the ACM 56(1), 116–124 (2013)

Extraction and Visualization of Cardiac Beat by Grid-Based Active Stereo

Hirooki Aoki[1], Ryo Furukawa[1], Masahito Aoyama[1], Shinsaku Hiura[1], Ryusuke Sagawa[2], and Hiroshi Kawasaki[3]

[1] Hiroshima City University, Japan
{haoki,ryo-f,masa,hiura}@hirishoma-cu.ac.jp
[2] National Institute of Advanced Industrial Science and Technology, Japan
ryusuke.sagawa@aist.go.jp
[3] Kagoshima University, Japan
kawasaki@ibe.kagoshima-u.ac.jp

Abstract. We propose a method to observe cardiac beat from 3D shape information of body surface by using grid-based active stereo, and report preliminary experiments to evaluate validities of the proposed method. By comparing results of our proposed with those of electrocardiogram (ECG), we confirmed sufficient correspondences between peak intervals of inter-frame depth changes measured by the active stereo and R-R intervals measured by ECG. We tried the visualization of the spatial distribution of inter-frame depth change plotted on the 3D shape of chest region. And, the shape change by cardiac beat is mainly found on the left side of the chest region.

Keywords: Active stereo, Cardiac beat measurement, Non-contact physiological measurement.

1 Introduction

Some researchers proposed cardiac beat measurement without contact by applying the thermal imaging [1] and the microwave reflectometry [2] in order to decrease the discomfort of examinees by attaching sensing devices on their body. These methods need expensive measurement devices. Novel measurement method by using webcam was proposed as feasible solution with low-cost devices [3]. However, in this method, it is considered that the cardiac rate can be measured, but the waveform of cardiac beat cannot be measured accurately.

Hence, we propose a non-contact measurement of cardiac beat by applying a 3D measurement method based on active stereo. In the method, minim shape change of chest wall caused by cardiac movement. We consider that non-contact cardiac beat measurement by 3D image sensor has one advantage of obtaining spatial distribution of cardiac beat. We expect that the spatial distribution of cardiac beat change enable us to assess the cardiac function.

The active stereo systems that consist of cameras and video projectors have been widely used for 3D measurements. However, determining correspondences between

X. Jiang et al. (Eds.): WDIA 2012, LNCS 7854, pp. 146–157, 2013.

the projected 2D pattern and the captured image is a difficult problem. A stable solution that produces precise results is projecting multiple patterns (e.g., Gray code patterns [8], or phase shift methods [9]).

The active stereo using static-pattern light projection are suitable for capturing moving objects and have been widely researched [4, 5, 8, 12]. The methods are usually categorized into two types: temporal-encoding methods and spatial-encoding methods. Since a spatial-encoding method just requires a single input for reconstruction (a.k.a. one-shot scan), it is ideal to capture moving objects with high rate. Therefore, many researches have been involved in spatial-encoding methods [13]. However, since they require certain areas to encode information on object surfaces, the resolution tends to be low and reconstruction becomes unstable.

One of the approaches to encode information in efficient ways is to use a color code. By using multiple colors, multiple bits of information can be assigned to each pixel of the camera image. A color-based coding is suitable for spatial encoding [5, 6, 10, 14, 15]. However, it has some limitations and problems. The surface of the target objects must sufficiently reflect each color of the pattern. And, since the RGBs of off-the-shelf video projectors have overlapped spectral distribution, errors in determining colors of pixels are inevitable. To avoid those problems, several methods are proposed for efficient spatial encoding without using colors, such as dot patterns or grid patterns. Even though, there still remain several problems, i.e., ambiguities on correspondences and sparse reconstruction.

In this paper, we apply a new one-shot scanning method proposed by us [16] for making measurement of minute change in body surface caused by cardiac movement. The method uses an active stereo with a wave grid pattern that consists of vertical and horizontal sinusoidal lines. From each image captured by a high-frame-rate camera, 3D shape is reconstructed using multiple epipolar constraints of a connected grid pattern [4]-[7]. By using multiple epipolar constraints and continuity of a grid patterns, these types of methods have sufficient stability and density of measurement points.

We also made preliminary experiments about our proposed cardiac measurement and evaluated the validity of the measurement.

2 System Configuration

The 3D measurement system used in the present work consists of a camera and a projector (Fig. 1(a)). Parameters of the camera and the projector such as the focal length, aspect ratio, or angle of view are assumed to be known by calibration. The system uses a static pattern emitted from the projector, and no synchronization is required between the camera and the projector.

The projector casts a static pattern, which is shown in Fig. 1(b), on the target surface. The pattern is configured with vertical and horizontal sinusoidal curves to create grid shape (details are described in next section). And, it is captured as a series of images by the camera. By processing the images frame by frame, the dynamic shape of the target surface is reconstructed. Since the projected pattern is static with single color, no synchronization is required, high frame rate scanning is possible.

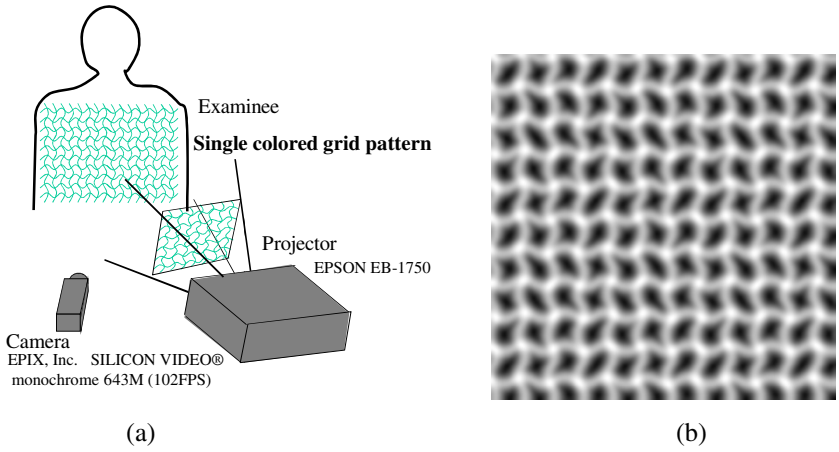


Fig. 1. (a) System configuration and (b) projected grid pattern

To make measurement of cardiac beat of a subject, the subject sits still on a chair and the pattern is cast to the breast surface from the front of the subject. The camera is also set in front of the subject, but the distance between the camera and the projector (the baseline) is set to be long enough so that the precision of the 3D measurement can be sufficiently high.

3 Method

3.1 3D Shape Reconstruction

In the cardiac beat measurement proposed by us, 3D shape reconstruction with high sensitivity and high spatial density is essential for detection of minim shape change of chest wall. In this paper, we apply the one-shot scanning method [16] which can solve the problem with the following 3D shape reconstruction. Overview of algorithm of the 3D shape reconstruction is shown in Fig. 3.

First, we detect curves from a captured image. We use the curve detection method using belief propagation method proposed by Sagawa et al. [5]. With the method, vertical and horizontal lines are robustly detected from a grid pattern from a single color. From the detected curves, intersection points are calculated, and a graph is also constructed by using intersection points as nodes of the graph. Then, for each intersection point, epipolar line on the projected pattern is calculated to find a correspondence. Since multiple candidates of correspondences are usually found, one solution can be determined by our belief propagation based technique. Finally, the depths for all the pixels are interpolated by matching between the pattern and the captured image and 3D shapes are densely reconstructed.

In the 3D reconstruction method, by using a wave-shaped grid pattern as shown in Fig. 1(b), the intersection points can be used as features for matching. Instead of

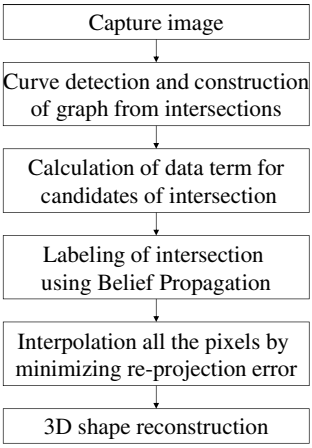


Fig. 2. Algorithm of 3D shape reconstruction

explicitly encoding the positional information of a structured light, the proposed pattern implicitly gives information which can make the order on the candidates of corresponding points.

To obtain unique correspondences between the camera and projector images by spatial encoding, a complicated pattern of large window size have been required in previous methods. Moreover, while the wider baseline is desirable to improve accuracy, the observed pattern will be more distorted, which makes it difficult to decode the pattern in practical cases. Therefore, we use a simple but informative pattern that is easy to detect and decode.

In this paper, we apply a pattern that gives information which can make the order on the candidates of corresponding points rather than get the unique correspondence through decoding process. The proposed pattern consists of vertical and horizontal directions of wave lines, which forms a grid pattern. Because each wave line is simple, it is easy to detect curves, and the position of a curve can be calculated in subpixel accuracy by detecting peaks of intensities of the curve.

The wave line is a sinusoidal pattern, which is periodic and self-recurring. The grid of wave lines, however, can give information for finding correspondences. In the method the intersection points of vertical and horizontal wave lines are used as feature points. The arrangement of intersection points is determined by the intervals and the wavelength of the wave lines. In the method, we use the same interval and wavelength for all the vertical and horizontal wave lines. However, as described in the following, because the interval of the vertical wave lines is not equal to the integral multiple of the horizontal wavelength, the intersection points appear at the different phases on the wave pattern; it means that the local pattern around an intersection point has local uniqueness, and it can be used as a discriminative feature. In this paper, we also use ‘wave patterns’ to refer to the wave lines.

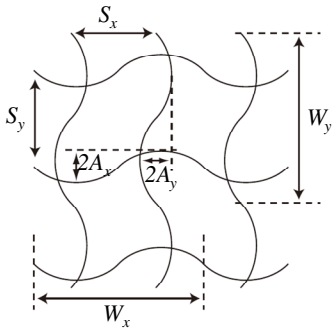
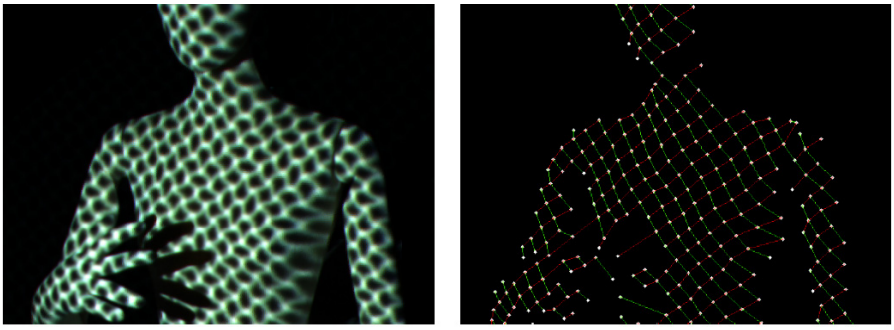


Fig. 3. Parameters of wave grid: S_x and S_y are the intervals between adjacent wave lines, W_x and W_y arethe wavelengths of a wave line, Ax and Ay are the amplitudes of waves with respect to vertical and horizontal lines, respectively



(a) (b)

Fig. 4. (a) input image, (b) detected grid

The local pattern around an intersection point is not globally-unique in the whole pattern and periodic. Therefore, the same pattern occurs at every Nx and Ny wave lines along the horizontal and vertical axes, where N_x = LCM(S_x, W_x) / S_x, N_y = LCM(S_y, W_y) / S_y where LCM(a, b) is the least common multiple of a and b. Hereafter, subscript letter x means the symbol describes values about horizontal axis, and y about vertical axis. S_x and S_y are the intervals between adjacent wave lines, and W_x and W_y are the wavelengths, as shown in Fig.3. The patterns, however, can be discriminative in each cycle.

In stereo matching, the candidates of corresponding points are restricted to the points on the epipolar line, as shown in Fig. 5. If an intersection point is located within a certain distance from the epipolar line, it is chosen as a candidate. The number of candidates depends on the position of intersection points in the camera image. Since the candidates are sparsely located in the projector image, the number of candidates is much smaller than the case of usual pixel-based stereo without restricting the search range.

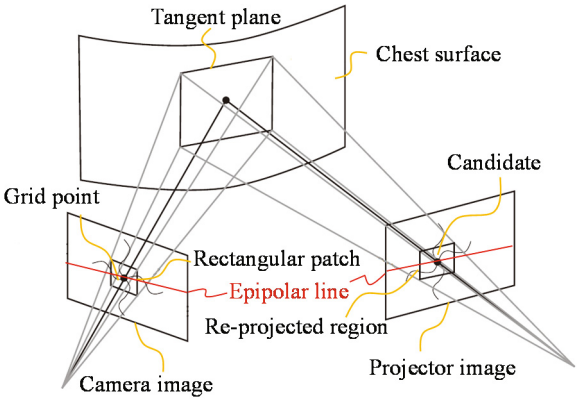


Fig. 5. The rectangular patch around a grid point is re-projected onto the projector's image plane to correspond the grid point with candidates by calculating matching costs

To find the best combinations of correspondences, a new optimizing method using regularization with matching cost of local patterns is introduced. In the method, the grid of wave patterns is detected by curve detection shown in Fig.4. The 3D shape reconstruction can be understood as an extension of a pixel-based stereo method for a camera-pair system to a grid-based stereo for a projector-camera system. With this method, as long as connected curves are detected on captured image, global optimization can be realized. And, to solve a sparse reconstruction because of grid pattern, we propose a (quasi-)pixel-wise interpolation and optimization technique based on image matching to estimate depth for all the pixels.

First, we calculate the matching costs for all the candidates as the data term for energy minimization. The cost is computed by Sum of Squared Differences (SSD) between the captured image and the projector image. Though, since the position of a grid point has some error and the pattern observed by the camera is distorted by the surface geometries, the simple SSD with rectangular patch is unsuitable for the data term. Therefore, we use the tangent plane of patch around the grid point to calculate a better matching cost and determine the correspondence with each candidate in sub-pixel accuracy.

The correspondences for sparse grid points are obtained by the grid-based stereo. The next step is to obtain dense correspondences by using all the pixels. We first calculate depth values of densely resampled pixels by interpolating the grid points using estimated local planes of surrounding grid points for each pixel. Then, the densely resampled depth values are optimized by minimizing the difference of intensity for all the pixels between camera image and projector image. In this work, independent depth estimation for each pixel is achieved by (quasi-)pixel-wise optimization based on photo-consistency.

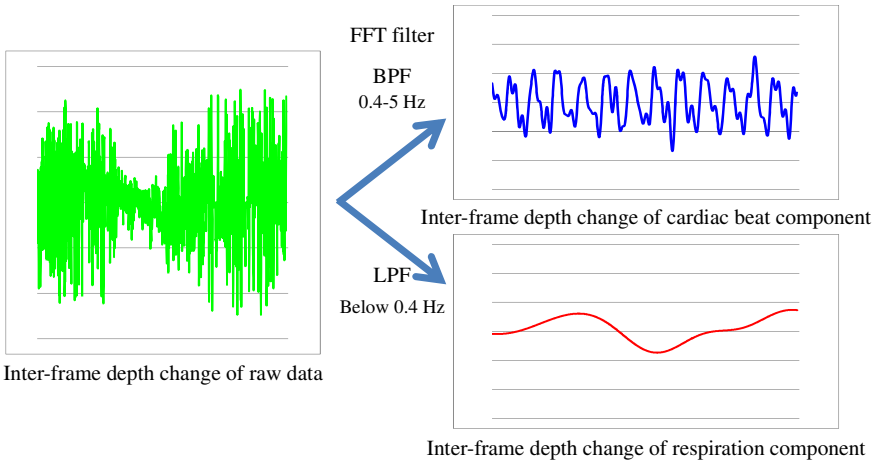


Fig. 6. Extraction of vital sign: Cardiac beat component and respiration component is extracted by applying FFT filter

3.2 Cardiac Beat Detection

Cardiac beat is extracted from time-series change of the point cloud reconstructed by above mentioned method. Since the reconstructed shapes are consist of unorganized vertices, it is not a simple process to compute the inter-frame correspondences for obtaining the time sequence of shapes. To obtain inter-frame correspondences, the point cloud is re-sampled at fixed 2D grid points arranged in xy-coordinates, where the z-coordinate of the re-sampled points are the depth values from the camera (here, it is assumed that the front direction from the camera is the z-axis). Then, the vertices sampled at the same xy-coordinate are set to be a set of corresponding points. In the algorithm for re-sampling the point cloud, 3D shape interpolation at the fixed xy-coordinates is required. In this work, Delaunay triangulation with linear interpolation was used to get the interpolated vertices [11].

Then, the time-series dataset of the depth change between frames in each re-sample vertices is computed. As shown in Fig. 6, the waveform of the time-series dataset is very noisy because of the camera image noise caused by image capture with high speed rate. However, by applying FFT band-pass filter which passes 0.4-5 Hz to the waveform of inter-frame depth change, the cardiac beat component is extracted. Here, the inter-frame depth change means the difference of depth between current frame (t) and previous frame (t-1). In a similar way, by applying the FFT low-pass filter which passes below 0.4 Hz to the inter-frame change of depth, respiration component is extracted.

4 Experiment and Results

Actual measurement by experimental system is executed to examine the validity of our proposed method. In experimental system, the SILICON VIDEO® monochrome

643M, manufactured by EPIX inc., is used as the high speed camera. The 643M provide a maximum of 211 FPS at VGA resolution. In this experiment, the frame rate is set at 100 FPS. The focal length of camera lens is 8mm. The EB-1750, which is manufactured by EPSON Corporation, is used as the pattern projector. The distance between the camera lens and the projector lens is set at 600mm.

Examinees are two health male (examinee A: age: 41 years old, body height: 171 cm, body weight: 62 kg/ examinee B: age: 39 years old, body height: 173 cm, body weight: 69 kg). Prior to the measurement, we obtained the consent document on the measurement execution from the examinees. In the measurement, the examinees wear a white T shirt. The measurement time is set at 30 seconds. In the measurement, at first, examinees stop breathing during about 10 seconds, and take breathing during last seconds.

Here, the experimental result about examinee A is shown below. Fig. 7 shows image. Fig. 8 shows point cloud reconstructed from projected pattern image and re-sampled point cloud. Fig. 9 shows a contour map computed by the re-sampled point cloud.

We examine the relationship between the periodicity of the filtered waveform by simultaneous measurement with ECG. The compact-type wireless ECG logger manufactured by LOGICAL PRODUCT CORPORATION is conducted in the simultaneous measurement. The electrodes of ECG are set on left breast region of the examinee. The sampling rate of ECG is set as 1000Hz. The graph shown in Fig. 10 is the raw waveform and the filtered waveform at point which are shown as x-mark in Fig. 9. The raw waveform includes much noise component. However, the filtered waveform periodically changes. The blue line shows the cardiac-beat waveform. And the red one shows the respiratory waveform obtained by applying low-pass filter which passes under 0.4Hz. The amplitude of filtered waveform is very small of an order of sub-millimeter per one cardiac-beat.

Fig. 7. Input image

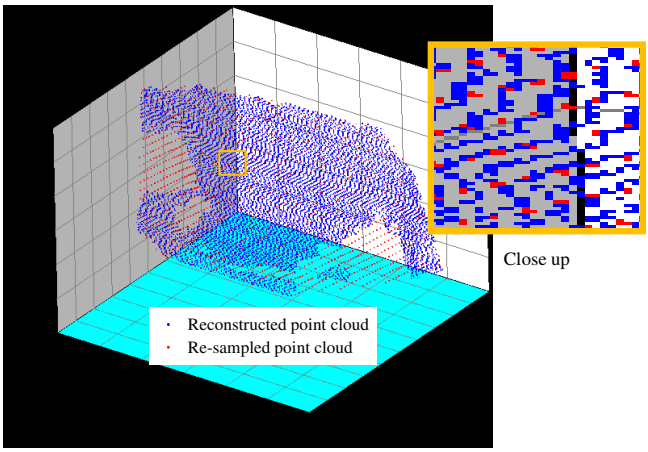


Fig. 8. Reconstructed point cloud and re-sampled point cloud

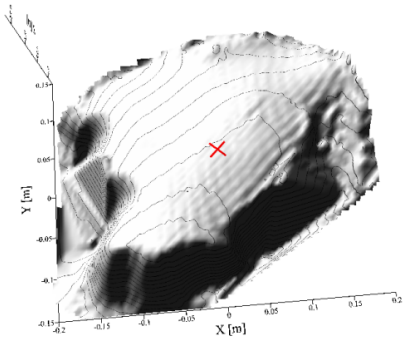


Fig. 9. Reconstructed 3D shape of chest region

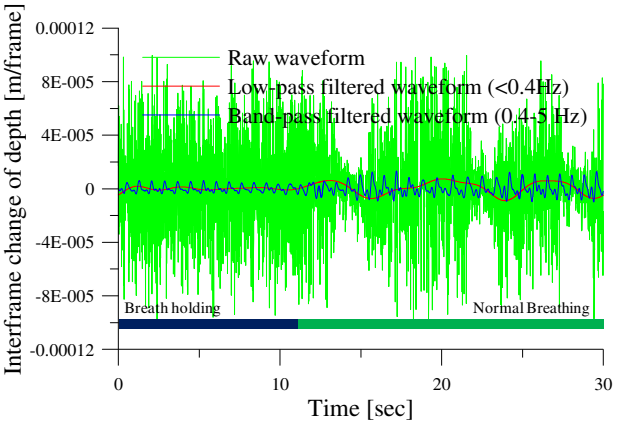


Fig. 10. Raw waveform and the bandpass-filtered waveform

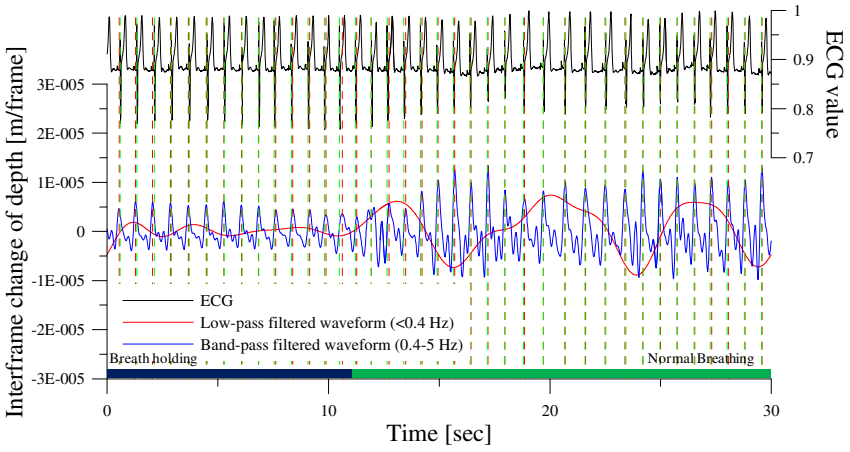


Fig. 11. Simultaneous measurement by our proposed method and ECG

We examine the relationship between the periodicity of the filtered waveform by simultaneous measurement with ECG. The compact-type wireless ECG logger manufactured by LOGICAL PRODUCT CORPORATION is conducted in the simultaneous measurement. The electrodes of ECG are set on left breast region of the examinee. As shown in Fig. 10, the R peaks in the ECG waveform basically correspond the peaks of inter-frame depth change measured by our system. Especially, there is sufficient correspondence during breath holding. Both peaks correspond during a large part of normal breathing, although unstable waveform appears in the inter-frame depth change during the early part.

The relationship between R-R interval of ECG waveform and peak interval of inter-frame depth change waveform is examined by the Bland-Altman plot, as shown in Fig. 12. Here, the R-R interval means the peak interval between continuing two R-peaks. The 95% coefficient interval (95%CI) in normal breathing is 0.001236+/-0.03830. And, 95%CI in breath holding is -0.005383+/-0.02561. This plot suggests that there is sufficient correspondence between both peak intervals, and is not severe systematic error. The value of difference in breath holding is smaller than in normal breathing. Therefore, we think that respiratory body movement influences the calculation of the depth change waveform. The reduction of influence by respiratory movement is one of future subjects.

Fig. 13 shows the spatial distribution of inter-frame depth change plotted on the 3D shape of chest region. The time-series variation corresponds to single cardiac beat. The shape change by cardiac beat is mainly found on the left side of the chest region. We expect that the visualization of minute shape change occurred by cardiac beat is realized by imaging the spatial distribution of inter-frame depth change with higher time resolution.

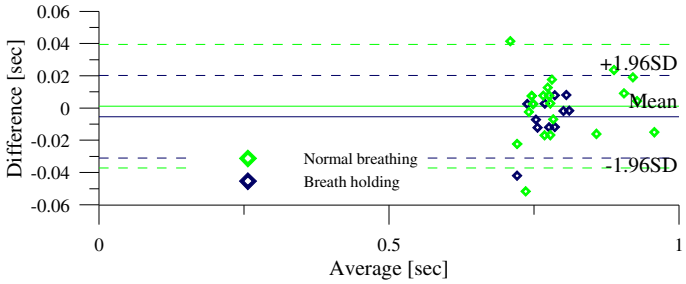


Fig. 12. Bland-Altman plot between R-R interval and peak interval of depth-change waveform

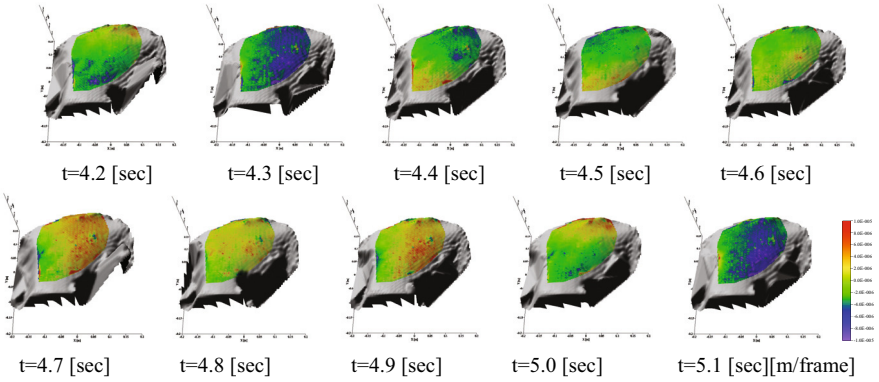


Fig. 13. Spatial distribution of interframe change of depth plotted on 3D shape of chest region

5 Conclusion

We propose the extraction of cardiac beat from 3D shape information of body surface by using grid-based active stereo, and basically examine the validity of proposed measurement. By simultaneous measurement with our proposed measurement and ECG, there are sufficient correspondence between peak interval of inter-frame depth change measured by our method and R-R interval measured by ECG. This result suggests that non-contact measurement of cardiac beat is realized by the active stereo. We tried the visualization of the spatial distribution of inter-frame depth change plotted on the 3D shape of chest region. And, the shape change by cardiac beat is mainly found on the left side of the chest region.

This work was supported in part by Strategic Information and Communications R&D Promotion Programme (SCOPE) No.101710002, KAKENHI No.21200002/No. 23700576, and Funding Program for Next Generation World-Leading Researchers No. LR030 in Japan.

References

1. Garbey, M., Nanfei, S., Merla, A., Pavlidis, I.: Contact-Free Measurement of Cardiac Pulse Based on the Analysis of Thermal Imagery. IEEE Transactions on BME 54(8), 1418–1426 (2007)
2. Nagae, D., Mase, A.: Measurement of vital signal by microwave reflectometry and application to stress evaluation. In: APMC 2009, Asia Pacific, pp. 477–480 (2009)
3. Poh, M., McDuff, D.J., Picard, R.W.: Advancements in Noncontact, Multiparameter Physiological Measurements Using a Webcam. IEEE Transactions on BME 58(1), 7–11 (2011)
4. Kawasaki, H., Furukawa, R., Sagawa, R., Yagi, Y.: Dynamic scene shape reconstruction using a single structured light pattern. In: CVPR, pp. 1–8 (2008)
5. Sagawa, R., Ota, Y., Yagi, Y., Furukawa, R., Asada, N., Kawasaki, H.: Dense 3D reconstruction method using a single pattern for fast moving object. In: ICCV (2009)
6. Sagawa, R., Kawasaki, H., Furukawa, R., Kiyota, S.: Dense One-shot 3D Reconstruction by Detecting Continuous Regions with Parallel Line Projection. In: ICCV (2011)
7. Ulusoy, A.O., Calakli, F., Taubin, G.: One-Shot Scanning using De Bruijn Spaced Grids. In: The 7th IEEE Conf. 3DIM (2009)
8. Sato, K., Inokuchi, S.: Range-Imaging System Utilizing Nematic Liquid Crystal Mask. In: Proc. Int. Conf. on Computer Vision, pp. 657–661 (1987)
9. Zhao, H., Chen, W., Tan, Y.: Phase-unwrapping algorithm for the measurement of three-dimensional object shapes. Applied Optics 33(20), 4497–4500 (1994)
10. Zhang, L., Curless, B., Seitz, S.: Rapid Shape Acquisition Using Color Structured Light and Multi-Pass Dynamic Programming. In: 3DPVT, pp. 24–36 (2002)
11. Lee, D.T., Schachter, B.J.: Two Algorithms for Constructing a Delaunay Triangulation. International Journal of Computer and Information Sciences 9(3), 219–242 (1980)
12. Microsoft. Xbox 360 Kinect (2010), `http://www.xbox.com/en-US/kinect`
13. Salvi, J., Pages, J., Batlle, J.: Pattern codification strategies in structured light systems. Pattern Recognition 37(4), 827–849 (2004)
14. Salvi, J., Batlle, J., Mouaddib, E.M.: A robust-coded pattern projection for dynamic 3D scene measurement. Pattern Recognition 19(11), 1055–1065 (1998)
15. Je, C., Lee, S.W., Park, R.-H.: High-contrast color-stripe pattern for rapid structured-light range imaging. In: Pajdla, T., Matas, J. (eds.) ECCV 2004, Part I. LNCS, vol. 3021, pp. 95–107. Springer, Heidelberg (2004)
16. Sagawa, R., Sakashita, K., Kasuya, N., Furukawa, R., Yagi, Y.: Grid based Active Stereo with Single-colored Wave Pattern for Dense One-shot 3D Scan. In: 3DIM/3DPVT 2012, pp. 363–370 (2012)

An Accurate and Efficient Pile Driver Positioning System Using Laser Range Finder

Xiangqi Huang[1], Takeshi Sasaki[2], Hideki Hashimoto[3], Fumihiro Inoue[4], Bo Zheng[1], Takeshi Masuda[5], and Katsushi Ikeuchi[1]

[1] Institute of Industrial Science, the University of Tokyo
[2] Shibaura Institute of Technology
[3] Chuo University
[4] Technical Research Institute, Obayashi Corporation
[5] The National Institute of Advanced Industrial Science and Technology

Abstract. Real-time positioning a pile for accurate pile driving is desirable for modern construction foundation work, but it suffers from the deficiency of the traditional systems because surveying instruments are manually used to mark the pile positions in which the accuracy heavily depends on the worker's experience. The paper confronts this problem by proposing a highly efficient positioning system using a Laser Range Finder (LRF). Over the traditional systems ours is superior to automatically detect the position of the pile or pile driver in real time with high accuracy. To this end, we first develop LRF based surveying system to scan the construction site in real time and gather the 2D laser point data. Then we detect target object such as pile or pile driver by fast fitting a circle-like geometric model to the data based on Maximum Likelihood Estimation (MLE) inference. The performance of the algorithm is validated by both synthesized and real data set. The results demonstrate the potentials on feasibility of our method in future construction field.

1 Introduction

1.1 Motivation

Nowadays, various pile drivers are widely used in the construction sites. The piles are one of most crucial parts that provide foundation support for a building once they are driven into soil at correct positions. Thus an accurate and efficient method for pile driving is always desirable for a modern construction yard.

The most common method of pile driving is based on the beforehand positioning using survey instruments. However, it is deficient due to a manual procedure: 1) placing a marker on the designed pile position using survey instruments, 2) digging a hole with certain radius tolerance at the marked position, and 3) driving the pile into the hole. The main disadvantages of this procedure are: i) at least 3 workers and 1 operator are needed for measuring and adjusting the pile position; ii) they are required to be well trained for collaboration to increase the work efficiency; iii) long operation time is commonly needed which may degrade the accuracy and safety for workers.

In this paper, we propose a highly efficient and accurate pile driver positioning system. Instead of measuring the preset markers using traditional instruments, we propose

X. Jiang et al. (Eds.): WDIA 2012, LNCS 7854, pp. 158–167, 2013.

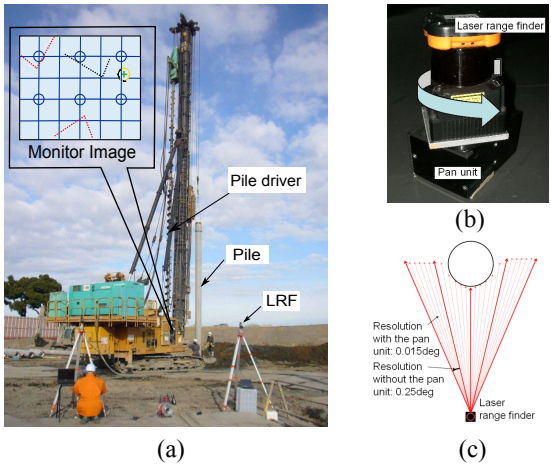


Fig. 1. (a) System sketch: LRS scans the construction workspace and transfers data to processor. Then the estimated current pile position which is relative to a construction map is displayed on a monitor to help the operator for adjustment. The red points are static background. The dark points are measured data points of the pile driver and yellow circle stands for the estimated pile. Green cross is the estimated center position of pile. The blue circles stand for the designed pile positions. (b) The LRF rotates under the drive of the pan unit. (c) The bold red lines stand for an original scan of the LRF. The thin red lines stand for the increased measurements.

to utilize highly accurate Laser Range Finder (LRF) to scan the whole construction field in real time, and simultaneously detect the moving pile position from the range scans. Fig. 1 illustrates our system in which only one operator is required to operate the pile driver to adjust pile position, navigated by a display which shows the moving pile position on a construction field map. This system provides a more convenient way that helps the operator to evaluate the errors on constriction map globally without any help of additional workers. Therefore our system dramatically increases the efficiency and accuracy for pile driving.

1.2 Related Work

On one hand, in a construction site, the survey technology and its instrument have progressed rapidly in a very short time period. With advanced measurement instruments including an automated transit and a total station which use light wave distance method, the measurement time is significantly shortened and positioning accuracy can achieve to millimeter level [1]. Moreover, in civil engineering work where the work area is very wide and there are few buildings around the area, the Global Positioning System (GPS) is frequently employed for sufficient survey accuracy and efficient work [2]. However they suffer from the disadvantages: i) these instruments are expensive, ii) these instruments might be unavailable in some places due to environmental conditions, iii) multiple objects cannot be tracked simultaneously, vi) it is impossible to track a target in real-time, and v) at least two workers are needed for the measurement.

On the other hand, the recent development of consumer-grade sensors has attracted increasing attention due to their applicable potentials for positioning system. Many real-time positioning systems using distributed devices including cameras [3–5], ultrasound sensors [6, 7] and Laser Range Finders (LRF) [8, 9] have been proposed. Among these measuring devices, LRFs are superior in the areas such as real-time scan, high accuracy, large covering area, robustness against poor illuminations, low noise-to-signal ratio and simple installation [10–12]. For instance, LRFs have already been used for real-time position measurement in large outdoor areas [13].

Laser scanners have been widely used in engineering surveys since the mid-1990s, such as terrestrial survey in high way constructions [14]. Recently in construction sites, some researchers are making effort to use LRFs in real-time to improve the working safety and efficiency. [15, 16] build laser scanner based tracking systems for rope shovels. Those systems work in short distance and give locally relative positions between the machine and objects. [17] uses a LRF for surveying task.

1.3 Overview and Contribution

Our system consists of three main processes: 1) data acquisition from LRF, 2) position estimation process and 3) visualization of results. It first gathers 2D LRF data from a construction site in real time; then simultaneously the pile position is estimated based on the obtained data; finally the estimated pile position is visualized in construction map together with the designed one to assist workers making decision for pile operation.

After data acquisition, our challenge is the difficulty of real-time position estimation for the moving pile from the range scans. To overcome this, the position estimation process includes two main sub-steps: 1) Target Detection based on circle model clustering; 2) Center position refinement using Maximum Likelihood Estimation (MLE).

Over the traditional positioning method, the main contributions of our system are: 1) instead of surveying in advance using the traditional instruments, we develop real-time surveying system using the highly accurate LRF; 2) we propose a fast detection algorithm for real-time and accurate pile positioning from range data; 3) our system simultaneously tracking and navigating of the pile driver provides the more efficient, safer, cheaper and easier way to the task of pile driving.

This paper is organized as: we present the system configuration in Section 2 and the detailed algorithms of position estimation in Section 3. Section 4 shows experimental results on both simulation and real data, followed by the conclusion in Section 5.

2 Configuration of LRF Sensing System

We first design a sensing system using LRF for positioning the pile in high accuracy. To this end, we employ UMT-30LX LRF [18] with high accuracy in depth direction (see Table 1) which horizontally scans a specific area in construction yard. However our challenge is that accuracy is difficult to achieve due to sparsity of the range scans. As shown in Fig. 1 (c), the original scan lines (shown in bold red arrow from left to right) are too sparse that only one point can return back when it reached a cylinder object. Such a sparse point cloud data is difficult to be used for accurate position estimation.

Table 1. Specification of UTM-30LX

Model Number	UTM-30LX
Measurable area	$0.1[m] \sim 30[m], 270[deg]$
Measurement accuracy	$0.1[m] \sim 10[m]$: $\pm 30[mm]$ $10[m] \sim 30[m]$: $\pm 50[mm]$
Angular resolution	$0.25[deg]$
Scan time	$25[msec]/scan$

To overcome this problem, we propose to mount UMT-30LX LRF to a pan unit for densely scanning [17]. As shown in Fig. 1 (b), we mount the LRF on a SPU-01 pan unit [19] which can drive the LRF to rotate in a very small angle (e.g. 0,015 degree in this case) after each scan of LRF. Therefore, dense points can be obtained by combining the scans captured at different angles of rotation of the unit pan. As shown in Fig. 1 (c), the resulted data points can be viewed as sampled in each 0.015 degree, where the density is much improved (at most 17 fold) compared to the original scan in each 0.25 degree. The system works in a data acquisition frequency between $2Hz - 40Hz$. The working speed of pile driver is between $0 - 2 \times 10^3 mm/s$. The speed in the final stage of the piling procedure, which especially needs high positioning precision, is around $0 - 50mm/s$. For this slow movement, the system data acquisition frequency is satisfied for the real-time requirement of the application of assistance of pile driver.

3 Pile Position Detection

Given the dense point data captured by above sensing system, our next task is how to track the pile position accurately and efficiently. This position estimation method consists of two major steps: 1) pile position detection, 2) center refinement. The former fast and coarsely detects circle-like models in range scans by a voting algorithm. The later accurately estimates the circle center by a refinement using MLE algorithm. We describe these two major steps in the following subsections respectively.

3.1 Pile Detection

Since the pile is always perpendicular to ground and scanned by LRF horizontally, the intersection by scan plane is always a circle. We design a circle model clustering algorithm for fast but coarsely detecting the pile position (see Algorithm 1).

Since the reference target is pile, from the viewpoint of LRF, the contour of the pile is convex, which means the center of object should not be observed. Under this observation, the constraint for rejecting the non-convex samples in Algorithm 1 can be described as:

$$(k_i \cdot x_{LRF} + t_i - y_{LRF})(k_i \cdot x_{new} + t_i - y_{new}) < 0, \tag{1}$$

where $l_i{:}y = k_i \cdot x + t_i$, $i = 1, 2, 3$ are lines determined by two points from the current subset of data. (x_{LRF}, y_{LRF}) is the position of LRF and $P_{new} = (x_{new}, y_{new})$ is the currently

Algorithm 1. Pile detection

1) **Input**: range data S, radius r of pile.
2) **Output**: a set of points $\mathcal{A} = \{x_i, y_i\}$ on pile boundary.
3) **Initialization**: null clusters $\{C_j\}$ each of them stores the circles in one cluster; null array V whose element v_j stores number of votes for cluster C_j.
4) **Loop**
5) Randomly sample a triangle by selecting three non-collinear points from S;
6) Calculate circumscribed circle A_i of the triangle (with radius R_i);
7) Reject a too big or small circle by radius thresholding: **if** $|R_i - r| > T_r$, **Go to** 4);
8) Reject a non-convex sample by Eq. (1);
9) **if** the center of A_i is close to an averaged center of cluster C_j, then $v_j \leftarrow v_j + 1$; otherwise create new cluster for A_i;
10) **if** $v_j < T_v$, then **Go to** 4);
11) Calculate inlier points $\{x_k, y_k\}$ which are near to the boundary of averaged circle of C_j
12) **if** enough inliers, then stop;
13) **End**

calculated center position. Eq. (1) means that the LRF and center of target should be on a different side of the line determined by points from the data subset. In Algorithm 1, the thresholding parameters are set according to the specification of LRF.

3.2 Refinement for Accurate Position Estimation

Since Algorithm 1 only provides a cluster of proposals of pile boundary points, we need to accurately estimate the pile position according to these proposals. Therefore the next task can be viewed as: given the proposals of boundary points $\mathcal{A} = \{x_i, y_i\}_1^K$ and a circle model $(x-a)^2 + (y-b)^2 = r^2$, how we can accurately estimate parameters a and b. To this end, a common but effective selection is to use Maximum likelihood estimation (MLE) algorithm that estimates the parameters which can maximize the likelihood for each proposal. The likelihood can be expressed as

$$p(\mathcal{A}) = \frac{\exp\Big[-\sum_{i=1}^{K}\left[(x_i - \bar{x}_i)^2 + (y_i - \bar{y}_i)^2\right]/2\sigma^2\Big]}{\sqrt{(2\pi\sigma^2)^{2K}}}, \tag{2}$$

where $(\bar{x}_i, \bar{y}_i)$ is the true position of (x_i, y_i), and the sensor noise is assumed as Gaussian noise with variance σ^2. To maximize this likelihood $p(\mathcal{A})$, we minimize $-\log[p(\mathcal{A})]$. We first take a logarithm of both sides of Eq.2 and then remove a constant term which does not contribute to minimization.

Here we fulfill circle model as a constraint of x_i and y_i. We remove the restraint condition by using Lagrange's method of undetermined multipliers. Finally the MLE for a circle is equal to estimate parameters a, b, and r which will minimize J_{ML} expressed as:

$$J_{ML} = \sum_{i=1}^{K} \frac{\left(x_i^2 + y_i^2 - 2ax_i - 2by_i + a^2 + b^2 - r^2\right)^2}{x_i^2 + y_i^2 - 2ax_i - 2by_i + a^2 + b^2}. \tag{3}$$

In our system, parameters which should be estimated are only a and b, since the radius of the reference target is given in advance. In this case, however, the MLE becomes a non-linear problem. To solve this non-linear equation, we apply the Newton-Raphson method since it is known to have a faster convergence rate than other Gradient methods, such as Conjugate gradient or Levenberg-Marquardt, if its initial value is close to the true value of a and b [17]. Here we use the detected center position from the previous detection result as the initial value of the Newton-Raphson method.

It is worth noticing that for further speeding up two simple pre-processes are adopted in our method: 1) Assuming the background scene is static, moving parts of data are extracted as foreground objects using a background subtraction algorithm, and then 2) all the foreground data points are clustered using a neighborhood verification method.

4 Experiment Results

To verify our proposed circle detection and fitting algorithm, we first test it by simulation experiments. Before the experiment at actual construction site, we conduct an indoor experiment on a small scale model of pile driver to test the proposed positioning system. At last, an outdoor experiment at the actual construction site is performed.

4.1 Simulation Experiment

In this simulation, we simulate the scene and devices of previous experiment at an actual construction site [20] by combining circles and lines. We assume that with the cylindrical reference target there are other objects with cross-sections shaped like rectangle and trapezoid in the scenario whose contour is similar to arc especially with large sensor noise.

The noise of LRF data is assumed to be independent Gaussian noise with the standard deviation $\sigma = 50mm$ in the range of distance $d_{max} = 30 \times 10^3 mm$. The angle resolution of LRF is $\theta_{reso} = 0.05°$. The radius of cylindrical reference target is $R = 1350mm$. The distance from LRF to the center of cylinder is $d = 9000mm$. Values of thresholds used in Algorithm 1 (line 7, 9, 11) are empirically determined as σ.

All results are average values of 1000 times simulations. To make the system work in real time and always process the latest data, the algorithm will abort and be considered as a failed detection if it cannot establish an acceptable model until maximum iteration time.

Evaluation of the Circle Detection Algorithm. To evaluate the algorithm, the detection rate, false detection rate, iteration times of random sampling and error of estimated center are used here. Fig.2 (a) shows that when votes are 2 and 3 the detection rate is almost 100%. But the value decreases when more votes are required. The main reason is the maximum iteration time limits the performance. Fig.2 (b) shows that the false positive rate is almost zero if T_v is set larger than one.

Results with $T_n = 0.1N$ and $T_n = 0.6N$ are compared in figures, where N is the roughly estimated number of inliers as shown in:$N = \frac{2}{\theta_{reso}} \cdot (\arccos \frac{R}{D})$, where D is the distance from LRF to the center of the data cluster. Parameter T_n is the number of data

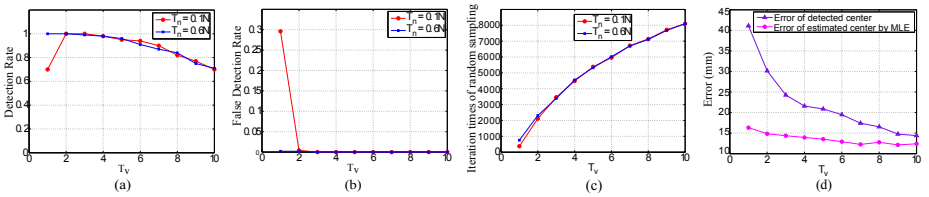


Fig. 2. (a) Detection rate. (b) False detection rate. (c)Iteration times of random sampling. (d)Error of estimated position and detected position ($T_n = 0.1N$).

points required to claim that the hypothetical model can be accepted. It means that the voting procedure makes the proposed algorithm still work well even when occlusion or significant noise degrades the data.

The average iteration times of random sampling increase while the number of votes becomes larger, as shown in Fig.2 (c). It also shows that choosing a small T_n would not increase the computation cost.

Position Estimation Precision. We evaluate estimation errors of the aforementioned two methods: the non-linear MLE and the proposed detection algorithm with known radius. Estimation error E is defined as $E = \sqrt{(X_L - a)^2 + (Y_L - b)^2}$, where (a, b) is the estimated center position of the cylindrical reference bar and (X_L, Y_L) is a position where we put the cylindrical reference bar.

Fig.2 (d) gives us the error of estimated center position using MLE and the error of detected center position. It shows that the number of votes T_v doesn't have a significant effect on the estimation result while the number of iterations increases dramatically with more votes. So a small T_v would be a good trade off between estimation accuracy and computation cost.

Considering the detection rate, iteration time of random sampling, error of estimated center position and the flexibility of algorithm, $T_v = 2$ is the best choice of votes under the given conditions.

4.2 Indoor Experiment

To verify the feasibility and estimation error of our proposed system, before implementing it in the actual construction field, we made the indoor experiment. As shown in Fig.3 (a), a cylinder with radius of 250*mm* is used as the reference bar. Two cardboard boxes are used to simulate the pile driver in the application scene. Fig.3 (b) shows an example of the scanned data from LRF of the experiment scene.

We firstly put the reference bar at a known position and estimate the center position only using MLE method. While keeping the cylinder at the same place, we put the other shaped objects near to it and estimate the position of cylinder again with the proposed positioning algorithm, to see if we can achieve the same estimation accuracy or not. Experiment with this set up is repeated at every 500*mm* from the distance 1000*mm* to 7500*mm*.

As shown in Fig.3 (c), the two curves of estimation error are almost the same, with the maximum difference of 3.4*mm*. The experiment result proved that our proposed

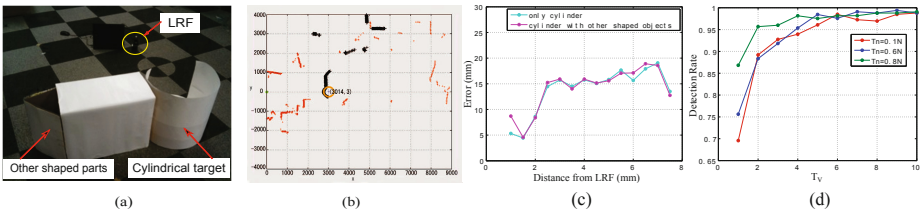


Fig. 3. (a) Indoor experiment scene. (b) An example of measuring the center position of the cylindrical reference bar. (c) Estimation error of MLE and the proposed positioning algorithm of indoor experiment. (d) Detection Rate of indoor experiment.

circle detection algorithm can effectively extract data points of circle from the scene with other shaped objects.

We vary the parameter T_v from 1 to 10 to see the effect on detection rate. When votes are $T_v \geq 3$, as Fig.3 (d) shows, the detection rate exceeds 90% even with small value of T_n. If there is no voting procedure, meaning $T_v = 1$, the detection rate is barely acceptable. This low detection rate is caused by the junction of triangle and rectangle boxes, whose contour is similar to an arc. This result proves the feasibility of the proposed pile detection algorithm and shows the possibility that it can still work well with occlusion or large noise present.

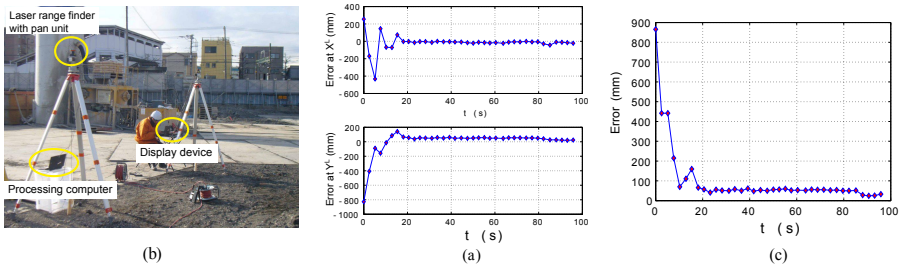
4.3 Experiment in the Construction Field

We tested the proposed system in an actual construction field. Our purpose is to measure the position of the pile which is being put into the drill hole. Tracking the position of pile can help to position it at the expected place, which requires an accuracy of $100mm$. As shown in Fig.1 (a), the pile was lifted by the pile driver and being slowly put down into the hole. Currently in most construction fields, the pile is kept to the expected position by manual work. It needs three well trained workers using sticks to measure whether the pile is in right place or not.

As shown in Fig. 4 (a), LRF and pan unit are placed on the tripod which can be adjusted to keep the scan plane of LRF horizontal. The height of tripod also helps to keep other moving objects, like humans, from the scan range. The object to be measured is the pile shown in Fig.1 (a), with a radius of $200mm$. Limited by the arrangement of construction field, the distance between the measured object and LRF is about $15 \times 10^3 mm$.

To calibrate the LRF, a thin metal stick (cross section: $30mm \times 1mm$) which has a highly reflectable surface is used. We first use total station to position several points with a precision around 2mm/km. We then put the calibration stick at those points and obtain the scan data of stick from LRF. We use an average value of scan data of stick to estimate its position. Transformation matrix between world coordinate and LRF coordinate can be calculated using least square estimation method.

Result of the Construction Field Experiment. The procedure of pile driving was recorded by the LRF. It started when the pile was moved near to the expected position



(b) (a) (c)

Fig. 4. (a) Scene of position measurement of pile driver. (b) Errors at X^L and Y^L(LRF coordinate system) of construction field experiment. (c) Error to the expected pile position.

about $1m$. After arriving the top of the hole, the pile was kept going down at a position with error around $50mm$. The placement of pile was adjusted to be more accurate at the final 10sec. The position adjustment by workers can be seen in Fig.4 (b). Here error at X^L is defined as $(X_L - a)$ and error at Y^L as $(Y_L - b)$. The errors between the measured center position and the expected design position of pile are shown in Fig.4 (c). The final position error given by proposed system is around $25mm$.

Currently there is no other direct way to measure the center position of pile while it moves. The accuracy of the traditional surveying system is $10mm$. In addition, from this experiment, it is certain that the pile was driven within the allowable range. We can estimate that the construction error is within the range of $15 - 35mm$. Considering the manual adjustment process, the result of our system is considered reasonable. The proposed system can measure the pile driving position directly which cannot be measured by conventional surveying instruments.

5 Conclusion and Future Work

In this paper, based on pile detection and fitting, we propose an novel real-time pile driver positioning system using laser range finder. Taking advantages of LRF's strengths, such as high accuracy, fast data acquisition and large covering area, and utilizing the orientation-invariant property of the cylindrical target, a new surveying technique is presented. To extract the pile target from pile driver, we propose a pile detection algorithm based on circle model clustering. Then the MLE method is adopted to accurately estimate the pile position.

The simulation and indoor experiment prove the reliability and flexibility of the proposed detection algorithm. The experiment on the actual construction field shows that the proposed system can keep tracking the pile position in real-time while the pile driver works, which is impossible for the conventional surveying methods.

For the purpose of verifying the feasibility of proposed system, we only use data from a single LRF currently. Since multiple sensors could increase the amount of information and enlarge the positioning range, multiple sensor system will be investigated and implemented. In the current implementation, the number of needed votes is empirically determined. The robustness of this parameter should be investigated.

References

1. Kavanagh, B.: Surveying: principles and applications. Prentice Hall (2003)
2. Odijk, D., Teunissen, P., Zhang, B.: Single-frequency integer ambiguity resolution enabled gps precise point positioning. Journal of Surveying Engineering 138(4), 193–202 (2012)
3. Gu, Y., Lo, A., Niemegeers, I.: A survey of indoor positioning systems for wireless personal networks. IEEE Communications Surveys Tutorials 11(1), 13–32 (2009)
4. Shivappa, S., Trivedi, M., Rao, B.: Hierarchical audio-visual cue integration framework for activity analysis in intelligent meeting rooms. In: CVPRW, pp. 107–114 (June 2009)
5. Morioka, K., Hashimoto, H.: Appearance based object identification for distributed vision sensors in intelligent space. In: IROS, vol. 1, pp. 199–204 (2004)
6. Loke, Y., Gopalai, A., Khoo, B., Senanayake, S.: Smart system for archery using ultrasound sensors. In: IEEE/ASME International Conference on Advanced Intelligent Mechatronics, pp. 1160–1164 (July 2009)
7. Das, S., Gleason, C., Shen, S., Goddard, S., Perez, L.: 2d tracking performance evaluation using the cricket location-support system. In: 2005 IEEE International Conference on Electro Information Technology, 6 p. (May 2005)
8. Abeles, P.: Robust local localization for indoor environments with uneven floors and inaccurate maps. In: IROS, pp. 475–481 (September 2011)
9. Even, J., Heracleous, P., Ishi, C., Hagita, N.: Multi-modal front-end for speaker activity detection in small meetings. In: IROS, pp. 536–541 (September 2011)
10. Huang, X., Sasaki, T., Hashimoto, H., Inoue, F.: Circle detection and fitting based positioning system using laser range finder. In: 2010 IEEE/SICE International Symposium on System Integration (SII), pp. 442–447 (December 2010)
11. Huang, X., Sasaki, T., Hashimoto, H., Inoue, F.: Circle detection and fitting using laser range finder for positioning system. In: 2010 International Conference on Control Automation and Systems (ICCAS), pp. 1366–1370 (October 2010)
12. Sasaki, T., Huang, X., Hashimoto, H., Inoue, F.: Position measurement of piles using a laser range finder for accurate and efficient pile driving. In: IEEE/ASME International Conference on Advanced Intelligent Mechatronics, pp. 241–246 (July 2011)
13. Zhao, H., Shibasaki, R.: A novel system for tracking pedestrians using multiple single-row laser-range scanners. IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans 35(2), 283–291 (2005)
14. Johnson, W., Johnson, A.: Operational considerations for terrestrial laser scanner use in highway construction applications. Journal of Surveying Engineering 138(4), 214–222 (2012)
15. Kashani, A.H., Owen, W.S., Himmelman, N., Lawrence, P.D., Hall, R.A.: Laser scanner-based end-effector tracking and joint variable extraction for heavy machinery. The International Journal of Robotics Research 29(10), 1338–1352 (2010)
16. Dunbabin, M., Corke, P.: Autonomous excavation using a rope shovel. Journal of Field Robotics 23(6-7), 379–394 (2006)
17. Tamura, H., Sasaki, T., Hashimoto, H., Inoue, F.: Circle fitting based position measurement system using laser range finder in construction fields. In: IROS, pp. 209–214 (October 2010)
18. Hokuyo, `http://www.hokuyo-aut.jp/02sensor/07scanner/utm_30lx.html`
19. Pan unit, `http://www.sustainable-robotics.com/products_panunit.html`
20. Hajime, T.: Position measurement system using laser range finder in construction fields – application for pile position measurements. Master Thesis, The University of Tokyo, pp. 115–116 (2010)

Kitchen Scene Context Based Gesture Recognition: A Contest in ICPR2012

Atsushi Shimada[1], Kazuaki Kondo[2], Daisuke Deguchi[3], Géraldine Morin[4], and Helman Stern[5]

[1] Department of Advanced Information Technology, Kyushu University, Japan
[2] Academic Center for Computing and Media studies, Kyoto University, Japan
[3] Strategy Office, Information and Communications Headquarters, Nagoya University, Japan
[4] IRIT, University of Toulouse, France
[5] Department of Industrial Engineering and Management, Ben-Gurion University of the Negev, Israel

Abstract. This paper introduces a new open dataset "Actions for Cooking Eggs (ACE) Dataset" and summarizes results of the contest on "Kitchen Scene Context based Gesture Recognition", in conjunction with ICPR2012. The dataset consists of naturally performed actions in a kitchen environment. Five kinds of cooking menus were actually performed by five different actors, and the cooking actions were recorded by a Kinect Sensor. Color image sequences and depth image sequences are both available. Besides, action label was given to each frame. To estimate the action label, action recognition method has to analyze not only actor's action, but also scene contexts such as ingredients and cooking utensils. We compare the submitted algorithms and the results in this paper.

1 Introduction

An action has a strong relationship not only with an actor's motion, but also with the situation surrounding the actor. Traditionally, most action recognition methods focus on motion features only, and assign the action label based on the discriminative analysis[1,2]. There are, however, many actions which cannot always be uniquely determined by using motion features alone. For instance, considering actions in a kitchen, "mixing something" and "baking something" would be done by similar motion sequences of a hand, which moves around in a circular motion. If he/she uses a bowl, the action label should be "mixing something in a bowl". Likewise, if he/she takes a frying pan, the label would be regarded as "baking something in a frying pan". Thus, the action label should be estimated by consideration of both the motion features and the context situation in which the action is performed.

Scene context based action recognition can be applied for various uses in daily life: real-time analysis of a cooking scene or other types of scenes will

X. Jiang et al. (Eds.): WDIA 2012, LNCS 7854, pp. 168–185, 2013.

enable to advise a beginner what he/she should do at the next step including recovery of mistakes. Scene analysis and classification of recorded videos can also provide context-based segmentation of image sequences, and facilitate automated scene annotations for video databases. Moreover, the context-based approach is applicable to other domains, for example; hospital operating rooms in medical practices, agricultural and manufacturing operations, etc.

To encourage and evaluate such a context-based approach, we created the Actions for Cooking Eggs (ACE) Dataset[1] and decided to hold a contest "Kitchen Scene Context based Gesture Recognition", in conjunction with ICPR2012[2]. The dataset consists of cooking actions in a natural kitchen environment. It includes 25 video sequences from five kinds of cooking menus. Each menu was performed by five actors. The video sequence was recorded by a Kinect sensor. Both of color image and depth image were captured. More details of the dataset will be explained in section 3. We also provide labeling results on the sequences of the dataset.

The remainder of this paper is organized as follows. Section 2 reviews open datasets related to action recognition. The detailed explanation of our dataset will be explained in section 3. Section 4 explains how the dataset was acquired. The contest tasks and evaluation criteria will be described in section 5 followed by contest results in section 6. Section 7 introduces two submitted algorithms in more detail. Finally, we conclude the report of contest with an outlook.

2 Related Action Datasets

There are several public datasets released for evaluating action recognition methods. The KTH dataset[3] and the Weizmann dataset[4] are often used for evaluation. These datasets consist of simple actions like "walking", "jogging", "running", "waving with the arms" or "jumping", captured by a single camera. The IXMAS dataset is similar to these datasets, but the scene was captured by multiple cameras. The characteristics of these three datasets are of limited relevance to practical applications.

The UT dataset[5] was used for the human activity recognition contest and it has been tested by several state-of-the-art methods. The UT dataset contains six interaction activities: "hand shake", "hug", "kick", "point", "push" and "punch", which are more complex than simple actions mentioned above. However, available motions are not natural activities in daily life.

From the viewpoint of natural actions in daily life, the CMU Multi-Modal Activity Database[3] contains multi-modal measures of the human activity of subjects performing the tasks involved in cooking and food preparation. Five kinds of modalities were recorded: "video", "audio", "motion capture", "internal measurement units" and "wearable devices". The TUM kitchen Data Set[6] also contains activity sequences recorded in a kitchen environment. The recorded

[1] http://www.murase.m.is.nagoya-u.ac.jp/KSCGR/
[2] http://www.icpr2012.org/
[3] http://kitchen.cs.cmu.edu/

data consists of observation of naturally performed manipulation tasks. These kitchen datasets observed whole of the kitchen: the kitchen sink, the kitchen table, stove, etc., by using multiple sensors including video sequences, full-body motion capture data, or other reading sensors.

In contrast to the CMU Multi-Modal Activity Database and the TUM kitchen Data Set, our dataset focused on cooking activities just around the kitchen sink and the counter. Cooking actions of five different actors were captured by a single Kinect sensor mounted above the kitchen(see Fig. 2), which is more simple and realizable kitchen environment than above two kitchen datasets.

3 Actions for Cooking Eggs (ACE) Dataset

3.1 Concept

Our dataset would like to provide actions which cannot be determined by motion analysis alone. Additional information such as "what kind of cooking utensil the actor is using" and "what ingredient is being cooked" should be considered to perform final decision of action recognition. Let's consider the following examples. If the actor repeatedly moves the hand from side to side, there are some candidates of action label such as "cutting", "baking", "boiling" or so on. If the actor is using the knife on the cutting board, the action label should be "cutting". The "boiling" label comes from the combination of saucepan and eggs. Therefore, it is important to focus not only on the motions but also on context information of ingredients and cooking utensils. Actually, there are five menus of cooking eggs in our ACE dataset. In the following subsections, brief recipe of each menu, the list of ingredients and cooking utensils used in the dataset are introduced. Then, eight kinds of cooking actions performed by actors will be defined.

3.2 Menus

Following five menus are selected. The first four menus are very popular served as a breakfast meal. The fifth menu is often used to decorate food in Japan. The basic recipe of each menu is briefly introduced as follows.

ham and eggs Brown some slices of ham, then break eggs on the ham. Season with salt if necessary.

omelet Break eggs into a bowl and mix together. Add salt, milk if necessary, and beat again. Pour the egg mixture into the pan and cook for a while until egg mixture is set. Shape the egg with a spatula.

scrambled egg Break eggs into a bowl and beat them until they turn a pale yellow color. Add the ham, salt and/or milk if necessary, and mix them again. Pour in egg mixture. As eggs begin to set, gently pull the eggs across the pan with a spatula until thickened and no visible liquid egg remains.

boiled egg Place the raw egg in a saucepan. Run cold water into the saucepan. Boil the water for several minutes. Peel the egg shells.

Kinshi-tamago shredded egg crepes, one of Japanese egg recipes. Break eggs into a bowl and mix together. Pour in egg mixture and make a crepe on the pan. With a sharp knife, cut the crepe into thin strips.

3.3 Ingredients

Ingredients used in the dataset are summarized as follows.

egg used by all menus. Some actors use an egg, the other use two eggs for cooking.
ham used by ham and eggs. Some actors use the ham in omelet and scrambled egg.
milk used by omelet and scrambled egg if an actor chooses it.
oil used by menus with a pan (except for boiled egg), if necessary.
salt used by menus with a pan (except for boiled egg), if an actor chooses it.

3.4 Cooking Utensils

Following cooking utensils are used in ACE dataset.

frying pan used by menus of ham and eggs, omelet, scrambled egg and Kinshi-tamago.
saucepan used by the menu of boiled egg.
bowl basically used by all menus, but some actors don't use it in the menu of ham and eggs. Besides, some actors put egg shells in the bowl in the menu of boiled egg.
cup used to pour water into a pan.
plate used by all menus when finishing the cooking.
chopsticks used to mix something in the bowl, the frying pan and the saucepan.
spatula used to turn something in the frying pan. Some actors use it to cut omelet.
knife used by all menus.
cutting board used by all menus.

3.5 Cooking Actions

The labels of cooking actions defined in the dataset are as follows. Strictly speaking, the labels can be divided more precisely. Besides, some labels are not necessarily relevant to the ones called in recipe books. However, we are sure that the following labels support essential actions in cooking above menus.

breaking used when an egg is cracked
mixing used when mixing something in the bowl or in the pan.
baking used when cooking something on the pan.
turning used when turning something in the pan.
cutting used when cutting something on the cutting board.
boiling used when boiling water.
seasoning used when seasoning with salt.
peeling used when peeling egg shells.

Fig. 1. Example shot of each action

4 Data Acquisition

Each menu was cooked by five different actors; that is, five cooking scenes for training are available for each menu. Besides, two cooking scenes were also captured as testing video for each menu. The scene was captured by a Kinect sensor providing synchronized color and depth image sequences. Each of the videos was from 5 to 10 minutes long containing 2,000 to 12,000 frames. A cooking motion label was assigned to each frame, indicating the type of action performed by the actors. In the following subsections, more details will be explained.

4.1 Kitchen Sensing Environment

A Kinect sensor was mounted above the Kitchen as shown in Fig. 2. The sensor recorded both color images and depth images at 30fps, and the image size was 320 × 240 pixels. The depth image consists of 8-bit gray scale information (256 levels). The depth unit of each level is 8mm. Therefore, the depth is available up to 2048mm (8mm x 256level). An example shot is also shown in Fig. 2. All ingredients and cooking utensils are visible within the field of view. All of them were arranged on the initial positions like as Fig. 3. Such information is helpful to initialize a tracking procedure of ingredients and cooking utensils.

The Kinect sensor also captured the hand area of an actor. Not only color images but also depth images are available to acquire hand positions and/or hand motions. For example, combination of skin color detection and restriction by distance from the sensor is one of the seasonable ways to find the hand area.

4.2 Cooking Scenarios

Five different actors followed the recipes roughly written by us, but the detailed procedure of cooking depended on each actor. Fig. 4 shows the actual recipes.

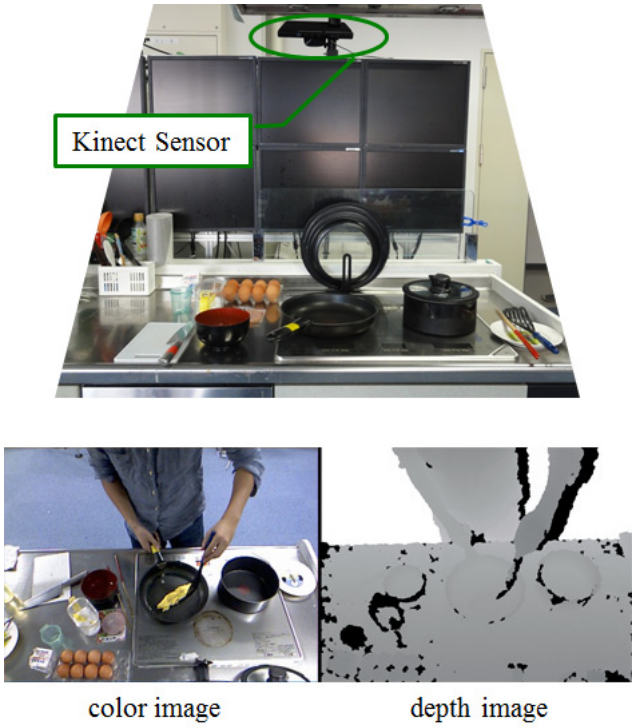


Fig. 2. Top: A Kinect sensor mounted above the kitchen. Bottom: An example frame.

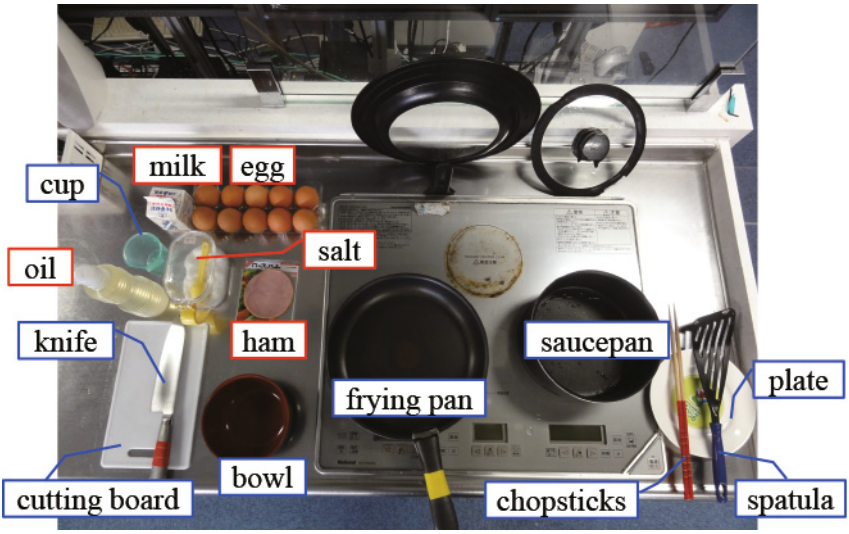


Fig. 3. Initial arrangement of ingredients and cooking utensils

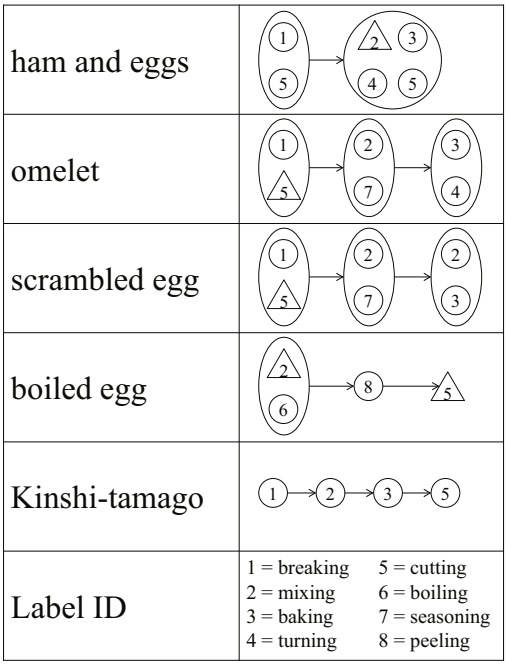


Fig. 4. Rough flow of each cooking menu

The number in circular shapes and triangular shapes corresponds to each action label written in the bottom right column. The circular shape means that the action was performed by all actors. Meanwhile, the triangular shape means that the action was performed by some actors (not all). The large ellipse surrounding two or four shapes represents unordered actions, i.e., the order of actions depends on each actor. For example, an actor performed "breaking an egg" followed by "cutting hams", and another actor was first to do "cutting hams".

4.3 Labeling

The labeling was achieved by the manner listed in Table 1. Actually, the label in each video frame was decided by a majority voting. Three people assigned a label to each video frame according to the rule. Then, the label with the most votes was regarded as the majority.

5 Description of Contest

5.1 Competition Tasks

The contestants are expected to evaluate the human gestures in a kitchen from continuous video sequences. The candidates of the cooking menus are "ham

Table 1. Labeling manner to video sequences

Label	Start	End
breaking	An actor starts to break an egg.	An actor puts the shell on somewhere.
mixing	An actor starts to mix something (in a bowl, in a frying pan, etc.).	An actor stops mixing.
baking	An actor starts to put something in a frying pan to bake it.	An actor removes the target from the pan.
turning	An actor starts to turn a target with a cooking utensil	An actor finishes turning the target.
cutting	An actor starts to cut something with a cooking utensil	An actor finishes cutting.
boiling	An actor puts an egg in a pan.	An actor removes the egg from the pan.
seasoning	An actor picks up seasonings.	An actor brings back it.
peeling	An actor picks up an egg to peel the shell.	An actor finishes peeling.

* The label "baking" is overwritten with other labels (mixing, turning, cutting, and seasoning)
* Milk is not included in the category of seasoning.

and eggs", "omelet", "scrambled egg", "boiled egg" and "Kinshi-tamago". An actor cooks one of cooking menus. Each video contains several cooking motions including eight actions, i.e. breaking, mixing, baking, turning, cutting, boiling, seasoning and peeling. Using the testing videos, contestants have to establish the relationships between motion features and scene features (i.e. scene contexts) and achieve scene context based cooking motion recognition by assigning a correct cooking motion label to each video frame.

Using the scene context is not an indispensable condition, but the organizers strongly recommended utilizing it. The following figure shows a brief example of cooking an "omelet". The task of the contest is to recognize the "cooking motion label" in the middle layer of cooking hierarchy. Note that the contestants have to estimate the action label located in the middle layer in Fig. 5 (Not the label of cooking menu in the top layer).

5.2 Evaluation Criteria

The cooking action labels assigned into the testing video is evaluated by the accuracy score calculated from precision and recall manner. The precision and recall are given by

$$Precision = \frac{TP}{TP + FP} \tag{1}$$

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

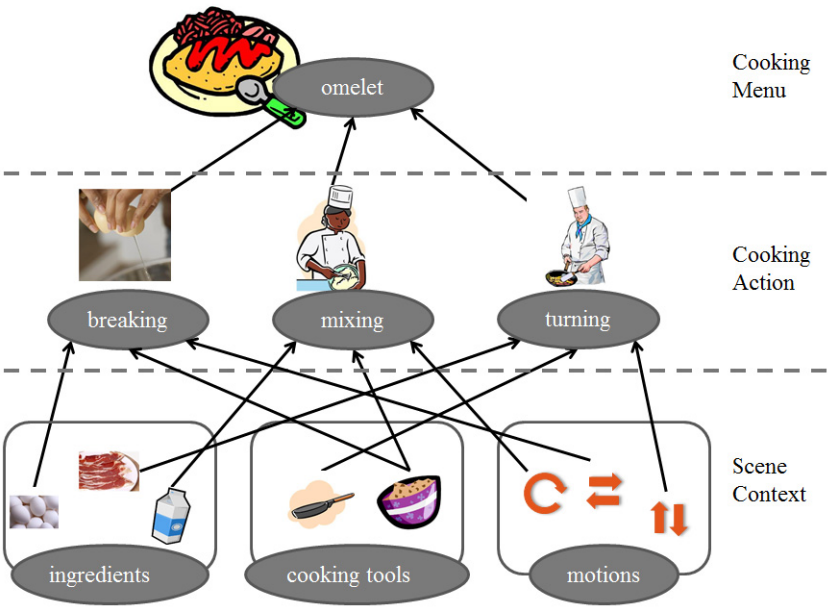


Fig. 5. Hierarchy relation among scene context, cooking action and cooking menu

where TP, FP and FN denote the number of labels assigned correctly, assigned wrongly, unassigned wrongly respectively. Then, their harmonic mean ($F - measure$) is calculated by following formula.

$$F = 2/\left(\frac{1}{Precision} + \frac{1}{Recall}\right) \tag{3}$$

Actually, the F-measure is calculated for each cooking motion label. Suppose the result is composed of N kinds of cooking motion labels, the final score is given by averaging all F-measures,

$$FinalScore = \frac{1}{N}\sum_{i=1}^{N} F_i \tag{4}$$

where F_i is a F-measure for each cooking motion label.

6 Summary of Results

6.1 Participation

The contest web-site attracted more than 3,000 visitors by the end of contest. The training dataset has been downloaded more than 340 times during the period

of 13 Dec. 2011 to contest date. The test dataset was open to public 15 Aug. 2012, and has been downloaded about 40 times. Finally, 6 teams submitted their results.

6.2 Submitted Algorithms

The submitted algorithms are summarized into two types: training based approaches and heuristic approaches. The training based approaches consist of preprocessing, feature extraction, classification and post-processing.

Preprocessing finds table and/or floor using depth information. Some contestants use the plane information in order to judge if a cooking utensil is on the plane or not.

Preprocessing finds table and/or floor using depth information. Some contestants use the plane information in order to judge a cooking utensil is on the plane or not.

Feature extraction is achieved by finding hand region, cooking utensils and ingredients. The position of object or state of object (i.e. in Use or not in Use) is used for feature representation. Some other contestants extract spatio-temporal features or local features from video sequence.

Classification is performed by SVM(Support Vector Machine) and/or HMM based approach. Subspace learning method is also used by one of contestants.

Post-processing is applied for smoothing the label estimation result. Majority, rule-based smoothing or Markov Random Field based smoothing is used.

On the other hand, heuristic approach makes some rules to classify the features into eight action labels. Position relationship among hand, cooking utensils and ingredients are carefully explored by the contestant in advance. The order of label assignment is also reflected to the rule.

6.3 Results

The total performance (i.e. the final score) is shown in Fig 6. There are six results named "Team-01" to "Team-06". The scores are widely distributed from 0.2 to 0.8. The approach of each team could be divided into three groups based on training based approach with object detection and tracking, training based approach with local features, and heuristic approach. The correspondence between the approach and the team ID is as follows.

Group-1 "Team-01" and "Team-02" belong to the training based approach with object detection and tracking.

Group-2 "Team-03" and "Team-04" belong to the training based approach with local features.

Group-3 "Team-05" and "Team-06" belong to the heuristic approach.

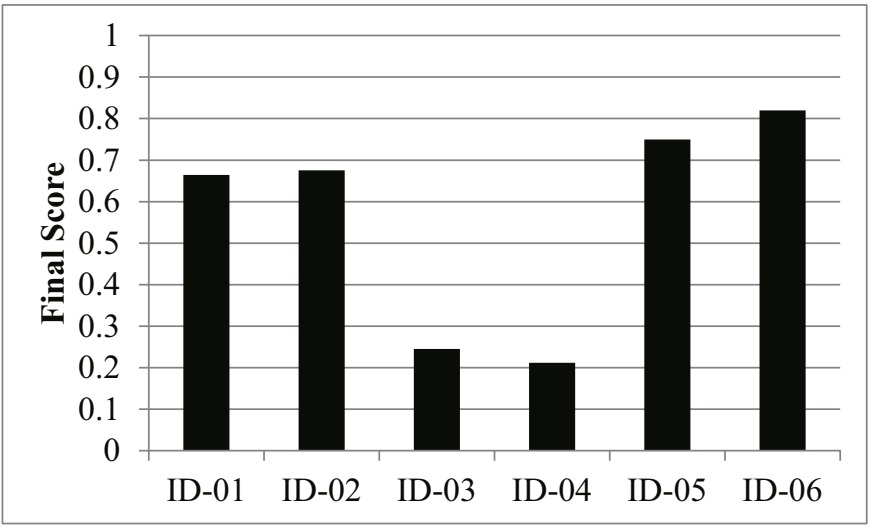


Fig. 6. Final Score of each team

Interestingly, heuristic approaches outperformed the training based ones. Generally speaking, heuristic approach is discussed that it would not ensure the generality so that it has been avoided by most academic studies. However, we have no option but to accept that the heuristics is effective from the viewpoint of the practical use.

With regard to the results of training based approaches, the final scores are definitely different between "Group-1" and "Group-2". Local features are directly extracted from spatio-temporal domain in the video sequence without any object detection processing. Meanwhile, object detection and tracking based approach can capture the spatio-temporal features of each object more precisely than the local feature based approach. We guess that's why the Group-1 got better results than Group-2.

Finally, F-measures of each action label are shown in Fig. 7. We can see the similar tendency with the final scores, i.e. heuristic approach provides better results than others. There is difficulty to recognize the action of "mixing" and "turning". The "mixing" action was performed in the bowl, in the frying pan and in the saucepan. Each cooking utensil was located at different position. One of the strategies to enhance the result is to use the location information of each utensil. However, it still has difficulty to recognize the action by using location information alone since the utensils are used for another purpose(e.g. the frying pan is used for the action of "baking" and "turning"). On the other hand, the "turning" action has another aspect of difficulty that the action is performed in short period of time during another action. Therefore, the action "turning" might be overridden by another action label by the post-processing.

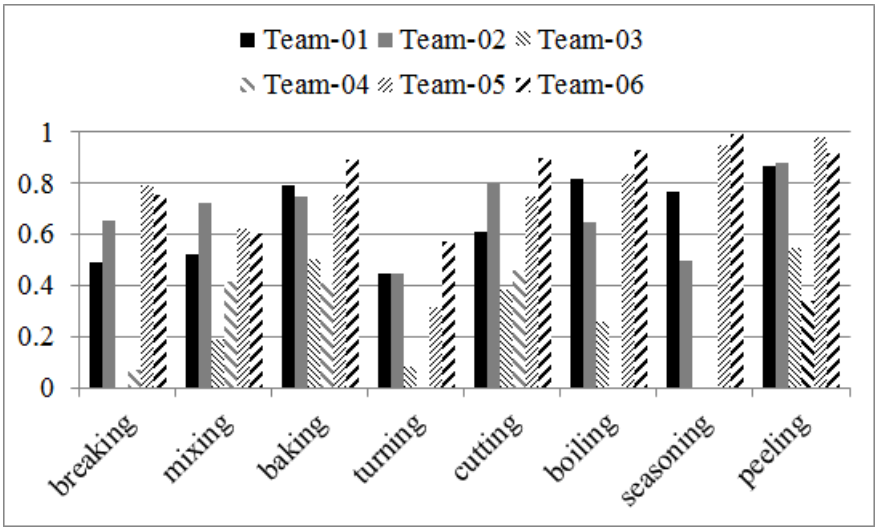


Fig. 7. Recognition ratio of each action

7 Introduction of Submitted Algorithms

In this section, we introduce two algorithms submitted to the contest. One is the training based approach, and the other is the heuristic approach.

7.1 Training Based Approach with Object Detection and Tracking

Traditional action recognition frameworks extract features from motion in the videos and perform discriminative analysis for action classification. However, in ACE dataset, it is argued that many actions concerned cannot always be uniquely determined by using motion features alone, such as action "boiling egg". Their work hence proposes to analyze the cooking video by separately modeling the human motion and the context.

Most cooking actions can be characterized by certain motion pattern of the arms of the actors, for example, mixing or cutting. Thus the motions of the actors' arms are first modeled. As observed from the videos, the skin area of the actors' arms can be detected. Using the prior of the human skin area location, more accurate motion description and modeling can be obtained.

As analysis to the kitchen scene video dataset, the five cooking menus "ham and eggs", "omelet", "scrambled egg", "boiled egg" and "Kinshi-tamago" use three kinds of ingredients: egg, milk, and ham, which is directly related to action "breaking", "peeling" and "seasoning". Also cooking tools chopstick, and saucepan are used, which can be used to identify action "mixing", "turning". All the aforementioned items are placed on the kitchen table initially. Therefore they propose to 1) obtain the geometry of the kitchen table from the depth image, which is feasible since the table can fit well to a plane, 2) locate the

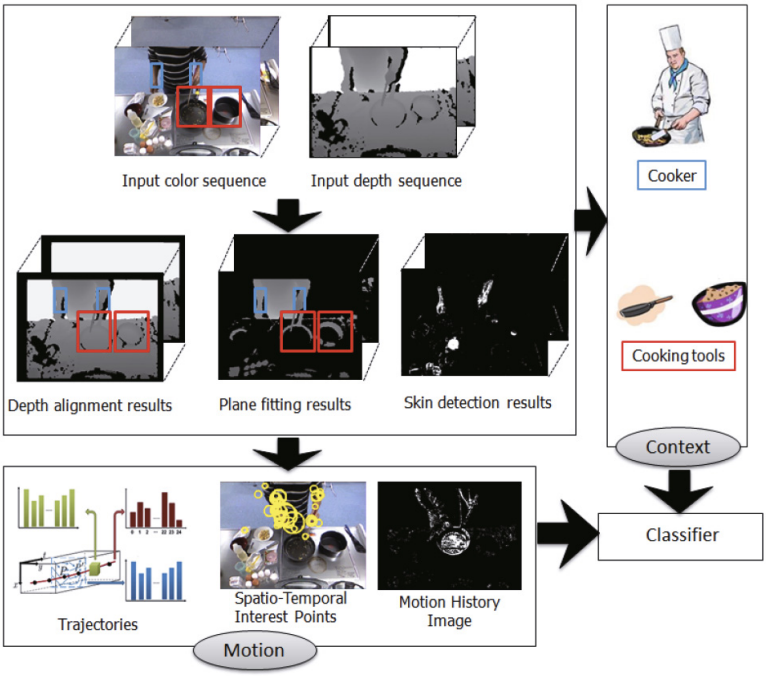


Fig. 8. The proposed framework of cooking action recognition system

objects placed on the kitchen table by their depth information and model their appearance using the color image.

In practice, they implement the cooking action recognition system as shown in Fig. 8.

Preprocessing. The depth and color images are aligned following the work[7]. Then, they apply RANSAC to find the geometry of the kitchen table and floor and label every pixel of the image whether it is on the table/floor or not by the its depth. Skin area is also obtained by classification on the color of each pixel.

Feature Extraction. In order to handle multiple actions in one video, they split the video into small clips. The label of each clip is decided by the major voting of labels of frames.

Motion History Image Motion History Image (MHI)[8] uses a single image to represent the information of how an object has moved (spatially and temporally) in an image sequence. They extract MHI from the video clips and resize the MHI image to 40×30 pixels image which thus forms a 1200- dimensional feature vector.

Spatio-Temporal Interest Point Description Spatio-temporal interest point (STIP) description is widely used for action recognition. They employ the most representative versions of STIP, which uses the Harris 3D detector for the interest point detection[9]. Histogram of Oriented Gradient (HOG) and Histogram of Optical Flow (HOF) descriptors are then extracted to form STIP descriptor and are encoded using Vector Quantization (VQ). A 1000-word codebook is learned from the training set, thus each clip forms a 1000- dimensional feature vector.

Trajectories They adopt a simplified version of the trajectory representation scheme proposed in[10]. First note that they only extract the trajectories related to the detected skin area and is not on the kitchen table and floor. Then for each video clip, they quantize all motion trajectories (short trajectories are discarded) into 45 histogram bins according to both depth (5 quantization levels) and orientation (9 quantization levels). The 45-dimensional representation for the trajectories are similarly encoded using VQ and a learned 1000-word codebook for the feature representation.

Context Information They only consider the pixel which is not on the kitchen table and floor for the context feature extraction. Also, only areas with motion is considered. They then split the pixels into skin region and non-skin region using the skin detection information. For both skin and non-skin region, they compute a 128-dimensional HSV color histogram of the pixels on 1×1 and 2×2 spatial pyramids as the feature descriptor. Thus a 1280-dimensional feature vector are used to represent the context information in total.

Action Model Learning. They train classifiers on the feature representation and labels extracted in the previous sections. For each action class, they train a one-vs-all linear SVM classifier as the action model. For testing, all classifiers are applied on the test sample, the label of the test sample is determined by the largest scored classifier.

Post-processing. In order to utilize the temporal information, they apply the post-processing step. They apply 1D Markov Random Field on the predicted class labels. Denote y_i is the label of the subsequence i, x_i is the corresponding extracted feature. The formulation is as follows.

$$\min_{y}\{E(y) = \sum_{i\in V} E_d(y_i) + \lambda \sum_{i,j\in N(i)} E_s(y_i, y_j)\} \tag{5}$$

The data term $E_d(y_i) = 1 - ((p(y_i|x_i) + p(y_{i+1}|x_{i+1}))/2$, where $p(y_i|x_i)$ is returned from the aforementioned one-vs-all linear SVM classifier. Meanwhile, the smooth term E_s follows Potts model $E_s(y_i, y_j) = \{0, y_i = y_j; 1, y_i \neq y_j$. λ is set as 0.1 in their experiment. They finally utilize graph cuts to solve above equation.

Table 2. Image features for cooking gesture recognition

Feature	Definition
x_in_pot	Whether something is in a pot or not
x_on_pan	Whether something is on a frying pan or not
x_on_board	Whether something is on a board or not
x_moved_in_pot	Whether something is moving in a pot or not
x_moved_on_pan	Whether something is moving on a frying pan or not
x_moved_on_board	Whether something is moving on a board or not
x_moved_in_bowl	Whether something is moving in a bowl or not
ham_on_board	Whether a ham is on a board or not
hand_above_bowl	Whether cook's hand is above/on a bowl or not
hand_above_pan	Whether cook's hand is above/on a frying pan or not
hand_above_board	Whether cook's hand is above/on a board or not
hand_around_dispenser	Whether cook's hand is around a dispenser or not
egg_yolk_in_bowl	Whether egg yolks is in a bowl or not
egg_yolk_on_pan	Whether egg yolks is on a frying pan or not
egg_yolk_on_board	Whether egg yolks is on a board or not
work_on_bowl_side	Whether the center of cook's hands is closer to that of a bowl or not
mixed_egg_on_pan	Whether mixed-egg is spread on a frying pan or not

7.2 Heuristic Approach

Their method recognizes cooking gestures in combination with simple and straightforward image features under the state transition constraint depending on each cooking menu.

Feature Definition. The image features used in their method are shown in Table 2. Note that each feature value is binarized (0 or 1) with several kinds of thresholds trained with training datasets. The process flow of the method is described below.

Pre-processing. The label for each frame is set to NONE. Also, the kitchen table region in an input video is extracted by thresholding the initial input depth image.

Cooking Menu Recognition. The cooking menu in the video is sequentially recognized in the following steps.

Step 1 "Boiled-egg" if the number of frames where $x\_in\_pot = 1$ is larger than that of frames where $x\_on\_pan = 1$ in the whole video.
Step 2 "Kinshi-tamago" if the number of frames where $x\_on\_board = 1$ and $work\_on\_bowl\_side = 0$ after a BAKING scene is significantly large.
Step 3 "Ham-egg" if the number of frames where $mixed\_egg\_on\_pan = 1$ is significantly small in the whole video.
Step 4 "Omelette" or "Scramble-egg" if all the above conditions are not satisfied.

Table 3. Scheme for cooking gesture recognition

Gesture	Definition
BREAKING (bowl)	Scene from the first frame with $hand\_above\_bowl = 1$ and $work\_on\_bowlside = 1$ to the subsequent frame with $egg\_yolk\_in\_bowl = 1$
BREAKING (frying pan)	Scene from the first frame with $hand\_above\_pan = 1$ and $work\_on\_bowlside = 1$ to the subsequent frame with $egg\_yolk\_on\_pan = 1$
MIXING (pot)	Scene from the first frame with $x\_moved\_in\_pot = 1$ to the first frame with $x\_moved\_in\_pot = 0$
MIXING (bowl)	Scene where $egg\_yolk\_in\_bowl$ or $x\_moved\_in\_bowl$ changes frequently
MIXING (frying pan)	Scene where $x\_moved\_on\_pan$ keeps 1
BAKING	Scene from the first frame with $x\_on\_pan = 1$ to the last frame with $x\_on\_pan = 1$
TURNING (ham)	Scene where $x\_moved\_on\_pan$ keeps 1
TURNING (egg)	Scene where the variance of the original (not binarized) value of $mixed\_egg\_on\_pan$ is significantly large
CUTTING (pot)	Scene from the first frame with $hand\_above\_board = 1$ to the first frame with $egg\_yolk\_on\_board = 1$
CUTTING (ham)	Scene from the first frame with $ham\_on\_board = 1$ and $work\_on\_bowl\_side = 0$ to the first frame with $x\_moved\_on\_board = 0$
CUTTING (board)	Scene where $hand\_above\_board$ and x_on_board keeps 1, and $work\_on\_bowl\_side$ keeps 0
BOILING	Scene from the first frame with $x\_in\_pot = 1$ to the last frame with $x\_in\_pot = 1$
SEASONING	Scene from the first frame with $hand\_around\_dispenser = 1$ to the subsequent frame with $hand\_around\_dispenser = 1$
PEELING	Scene where $hand\_above\_bowl$ keeps 1
NONE	Scene where all the other gestures are not specified

Cooking Gesture Recognition. The cooking menus can be separated into two categories: 1) pot-based menu ("Boiled-egg") and 2) frying pan-based menu ("Ham-egg", "Kinshi-tamago", "Omelette", and "Scramble-egg"). There must include BOILING scenes in an input video of a pot-based menu. On the other hand, there must include BAKING scenes in an input video of a frying pan-based menu. Their method at first detects and fixes BOILING scenes for a pot-based menu or BAKING scenes for a frying pan based menu. Then, their method detects the other cooking gestures considering the necessity of 1) the time spent on each cooking gesture and 2) the transition of the cooking gestures expected in the estimated cooking menu. Here, the transition of the cooking gestures in each cooking menu is constrained as follows.

Boiled-egg BOILING [⇔ MIXING] ⇒ PEELING [⇒ CUTTING]
Ham-egg (BREAKING ⇒ CUTTING) or (CUTTING [⇒ BAKING] ⇒ BREAKING) ⇒ BAKING [⇒ MIXING] [⇔ TURNING] ⇔ SEASONING
Kinshi-tamago BREAKING ⇒ MIXING ⇒ BAKING [⇔ TURNING] ⇒ CUTTING
Omelette or Scramble-egg BREAKING [⇒ MIXING] [⇒ CUTTING] or (CUTTING ⇒ BREAKING) ⇒ SEASONING ⇒ MIXING ⇒ BAKING ⇔ MIXING or TURNING

The cooking gestures are recognized according to the scheme shown in Table 3 under the gesture transition constraint.

Post-processing. To avoid unnatural labeling results, their method performs several kinds of label modification. For example, a short NONE section is inserted into the turn of the label, and successive short sections with the same label are merged.

8 Conclusion

This paper introduced a new action dataset "Actions for Cooking Eggs (ACE) Dataset" for evaluating action recognition methods, and reported a contest with using the dataset. Unlike the previous related action datasets, our dataset emphasizes a scene context which supports the determination of action. We hope that ACE dataset will be used by many researchers to discuss the importance of considering the scene context in gesture recognition.

Acknowledgments. The authors like to thank Tomo Asakura, Keizaburo Takase, Keisuke Yasuzawa, Kenta Matsui and Yusa Ko (Kyoto University in Japan) for cooking the menus, and Toshiki Sonoda (Kyushu University in Japan) for making illustrations.

References

1. Mitra, S., Acharya, T.: Gesture Recognition: A Survey. IEEE Transactions on Systems, Man, and Cybernetics - Part C: Applications and Reviews 37(3), 311–324 (2007)
2. Poppe, R.: A survey on vision-based human action recognition. International Journal of Image and Vision Computing 28(6), 976–990 (2010)
3. Schuldt, C., Laptev, I., Caputo, B.: Recognizing human actions: A local svm approach. In: Proceedings of the 17th International Conference on Pattern Recognition, vol. 3, pp. 32–36 (2004)
4. Gorelick, L., Blank, M., Shechtman, E., Irani, M., Basri, R.: Actions as space-time shapes. Transactions on Pattern Analysis and Machine Intelligence 29(12), 2247–2253 (2007)
5. Ryoo, M.S., Chen, C.-C., Aggarwal, J.K., Roy-Chowdhury, A.: An overview of contest on semantic description of human activities (SDHA) 2010. In: Ünay, D., Çataltepe, Z., Aksoy, S. (eds.) ICPR 2010. LNCS, vol. 6388, pp. 270–285. Springer, Heidelberg (2010)

6. Tenorth, M., Bandouch, J., Beetz, M.: The TUM Kitchen Data Set of Everyday Manipulation Activities for Motion Tracking and Action Recognition. In: IEEE International Workshop on Tracking Humans for the Evaluation of their Motion in Image Sequences (THEMIS), in conjunction with ICCV 2009 (2009)
7. http://nicolas.burrus.name/index.php/Research/KinectCalibration
8. Davis, J.W., Richards, W., Bobick, A.F.: Categorical representation and recognition of oscillatory motion patterns. In: CVPR, pp. 1628–1635 (2000)
9. Laptev, I.: On space-time interest points. International Journal of Computer Vision 64(2-3), 107–123 (2005)
10. Sun, J., Wu, X., Yan, S., Cheong, L.F., Chua, T.-S., Li, J.: Hierarchical spatio-temporal context modeling for action recognition. In: CVPR, pp. 2004–2011 (2009)

Results and Analysis of the ChaLearn Gesture Challenge 2012

Isabelle Guyon[1], V. Athitsos[2], P. Jangyodsuk[2], H.J. Escalante[3], and B. Hamner[4]

[1] ChaLearn, Berkeley, California
guyon@chalearn.org
http://chalearn.org
[2] University of Texas at Arlington, Texas
{athitsos,pat.jangyodsuk}@uta.edu
[3] INAOE, Puebla, Mexico
hugojair@inaoep.mx
[4] Kaggle, San Francisco, California
ben@benhamner.com

Abstract. The Kinect^{TM} camera has revolutionized the field of computer vision by making available low cost 3D cameras recording both RGB and depth data, using a structured light infrared sensor. We recorded and made available a large database of 50,000 hand and arm gestures. With these data, we organized a challenge emphasizing the problem of learning from very few examples. The data are split into subtasks, each using a small vocabulary of 8 to 12 gestures, related to a particular application domain: hand signals used by divers, finger codes to represent numerals, signals used by referees, Marshalling signals to guide vehicles or aircrafts, etc. We limited the problem to single users for each task and to the recognition of short sequences of gestures punctuated by returning the hands to a resting position. This situation is encountered in computer interface applications, including robotics, education, and gaming. The challenge setting fosters progress in transfer learning by providing for training a large number of subtasks related to, but different from the tasks on which the competitors are tested.

1 Introduction

Gesture recognition is an important sub-problem in many computer vision applications, including image/video indexing, robot navigation, video surveillance, computer interfaces, and gaming. With simple gestures such as hand waving, gesture recognition could enable controlling the lights or thermostat in your home or changing TV channels. The same technology may even make it possible to automatically detect more complex human behaviors, to allow surveillance systems to sound an alarm when someone is acting suspiciously, for example, or to send help whenever a bedridden patient shows signs of distress.

Gesture recognition also provides excellent benchmarks for Adaptive and Intelligent Systems (AIS) and computer vision algorithms. The recognition of continuous, natural gestures is very challenging due to the multi-modal nature of the visual cues (e.g., movements of fingers and lips, facial expressions, body pose), as well as technical limitations

X. Jiang et al. (Eds.): WDIA 2012, LNCS 7854, pp. 186–204, 2013.

such as spatial and temporal resolution and unreliable depth cues. Technical difficulties include tracking reliably hand, head and body parts, and achieving 3D invariance. The competition we organized helped improve the accuracy of gesture recognition using Microsoft Kinect^{TM} motion sensor technology, a low cost 3D depth-sensing camera. Examples of depth images are shown in Figure 1.

Much of the recent research in machine learning and data mining has sacrificed the grand goal of designing systems ever approaching human intelligence for solving tasks of practical interest with more immediate reward. Humans can recognize new gestures after seeing just one example (one-shot-learning). With computers though, recognizing even well-defined gestures, such as sign language, is much more challenging and has traditionally required thousands of training examples to teach the software. One of our goals was to evaluate whether transfer learning algorithms, which can exploit miscellaneous data resources, can improve the performance of systems designed to work on new similar tasks (e.g. recognize a new vocabulary of gestures). To see what the machines are capable of, ChaLearn launched in 2012 a competition with prizes donated by Microsoft.

Fig. 1. Kinect^{TM} data. Color rendering of depth images from the gesture challenge database were recorded with a Kinect^{TM} camera. Regular RGB images are recorded simultaneously (not shown). Kinect^{TM} can record videos with up to 30 frames per second.

2 Problem Setting and Data

We are portraying a single user in front of a fixed camera, interacting with a computer by performing gestures to play a game, remotely control appliances or robots, or learn to perform gestures from an educational software. We have collected a large dataset of gestures using the Microsoft Software Development Kit (SKD) interfaced to Matlab (Figure 2), which includes:

- over 50,000 gestures recorded with the Kinect^{TM} camera, including RGB and depth videos,
- with image sizes 240 x 320 pixels,
- at 10 frames per second,
- recorded by 20 different users,
- grouped in 500 batches of 100 gestures,
- each batch including 47 sequences of 1 to 5 gestures drawn from various small gesture vocabularies of 8 to 12 gestures,
- from 85 different gesture vocabularies.

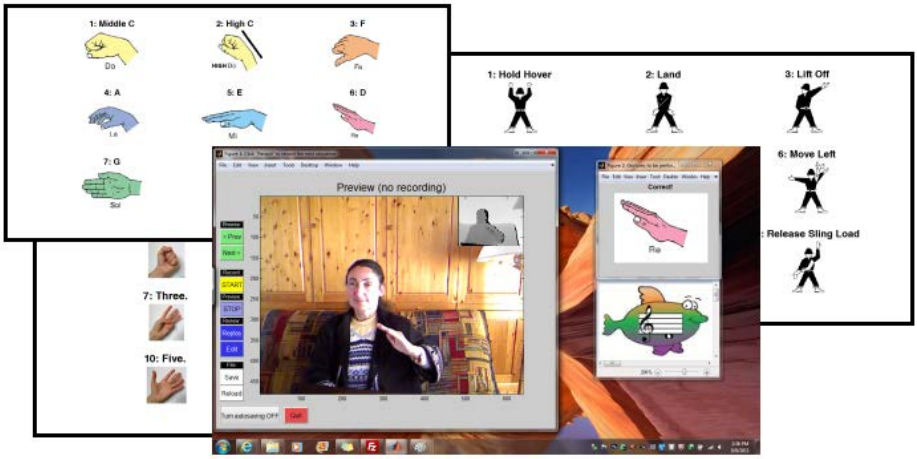


Fig. 2. Data collection. We recorded a large database of hand and arm gestures using the RGB+Depth camera Kinect^{TM}, which was made freely available. The figure shows some of the gesture lexicons and the user interface.

The data are available from in 2 formats[1]: A lossy compressed AVI format (5 GB) and a quasi-lossless AVI format (30 GB). It presents various features of interest (Table 1).

To get a sufficient spacial resolution, we framed only the upper body. Although Microsoft provided a full body skeleton tracker with the SDK used for data collection, at the time of data collection, it could not handle partial body occlusion and was not usable for our data. Rather, we provided manual annotations:

- all temporal segmentation for the devel01-20 batches into individual gestures;
- the position of the head, shoulders, elbows and hands for 400 frames sampled from the devel01-20 batches (Figure 3);
- image alignment between RGB and depth modalities.

We also provided code to browse though the data, a library of computer vision and machine learning techniques written in Matlab featuring examples drawn from the challenge datasets, and an end-to-end baseline system capable of processing challenge data and producing a sample submission. The dataset is described in details in a companion paper [9].

3 Task of the Challenge: One-Shot-Learning

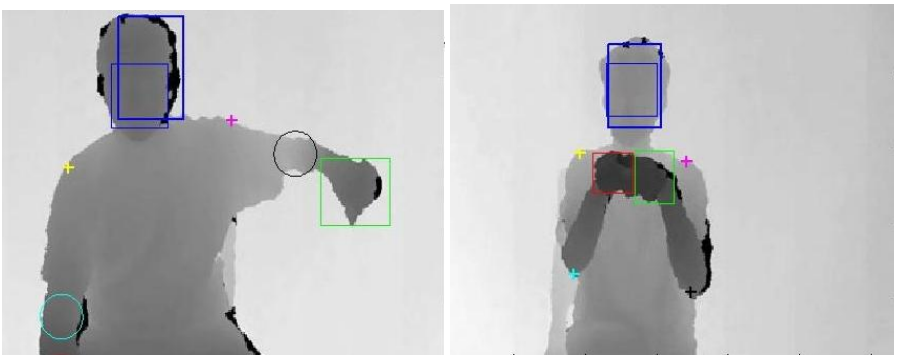
The data are organized in batches: development batches devel01-480, validation batches valid01-20, and final evaluation batches final01-20 (for round 1) and final21-40 (for round 2). For the devel batches, we provided all the labels. To evaluate the performances on "one-shot-learning" tasks, the valid and final batches were provided with

[1] `http://gesture.chalearn.org/data`

Table 1. Easy and challenging aspects of the data

Easy aspects
Fixed camera
Availability of depth data
Within a batch: single user, homogeneous recording conditions, small vocabulary
Gestures separated by returning to a resting position
Gestures performed mostly by arms and hands
Camera framing upper body (some exceptions)

Challenging aspects
Within a batch: only one labeled example of each gesture
Skeleton tracking data not provided
Between batches: variations in background, clothing, skin color, lighting, temperature, resolution
Some errors or omissions in performing gestures
Some users are less skilled than others
Some parts of the body may be occluded



(a) (b)

Fig. 3. Body part annotations. We provided body part annotations for development data, including head, shoulder, elbow and hand positions. Occluded parts are indicated by circles.

labels only for **one example of each gesture class** in each batch (training examples). The goal was to automatically predict the gesture labels for the remaining unlabeled gesture sequences (test examples).

Each batch includes 100 recorded gestures grouped in sequences of 1 to 5 gestures performed by the same user. The gestures are drawn from a small vocabulary of 8 to 12 unique gestures, which we call a "lexicon". For instance a gesture vocabulary may consist of the signs to referee volleyball games or the signs to represent small animals in the sign language for the deaf. We selected lexicons from nine categories corresponding to various settings or application domains (Figure 4):

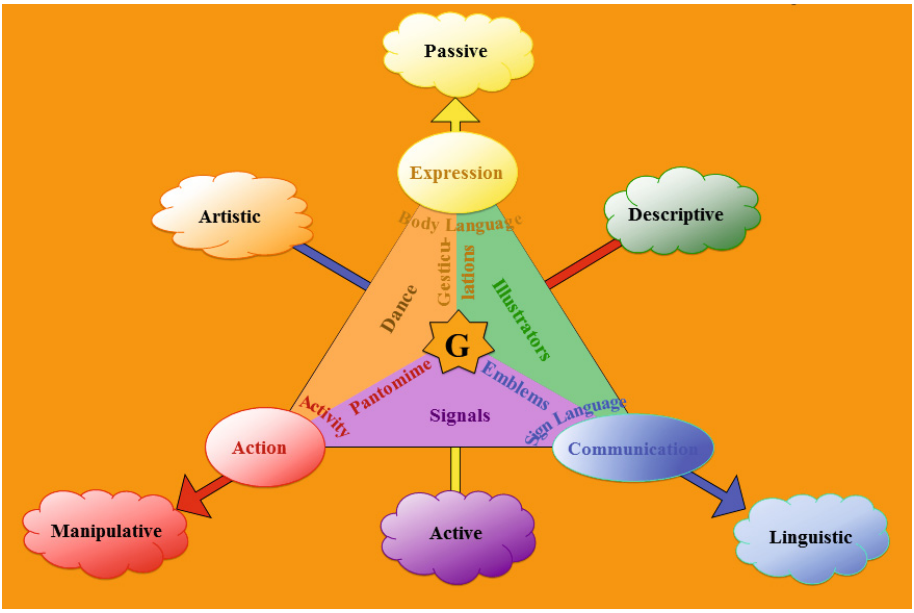


Fig. 4. Types of gestures. We created a classification of gesture types according to purpose defined by three complementary axes: communication, expression and action. We selected 85 gesture vocabularies, including Italian gestures, Indian Mudras, Sign language for the deaf, diving signals, pantomimes, and body language.

1. **Body language** gestures (like scratching your head, crossing your arms).
2. **Gesticulations** performed to accompany speech.
3. **Illustrators** (like Italian gestures).
4. **Emblems** (like Indian Mudras).
5. **Signs** (from sign languages for the deaf).
6. **Signals** (like referee signals, diving signals, or Marshalling signals to guide machinery or vehicle).
7. **Actions** (like drinking or writing).
8. **Pantomimes** (gestures made to mimic actions).
9. **Dance postures.**

During the challenge, we did not disclose the identity of the lexicons and of the users.

4 Protocol and Evaluation

Both rounds of the challenge consisted of two main components: a development phase (Dec. 7, 2011 to Apr. 6, 2012 for round 1 and May 7 to Sep. 6, 2012 for round 2) and a final evaluation phase (Apr. 7 to Apr. 10. 2012 for round 1 and Sep. 7 to Sep. 10 for round 2):

During the development phase (lasting approximately 4 months), the participants were asked to build a learning system capable of learning from a single training example a gesture classification problem. To that end, they received the development data to train and self-evaluate their systems. To monitor their progress they could use the validation data for which the test labels were withheld. The prediction results on validation data could be submitted on-line to get immediate feed-back. A real-time leaderboard showed to the participants their current standing based on their validation set predictions.

During the final evaluation phase (lasting four days), the participants performed similar tasks as those of the validation data on new final evaluation data revealed at the end of the development phase. The participants had only a few days to train their systems and upload their predictions. Prior to the end of the development phase, and BEFORE the final evaluation data was revealed, a software vault was made available so the participants could upload executable code for their best learning system, which they then used to train their models and make predictions on the final evaluation test data. This allowed the competition organizers to check their results and ensure the fairness of the competition. Note that participation was NOT conditioned on submitting code or disclosing methods. If any of the top ranking participants had opted not to submit their learning system for verification, an alternative verification method would have been offered. All top ranking participants submitted their code in round 1. In round 2, one top ranking participant did not submit himself to the validation procedure and forfeited his prize.

The submission and evaluation of the challenge entries was via the Kaggle platform. Post challenge submissions can still be made as an ever lasting benchmark. The official rules are provided on the website of the challenge[2].

In both rounds, participation was encouraged by donating free Kinect^{TM} sensors to the first ten entrants who outperformed the baseline method on the leaderboard and by offering prizes of $5000, $3000 and $2000 to the top three ranking participants, donated by Microsoft. The participants had the opportunity to present their results at the CVPR 2012 and ICPR 2012 conferences.

Metric of Evaluation

For each unlabeled video, the participants were instructed to provide an ordered list of labels R corresponding to the recognized gestures. We compared this list to the corresponding list of truth labels T i.e.the prescribed list of gestures that the user had to play during data collection. We computed the Levenshtein distance $L(R,T)$, that is the minimum number of edit operations (substitution, insertion, or deletion) that one has to perform to go from R to T (or vice versa). The Levenhstein distance is also known as "edit distance". For example: $L([124],[32]) = 2$.

The overall score is the sum of the Levenshtein distances for all the lines of the result file compared to the corresponding lines in the truth value file, divided by the total number of gestures in the truth value file. This score is analogous to an error rate. For simplicity, in what follows, we call it "error rate". However, it can exceed one.

Public score means the score that appears on the leaderboard during the development period and is based on the validation data. Private score means the score that was

[2] `http://gesture.chalearn.org/`

computed on the final evaluation data released at the end of the development period, which was not revealed until the challenge was over. The private score was used to rank the participants and determine the prizes.

5 Results

The first round of the challenge attracted 50 teams and the second round 35 teams. In total, 935 entries were made. This an unprecedented level of participation for a computer vision challenge requiring very specialized skills. For comparison, the popular Pascal2 VOC challenges attracted in 2011 between and 1 and 19 participants.

The results of the top ranking participants were checked by the organizers who reproduced their results using the code provided by the participants BEFORE they had access to the final evaluation data. All of them passed successfully the verification process. These results are shown in Tables 2 and 3.

Table 2. Results of round 1. In round 1 the baseline method was a simple template matching method (see text). For comparison, we show the results on the final set number 2 not available in round 1.

Team	Public score on validation set	Private score on final set #1	For comparison score on final set #2
Alfnie	0.1426	0.0996	0.0915
Pennect	0.1797	0.1652	0.1231
OneMillionMonkeys	0.2697	0.1685	0.1819
Immortals	0.2543	0.1846	0.1853
Zonga	0.2714	0.2303	0.2190
Balazs Godeny	0.2637	0.2314	0.2679
SkyNet	0.2825	0.2330	0.1841
XiaoZhuwWudi	0.2930	0.2564	0.2607
Baseline method 1	0.5976	0.6251	0.5646

Table 3. Results of round 2. In round 2, the baseline method was the "Principal Motion" method (see text).

Team	Public score on validation set	For comparison score on final set #1	Private score on final set #2
alfnie	0.0951	0.0734	0.0710
Turtle Tamers	0.2001	0.1702	0.1098
Joewan	0.1669	0.1680	0.1448
Wayne Zhang	0.2814	0.2303	0.1846
Manavender	0.2310	0.2163	0.1608
HIT CS	0.1763	0.2825	0.2008
Vigilant	0.3090	0.2809	0.2235
Baseline method 2	0.3814	0.2997	0.3172

Statistics on the Results

We show in Figure 5 the distribution of results. The figure represents the histograms of performance of all the entries made on validation data in rounds 1 and 2. The distribution was widely spread in round 1, indicating that the tasks of the challenge separated well the participants: they were challenging enough to require some effort to achieve good results, yet they were doable. The two top ranking participants and several others made no entry on the validation set until the very last days of the challenge, possibly in an attempt to avoid that other competitors would put additional effort to try to beat them. Their entries ended up cutting the error rate by almost a factor of two, reaching 10% error, narrowing down considerably the gap to human performance (which is under 2% error). The achievements of the top ranking participants are noteworthy considering that several participants did not succeed in outperforming the very trivial baseline method provided by the organizers. In round 1, the participants received sample code with the simple following method: The videos were first temporally segmented using equally spaced segmentation points estimated from the average isolated gesture duration. The video segments were then simply averaged to form a 2d representation, which was then "flattened" as a vector. The one-nearest-neighbor algorithm was then used for classification using the Euclidean distance. It is interesting to note that the experiments that we performed using methods that track hand position and describe the gestures as a hand trajectory perform worse then the baseline method, due to the difficulty of tracking the hand position.

In the second round, the organizers provided additional library functions as part of the samples. In particular, they provided code for dynamic time warping allowing easily the implementation of Hidden Markov models [19], which proved to work well in round 1 (See Appendix A). They also provided a better baseline end-to-end system than in round 1: the "Principal motion" method [4] based on principal components of low resolution motion histograms. The whole distribution of scores for the second round participants is therefore shifted upwards. The error rate of the winner further reduces the gap to human performance (7% error on the final evaluation set of round 2).

Figure 6 shows the correlation between the validation set error and the final evaluation data error in both rounds. We purposely made the final evaluation data slightly easier so the entrants would not eventually feel frustrated. Hence we expected that the entry points would be under the diagonal. This is confirmed for the top ranking participants. However, some lower ranking participants have entry points over the diagonal, which indicates a possible overfitting of the validation data.

Further analysis of the data batch by batch revealed that for three batches all the participants performed poorly, including the winner. Those are data batches for which details of the finger position is important (e.g. counting with fingers).

Survey of Methods Employed by the Participants

We asked the participants to fill out a survey about the methods employed. Twenty eight groups replied, among the top ranking participants. We briefly summarize the answers of the top ranking participants. More details including slides of the presentations are found on the website of the challenge.

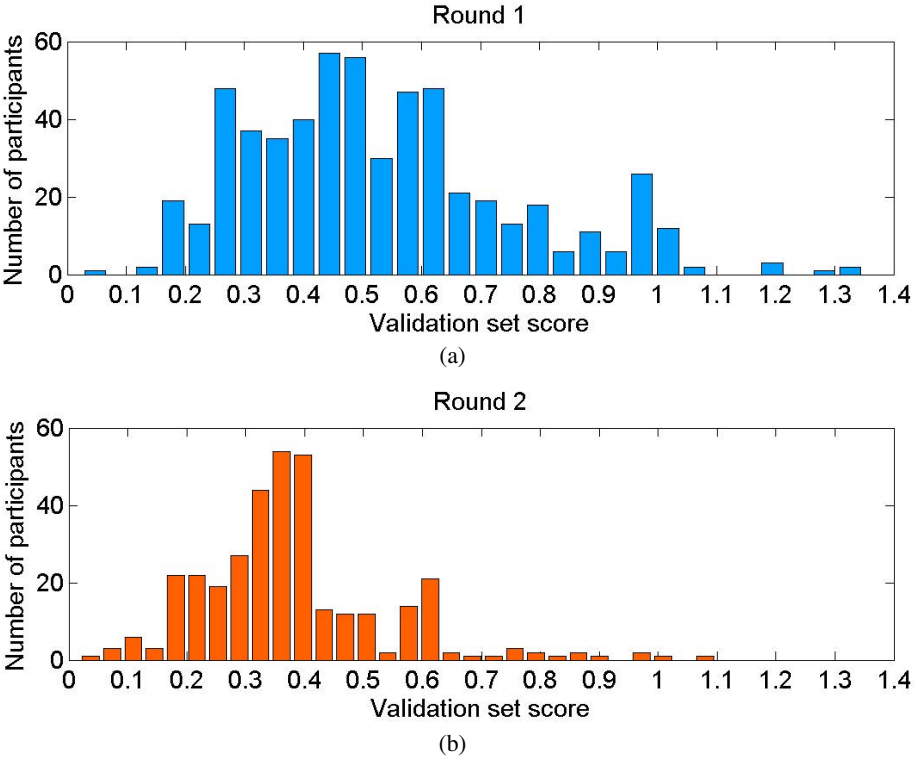


Fig. 5. Result distribution in both rounds. The figure shows histograms of score values on validation data.

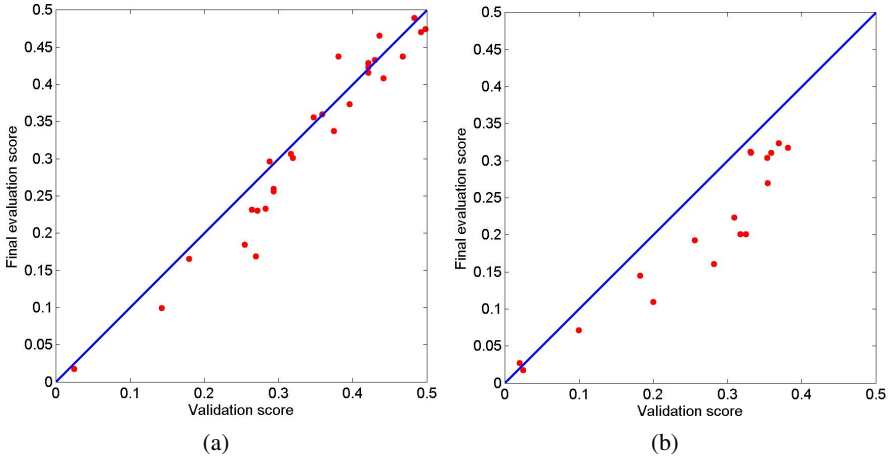


Fig. 6. Correlation between validation results and final evaluation. (a) Round 1. (b) Round2.

Preprocessing and Data Representation: Most of the participants employed image enhancement and filtering techniques, in majority denoising or outlier removal and background removal. Some reduced the image resolution for faster processing. Notably, some of the top ranking participants did not do any such low level preprocessing. The majority of the top ranking participants used HOG/HOF features [3,15] and/or ad-hoc hand crafted features, edge/corner detectors or SIFT/STIP features [14]. The latter use a bag-of-feature strategy, which ignores exact location of features and therefore provides some robustness against translations. The winner of both rounds of the challenge claims that his features are inspired by the human visual system. Very few participants resorted to using body parts or trained features. Most participants used the depth image only, but about one third used both RGB and depth images. Interestingly, the second place winner in round 1 used the RGB image only. About one third of the participants did no dimensionality reduction at all and one third resorted to feature selection. Other popular techniques included linear transforms (such as PCA, see e.g. [10]) and clustering.

Recognition: For temporal segmentation, most participants used candidate cuts based on similarities with the resting position or based on amount of motion. All the top ranking participants used recognition-based segmentation techniques (in which recognition and segmentation are integrated). As gesture representation, all highest ranking participants used a variable length sequence of feature vectors (sometimes in combination with other representations). To handle such variable length representations, the highest ranking participants used Hidden Markov Models (HMM), Conditional Random Fields (CRF) or other similar graphical models, see e.g. [12]. This is a state machine including skips and self-loops to allow for variation in the speed of the gesture execution. The most likely sequence of gestures is determined by a Viterbi search. Some highly ranked, but not top ranking, participants used a bag-of-word representation or image templates, including motion energy or motion history representations [1]. The corresponding classifiers were usually nearest neighbors (using as metric the Euclidean distance or correlation). One participant used a linear SVM, see e.g. [10]. Many participants made use of the development data to either learn features or gesture representations in the spirit of "transfer learning", see e.g. [18].

Implementation: Most participants claimed that the algorithmic complexity of their methods was linear in image size, number of frames per video, and number of training examples. The median execution time on the 20 batches of the final evaluation set was 2.5 hours, which is very reasonable and close to real time performance. However, there were a few outliers and it took up to 50 hours for the slowest code. Most participants, including the top ranking ones, claimed that their methods were simple and easy to implement. Several top ranking participants claimed that they had made novel contributions. Most participants developed their own code and may be willing to share it when it matures. Matlab was the most popular platform and was used by the 2 top ranking participants. "C" and derived languages were often used and sometimes in combination with Matlab, in particular making use of OpenCV libraries [2]. A few participants used Java. The majority of the participants developed under the Windows operating system (65%), the rest divided about evenly between MacOS and Linux. Two third of the

systems required less than 2GB of memory and all less than 8 GB. Parallelism was usually exploited via the use of multi-processor machines.

Development Effort: Two-third of the survey respondents spent more than 2 man-weeks of development effort. This is a lot of effort in comparison to the computer time generally estimated to a few hours of a few days. Half of the participants estimated that the challenge duration was sufficient to achieve their goals. One third declared that they will spend more time during the second round.

Algorithms Employed by the Top Ranking Participants

We summarize the descriptions provided by the top ranking participants in their fact sheets. Interestingly, all top ranking methods are based on techniques making no explicit detection and tracking of humans or individual body parts. This departs from the methods based on skeleton-tracking used in early applications of Kinect^{TM} to computer interfaces.

The winner of both rounds (Alfonso Nieto Castañon of Spain, a.k.a. alfnie) used a novel technique called "Motion Signature analyses", inspired by the neural mechanisms underlying information processing in the visual system. This is an unpublished method using a sliding window to perform simultaneously recognition and temporal segmentation, based solely on depth images. The method, described by the authors as a "Bayesian network", is similar to a Hidden Markov Model (HMM). It performs simultaneous recognition and segmentation using the Viterbi algorithm. The preprocessing steps include Wavelet filtering replacement of missing values and outlier detection. Notably, this method is one of the fastest despite the fact that he implemented it in Matlab (close to real time on a regular laptop). The author claims that it is linear complexity in image size, number of frames, and number of training examples.

The second best ranked participants (team Pennect of Universit of Pennsylvania, USA, in round 1 and team Turtle Tamers of Slovakia, in round 2) used very similar methods and performed similarly. Their methods are based on an HMM-style model using HOG/HOF features to represent movie frames. They differ in that Pennect used RGB images only while Turtle Tamers used both RGB and depth. Another difference is that Pennect used HOG/HOF features at 3 different scales while Turtle Tamers created a bag of features using K-means clustering from only 40x40 resolution and 16 orientation bins. Pennect trained a one-vs-all linear classifier for each frame in every model and used the discriminant value as a local state score for the HMM while Turtle Tamers used a quadratic-chi kernel metric for comparing pairs of frames in the training and test movie. As preprocessing, Pennect uses mean subtraction and compensates for body translations while Turtle Tamers replaces the missing values by the median of neighboring values. Both teams claim a linear complexity in number of frames, number of training examples, and image size. They both provided Matlab software that processes all the batches of the final test set on a regular laptop in a few hours.

The next best ranked participants (who won third place in round 2), the Joewan team [21], used a slightly different approach. They relied on the motion segmentation method provided by the organizers to pre-segment videos. They then represented each video as a bag of 3D MOSIFT features (integrating RGB and depth data) then used a

nearest neighbor classifier. Their algorithm is super-quadratic in image size, linear in number of frames per video, and linear in number of training examples. The method is rather slow and takes over a day to process all the batches of the final test set on a regular laptop.

The third best ranked team in round 1 (OneMillionMonkeys) also used an HMM in which a state is created for each frame of the gesture exemplars. His data representation is based on edge detection in each frame. Edges are associated with several attributes including the X/Y coordinates, their orientation, their sharpness, their depth and location in an area of change. To provide a local state score to the HMM for test frames, OneMillionMonkeys calculated the joint probability of all the nearest neighbors in training frames using a Gaussian model. The system works exclusively from the depth images. The system is one of the slowest proposed. Its processing speed is linear in number of training examples but quadratic in image size and number of frames per video. The method is rather slow and takes over a day to process all the batches of the final test set on a regular laptop.

Robustness to Translation and Scale

This section reports post-challenge experiments we conducted, using the code provided by the participants, to test the robustness of recognition to body translation and image scaling. In a variety of tasks, the user sits in a fixed position relative to a camera. Such is the setup that we used to collect the challenge data. However, it can happen that the user shifts his position or moves forward or backward. We emulated this situation by transforming a number of challenge batches as follows:

- We went back to the original data and selected batches including a large background area in which no gesture was taking place. This constitutes the *utran* data.
- We visually inspected the training videos to identify a cropping area including every important gesture parts. Once selected, the cropping size was fixed for the given batch. The aspect ratio was always always 4:3 (width:height), similar to the challenge data.
- For every test video in the batch, using the same cropping size, a different horizontal translation was applied. This was done by visual inspection to make sure no important gesture part was occluded. No vertical translation was applied. This constitutes the *tran* data.
- Similarly, we applied various scaling factors to generate the *scaled* data.

We selected 20 batches for these experiments, not coinciding with the sets of batches used for validation, and final testing because most of those batches included dynamic gestures covering the entire image area, therefore not leaving room for translations. The batches used are harder on average than those used for the challenge final evaluation, in particular because they include more static posture recognition. We ran experiments with the un-translated batches (utran) and with the translated batches (tran) and the scaled batches (scaled).

The results are summarized in Table 4.[3] We notice that the methods of the winner of both rounds (Alfnie) are robust against translation and scale while the second ranking methods (Pennect and Turtle Tamers) exhibit important performance degradation between utran and tran and scaled. This is not so surprising considering that both Pennect and Turtle Tamers use features rigidly positioned on image feature maps.

Methods robust against translation include those of Joewan [21] and Immortals/ Manavender (this is the same author under two different pseudonyms for round 1 and round 2) [16]. Their representations are based on a bag of visual words, inspired by techniques used in action recognition [14]. Such representations are inherently shift invariant. The slight performance loss in translated data may be due to partial occlusions.

Table 4. Comparisons of results for the top ranking methods in both rounds, on the validation set and final evaluation sets of each round, and on the untranslated (utran) and translated (tran) batches used in the translation experiments, and on the scaled data. We also indicate the time spent executing one batch in seconds on an Intel Core 2 Duo 2 GHz (desktop) with 4 GB of memory.

Name	valid	final1	final2	utran	trans	scaled	time (s/batch)
Alfnie1	0.1426	0.0996	0.0915	0.2316	0.2255	0.2573	71
Alfnie2	0.0995	0.0734	0.0710	0.1635	0.2310	0.2566	55
BalazsGodeny	0.2714	0.2314	0.2679	0.4347	0.5636	0.5526	93
HITCS	0.3245	0.2825	0.2008	0.4743	0.6640	0.6066	96
Immortals	0.2488	0.1847	0.1853	0.3594	0.3962	0.4152	925
Joewan	0.1824	0.1680	0.1448	0.2623	0.2612	0.2913	4099
Manavender	0.2559	0.2164	0.1925	0.3644	0.4252	0.4358	384
OneMillionMonkeys	0.2875	0.1685	0.1819	0.3633	0.4961	0.5552	21600
Pennect	0.1797	0.1652	0.1231	0.2589	0.4888	0.4068	287
SkyNet	0.2825	0.2330	0.1841	0.3901	0.4693	0.4771	120
TurtleTamers	0.2084	0.1702	0.1098	0.2896	0.5993	0.5296	383
Vigilant	0.3090	0.2809	0.2235	0.3817	0.5173	0.5067	294
WayneZhang	0.2819	0.2303	0.1608	0.3387	0.6278	0.5843	NA
XiaoZhuWudi	0.2930	0.2564	0.2607	0.3962	0.6986	0.6897	320
Zonga	0.2714	0.2303	0.2191	0.4163	0.4905	0.5776	159

6 Demonstration Competitions

We also hosted two demonstration competitions, one in June 2012 at the CVPR conference, and one in November 2012 at the ICPR conference. The goal in these demonstration competitions was to address qualitative factors of gesture recognition systems that were not addressed in the quantitative part of the gesture challenge. In particular, these qualitative factors included relevance and importance of the target application, user friendliness, quality of system design, and real time performance.

[3] Some results in Tables 2 or 3 may be slightly different from the results of Table 4 because the former comes from the participants entries on the submission website while the latter come from running the participants' submitted code on our computers. The differences did not affect the final ranking.

At CVPR 2012 we had eight participants. We summarize the methods of the winners:

- The winning entry was by a team from Turkey [11], and entailed a system performing articulated hand pose estimation, hand shape classification, and gesture recognition. The method uses randomized classification forests to assign class labels to each pixel on a depth image, and the final class label is determined by voting. The authors report in their paper experiments on American sign language and on a subset of the ChaLearn gesture challenge data, with very high success rates (larger than 90%).
- The entry winning second place was by a team from Italy [8]. Their system allowed a user to play a memory game against a robot. In the game, each player has to remember and repeat a sequence of gestures performed by the other player, and then add one more gesture to that sequence. The team also made a honorable entry in the ChaLearn gesture challenge (ranking 9 of 50 in the first round). The method uses features based on 3D HOG and HOF, followed by sparse coding and SVM classification.
- The third place system was by a team from Switzerland that performed real-time head pose estimation [6], for the purpose of improving the realism of sound simulation. The user wearing headphones received a 3D reconstruction of sound automatically adjusted to changes in head orientation, such that the apparent location of sources of sound remains accurate. The approach is based on discriminative random regression forests, which simultaneously classify image regions into whether they belong to the head region or not and cast probabilistic votes in a continuous space of head poses, defined as the 3D position of the nose and the Euler rotation angles.

All three winning systems demonstrated real time performance on standard hardware. Other entries included a system for gesture-based interfaces in operating rooms, two general-purpose systems allowing non-technical users to define gestures that the system should recognize, and two systems focusing on minimizing processing time and CPU load, so as to produce lightweight gesture recognition modules.

At the second demonstration contest, hosted at ICPR 2012, we had six participants. We summarize the winning entries:

- The winning entry was by a team from Greece, performing real-time articulated hand pose tracking [17]. The method used employs a hand model that produces hypotheses of hand postures that are matched with RGB-D image features. The matching scores drive an iterative an optimization process using Particle Swarm Optimization (PSO), carried out on a parallel processor. Although the topic is similar to that of the winning entry in the first round, the method is radically different because it is model-based.
- The system winning second place was by a team from Italy [7], and allowed users to navigate and visualize 3D medical images using gestures. They perform view-independent hand shape classification for a number of shapes used to drive the user interface (open and closed fist, L shape and finger pointing). From the depth image only, they represent the hand as a point cloud summarized by the Flusser moments of its 2 first principal component projections. Classification is performed by a SVM and hand velocity is used to translate gesture into action.

– Third place was won by a team from Spain [5], whose system also allows users to manipulate medical image data with hand gestures. The authors proposed a novel 3D hand point cloud description using a Spherical Blurred Shape Model (SBSM) descriptor. They also used a SVM for classification of 8 different hand poses. The overall system incorporates hand trajectory features to map gesture to action.

Except for the top ranking entry, the winning methods ran in real time on standard hardware. The other entries included a system allowing non-technical users to customize gesture recognition algorithms, a system allowing members of the general public to interact with a storefront display and view information about products on that display, and a system for monitoring the execution of recipes in a kitchen environment.

Importantly, the methods proposed in the demonstration competitions are complementary to those of the quantitative evaluation. They provided real time methods capable of recognizing subtle hand postures, which can be used to drive computer interfaces. In the quantitative evaluation, most gesture lexicons included dynamic gestures. This prompted the participants to de-emphasize the recognition of finger postures, which is important for gestures in which hand posture has an important semantic role. As previously mentioned, the participants of the quantitative evaluation all performed poorly on batches including static hand postures. The hope is that by using the methods proposed in the demonstration competition, performances on the quantitative evaluation could be boosted and approach human performance.

7 Conclusion

The ChaLearn gesture challenge helped narrow down the gap between machine and human performance on the task of one-shot gesture learning from 3D video data. In two rounds each lasting four months, the challenge attracted a total of 85 teams making 935 entries. They lowered the error rate, starting from a baseline method making more than 50% error to 7% error. The winner of the challenge, Alfonso Nieto Castañón, used a method he invented, which is inspired by the human vision system. His method is robust against user horizontal translation and image scaling, a feature exhibited in post-challenge tests we conducted. ChaLearn also organized demonstration competitions of gesture recognition systems using Kinect^{TM} in conjunction with those events. Novel data representations were proposed to tackle with success, in real time, the problem of hand and finger posture recognition. The demonstration competition winners showed systems capable of accurately tracking in real time hand postures in application to touch free exploration of 3D medical images for surgeons in the operating room, finger spelling (sign language for the deaf), virtual shopping, and game controlling. Combining the methods proposed in the demonstration competition tackling the problem of hand postures and those of the quantitative evaluation focusing on the dynamics of hand and arm movements is a promising direction of future research. For a long lasting impact, the challenge platform, the data and software repositories will remain available beyond the term of the project.

Acknowledgements. This challenge was organized by ChaLearn `http://chalearn.org` whose directors are gratefully acknowledged. The submission website was hosted by Kaggle `http://kaggle.com` whom we thank for wonderful

support. This effort was initiated by the DARPA Deep Learning program and is supported by the US National Science Foundation (NSF) under grants ECCS 1128436 and ECCS 1128296, and the EU Pascal2 network of excellence. Our sponsors include Microsoft (Kinect for Xbox 360) and Texas Instrument who donated prizes. We are very grateful to Alex Kipman and Laura Massey at Microsoft and to Branislav Kisacanin at Texas Instrument who made this possible. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the funding agencies and sponsors. We also thank the committee members and participants of the CVPR 2011 gesture recognition workshop and the Pascal2 reviewers who made valuable suggestions.

A HMM Implementation in the Sample Code

Since Hidden Markov Models (HMM) [19] were so prominently used by the top ranking participants in round 1, we provided an implementation in the distributed sample code[4] for round 2. Several top ranking participants in round 2 made use of it. We give a brief description.

The models we are interested in possess a finite number of states. States and state transitions are represented by a graph. In their original formulation, HMMs are data generating models implementing a double random process: there are probabilities of transition from state to state and probabilities of emission of a visible observation in every state. In application to gesture recognition, each state represents an archetypal body posture and visible observations are movie frames. Sub-models of sequences of postures representing individual gestures can be chained. When a test video is observed, it is matched to an optimum sequence of states using the Viterbi algorithm [20]. Other algorithms that are close cousins to Hidden Markov Models such as Conditional random Fields (CRF) perform a very similar function without a data generating interpretation [13].

Generally, transitions between states have a weight (transition probability or transition score), which is learned by example. However, since we have only one training gesture per class in this challenge, all transition scores in our models are set to the same value (one). Similarly, when a large number of training examples is available, each node is usually associated with a trained emission model or a trained discriminant function recognizing whether a given observation belongs to a given state. Such emission models or discriminant functions allow us to compute matching scores of test frames to model states. But, since we have only one example of gesture per class in this challenge, we obtain matching scores by simply computing the distance of the test observation (test movie frame) with the reference template (training example movie frame).

We remind the reader that the subjects who recorded the videos were instructed to return their hands to a resting position between gestures (arms along the body). Hence, we provide the option to force the model to return to a resting position between gestures.

We now describe more specifically our implementation. Let $L(i), i = 1 : P$ be the length of the P example videos in which training gestures are performed. We begin by calling the Matlab function `parents = simple_forward_model(L, N)`,

[4] `http://gesture.chalearn.org/data/sample-code`

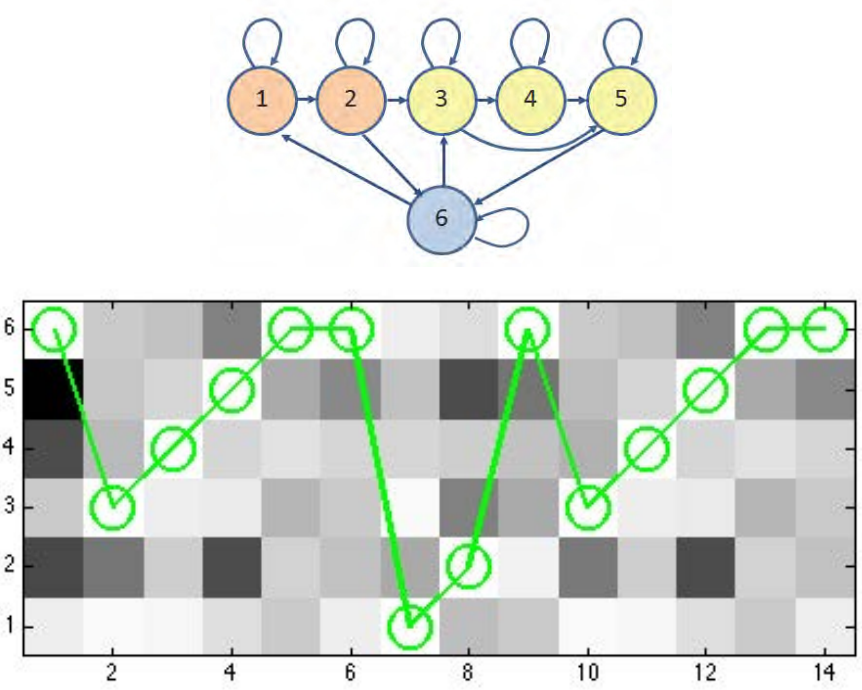


Fig. 7. Example of graph matching. Top: The graphical model for two gestures having 2 and 3 states respectively. Gesture 1 is the orange model and gesture 2 the yellow model. The blue state is the resting position. Bottom: The matrix of matching scores between the 6 states of the model associated each with a training example frame and 14 frames of a test movie. Lighter gray levels represent better scores. The best path is shown in green. The sequence identified is: gesture 2, gesture 1, gesture 2. The path goes through the resting position between every gesture.

where N is the length of the optional rest position model. The function returns an array "parents" containing the lists of the parent nodes of every node. For instance, if $L = [2, 3]$ and $N = 1$, the model returns the following lists of parents for every node:

```
1: 1 6
2: 2 1
3: 3 6
4: 4 3
5: 5 4 3
6: 6 2 5
```

This corresponds to a simple forward model (Fig. 7) in which the first training video has only 2 frames and the second one three frames, thus they are mapped with gesture models of 2 and 3 states respectively. The last node represents the resting position. Every state has a self loop to generate durations. Within a gesture model there are forward transitions to the immediate next state and to the following one. Skipping states

permits time compression. Hence, the model architecture allows us to perform an elastic matching of frame sequences.

We then need to compute the matching scores between all pairs of training frames and test frames. We use the function `local_scores=euclid_simil(TR, TE)`, which computes simply the negative Euclidean distance between preprocessed data frames. `TR` is the concatenation of the movies of all training videos plus a frame illustrating the resting position. `TE` is the test movie.

Finally, we compute the best matching path using the Viterbi algorithm with the function `best_score=viterbi(local_scores, parents)`. The algorithm computes global scores from the local scores as follows:

```
global_score = local_score + max(global_score_parents)
```

where `max(global_score_parents)` is the maximum global score of the parents of a given node. The algorithm also keeps track of which parent provided the maximum global score. The computation is initialized at the first frame of the test sequence and propagates until the last frame. The largest global score is then selected and using the pointers to the best parent node, the best path is back-tracked.

References

1. Bobick, A.F., Davis, J.W.: The recognition of human movement using temporal templates. IEEE Trans. Pattern Anal. Mach. Intell. 23(3), 257–267 (2001)
2. Bradski, G.: The OpenCV Library. Dr. Dobb's Journal of Software Tools (2000)
3. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: CVPR, pp. 886–893 (2005)
4. Escalante, H.J., Guyon, I.: Principal motion: PCA-based reconstruction of motion histograms. Technical report, ChaLearn (2012)
5. Escalera, S., Fornés, A., Pujol, O., Lladós, J., Radeva, P.: Circular blurred shape model for multiclass symbol recognition. IEEE Transactions on Systems, Man, and Cybernetics, Part B 41(2), 497–506 (2011)
6. Fanelli, G., Gall, J., Van Gool, L.J.: Real time head pose estimation with random regression forests. In: CVPR, pp. 617–624 (2011)
7. Gallo, L., Placitelli, A.P., Ciampi, M.: Controller-free exploration of medical image data: Experiencing the kinect. In: CBMS, pp. 1–6 (2011)
8. Gori, I., Fanello, S.R., Metta, G., Odone, F.: All gestures you can: a memory game. Technical report, Istituto Italiano di Tecnologia, Italy, Submitted to JMLR (2012)
9. Guyon, I., Athitsos, V., Jangyodsuk, P., Escalante, H.J.: The ChaLearn Gesture Dataset (CGD 2011). Submitted to Machine Vision and Applications (2013)
10. Hastie, T., Tibshirani, R., Friedman, J.H.: The elements of statistical learning: data mining, inference, and prediction: with 200 full-color illustrations. Springer, New York (2001)
11. Keskin, C., Kira, F., Kara, Y.E., Akarun, L.: Randomized decision forests for static and dynamic hand shape classification. In: CVPR Workshops, pp. 31–36. IEEE (2012)
12. Koller, D., Friedman, N.: Probabilistic Graphical Models: Principles and Techniques. MIT Press (2009)
13. Lafferty, J.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data, pp. 282–289. Morgan Kaufmann (2001)
14. Laptev, I.: On space-time interest points. International Journal of Computer Vision 64(2-3), 107–123 (2005)

15. Lucena, M., de la Blanca, N.P., Fuertes, J.M., Marín-Jiménez, M.J.: Human action recognition using optical flow accumulated local histograms. In: Araujo, H., Mendonça, A.M., Pinho, A.J., Torres, M.I. (eds.) IbPRIA 2009. LNCS, vol. 5524, pp. 32–39. Springer, Heidelberg (2009)
16. Malgireddy, M., Nwogu, I., Govindaraju, V.: Language-motivated approaches to action recognition. Submitted to JMLR (2013)
17. Oikonomidis, I., Kyriazis, N., Argyros, A.A.: Tracking the articulated motion of two strongly interacting hands. In: CVPR, pp. 1862–1869 (2012)
18. Pan, S.J., Yang, Q.: A survey on transfer learning. IEEE Transactions on Knoweledge and Data Engineering 22(10), 1345–1359 (2010)
19. Rabiner, L.R.: A tutorial on hidden markov models and selected applications in speech recognition. Proceedings of the IEEE, 257–286 (1989)
20. Viterbi, A.J.: Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. IEEE Transactions on Information Theory IT-13(2), 260–269 (1967)
21. Wan, J., Ruan, Q., Li, W.: One-shot learning gesture recognition from rgb-d data using bag-of-features. JMLR (in press, 2013)

Author Index