Active Monte Carlo Recognition

Felix v. Hundelshausen[1] and Manuela Veloso[2]

[1] Computer Science Department, Freie Universität Berlin, 14195 Berlin, Germany
hundelsh@googlemail.com
[2] Computer Science Department, Carnegie Mellon University
Pittsburgh, PA 15213, USA
veloso@cs.cmu.edu

Abstract. In this paper we introduce *Active Monte Carlo Recognition (AMCR)*, a new approach for object recognition. The method is based on seeding and propagating "relational" particles that represent hypothetical relations between low-level perception and high-level object knowledge. AMCR acts as a filter with each individual step verifying fragments of different objects, and with the sequence of resulting steps producing the overall recognition. In addition to the object label, AMCR also yields the point correspondences between the input object and the stored object. AMCR does not assume a given segmentation of the input object. It effectively handles object transformations in scale, translation, rotation, affine and non-affine distortion. We describe the general AMCR in detail, introduce a particular implementation, and present illustrative empirical results.

1 Introduction

As Computer Vision researchers, we are interested in processing and extracting information from a sequence of images. Since image sequences are subject to continuity constraints, *iterative methods* should be applied, rather than treating the images as being independent of one another.

In the field of state estimation by vision, e.g. recursively estimating the position and motion of a vehicle on a highway [4][3], the concept of iterative processing has been fully adopted. Until now, it has not been adopted for the task of object recognition. Almost all existing approaches for object recognition treat the images as being independent from one another, processing each in a pipeline of steps, not considering the results of earlier processing.

In this paper, we propose a new, iterative approach for object recognition. The framework deals with a sequence of input images of an object, and iteratively finds the best match in a given set of prototype objects. Besides recognizing the object, this approach finds the mapping that transforms the input object to the object stored in memory. It can handle differences in scale, translation, rotation, affine and even non-affine distortions.

The most important aspect of our approach is that recognition does not take place in a one-way pipeline: It is recursive and exploits feedback information.

C. Freksa, M. Kohlhase, and K. Schill (Eds.): KI 2006, LNAI 4314, pp. 229–243, 2007.

While the recognition runs, it can guide low-level operations according to the current recognition state. For instance, when the system cannot decide whether the input object is a camel or a dromedary, our method allows to focus attention on the back of the animal, thus finding out whether there are one or two humps.

2 Related Work

Our method is different and contrasts with two common approaches for object recognition: *Independent Feature Extraction* and *Directed Processing*. The first means running low-level image operations in each image of a video stream from scratch. In [6], for instance, building the scale space has to be done for each image anew. On a Pentium 4 3GHz processor just building the scale space for a 640×480 image takes approximately one second[1]. In object recognition from video rather than from photographs, this processing time is too long. Other examples for *Independent Feature Extraction* include Shape Contexts [2], Maximally Stable Extremal Regions [8], and affine interest point detectors [9].

Directed Processing means that current approaches treat object recognition as being solvable in a one-way sequence of processing steps, hopefully ending with the recognition of the object (for example, [6] builds the scale space, extracts keypoints, builds feature descriptors, matches these descriptors with descriptors in a database).

In contrast to *Independent Feature Extraction*, our approach allows to speedup processing by making use of earlier results. When a feature was detected in frame I_k the same feature is likely to occur at a close position in I_{k+1}.

In contrast to *Directed Processing*, our approach is iterative and does allow feedback loops. The current state of recognition can guide attention to parts and features that help to discriminate between objects.

3 Active Monte Carlo Recognition (AMCR)

In this section we introduce our new approach. We call it *Active Monte Carlo Recognition (AMCR)*, because it is based on sequential Monte Carlo filtering [11]. The word *active* stresses the fact that the approach allows the integration of information feedback. It can guide low-level feature extraction based on the current recognition state, as well as focus attention on important image locations. In this sense, the approach integrates a *visual behavior*, that is, a policy for guiding attention and feature extraction to parts of the object which allow its recognition.

Sequential Monte Carlo methods, or particle filters, have extensively been applied for robust object tracking. The best-known approach is the Condensation algorithm. For example, in [5] it is shown that a leaf can robustly be tracked in the presence of background clutter. Also, particle filtering has become the standard approach for mobile-robot localization [10]. Here, a probability distribution

[1] Using the C++ implementation of [6].

for the robot's position is approximated and propagated by a set of particles, each representing a hypothesis for the robot's position. Starting with a uniform distribution for example, the particles start to build clusters at highly likely points in a map, while the robot moves and observes its environment [10].

3.1 Analogy Between Object Recognition and Mobile Robot Localization

One inspiration for our approach came from realizing that object recognition and mobile robot-localization are essentially the same problem: When a robot finds its position in one of a set of maps (e.g. of different buildings), one could say that it recognized that its current environment matches one of the maps. When considering the environment as an object, finding the correct map actually means object recognition. Besides identifying the correct map, the correct position within this map is found, too. Hence, the correspondences between environment and map are also found. In the following section we will adapt and modify the Monte Carlo Localization approach to perform object recognition. Here, the main additional complication is, that we want to be not only invariant with respect to translation and rotation, but also to scale and affine distortion.

3.2 An Example

The initial task is a follows. Given are:

- $\mathcal{I} = I_1, ..., I_l$: a sequence of l input images
- $P = \{M_1, M_2, ..., M_r\}$: a set of r prototype images

The goal is to find the image M_k that corresponds to the image sequence. We assume that $\mathcal{I}$ shows an object of one of the prototype images M_i but that it can be arbitrarily scaled, translated, rotated or even sheared, in short, we want to allow any affine transformation. Furthermore, we do not assume that the object in I_k is already segmented from the background, i.e. we allow background clutter. However, we do assume that the prototype images are perfectly segmented. Besides identifying M_k we also want to find the affine transformation for a good match. Figure 1 shows an example of this setup.

3.3 Overview

We deal with two types of particles, *V-particles* and *M-particles*. The V-particles refer to positions in the input image, while the M-particles refer to positions in the prototype images. One V-particle is linked to several M-particles as shown in figure 2. One important property of the algorithm is that the particles move. While the V-particles move in the input image, the M-particles form moving clusters in the prototype images, at positions that correspond to the shape surrounding the V-particles.

To get an initial idea of how the algorithm works, Figure 3 illustrates the algorithm using only one V-particle. Initially, the M-particles are randomly distributed in the prototype images. While the V-particle moves, the M-particles

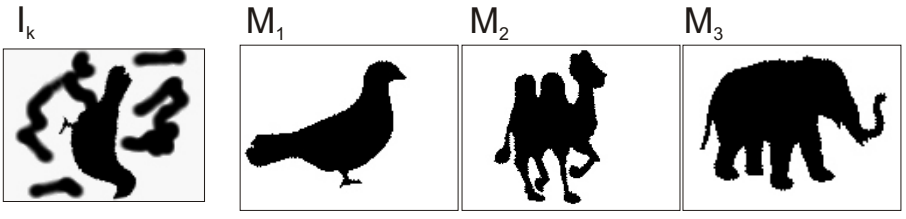

Fig. 1. The goal is to identify the bird shown in a sequence of input images I_k in the set of prototype images M_1, M_2, M_3 and to determine the affine transformation that maps the input object to its corresponding prototype object. Background clutter in the input image can be present. The shape shown in the image sequence can move. We assume that the movement is restricted to affine transformations.

move accordingly, subject to their estimate of the affine transformation. Step by step the M-Particles start to build clusters at probable locations and finally only one cluster remains in the prototype that corresponds to the input image. The resulting particles hold the correct affine transformation.

3.4 Definitions

To give a mathematically sound description of our algorithm we have to define several terms:

Definition 1. *A* ***V-particle*** $\mathbf{v}$ *is defined as* $\mathbf{v} := (\mathbf{p}, \mathcal{E}_a, \mathcal{E}_f, \mathcal{O}, \mathcal{Q}, \mathcal{F})$ *where* $\mathbf{p} = (x, y, \phi)^T$ *describes a position and orientation in the input image* I*,* $\mathcal{E}_a$ *is an affine estimator,* $\mathcal{E}_f$ *is a feature extractor,* $\mathcal{O}$ *is an observation model,* $\mathcal{Q}$ *is a motion model and* $\mathcal{F}$ *is a feedback strategy. (The latter five terms will be described with an example in section 4.)*

Definition 2. *A* ***prototype pose*** $\tilde{\mathbf{x}} = (x, y, \phi, i)$*,* $x, y \in I\!R$*,* $\phi \in [0, ..., 2\pi]$*,* $i \in \{1, ..., r\}$*, specifies a position* (x, y) *with orientation* ϕ *in the* i_{th} *of all* r *prototype images.*

Definition 3. *An* ***M-particle*** $\mathbf{m}$ *is defined as* $\mathbf{m} := (\tilde{\mathbf{x}}, \mathbf{A}, \pi)$ *where* $\tilde{\mathbf{x}}$ *specifies a pose in one of the prototype images. The* 3×3 *matrix* $\mathbf{A}$ *defines a 2D affine mapping in homogeneous coordinates. The value* $\pi \in [0, ..1]$ *is a probability.*

Definition 4. *A* ***particle configuration*** $\mathcal{C}$ *is a triple* $\mathcal{C} = (\mathcal{V}, \mathcal{M}, \mathcal{R})$*, where* $\mathcal{V} := \{\mathbf{v}_i\}, i = 1, ..., m$ *is a set of* m *V-particles,* $\mathcal{M} := \{\mathbf{m}_i\}, i = 1, ..., n$ *is a set of* n *M-particles and* $R : \mathcal{M} \longrightarrow \mathcal{V}$ *is a mapping relating each M-particle to a V-particle. The mapping is surjective but not injective, that is, different M-particles can be mapped to the same V-particle.*

The mapping R induces an equivalence relation in the set $\mathcal{M}$, dividing it in subsets whose elements are mapped to the same V-particle. The subset belonging to $\mathbf{v} \in \mathcal{V}$ is

$$R^{-1}(\mathbf{v}) = \{\mathbf{m} \in \mathcal{M} : R(\mathbf{m}) = \mathbf{v}\} \tag{1}$$

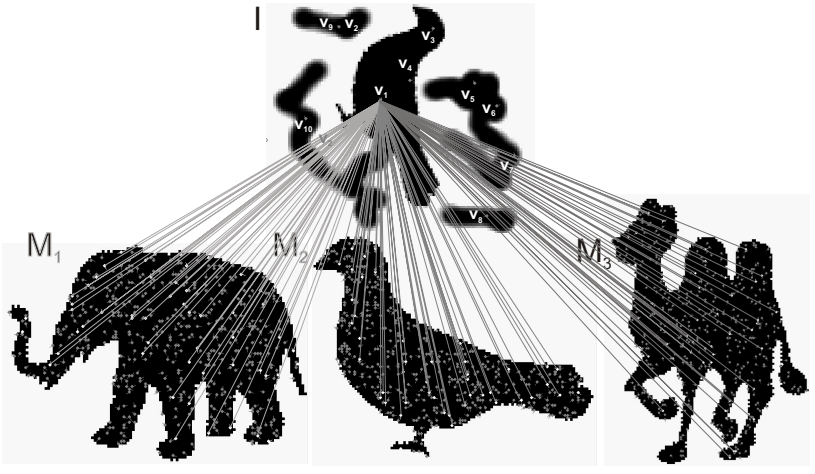

Fig. 2. Each of the $m = 10$ V-particles in the input image I is linked to 50 M-particles in each prototype image M_1, M_2, M_3. All particles are drawn in gray, except the $\mathbf{v_1}$ and the M-particles that are linked to it. The links are drawn by straight thin lines. Note, that only the connections of $\mathbf{v_1}$ are drawn, but that each V-particle has the same number of connections (to different M-particles).

Thus, each V-particle is linked to a whole set of M-particles and the sets of M-particles are disjoint. Relating our approach to mobile robot localization, each V-particle can be thought of a "simulated robot" discovering the input environment (the image I) and the related M-particles represent a probability distribution for the robot's position within a set of maps (the prototype images M_i). An example is shown in figure 2 where each of $m = 10$ V-particles is linked to 50 M-particles in each prototype image resulting in a number of $n = 10 \times 3 \times 50 = 1500$ M-particles. The set $R^{-1}(\mathbf{v})$ contains M-particles that are distributed over all prototype images. Often, we are interested in only the M-particles of a V-particle $\mathbf{v}$ that are in the same prototype image M_k. We will denote this set by

$$R_k^{-1}(\mathbf{v}) := \{\mathbf{m} = (i, \mathbf{p}, \mathbf{A}) \in \mathcal{M} : R(\mathbf{v}) = \mathbf{m} \wedge i = k\} \tag{2}$$

This set $R_k^{-1}(\mathbf{v})$ approximates the probability distribution for the corresponding position $\mathbf{v}$ in the prototype image M_k.

Often, we will consider an M-particle $\mathbf{m}$ in conjunction with its linked V-particle $\mathbf{v_m} := \mathcal{R}(\mathbf{m})$. Such a pair $(\mathbf{v_m}, \mathbf{m})$ represents a hypothetical relation between the input image and a prototype image. Thus we call the entity $(\mathbf{v_m}, \mathbf{m})$ a *relational particle*. The affine mapping $\mathbf{A}$, stored in $\mathbf{m}$, defines how the local neighborhood around $\mathbf{v_m}$ has to be transformed in order to match the local neighborhood around $\mathbf{m}$.

Definition 5. *A* ***relational particle*** $\mathbf{r}$ *is a tuple* $(\mathbf{v_m}, \mathbf{m})$*, with* $\mathbf{m} \in \mathbf{M}$ *and* $\mathbf{v_m} := \mathcal{R}(\mathbf{m})$.

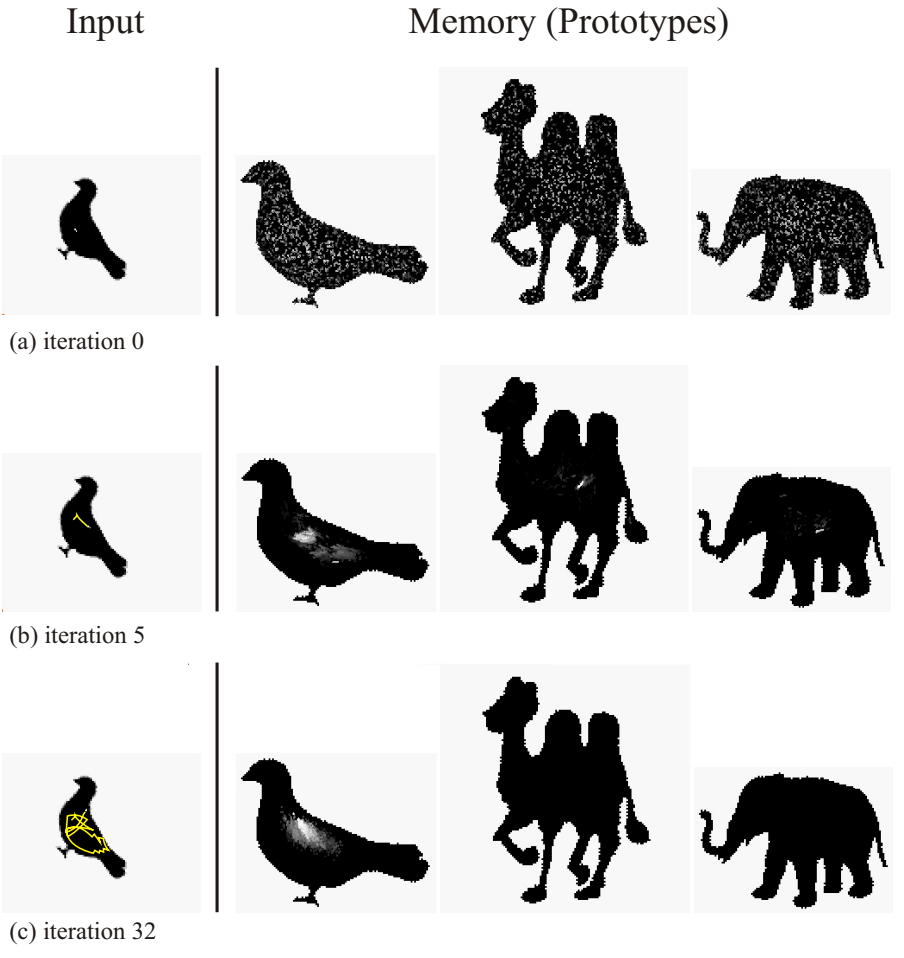

Fig. 3. This figure illustrates the algorithm using only one V-particle connected to 500 M-particles in each prototype image. (a) Initially, the M-particles are distributed randomly. (b)While the V-particle moves, the M-particles start to build clusters at probable locations. (c)Finally, only one cluster remains in the prototype image that corresponds to the input image. The trajectory of the V-particle corresponds to the thin curve in the input images. While the V-particles move, the M-particles move accordingly, subject to their current estimate of how the input and prototype images are related in terms of an affine transformation.

3.5 Probabilistic Formulation

For each V-particle $\mathbf{v}$ we reformulate the problem of object recognition as the problem of localizing $\mathbf{v}$. Here, localizing $\mathbf{v}$ means finding its corresponding prototype pose $\tilde{\mathbf{x}} = (x, y, \phi, i)$ (see definition 2), that is identifying the correct prototype image M_i and finding the corresponding pose (x, y, ϕ) within M_i. Since we do not assume that the object shown in the input images is segmented, the

algorithm will try to recognize whatever is at the position of $\mathbf{v}$. Since several V-particles exist, different objects can be recognized simultaneously.

With each new image I_k, each V-particle $\mathbf{v} = (\mathbf{p}, \mathcal{E}_a, \mathcal{E}_f, \mathcal{O}, \mathcal{Q}, \mathcal{F})$ will perform a movement and a measurement $\mathbf{z}_k$. The movement is controlled by the feedback policy $\mathcal{F}$ that returns a control command $\mathbf{u}_k$ in each iteration. It consists of a rotation and a translation. The measurement $\mathbf{z}_k$ is a feature descriptor returned as a result of applying feature extractor $\mathcal{E}_f$. We want to estimate the prototype pose $\tilde{\mathbf{x}}$ based on all measurements $Z^k = \{\mathbf{z}_k\}, i = 1, ..., k$, up to the current time, and knowledge about the movements and the initial state of $\tilde{\mathbf{x}}$. The initial knowledge about the state $\tilde{\mathbf{x}}_0$ is given by the apriori probability distribution $p(\tilde{\mathbf{x}}_0)$ over the space of $\tilde{\mathbf{x}}$. For instance, it could be that certain prototype images are known to be more likely, in a given context. However, we will often assume an initial uniform distribution. Thus, for each V-Particle, we are interested in constructing the posterior density $p(\tilde{\mathbf{x}}_\mathbf{k}|Z^k)$ of the current prototype pose $\tilde{\mathbf{x}}_\mathbf{k}$ conditioned on all measurements.

In analogy to mobile robot localization [10], to localize $\tilde{\mathbf{x}}_\mathbf{k}$ the corresponding position of the V-particle in time step k, we need to recursively compute the density $p(\tilde{\mathbf{x}}_\mathbf{k}|Z^k)$ at each time step, which is done in two phases, the prediction phase and the update phase:

– **Prediction Phase.** We use the *motion model* $\mathcal{Q}$ of the V-particle to predict $\tilde{\mathbf{x}}$ in the form of a predicted probability density function (PDF) $p(\tilde{\mathbf{x}}_k|Z^{k-1})$ taking only motion into account. In contrast to real mobile robot localization we know the effect of the control input precisely, without noise. However it still makes sense to include a noise model, since the overall approach will then be able to handle even non-affine distortions (to some limited degree). We assume that the current state $\tilde{\mathbf{x}}_\mathbf{k}$ is only dependent on the previous state $\tilde{\mathbf{x}}_{k-1}$ (the Markov assumption) and the known control input $\mathbf{u}_{k-1}$. The motion model is specified as a conditional density $p(\tilde{\mathbf{x}}_k|\mathbf{x}_{k-1}, \mathbf{u}_{k-1})$ and the predictive density over $\tilde{\mathbf{x}}_k$ is then calculated by integration:

$$p(\tilde{\mathbf{x}}_k|\mathbf{Z}^{k-1}) = \int p(\tilde{\mathbf{x}}_k|\mathbf{x}_{k-1}, \mathbf{u}_{k-1}) p(\tilde{\mathbf{x}}_{k-1}|Z^{k-1}) d\tilde{\mathbf{x}}_{k-1} \tag{3}$$

– **Update Phase.** In this phase we take a measurement $\mathbf{z}_k$ by applying feature extractor $\mathcal{F}_e$ around the V-Particle and use its *measurement model* $\mathcal{O}$ to obtain the posterior $p(\tilde{\mathbf{x}}_k|Z^k)$. We assume that the measurement $\mathbf{z}_k$ is conditionally independent from earlier measurements Z^{k-1} given $\tilde{\mathbf{x}}_k$ and the measurement model is given in terms of a likelihood $p(\mathbf{z}_k|\tilde{\mathbf{x}}_k)$. This term expresses the likelihood that the V-particle is at a location corresponding to $\tilde{\mathbf{x}}_k$, given that $\mathbf{z}_\mathbf{k}$ was observed. The posterior density over $\tilde{\mathbf{x}}_k$ is obtained using Bayes theorem:

$$p(\tilde{\mathbf{x}}_k|Z^k) = \frac{p(\mathbf{z}_k|\tilde{\mathbf{x}}_k) p(\tilde{\mathbf{x}}_k|Z^{k-1})}{p(\mathbf{z}_k|Z^{k-1})} \tag{4}$$

3.6 The AMCR-Algorithm

For each V-particle $\mathbf{v} = (\mathbf{p}, \mathcal{E}_a, \mathcal{E}_f, \mathcal{O}, \mathcal{Q}, \mathcal{F})$ the density $p(\tilde{\mathbf{x}}_k|Z_k)$ is approximated by its set $R^{-1}(\mathbf{v})$ of connected M-Particles. The overall algorithm is then summarized by the following procedure:

Algorithm 3.1. AMCR()

for each I_k

do
- **for each** V-particle $\mathbf{v} = (\mathbf{p}, \mathcal{E}_a, \mathcal{E}_f, \mathcal{O}, \mathcal{Q}, \mathcal{F})$
  - **do**
    - Get control command $\mathbf{u}_k = \mathcal{F}(\mathbf{v})$
    - Move $\mathbf{v}$ according to $\mathbf{u}_k$
    - **for each** $\mathbf{m} \in R^{-1}(\mathbf{v})$
      - **do**
        - PREDICT($\mathbf{m}$, $\mathcal{Q}$)
        - UPDATE($\mathbf{v}, \mathbf{m}, \mathcal{E}_a, \mathcal{E}_f, \mathcal{O}$)
    - RESAMPLE($R^{-1}(\mathbf{v})$)
- **every** w_{th} frame:
  - UPDATE(V)
  - RESAMPLE(V)
- Run other algorithms like tracking in parallel

Here, updating and resampling $\mathcal{V}$ is done only every wth frame ($w = 10$ in our implementation). To update $\mathcal{V}$, each $\mathbf{v} \in \mathcal{V}$ is assigned a weight proportional to the number of M-particles connected to $\mathbf{v}$. When resampling $\mathcal{V}$, particles that are created in several instances receive an exact copy of the M-Particles connected to the original V-particle. By resampling V the focus of attention is directed to parts of the input image that are interpretable in terms of the prototype images. The entire procedure of propagating the particles by sampling, reweighting, and resampling (Sampling/Importance Resampling, or SIR) is discussed in full detail in [12].

One important property of AMCR is that the input images are only accessed at the position of the V-particles. In each iteration algorithms such as determining the optical-flow can be processed in parallel and the V-particles can be moved according to the optical flow. In this way recognition can be distributed over several frames, even though the input object moves.

4 Radial-AMCR: AMCR for Shape Recognition

In this section, we describe a particular application of the AMCR-algorithm, allowing it to perform shape recognition. Here, we assume that all V-particles use the same affine estimator, feature extractor, motion and measurement model. We call this particular instantiation *Radial-AMCR* because it is based on radial edge scans.

4.1 The Affine Estimator and the Measurement Model

For Radial-AMCR the affine estimator $\mathcal{E}_a$ estimates an affine mapping for each M-particle by considering two triangles, one in the input image I_k and one in

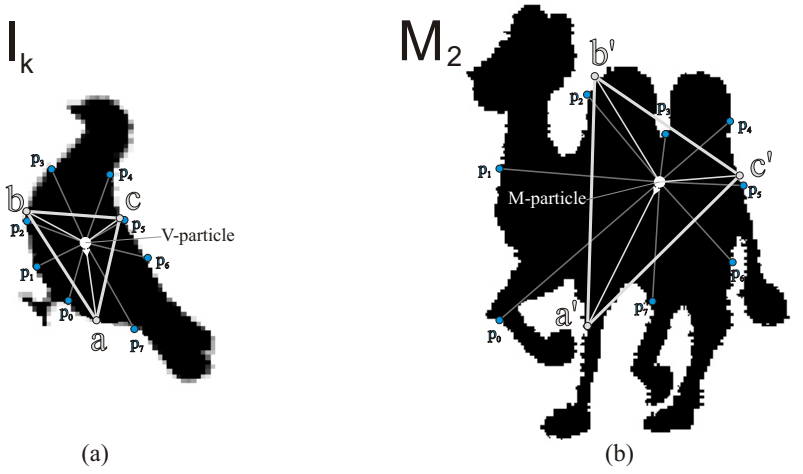

Fig. 4. Each pair $(\mathbf{v}, \mathbf{m})$ of particles creates a hypothesis for an affine transformation by constructing two triangles. The corner points of the triangles are found by three edge scans per particle. To compare the local shape around the particles 8 further points are determined by edge scans.

the prototype image of the M-particle. The three points of each triangle are found by applying an edge-detector along three scan lines, radially emerging from the V- and M-particle's position. The orientation of the V- and M-particle specifies the direction of the first scan line. The remaining angles are at 120 and 240 degrees relative to the first direction. In doing so, three points $\mathbf{a}, \mathbf{b}$ and $\mathbf{c}$ are determined for the V-particle $\mathbf{v}$ in I_k and three points $\mathbf{a}', \mathbf{b}'$ and $\mathbf{c}'$ are determined for the M-particle $\mathbf{m}$ in its prototype image. The pair $(\mathbf{v}, \mathbf{m})$, a relational particle, will now represent the hypothesis that the shape in the input image has to be transformed in such a way, that both triangles match. That is, we compute the affine transformation $\mathbf{A}$ that maps $\mathbf{a}$ to $\mathbf{a}'$,$\mathbf{b}$ to $\mathbf{b}'$ and $\mathbf{c}$ to $\mathbf{c}'$. Depending on the position of the V- and M-particles different hypotheses arise. Although, the affine mapping $\mathbf{A}$ is stored in the M-particle, it is an attribute of the whole relational particle $(\mathbf{v}, \mathbf{m})$. It is possible to store it in $\mathbf{m}$, because each M-particle can only be linked to one V-particle.

Based on the estimate of the affine transformation, we are now able to specify a measurement model. Similar to how the triangles are found, the feature extractor $\mathcal{E}_f$ performs radial edge scans along 8 rays. For each relational particle $(\mathbf{v}, \mathbf{m})$, two sets of 8 points are found, $\mathbf{z}_k = (\mathbf{p}_0, ...\mathbf{p}_7)$ in input image I_k (which constitutes the measurement of the V-particle) and $\mathbf{p}'_0, ...\mathbf{p}'_7$ in the prototype image of $\mathbf{m}$. Based on these points and the affine transformation we specify the measurement model $\mathcal{O}$, that is the likelihood $p(\mathbf{z}_k|\tilde{\mathbf{x}})$. The underlying consideration is that if the current M-particle was in the correct prototype image (the one that corresponds to the input images) and if its position and orientation exactly corresponded to the position of the V-particle in the current input image, then, when transforming the points $\mathbf{p}_0, ..., \mathbf{p}_7$ using $\mathbf{A}$, they would exactly match the points $\mathbf{p}'_0, ..., \mathbf{p}'_7$. We then calculate the deviations $\mathbf{w}_j = \mathbf{p}'_j - \mathbf{A}\mathbf{p}_j, j = 0, ..., 7$

and assume that each deviation is either an outlier or subject to a Gaussian distribution,

$$p(||\mathbf{w}_j||) = p_{outlier} + (1 - p_{outlier})\frac{1}{\sigma_w\sqrt{2\pi}}e^{-\frac{||w_j||^2}{2\sigma_w^2}}. \quad (5)$$

Here, $p_{outlier}$ is the probability of an outlier, in case of which we assume a uniform distribution over all ranges of $||\mathbf{w}_j||$ and σ_w is a constant ($p_{outlier} = 0.1$ and $\sigma_w = 10$ pixel in our implementation). Assuming that the individual scan line measurements are independent the measurement model then is

$$\pi = p(\mathbf{z}_k|\tilde{\mathbf{x}}_k^i) := \Pi_{j\in I_{\text{valid}}}\, p(||\mathbf{w}_j||)\Pi_{j\in I_{\text{invalid}}}\, p_{\text{invalid}}. \quad (6)$$

Here, I_{valid} is the subset of indices $\{i = 1..m\}$ for which both v_i and v_i' indicate the validity of the ith edge measurement, and I_{invalid} is its complementary set. The constant p_{invalid} is the modeled probability of an invalid range measurement ($p_{\text{invalid}} = 0.1$ in our implementation).

4.2 The Motion Model

The motion model $\mathcal{Q}$ defines how the M-particles move, when a V-particle moves as a response to its control command $\mathbf{u}$. It is specified in terms of the conditional density function $p(\tilde{\mathbf{x}}_k|\tilde{\mathbf{x}}_{k-1}, \mathbf{u}_{k-1})$. Rather than explicitly specifying this function we specify how to sample from it. Given $\mathbf{u}$ specifying the translation and rotation of the V-particle we transform the movement by the M-particle's current estimate of the affine transformation and add zero-mean Gaussian noise to both translation and rotation. The M-particle will then perform this transformed movement. When the prototype image is not an exact affine transformation of the input image, non-affine distortions can be compensated by the noise in the M-particle's movement.

4.3 Feedback Loops

There are two different feedback loops: *Feature Feedback* and *Attention Feedback*. Feature feedback means that different V-particles can have different feature extractors and that the extractors are selected depending on the recognition state. For instance, when recognition is ambiguous at a given iteration, feature extractors could change from edges to texture, if texture would better discriminate among the hypotheses. This aspect is not dealt with in more detail in this paper. Attention feedback has two mechanisms: One automatically occurs during V-resampling. V-particles with many connected M-particles will be reproduced more likely, which lets the V-particles concentrate on interpretable parts of the input shapes. V-particles which cover non-interpretable background clutter, will vanish automatically. The second mechanism involves the motion guidance of the V-particles.

Consider the case where a V-particle determines its movement solely based on the input image. For instance, a V-particle could always move forward and be

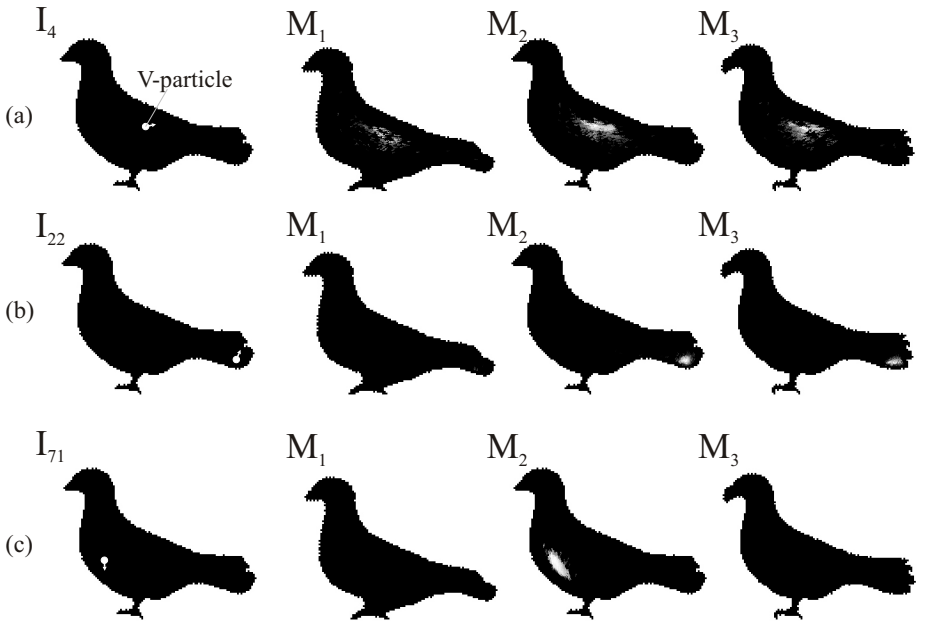

Fig. 5. Even if the prototype images are very similar the method converges. The figures shown here, are snapshots of a real experiment. However, convergence is slow in the case of high similarity. To speed it up, a feedback policy is needed that guides the V-particles to parts of the figures that let discriminate them (e.g. the feed, tail and head).

reflected at an edge in the image. This simple behavior would let the V-particle explore its input shape. Although the approach works, it is not optimal. To see why, consider the case where the prototype images are very similar, i.e. as in the scenario shown in figure 5. Since the birds are very similar, it takes 71 iterations till the method converges to only one remaining cluster in the correct prototype image. The reason is that the M-particles often scan the shapes at parts that are very similar. A strategy of guiding the V- and M-particles to parts that help to discriminate the shapes is required to increase the performance. In the situation shown in figure 5b) the recognition process is not sure, whether the input image is M_2 or M_3 and a feedback control that would move the V-particle to a part that best discriminates M_2 and M_3 (i.e the head of the birds) is desirable in this situation. One difficulty in implementing such an approach is to automatically compare all pairs of prototype images and to find the discriminative locations. This issue will be subject of a separate paper.

4.4 Lookup Tables

In each iteration, each M-particle has to apply its feature extractor in its prototype image. But since, the prototype images (the memory) is fixed, we can

pre-compute lookup tables to speed up the extraction process. For instance, for Radial-AMCR each pixel in each prototype images holds a pre-computed radial scan, and the edge points can simply be looked up. Thus, while the V-particles actually have to access the input image, the M-particles will just operate on lookup tables. In this sense, the prototype images M_i are rather feature lookup tables, instead of the actual prototype images.

4.5 The Focus of Attention

In our approach, the positions of the V-particles constitute the focus of attention. Since we do not assume that the input image is segmented, the V-particles will try to recognize whatever is at their position. Consider i.e an image showing different shapes the same time. Starting with 100 V-particles, initially randomly distributed in the image, some of the particles will lay within the figures, some between the figures, and some at image parts that contain background clutter. V-particles that are in figures, that are represented in the prototype images, will produce clusters after several iterations, the other V-particles will remain unsuccessful, that is they will not yield a focussed cluster of M-particles in a prototype image. This implies, that different shapes can be recognized simultaneously. Context based recognition can be achieved through the initial distribution of M-particles in the prototype images. If a prototype has no M-particles no processing power is consumed in trying to recognize it. That is, depending on the situation in which recognition takes place, certain prototypes can be biased.

5 Experimental Results

It is difficult to compare our algorithm against other methods for shape or object recognition, because they typically are not iterative. But the iterativeness is one of the most important properties of AMCR, because it allows to distribute the computational load over several frames. Even though recognition might take a second, the video input can be processed at full frame rate during this second. Thus other algorithms like tracking can run in parallel. This is the important philosophy behind our approach.

Despite this difficulty, we performed a comparison against shape contexts [1] by repeating the iterations over static images. Some of the input images that were correctly classifed are shown in figure 6. These had to be identified within

Fig. 6. Some examples of input images that were correctly classified

a database of 17 prototype shapes. Because of the severe background clutter our method performed clearly better than shape contexts. However, this is not the point, because shape contexts themselves could be integrated in AMCR by defining an appropriate observation model. Thus, our approach is more a framework that allows to incorporate existing approaches, rather than being opposed to them.

With the following example we try to show all properties of AMCR. We work with a prototype memory of only three shapes. We printed two of them on a sheet of paper, together with outlier shapes, such that the figures of interest cannot easily be segmented from the background clutter (see figure 7a). Then we took a video, while moving the sheet around. Running Radial-AMCR with 30 V-particles and 60 M-particle per prototype and V-particle, our method iteratively recognizes the given shapes. Using the lookup tables, one iteration takes approximately 61 milliseconds on a Pentium IV, 3 GHz processor. Splitting the update phases over two frames, the processing time per frame can be reduced to 30 milliseconds, which allows to process the input video at a rate of 30 frames per second. While recognition runs, we simultaneously determine the optical flow (using Lukas Kanade [7]) in the input sequence, and move the V-particles according to the flow, such that the relative position between the input shapes

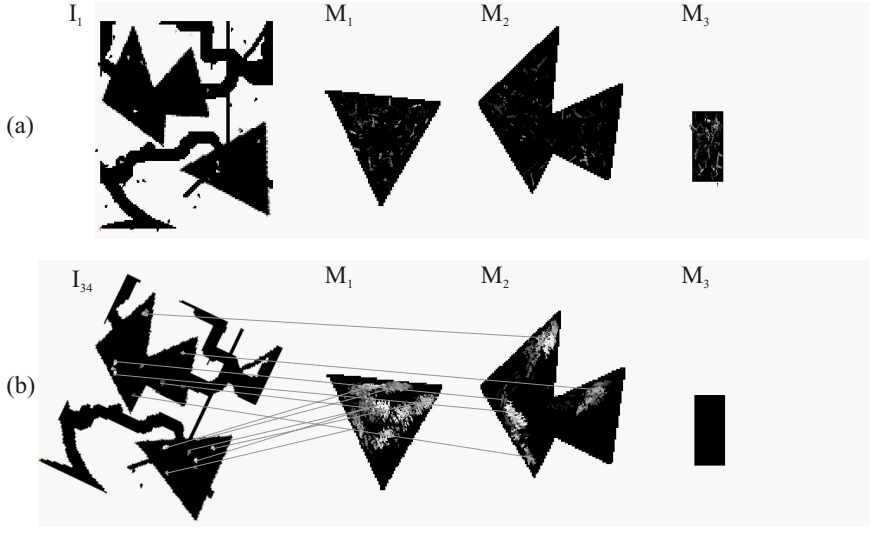

Fig. 7. At the beginning (a) the V and M-particles have random positions in the input image and the prototype images. After 34 iterations only V-particles remain, that are interpretable in terms of the prototype images. The M-particles form clusters at corresponding positions. The thin straight lines show these correspondences. While recognition runs, the input image is rotated and translated randomly and the V-particles are moved according to the optical flow. Thus, while tracking, the process of recognition is distributed over the sequence of images.

and the V-particles remains the same. This example is illustrated in figure 7. It shows how recognition and tracking can be performed simultaneously, how the process of recognition can be distributed over several frames and how several objects can be recognized simultaneously.

6 Conclusions

In this paper we have introduced *Active Monte Carlo Recognition (AMCR)* as a new framework for object recognition and a result of our realization of the similarity between mobile robot localization and object recognition. In relation to existing approaches, AMCR is based on an image sequence rather than on single images, and it includes a feedback loop integrated in the recognition process to guide attention to discriminative parts. At the core of AMCR are *relational particles* that represent hypotheses for relations between an input image and a set of prototype images. Low-level image access is hence performed with a relationship to high-level memory. We have shown the potential of our approach by implementing a particular instantiation of the algorithm for shape recognition. In summary, our approach has several contributions, including:

- **Iterativeness.** The process of recognition is distributed over a sequence of images showing the same object. Tracking can be performed simultaneously with recognition.
- **Local image access.** The input images are only accessed at the position of the V-particles. By moving the V-particles recognition can be combined with tracking. For instance, the V-particles can be moved according to the optical flow such that the relative position between V-particles and the object remains the same.
- **Multi-Modality.** AMCR maintains several hypothetical interpretations during its iterations. Also, several objects can be recognized simultaneously.
- **Simultaneous Segmentation and Recognition.** Our method does not require the object to be segmented from the background. Rather, segmentation and recognition are performed simultaneously.
- **Integration of Feedback Loops.** During iterations, an object might not be uniquely classified. Our approach allows to guide attention to parts and features that help discriminate the different hypotheses.

Future work will concentrate on the simultaneous application of different feature-extractors, a hierarchical organization of the prototype objects and the learning of feedback strategies.

References

1. S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts, 2001.
2. S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(4):509–522, 2002.

3. E. D. Dickmanns. The 4d-approach to dynamic machine vision. In *Proceedings of the 33rd IEEE Conference on Decision and Control*, volume 4, pages 3770–3775, 1994.
4. E. D. Dickmanns and B. D. Mysliwetz. Recursive 3-d road and relative ego-state recognition. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 14:199–213, February 1992.
5. M. Isard and A. Blake. Condensation: conditional density propagation for visual tracking. *International Journal of Computer Vision*, 1998.
6. D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.
7. B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *IJCAI81*, pages 674–679, 1981.
8. J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In *BMVC*, pages 384–393, 2002.
9. K. Mikolajczyk and C. Schmid. Scale and affine invariant interest point detectors. *International Journal of Computer Vision*, 60(1):63–86, 2004.
10. W. B. S. Thrun, D. Fox and F. Dellaert. Robust Monte Carlo Localization for Mobile Robots. *Artificial Intelligence*, 2001.
11. A. Smith, A. Doucet, and N. de Freitas. *Sequential Monte Carlo Methods in Practice*. Springer, New-York, 2001. ISBN 0-387-95146-6.
12. A. F. M. Smith and A. E. Gelfand. Bayesian statistics without tears: A sampling-resampling perspective. *American Statistician*, 46(2):84–88, 1992.