

MODELING AND RECOGNITION OF HUMAN ACTIONS USING A STOCHASTIC APPROACH

Esther B. Koller-Meier and Luc Van Gool

Communication Technology Lab, Image Science

Swiss Federal Institute of Technology (ETH)

CH-8092 Zurich, Switzerland

{ebmeier,vangool}@vision.ee.ethz.ch

Abstract This paper describes a self-learning prototype system for the real-time detection of unusual motion patterns. The proposed surveillance system uses a three-step approach consisting of a tracking, a learning and a recognition part. In the first step, an arbitrary, changing number of objects are tracked with an extension of the CONDENSATION algorithm. From the history of the tracked object states, temporal trajectories are formed which describe the motion paths of these objects. Secondly, characteristic motion patterns are learned by clustering these trajectories into prototype curves. In the final step, motion recognition is then tackled by tracking the position within these prototype curves based on the same method, the extended CONDENSATION algorithm, used for the object tracking.

Keywords: Tracking, Condensation algorithm, spatio-temporal grouping, human motion recognition

1. Introduction

The analysis of human movements or activities from image sequences is an important problem especially for surveillance applications. This paper focuses on the surveillance of bank lobbies where security staff is required to observe human activities on monitors to detect the occurrence of unusual events. The support of automatic video surveillance systems should relieve the operators by directing their attention to the unusual cases. We are primarily interested in detecting unusual events such as vandalism, panic or overcrowded areas including *unusual movements* as well as *unlikely object positions*. For this purpose we have developed a self-learning framework for the real-time detection of such

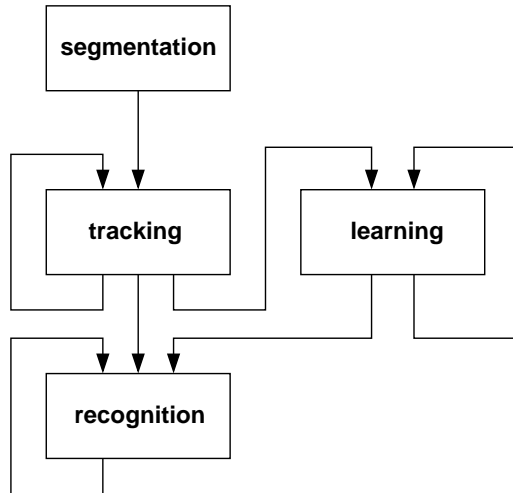


Figure 1. Information flow of the proposed surveillance system.

unusual motion patterns consisting of a tracking, a learning and a recognition part (see Fig. 1).

There is an extensive amount of computer vision work in the area of motion models. Such models are generally first constructed for the motion paths and the detected motions are then classified on the basis of these models. The most frequent attempt based on a state-space approach is the Hidden Markov Model (HMM) [3, 10, 11]. In [2] the CONDENSATION-based Trajectory Recognition (CTR) is proposed which can be seen as a generalization of HMM's. Alternative methods based on template techniques convert an image sequence into a static shape pattern. The most commonly used features for this technique are 2D meshes [8]. Furthermore, these approaches also comprise motion-energy images (MEI) and motion-history images (MHI) which are presented in [4]. A detailed review of human motion analysis can be found in [1].

The outline of this paper is given as follows. In the next section we introduce the tracking approach which is based on the CONDENSATION algorithm [5] to trace an arbitrary, changing number of objects. In Section 3 we propose a self-learning clustering method using the tracking results to build prototype curves as models for the motion paths. The recognition part where the tracking results are matched to the learned prototype curves is then explained in Section 4. Finally, we present some experimental results in Section 5 for the surveillance of bank lobbies.

2. Tracking

First of all we focus on the object tracking in video sequences. We use the extension of the CONDENSATION algorithm which we have developed to track an arbitrary, changing number of objects for mobile robots [7].

In principle, we are interested in recursively constructing the *a posteriori* probability density of an object state conditioned on all measurements $\mathcal{Z} = \{z_1, \dots, z_t\}$ which are obtained from a segmentation. The key idea of the CONDENSATION algorithm [5] is to approximate the probability distribution by a weighted sample set $S = \{(s^{(j)}, \pi^{(j)}) | j = 1 \dots N\}$. Each sample consists of an element s which represents the state of an object and a corresponding discrete sampling probability π . The evolution of the sample set is described by first propagating each sample according to a system model. Secondly, each element of the set is weighted in terms of the measurements and finally, a particular sample is drawn with replacement, by choosing it with probability π .

In our application it is necessary to track *multiple* objects simultaneously and to handle *newly appearing* objects. For that purpose, we use a single CONDENSATION tracker for multiple objects such that several object states are represented simultaneously with one sample distribution. Furthermore, we apply an initialization at every iteration step to incorporate newly appearing objects into the tracking process. The programming details of the extended CONDENSATION algorithm for one iteration step are given in Fig. 2.

The measurements for our application are obtained by subtracting two successive images. From the resultant motion blobs we extract the central points described by the x and y image coordinates. Accordingly, we define the state of an object blob at time t by

$$s_t = (x_t, y_t, u_t, v_t) \tag{1}$$

where u and v describe the velocities in the corresponding coordinate directions. The propagation of each sample is then given as

$$\begin{aligned} x_t &= x_{t-1} + u_{t-1} \Delta t + n_{t-1}^{(x)} \\ y_t &= y_{t-1} + v_{t-1} \Delta t + n_{t-1}^{(y)} \\ u_t &= u_{t-1} + n_{t-1}^{(u)} \\ v_t &= v_{t-1} + n_{t-1}^{(v)} \end{aligned} \tag{2}$$

where $n = (n^{(x)}, n^{(y)}, n^{(u)}, n^{(v)})$ is a vector of random variables with a known distribution and Δt is the time interval between two consecutive measurements. Currently, a first order system model is applied for computational reasons, but we plan to expand this model to second order

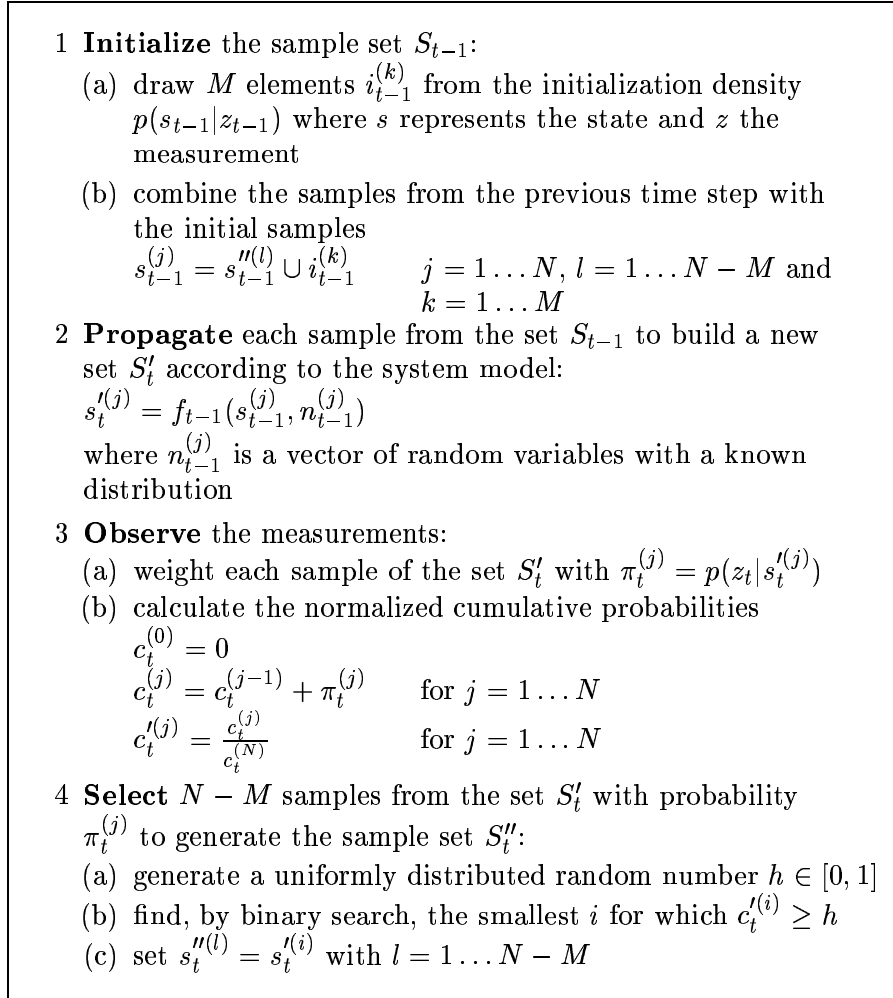


Figure 2. An iteration step of the extended CONDENSATION algorithm.

by additionally incorporating the acceleration into the object state. We expect that for example the queuing in front of the cash machines can then be modeled more precisely. By keeping the history of each sample a trajectory is formed as a sequence of states.

An example where two persons are tracked simultaneously with a single sample distribution is shown in Fig. 3. A bank lobby is observed where three cash machines are located on the left side. The bright grey spots show the current sample distribution while in darker grey the corresponding trajectories are illustrated. The size of the sample set



Figure 3. The sample distribution is shown as bright grey spots while in darker grey the corresponding motion trajectories are illustrated.

is dependent on the number of objects, for our application we choose $N = 400$ and $M = 40$. The computational cost for the tracking is 0.09 seconds on a PC with a 450 MHz Pentium II processor.

3. Learning

The learning phase starts by building prototypes for the trajectories of the tracked object states. The trajectories received from the tracking process are typically short-lived, partially overlapping in space and time and are possibly produced by different objects. Consequently, we cluster these trajectories into long-lived prototype curves each corresponding to a characteristic movement.

The clustering is done in the following steps:

- 1 All trajectories that are sufficiently long are added as new prototype curves.
- 2 Prototype curves that are overlapping and sufficiently close are merged into a new curve.

In each iteration step the clustering method is applied to all new prototype curves as well as to the already existing curves. As a result we obtain a set of prototype curves of the monitored environment. A related clustering method, where feature points are grouped to form object trajectories is described in [9].

Each prototype curve describes the spatio-temporal path of an object and is modeled by a sequence of nodes

$$b = (x, y, u, v, w) \quad (3)$$

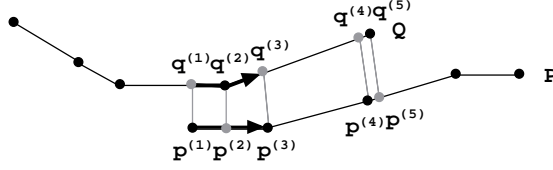


Figure 4. Finding point pairs with equal distance in arc length to the initial points $p^{(1)}$ and $q^{(1)}$.

where x, y describe the spatial localization and u, v the motion. The parameter w indicates the specific weight of this node and is increased with every trajectory that contributes to the node. This weight is used when two curves are merged, to interpolate the nodes of the resulting curve. The initial value is given by $w = 1$.

For the comparison of two prototype curves it is necessary to analyze their distance and their spatial overlap. Let us assume that we have two prototype curves P and Q . We first look at the starting point of curve P and find the closest point on curve Q . From these *initial points*, we now find more corresponding point pairs on the two curves. A point pair consists of a node on one curve and a corresponding point on the other curve. Such pairs have always the same distance in arc length to the initial points (see Fig. 4). We continue to find pairs until we reach the end of one curve.

Two prototype curves are merged if the mean distance of the point pairs

$$d = \frac{1}{K} \sum_{k=1}^K (x_q^{(k)} - x_p^{(k)})^2 + (y_q^{(k)} - y_p^{(k)})^2 + \lambda ((u_q^{(k)} - u_p^{(k)})^2 + (v_q^{(k)} - v_p^{(k)})^2) \quad (4)$$

is sufficiently small and if they have a large enough overlap given by the distance in arc length between the first and the last overlapping point found on the curves. The factor λ is used to normalize the velocities and the parameter K indicates the number of point pairs. The merged points on curve R are then calculated by a weighted interpolation of the points on curve P and Q in the overlapping area (see Fig. 5).

In order for the merged prototype curve R not to acquire too many nodes, we check for nearby nodes with low curvature values that can be merged into one node without significantly changing the shape of the prototype curve. Furthermore, as relatively unlikely trajectories can

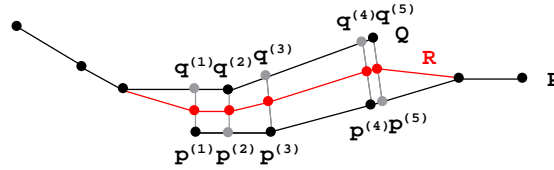


Figure 5. Generate an interpolated prototype curve R from the curves P and Q .

also provoke prototype curves we replace weak curves by more dominant ones when a maximal number of prototype curves has been reached.

In Fig. 6 a new prototype curve is formed by the person crossing the bank lobby. In grey we see as before the tracking results while in white the prototype curves are shown. The computation power for the learning is dependent on the number of prototype curves. By the chosen maximal number of 30 curves the processing of the learning can temporarily reach up to 1.0 second.

4. Recognition

In the real-time recognition mode movements are identified on the basis of the prototype curves. In [2] the authors describe how the basic CONDENSATION algorithm [5] was successfully used to solve different recognition tasks. We adapted the idea of this CONDENSATION-based Trajectory Recognition to our extended CONDENSATION method specified in Fig. 2.



Figure 6. The development of a new prototype curve caused by the person crossing the bank lobby is illustrated in white.

Instead of the segmentation results, the output of the tracking is now used as the measurement input. However, we usually use less samples for the motion recognition than for the tracking. Therefore, the tracking results are first clustered according to their distance. For each cluster, a mean sample is calculated that consists of a position and a velocity. This sample is then used as measurement input z for the motion recognition.

As we want to detect if and where our objects are positioned on one of the learned prototype curves, the state of each sample is given as

$$s_t = (\mu_t, \phi_t) \quad (5)$$

where μ indicates the prototype curve and ϕ specifies the current position on this curve.

The object is assumed to move along the prototype curve, so the state evolution of each sample is described by

$$\begin{aligned} \mu_t &= \mu_{t-1} \\ \phi_t &= \phi_{t-1} + \Delta t + n_{t-1} \end{aligned} \quad (6)$$

where n is a random variable with a known distribution and Δt is the time interval between two consecutive measurements.

In the first step of the extended CONDENSATION algorithm, we initialize new samples from the measurements. Such a new sample is specified by finding the closest point on a prototype curve μ and by calculating the corresponding temporal position ϕ . If the distance to the closest curve is too far, we assign this sample to a *residual curve* with $\mu = 0$. So, to this virtual curve are assigned all movements at unusual places.

In Fig. 7 the recognition results for an example sequence are indicated. The tracking results are shown in grey while in white the learned set of prototype curves are illustrated. From the sample distribution of the recognition we have recovered the corresponding object positions which are shown as black spots on the prototype curves. The number of samples required for the recognition is dependent on the number of prototype curves. For our application we choose $N = 300$ and $M = 30$. The recognition can be performed within 0.04 seconds.

5. Experimental Results

We give three examples, one for an unlikely object position and two for unusual movements. For the recognition of movements at unusual places we consider the samples of the residual prototype curve. An alert is triggered when a minimal percentage of samples belong to the residual curve. In the left image of Fig. 8 such a situation is illustrated where two unauthorized people improperly occupy the bank lobby using it as

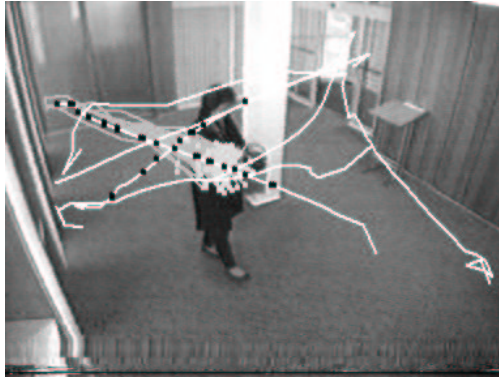


Figure 7. Recovered object positions of the recognition samples are shown in black on a learned set of prototype curves.

a dormitory. The corresponding alert seen by the security staff is shown in the right image. Of course, this case could even be detected by a static analysis that does not use tracking, however it is convenient that the result is also provided by our framework.

When a person stays in a common place but his behavior is not equivalent to one of the learned prototype curves, e.g. because he has an unusually high speed, we call this an unusual motion event. In that case



Figure 8. Detection of an unusual motion event caused by the two unauthorized people which improperly occupy the bank lobby using it as a dormitory. In the left image, the current situation of the surveillance framework is illustrated while in the right image the alert seen by the security staff is shown.



Figure 9. Detection of an unusual motion event caused by rapid movements of an act of vandalism.

the tracked position of the person will be too far away from the expected position on the prototype curve. This results in a lower weight for the sample, so that we can detect this event by analyzing the proportion of the recognition samples assigned to one prototype curve before and after the selection step. In Fig. 9 such an event caused by an act of vandalism is illustrated.

False alarms produced for example by playing children (see Fig. 10) can not be excluded as the system is unable to differentiate between not learned or unusual behavior patterns on the one hand and because some distinctions between normal and abnormal behaviors simply require too high a level of semantic interpretation. Of course, in the end it is the operator who will assess the situation.

6. Conclusion

We have described a self-learning framework consisting of a tracking, a learning and a recognition part for detecting unusual motion events in real-time. Such a surveillance system is primarily intended to be used in security-sensitive areas such as airports, railway stations, banks or public building lobbies. The tracking approach is solved with an extension of the CONDENSATION algorithm which can deal with an arbitrary, changing number of objects. Motion trajectories are formed by keeping the history of the tracked object states. Using the clustering we can generate long-lived prototype curves by grouping the trajectories. This processing step is generally the most time-consuming part as the compu-



Figure 10. Detection of an unusual motion event caused by the child running around the pillar.

tation power increases with the number of prototype curves. But since the clustering is only done during the training phase, it does not add to the computational cost during normal operation. For the recognition task the tracking results are then assigned to the prototype curves. The idea of the CONDENSATION-based Trajectory Recognition is applied to the extended CONDENSATION tracker so that the recognition and tracking are both based on the same probabilistic framework. Finally, the detection of unusual movements is done by analyzing the sample distribution.

If we have more than one person in the scene, the identification and localization of the object which is responsible for the alert is not directly provided by the sample distribution. On the other hand, we know which prototype curves are active and where in these curves the samples are located.

To improve the accuracy, tracking could be considered across multiple cameras. In addition, the prototype curves which represent the usual events could evolve. Motion patterns could possibly be included or removed from the set of prototype curves over time, for example when the bank closes its counter at 5pm, the corresponding curve could be deleted.

Acknowledgments

This research was partially supported by the Swiss Commission for Technology and Innovation (KTI). We would also like to thank our in-

dustrial partner ASCOM Systec AG for providing us with a video surveillance system including their software [6].

References

- [1] J. K. Aggarwal and Q. Cai. Human Motion Analysis: A Review. *Journal of Computer Vision and Image Understanding*, Vol. 73, No. 3, March, pp. 428-440, 1999.
- [2] M. J. Black and A. D. Jepson. A Probabilistic Framework for Matching Temporal Trajectories: Condensation-Based Recognition of Gestures and Expressions. *5th European Conference on Computer Vision*, Vol. 1, pp. 909-924, 1998.
- [3] Ch. Bregler. Learning and Recognizing Human Dynamics in Video Sequences. *Computer Vision and Pattern Recognition*, pp. 568-574, 1997.
- [4] J. W. Davis and A. F. Bobick. The Representation and Recognition of Human Movement Using Temporal Templates. *Computer Vision and Pattern Recognition*, pp. 928-934, 1997.
- [5] M. Isard and A. Blake. Contour Tracking by Stochastic Propagation of Conditional Density. *4th European Conference on Computer Vision*, Vol. 1, pp. 343-356, 1996.
- [6] R. Mattone, A. Glaeser, and B. Bumann. SEDOR: A Self-learning Event Detect-OR for video-based surveillance. *10th International Conference on Image Analysis and Processing*, 1999.
- [7] E. B. Meier and F. Ade. Using the Condensation Algorithm to Implement Tracking for Mobile Robots. *3rd European Workshop on Advanced Mobile Robots*, pp. 73-80, 1999.
- [8] R. Polana and R. Nelson. Low Level Recognition of Human Motion (Or How to Get Your Man Without Finding his Body Parts). *IEEE Computer Society Workshop on Motion of Non-Rigid and Articulated Objects*, pp. 77-82, 1994.
- [9] J. Segen and S. Pingali. A Camera-Based System for Tracking People in Real Time. *13th International Conference on Pattern Recognition*, Vol. 3, Track C, pp. 63-67, 1996.
- [10] T. Starner and A. Pentland. Visual Recognition of American Sign Language Using Hidden Markov Models. *International Workshop on Automatic Face and Gesture Recognition*, pp. 189-194, 1995.
- [11] J. Yamato, J. Ohya, and K. Ishii. Recognizing Human Action in Time-Sequential Images using Hidden Markov Model. *Computer Vision and Pattern Recognition*, pp. 379-385, 1992.