

# Tracking of Individuals in Very Long Video Sequences

P. Fihl<sup>†</sup>, R. Corlin<sup>†</sup>, S. Park<sup>‡</sup>, T.B. Moeslund<sup>†</sup>, and M.M. Trivedi<sup>‡</sup>

<sup>†</sup> Laboratory of Computer Vision and Media Technology  
Aalborg University, Denmark.  
Email: pfa@cvmt.dk , tbm@cvmt.dk

<sup>‡</sup> Computer Vision and Robotics Research Laboratory  
The University of California, San Diego, USA  
Email: parks@ucsd.edu , mtrivedi@ucsd.edu

**Abstract.** In this paper we present an approach for automatically detecting and tracking humans in very long video sequences. The detection is based on background subtraction using a multi-mode Codeword method. We enhance this method both in terms of representation and in terms of automatically updating the background allowing for handling gradual and rapid changes. Tracking is conducted by building appearance-based models and matching these over time. Tests show promising detection and tracking results in a ten hour video sequence.

## 1 Introduction

Visual analysis of humans has a number of applications ranging from automatic surveillance systems to extracting pose parameters for realistically character animation in movies. Automatic surveillance systems observe humans at a distance and in various environments. Furthermore, these systems should, as opposed to e.g., motion capture systems, work completely autonomous and for long periods of time.

The foundation of many surveillance systems is a good detection and tracking of humans in a video sequence. These issues have received much attention in the last decade or so [1–4]. The detection problem (aka the figure-ground segmentation problem) is typically done using shape, motion, depth, background detection, or appearance [5–10]. When the scene of interest contains individuals that are allowed to occlude each other, the tracking of individuals is inherently difficult and using an appearance-based model for each individual is often the preferred approach.

In this work we consider situations where occlusion can occur and we therefore follow the appearance-based approach. Our aim is continuous detection and tracking over very long periods of time as opposed to other approaches mostly evaluated on short video sequences. Concretely, we first develop and use an advanced background subtraction algorithm in order to handle the figure-ground segmentation problem. The result is a silhouette of each individual in the scene - section 2. Next we use an appearance-based model to represent each individual. A good model is obtained by using some of the results from research on modeling interacting people. We then present a scheme for matching appearance-based models over time - section 3. In section 4 we present tracking results of several hours of continuous video and in section 5 a conclusion is given.

## 2 Figure-Ground Segmentation

The first step in our tracking algorithm is to separate the foreground (humans) from the background, i.e., the figure-ground segmentation problem. We do this using a background subtracting approach inspired by [11].

### 2.1 Background Representation

We apply the Codeword approach of [11], which has shown to perform better than Gaussian mixture models [9] and other well-known methods [12, 1] in terms of both speed and sensitivity [13].

The representation of a background pixel in the codeword approach [11] is based on the representation from [14]. Here color and intensity are represented independently and a background pixel is represented as a vector in the RGB-cube,  $\mu$ . The distance, in terms of color,  $\rho$ , from a new pixel,  $x$ , to the background model is measured as the perpendicular distance to the vector. The difference in intensity is measured along the vector and denoted,  $h$ , see figure 1. In the work by [11] a cylinder centered around the

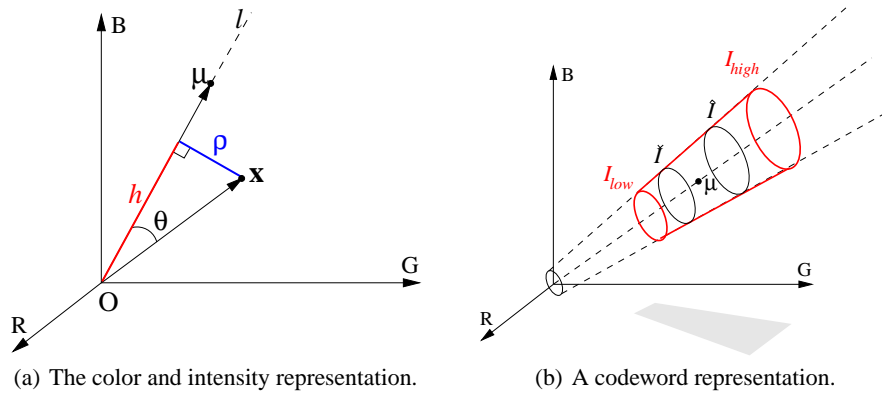


Fig. 1. Illustration of the representations used in the background subtraction.

vector represents a codeword for this particle pixel and all pixel values inside the cylinder are classified as background. During a training phase a number of codewords are learned for each pixel and together these are denoted the codebook for this particular pixel. This is a fast and robust approach due to the multi-mode nature of the representation. However, since all color vectors go through the origin of the color-cube a more correct representation is to form a truncated cone around each learned background vector. In this way the different colors inside the codeword actually corresponds to the same colors with different intensity. In figure 1 our representation of a codeword is shown, where  $\tilde{I}$  and  $\hat{I}$  are the minimum and maximum values found during training, and  $I_{low}$  and  $I_{high}$  are where the cone is truncated in order to allow additional values, e.g., due to shadows [15].

## 2.2 Background Initialization

A key issue in successful background subtraction is to learn a good model of the background during an initialization phase. If no moving objects are present in the scene this is obviously easier. But a more general approach is to allow for moving objects. If a pixel is covered by moving objects in less than 50% of the learning period then a median filter can be applied [1]. A different method is first to divide the training sequence into temporal subintervals with similar values - assumed to belong to the background and then find the subinterval with the minimum average motion and only use these pixels for model initialization [16, 17]. In this work we follow the approach by [11], which also works along these lines of reasoning.

During the initialization phase each new pixel is either assigned to an already existing codeword (which is updated accordingly) or a new codeword is created. This will produce codewords for non-background pixels, but these are handled by temporal filtering using the so-called *maximum negative run-length* (MNRL). This measures the longest time interval where no pixel values have been assigned to the codeword. After the training period all codewords with a too large maximum negative run-length will be removed from the background model, i.e., this process allows for moving objects during the training period, see [11] for details.

## 2.3 Background Updating

Using multiple codewords for each pixel allows modeling of very dynamic scenes, but only the variation that is present in the training period will be modeled by the codebook background method as described in [11]<sup>1</sup>. For the background subtraction to work for several hours it is necessary also to handle the changes in the background that occurs after the initialization phase. Two different types of changes need to be handled.

- **Gradual changes** do not change the appearance of the background much from one frame to the next. The accumulated change over time can however be large, e.g., the effect of the changing position of the sun during a day.
- **Rapid changes** cause significant changes in the background from one frame to the next. Background objects that are moved or significant changes in the motion patterns of vegetation caused by gusting winds will for example cause rapid changes.

To handle the gradual changes we apply a simple continuous model:

$$\mu_{t+1} = (1 - \alpha) \cdot \mu_t + \alpha \cdot x_t \quad , \quad 0 \leq \alpha \leq 1 \quad (1)$$

where  $\mu_t$  is the codeword,  $\mu_{t+1}$  is the updated codeword, and  $x_t$  is the current pixel value. Based on experiences in dynamic outdoor environments  $\alpha$  is typical 0.05 to 0.15.

Only the activated codeword in each codebook is updated with this process. Consider the situation where a pixel is sometimes occupied by a green tree branch and sometimes occupied by a red wall, e.g., due to wind. Only the codeword modeling the

<sup>1</sup> It should be noted that at the time when this work was finished [15] the authors of [11] published a more advanced version of their codebook algorithm [18], which is somewhat similar to our background subtracting approach.

branch should be updated when the branch occupies the pixel, since the color of surfaces with different orientation, texture, and color change in different ways with changing illumination. The change of the codeword that models the wall cannot be found from the color change of the branch.

Pixels falsely classified as background will lead to codewords that are updated to model foreground instead of background. Therefore only pixels describing stable background will be updated. Stable background is defined as a pixel that has been classified as background for the last  $j$  frames, where  $j$  typically is 10-15. As for  $\alpha$  this interval is based on experiences in dynamic outdoor environments. The performance of the background updating is not very sensitive to neither  $j$  nor  $\alpha$  within these intervals.

Equation 1 cannot handle rapid changes and therefore new codewords are learned during run-time. For example, in a situation when a car is parked in the scene it is treated (correctly) as a foreground object. However, while the car is parked we want it to become part of the background so we can detect new foreground objects that move in front of the car. We do this in the following way. Each time a pixel is classified as foreground we create a new codeword, denoted a *training codeword*. If this codeword has a small MNRL within the next  $n$  frames we conclude that this codeword does indeed represent a new background and we make it a *temporary codeword*<sup>2</sup>. If the MNRL is big we delete this training codeword. Temporary codewords that become inactive (measured by their MNRL) are deleted, e.g., if the parked car starts to move again. So in each frame a pixel value is matched against the codewords from the "real" background (learned during initialization), the temporary codewords and the training codewords, in that order. If a match is found, the respective codeword is updated using equation 1.

## 2.4 Bounding Box Representation

After an image is processed by the background subtraction process we remove noise (false positives) using a median filter. Sometimes false negatives result in the silhouette being split into smaller blobs. We therefore investigate the size and proximity of the bounding boxes of each blob and try to merge them into bounding boxes each representing one human [15]. The silhouettes in the merged bounding boxes are compared to a simple body model to distinguish humans from small blobs of noise. The silhouette of a person can roughly be described by an ellipse, and our body model defines limits for the ratio between the major and minor axes of the ellipse, the slope of the major axis, the fidelity between the ellipse and silhouette, and the area of the silhouette. A silhouette complying with all limits of the body model will be accepted as a person otherwise it is considered as noise. To avoid the problem with a person producing multiple enter/exit results when the area of the silhouette is close to the limit of the body model we utilize a hysteresis threshold [15]. To get a correct initialization of our appearance model we need to ensure that a person is completely inside the field-of-view before we accept the new person and we therefore introduce an entry zone around the image border [15]. To summaries, after the above processes we are left with a number of bounding boxes each containing the silhouette of one person.

<sup>2</sup> Pixels that belong to a training codeword are classified as foreground whereas pixels belonging to a temporary codeword are classified as background.

### 3 Tracking

#### 3.1 Representation

We model the appearance of a person by dividing it into two regions which are modeled separately: the upper body (not including the head) and the lower body [4]. Due to the nature of the method the regions are not simply found as a ratio of the bounding box as seen in, e.g., [10, 19, 20] but are found by dividing the body into a set of blobs that are similar in color and spatially connected, and then grouping these blobs into an upper body and a lower body using a ratio of the bounding box as a guideline.

The blobs are initialized by labeling pixels with similar color in the foreground to the same class. To do this the Expectation Maximization (EM) algorithm is used to first learn the gaussian distributions of color classes in the foreground followed by a classification of the pixels to these classes. The labeling of pixels to color classes is carried out by a Maximum Likelihood estimation.

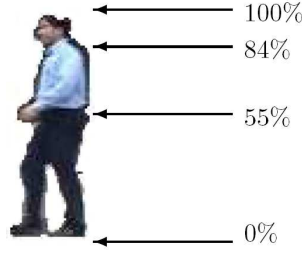
When the pixels have been classified in this way the classes are not necessarily spatially connected, e.g. the dark hair of a person could be assigned to the same class as a dark pair of shoes or simply a checkered shirt could consists of many spatially disconnected classes of the same color. To make sure that each blob represents a region of connected pixels a relabeling is done by making a connected component analysis. This is done by finding the contours of all disconnected regions for the pixels in each color class separately and giving all pixels within the boundary of these contours a unique label.

To avoid over-segmentation of the foreground similar blobs are merged. Blob similarity is evaluated using four criteria [15] and two blobs are merged if either the first criterion is true or if the three remaining criteria are all true: 1) a blob is completely surrounded by another blob, 2) two blobs are adjacent, 3) two blobs share a large border, 4) two blobs are similar in terms of color. By use of these criteria the number of blobs is reduced considerably and a set of blobs that are expected to represent relatively stable parts of the foreground is obtained.

To define the merged blobs as either upper body or lower body we use ratios of the bounding box as a guideline. The bounding box is divided into three regions as shown in figure 2 [15]. All blobs with centroid in the range from 0 to 0.55 times the height of the bounding box will define the lower body and blobs with centroid from 0.55 to 0.84 times the height of the bounding box will define the upper body. This way the border between upper and lower body will not be a straight line but follow the borders of dissimilar blobs. The final features representing a person (or silhouette) are listed in the feature vector  $\mathbf{m}$ :

$$\mathbf{m} = [\mu_x, \mu_y, \mu_{H\_upper}, \mu_{S\_upper}, \mu_{H\_lower}, \mu_{S\_lower}]^T \quad (2)$$

where  $\mu_x$  and  $\mu_y$  are the mean position or center of mass of the given person. The last four parameters represent the mean color of the upper and lower body respectively in terms of hue and saturation [15].



**Fig. 2.** Ranges for partitioning the silhouette of a person into head, upper body and lower body in relation to the height of the person. Note that the hand is assigned to the upper body even though its located below the 0.55 line. This is due to the complex merging process described above.

### 3.2 Matching

The matching of identities is performed by calculating the dissimilarity in terms of the Mahalanobis distance<sup>3</sup> between all extracted silhouettes in the current frame, indexed by  $i$ , and all known identities that have been tracked to this frame, indexed by  $j$ :

$$\Delta_{ij} = (\mathbf{m}_i - \mathbf{m}_j)^T (\mathbf{\Pi}_i + \mathbf{\Pi}_j)^{-1} (\mathbf{m}_i - \mathbf{m}_j) \quad (3)$$

where  $\mathbf{\Pi}_i$  represents the between class covariance of all body models in the current frame and  $\mathbf{\Pi}_j$  the between class covariance for the identities that the body models are being matched to. These are pooled in order to compensate for the differences in the variations of the body models and the identities they are being matched to. To simplify the calculations only the diagonal of these covariances have been used. The between class variances have been calculated as follows:

$$\mathbf{\Pi} = \frac{1}{k} \sum_k (\boldsymbol{\mu}_k - \boldsymbol{\mu}_{all})(\boldsymbol{\mu}_k - \boldsymbol{\mu}_{all})^T \quad (4)$$

where  $\boldsymbol{\mu}_k$  is the mean value of the  $k^{th}$  body model/identity and  $\boldsymbol{\mu}_{all}$  is the mean of all body models/identities present in the given region [15].

## 4 Results

The presented system has been tested at two levels. The figure-ground segmentation have been tested to show its performance on very long video sequences (10 hours). The bounding box representation and the tracking have been tested on the output of the figure-ground segmentation to show the system's overall capability to track people.

<sup>3</sup> Note that a weight is assigned to each feature in order to balance the positional features and the appearance features. I.e., 0.3 for the positional features and 0.1 for the appearance features. The weights are omitted from the equation for clarity.

#### 4.1 Test of Figure-Ground Segmentation

The video used for the test is a 10 hour video with a frame rate of 30 frames per second. The video is captured from 9.15 AM to 7.15 PM. and the scene contains several challenging situations in the context of figure-ground segmentation, i.e. illumination changes, non-static background, shadows, puddles, and foreground camouflage (see figure 4).

To calculate the false rejection rate (percentage of foreground pixels falsely classified as background) and the false acceptance rate (percentage of background pixels falsely classified as foreground) a set of 93 frames containing foreground objects (people) have been sampled from the whole time span. The images used are the binary foreground mask obtained from the figure-ground segmentation filtered with the median filter. For each of the sampled frames the foreground region was marked by hand (based on the original input frame) and used as the ground truth.

The calculation of the false acceptance rate is based on the above mentioned frames in addition to a set of frames containing only background. The frames containing only background were added since only a limited number of frames actually contain people, and the additional frames would give a more representative result for the whole video. The frames containing only background were sampled every 1000 frames. When sampling every 1000 frames some of the frames contained people, but these frames were discarded from the set giving a total of 971 frames.

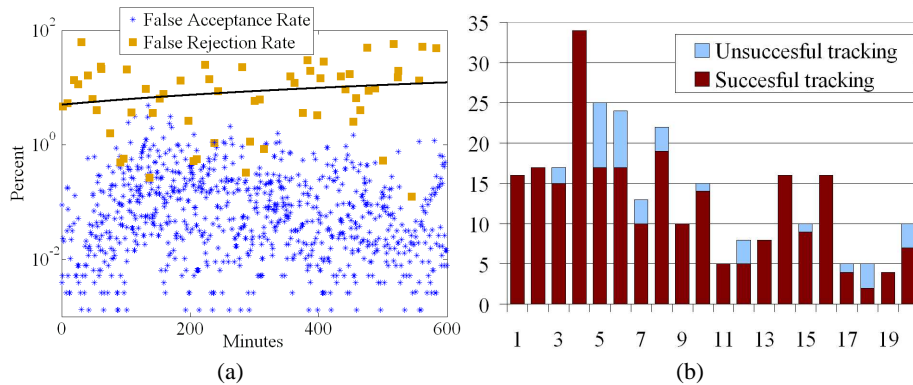
Figure 3(a) shows FRR (false rejection rate) and FAR (false acceptance rate) in percent as a function of time. FRR is in the range [0%;62.4%] with mean value 8.45% and standard deviation 13.43. The black line represents the best linear fit of the FRR samples in the least-squares sense and shows a slightly increasing tendency in FRR. The increase does however not mean that the performance of the background subtraction decreases over time. The increase in FRR is caused by the low overall illumination level at the end of the day which causes the problem of foreground camouflage to increase. The FAR is in the range [0%;4.9%] with mean value 0.14% and standard deviation 0.33. The FAR shows a slightly decreasing tendency over time. The mean FAR of 0.14% shows that the background subtraction in general effectively models the background and adapts to the changes present in the test video. The performance of the background subtraction in terms of FAR is not dependent on how many hours it has been running, but on the type of changes that happens in the scene, and the background subtraction automatically recovers from changes that are not directly handled by the model.

#### 4.2 Test of Tracking

The 10 hours of test video contains 267 persons that move through the scene<sup>4</sup>. The tracking result of each person has been evaluated to see if the system successfully identifies each person and tracks the person.

The system identifies and tracks 247 persons successfully. 20 persons are not tracked correctly and the system further identifies 15 blobs of noise as persons which gives a

<sup>4</sup> People that never enter the field-of-view completely or people that move through the scene in groups are not included in this number or the test.



**Fig. 3.** Results from the 10 hour test video. (a): The FAR and FRR in percent as a function of time. The black line is the best linear fit of the FRR samples in the least-squares sense. Note that the y-axis is logarithmic. (b): The number of successful and unsuccessful tracks. Each column covers 30 minutes.

total of 35 errors. The overall successful tracking rate yields 86.9%. Figure 3(b) shows the tracking result for each 30 minutes interval.

Figure 3(b) indicates that the performance of the tracking is independent of the number of hours the system has been running. The number of tracking errors is most remarkable in the 3rd and 9th hour. This is due to rapid background variations and low overall illumination, respectively, and not because the system has been running for several hours. Figure 4 shows examples of tracking results (both successful and unsuccessful) when multiple people are in the scene.

The errors that occur during tracking can be explained by either *noisy foreground segmentation* (24 errors) or *insufficient tracking or appearance model* (11 errors). The errors originating from noisy foreground segmentation are mainly due to moving vegetation and strong shadows that gets identified as humans. The errors originating from insufficient tracking or appearance model often occur when two persons move close past each other (resulting in switching identities) or when strong shadows makes the silhouette of a person non-elliptic. The effect of both types of errors can possibly be reduced by including temporal information.

## 5 Conclusion

In this paper we have presented a system to do figure-ground segmentation and tracking of people in an outdoor scene continuously for several hours.

The system has been thoroughly tested on 10 hours of continuous video containing multiple difficult situations. The system was able to automatically update the background model allowing for tracking people with a success rate of 86.9%, and we believe that this number can be increased with relatively simple improvements to the tracking algorithm. To our knowledge this is the first system to present results on continuous tracking in very long video sequences.



(a) 9.25 AM. The box shows an example of a region with foreground camouflage. Notice the puddles on the ground.



(b) The two persons are tracked correctly even though they move near each other.



(c) 3.05 PM. The boxes show examples of regions with strong shadows. Furthermore, other shadows move rapidly because of the wind.



(d) The three people are tracked correctly until the two persons to the left get too close to each other.



(e) 7.05 PM. The overall illumination level changes significantly from morning to the afternoon and again from the afternoon to the evening



(f) The tracks of the two persons to the left are swapped.

**Fig. 4.** Left: Examples of the changes in the scene during the 10 hours test video. Right: Tracking results.

## References

1. Haritaoglu, I., Harwood, D., Davis, L.: W4: Real-Time Surveillance of People and Their Activities. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **22** (2000)
2. McKenna, S., Jabri, S., Duric, Z., Wechsler, H.: Tracking Interacting People. In: The fourth International Conference on Automatic Face and Gesture Recognition, Grenoble, France (2000)
3. Zhao, T., Nevatia, R.: Tracking Multiple Humans in Crowded Environments. In: *Computer Vision and Pattern Recognition*, Washington DC, USA (2004)
4. Park, S., Aggarwal, J.: Simultaneous tracking of multiple body parts of interacting persons. *Computer Vision and Image Understanding* **102** (2006)
5. Leibe, B., Seemann, E., Schiele, B.: Pedestrian Detection in Crowded Scenes. In: *Computer Vision and Pattern Recognition*, San Diego, CA, USA (2005)
6. Viola, P., Jones, M., Snow, D.: Detecting Pedestrians Using Patterns of Motion and Appearance. *International Journal of Computer Vision* **63** (2005)
7. Sidenbladh, H.: Detecting Human Motion with Support Vector Machines. In: *International Conference on Pattern Recognition*, Cambridge, UK (2004)
8. Hayashi, K., Hashimoto, M., Sumi, K., Sasakawa, K.: Multiple-Person Tracker with a Fixed Slanting Stereo Camera. In: *International Conference on Automatic Face and Gesture Recognition*, Seoul, Korea (2004)
9. Stauffer, C., Grimson, W.: Adaptive Background Mixture Models for Real-Time Tracking. In: *Computer Vision and Pattern Recognition*, Santa Barbara, CA, USA (1998)
10. Roth, D., Doubek, P., Gool, L.: Bayesian Pixel Classification for Human Tracking. In: *IEEE Workshop on Motion and Video Computing (MOTION'05)*, Breckenridge, Colorado (2005)
11. Kim, K., Chalidabhongse, T.H., Harwood, D., Davis, L.: Background modeling and subtraction by codebook construction. In: *IEEE International Conference on Image Processing (ICIP)*. (2004)
12. Elgammal, A., Harwood, D., Davis, L.: Non-Parametric Model for Background Subtraction. In: *European Conference on Computer Vision*, Dublin, Ireland (2000)
13. Chalidabhongse, T., Kim, K., Harwood, D., Davis, L.: A Perturbation Method for Evaluating Background Subtraction Algorithms. In: *Int. Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, Beijing, China (2005)
14. Horprasert, T., Harwood, D., Davis, L.: A Statistical Approach for Real-time Robust Background Subtraction and Shadow Detection. In: *IEEE ICCV'99 FRAME-RATE WORKSHOP*, Corfu, Greece (1999)
15. Andersen, P., Corlin, R.: Tracking of Interacting People and Their Body Parts for Outdoor Surveillance. Master's thesis, Laboratory of Computer Vision and Media Technology, Aalborg University, Denmark (2005)
16. Gutchess, D., Trajkovic, M., Solal, E., Lyons, D., Jain, A.: A Background Model Initialization Algorithm for Video Surveillance. In: *International Conference on Computer Vision*, Vancouver, Canada (2001)
17. Wang, H., Suter, D.: Background Initialization with a New Robust Statistical Approach. In: *Int. Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, Beijing, China (2005)
18. Kim, K., Chalidabhongse, T., Harwood, D., Davis, L.: Real-Time Foreground-Background Segmentation using Codebook Model. *Real-Time Imaging* **11** (2005)
19. Yang, C., Duraiswami, R., Davis, L.: Fast Multiple Object Tracking via a Hierarchical Particle Filter. In: *International Conference on Computer Vision*, Beijing, China (2005)
20. Mittal, A., Davis, L.: M2Tracker: A Multi-View Approach to Segmenting and Tracking People in a Cluttered Scene. *International Journal of Computer Vision* **51** (2003) 189–203