

LIKELIHOOD-BASED OBJECT DETECTION AND OBJECT TRACKING USING COLOR HISTOGRAMS AND EM

Paul Withagen^{1,2}, Klamer Schutte¹ and Frans Groen²

¹ TNO Physics and Electronics Laboratory, P.O. Box 96864, 2509 JG The Hague, The Netherlands

² IAS group, University of Amsterdam, Kruislaan 403, 1098 SJ Amsterdam, The Netherlands

Email: Withagen@fel.tno.nl

ABSTRACT

The topic of this paper is the integration of Expectation Maximization (EM) background modeling and template matching using color histograms as templates to improve person tracking for surveillance applications.

The tracked objects are humans, which are not rigid bodies. As such shape deformations of the objects must be allowed. For each frame, the decision has to be made which pixels belong to an object, and which do not.

The integration of detection and tracking is done using a likelihood-based framework. This way the classification of pixels between background and object can be based on comparing likelihoods rather than separate thresholds. A demonstration of the proposed algorithm will be given.

1. INTRODUCTION

A surveillance application ([1, 2, 3]) usually consists of some sort of moving object detection, object tracking, and higher order processing, like the detection of persons entering a prohibited area, face recognition for identification, or gesture recognition to determine what a person is doing. This paper concentrates on the two initial steps, moving object detection and object tracking.

Often, background modeling is used to detect moving objects. Toyama compares a number of these algorithms in [2]. A popular algorithm is Expectation Maximization (EM, [4]). Because of increasing speed of computers, it is now feasible to make real-time implementations (using the online version of this algorithm described in [5]). The algorithm models the background by creating for each pixel a model of the appearance of colors over time. This model is described by a mixture of Gaussian kernels. For every new frame, the new color value of each pixel is compared to the mixture model, and this way the likelihood that this pixel is background can be calculated. Because this approach is adaptive, initialization with an empty scene is not necessary, which is a considerable advantage over non-adaptive algorithms.

By thresholding the likelihood image produced by the background estimation, new foreground objects can be detected. The detected objects need to be tracked. This is possible by using a Kalman filter [6] on their positions, letting the Kalman filter solve the correspondence problem between the frames. However, humans easily change direction and speed, and tend to interact with each other. That is why additional features are necessary to keep the different objects separated. Two frequently used features are shape and color. We will use the color histogram of the object as feature. The use of color histograms for object location was introduced in [7] and is frequently used to find objects in image achieves.

Histogram-based object recognition ([7]) has the advantage that it is invariant to limited deformations of the object itself. It also allows for an efficient algorithm. The histogram-based “template matching” algorithm (in the remainder of this paper called histogram matching) does not use least squares to compare the image and the template, but compares the two histograms. It can be implemented very efficient because no spatial information is incorporated in the model. So when the region of interest is shifted over the image, for each new shift position, only the pixels which “shift in” and those that “shift out” need to be processed to find the new match-error at that position.

Section 2 shows how the tracking of objects results in a likelihood whether a pixel belongs to a certain object. Using these likelihoods, and the likelihood for background, all pixels are classified without the use of thresholds. The specifics of our implementation will be explained in section 3. In section 4 we will describe some experiments performed with the algorithm and finally, conclusions will be given in section 5.

2. LIKELIHOOD INTEGRATION

This section describes the idea of likelihood integration. For an overview of the entire system, see section 3.

As a result of the background modeling, for each pixel we predict how likely it is that the pixel belongs to the back-

ground. Provided that we also have a likelihood for the pixel being object, no thresholds will be necessary in the classification of the pixels between background and any of the objects: we assign the pixels to the class with the highest likelihood.

Looking at the physics of the objects we want to track, we notice that between two frames, objects can move only a certain distance, and also their shape can modify slightly. Thus, the object-core (by the word object-core we mean the entire object except for a layer of certain thickness at the boundary) will not change significantly by deformation of the object, and because of the limited speed of the objects, it will not be occluded by other objects. As such we can obtain the new position of the object-core by histogram matching. At the new location, all pixels belonging to the object-core are assumed to belong to the object, so their object likelihood is close to one, depending on the match-error.

For the pixels close to the boundary of the object, this will not hold. The likelihood that they belong to the object is determined by three factors:

- Distance to the object-core.
- Likelihood of a pixel with this color, given the object histogram.
- Likelihood of a pixel with this color, given the object histogram and the pixels belonging to the object-core.

The first item can easily be calculated by a distance transform [8] of the object-core (see figure 1(b)). The second item is given by the count in the associated bin of the object histogram \mathbf{H} in relation to the total number of pixels in the histogram (see figure 1(c)). The third item is more complicated. We already have all pixels belonging to the object-core assigned to belong to the object. These can be removed from the object histogram by subtraction of the object-core histogram \mathbf{C} . This results in a wish-histogram \mathbf{W} of missing values. Normalization so the values lie between zero and one gives:

$$\mathbf{W} = \frac{1}{2} + \frac{1}{2} \frac{\mathbf{H} - \mathbf{C}}{\mathbf{H} + \mathbf{C}}. \quad (1)$$

Locations where \mathbf{W} has a high value are the color-values we still need to make our object histogram the same as in the previous frame (see figure 1(d)).

The three likelihoods mentioned above can be combined by multiplying them to calculate one object likelihood, so

$$L(x, y, \vec{c}) = L_D(x, y) \cdot L(\vec{c}|\mathbf{H}) \cdot L(\vec{c}|\mathbf{W}) \quad (2)$$

with L_D a likelihood related to the distance and \vec{c} the vector of color values of the pixel at location x, y .

The maximum speed and deformability of the objects determines which pixels need to be processed (the width of the boundary). All other pixels get an object likelihood

equal to zero for this object. After calculating a likelihood-image for each object, classification can be performed by taking the maximum for each pixel of the set of background likelihood and object likelihoods. Subsequently some post-processing is necessary in the case of objects which (for example due to occlusion) split, and to deal with small isolated regions of one class inside an other class.

3. IMPLEMENTATION

In this section some details of our implementation will be described. First the overall pseudo-code will be given, then some parts of the implementation will be explained in more detail.

Pseudo-code of the algorithm for each frame:

1. Calculate the object-core of known objects.
2. Read new image.
3. Get Kalman prediction for known objects.
4. Find object-core location using histogram matching
5. For all pixels close to an object-core, calculate the likelihood for their object and the background.
6. Classify the pixels based on their likelihoods.
7. Detect new objects by thresholding the likelihood of pixels classified as background.
8. Perform post-processing.
9. Update the background for those pixels classified as background.
10. Update object and Kalman parameters.

3.1. Background modeling

The EM background modeling algorithm we used is described in more detail in [9]. This work is closely related to the work of Stauffer and Grimson [10], but as opposed to Stauffer and Grimson, it updates all kernels. This results in a more accurate estimate of the variance of the kernels [11].

In order to reduce artifacts from shadows and changes in the light-intensity, we use the r and g channels from normalized rgb as features of the background model. An additional advantage is that this normalized color space is two dimensional and practically without any correlation, which results in a faster algorithm, as we treat the two channels as independent of each other. In the (r, g) -space, we use the EM algorithm to model the data per pixel by two separate 1D mixtures of Gaussian kernels (although the use of 2D kernels is being considered), one for r and one for g . Four kernels are used per color, one to describe the background, and three to prevent this kernel from changing too fast when an object occludes the background. The kernel with the highest prior is regarded to describe the background, the other kernels the foreground. As initialization, 100 frames (4 seconds) are processed before object detection and tracking is started. After the initialization period, the background

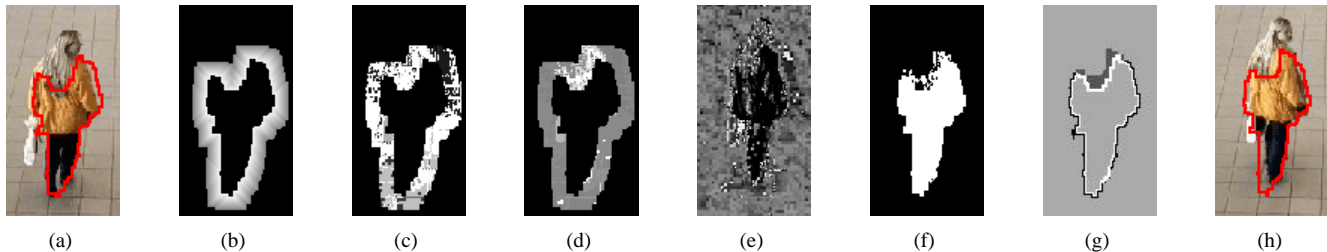


Fig. 1. Step-by-step algorithm description. A shows the object in frame t . B, c, d and e show the likelihood based on distance, histogram, wish-histogram, and background respectively. F shows where the object likelihood is maximal. G shows in light-gray the background and object-core, in white the pixels which were kept, in black the pixels which were removed, and in dark-grey the pixels which were added. Finally h shows the object in frame $t + 1$. (Actually, Some frames were skipped to show the difference more clearly.)

model is updated each frame for only those pixels which are labeled as background.

3.2. Object modeling

Similar to the background model, the object templates (a color histogram in our case) need to be updated. In order to prevent bad templates from one bad estimate of the location or occlusion, the templates are updated with only a small fraction γ of the object in the new image. The update of the color histogram is done by:

$$\mathbf{H}_{t+1} = (1 - \gamma)\mathbf{H}_t + \gamma\mathbf{N}, \quad (3)$$

where \mathbf{H} is the template histogram and \mathbf{N} is the histogram of the object in the new frame. Here we choose to use the full RGB histogram, and not use normalized rgb, because of its higher selectivity. We can do so, because the update speed for the objects is much greater than that for the background model, as the appearance of objects tends to change faster than the background. The number of bins of the histograms used is eight for each of the three dimensions.

As shown, there are some differences between modeling the background and the objects. As the number of objects in the data is limited, it is feasible to memorize the full histograms of the objects, as opposed to memorizing a full histogram for each pixel, which would require too much memory. The update speed of the background should be much slower than the update of the histograms using γ . That is because different time-scales are involved. The modeling of the background should not be affected by passing objects, so updating is done slowly. The update of the objects on the other hand, should be much faster, as mentioned before.

3.3. Object tracking

Likelihood calculation of the objects is done in two stages. First, histogram matching is used to find the best location of the object-core. The second stage is determining which pixels could belong to the object and calculate their object likelihood.

To perform histogram matching, a binary erosion of the object in the previous frame is applied (see figure 1(g)). The size of the structuring element is based on the maximal speed of the objects, and is in our case 5x5 pixels. This provides the set of object-core pixels. At different locations in the new image, the histogram of the object-core is compared to the histogram we would obtain if the object-core would be at this location. The sum of absolute differences is used as measure of fit. The location of the best fit is chosen as the new object-core location, and the pixels given by the object-core at this location are assigned an object likelihood of one. The histogram of the object-core is updated using equation (3).

In order to improve the speed of the histogram matching, an estimate of the object location is calculated by using a Kalman filter. The new location is used to update the Kalman filter. This reduces the area we have to perform this (histogram based) template matching in.

Using a binary dilation on the object-core (with twice the size of the erosion which has been used to create the object-core from the object) gives us those pixels which could belong to the object. For those pixels we calculate the object likelihood as described in section 2.

3.4. Object detection

New objects are detected using a threshold on the likelihood of pixels classified as background. The physics of the objects determines the minimum size of an object, providing a criterium whether or not a blob can be an object. Objects which are too small are re-labeled as background.

4. EXPERIMENTS

For our experiments we used data recorded in the main hall of an airport using a digital video camera. The camera was located at the second floor. We used an background update of 0.01, a foreground update (γ) of 0.1, a threshold on the background likelihood of 5 on a scale of 0-255, and a minimum object-size of 300 pixels. These parameters were

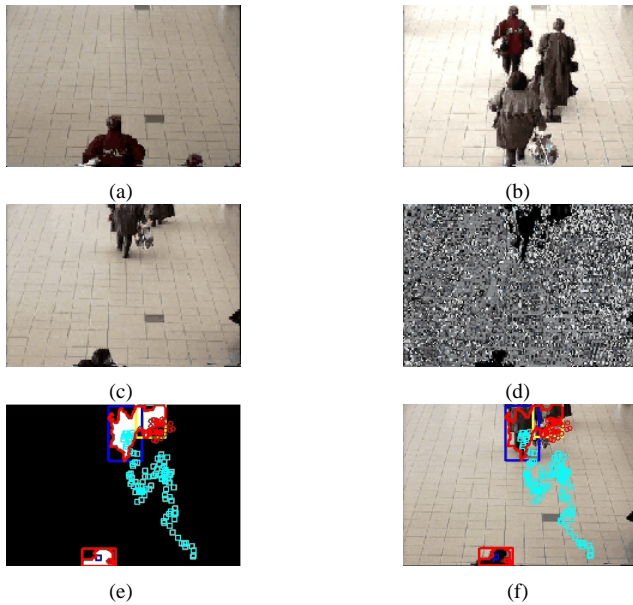


Fig. 2. Results of the background estimation and tracking after 337 frames. A and b are two images from earlier in the sequence (frames 235 and 285). C shows the current input image (frame 337) and d the related background likelihood. E shows three tracked blobs (white) with their boundary (colored), a bounding rectangle, and the path the objects have made in the past (squares and circles). F shows the input frame with the same information as in e overlaid.

determined by a visual inspection of the result. The size of the images was 392x270 pixels.

In figure 1 we show only part of the image, and we skipped some frames to make the difference between the old and new object more clear. Figures 2(a), 2(b) and 2(c) show different frames in a sequence. In Figure 2(c), the frame which is being processed is shown. Figure 2(d) shows the background likelihoods, while figure 2(e) shows the segmentation and track results using our integration algorithm. Finally, figure 2(f) shows the input image with tracking info.

5. CONCLUSIONS

We have shown that two important steps of video surveillance, object detection and tracking, can be integrated by a likelihood-based algorithm. This approach has the advantage that no thresholds are necessary to re-find previously detected objects. This also leads to better object segmentation and easier detection of (partial) occlusion.

The object-core is tracked using template matching. Pixels near the object boundary are classified in each frame using a likelihood-based algorithm, which strives to keep the color information in the object constant, while allowing the object pixels to shift relative positions.

6. REFERENCES

- [1] D. Harwood I. Haritaogly and L.S. Davis, "W⁴: Real-time surveillance of people and their activities," *IEEE Transactions on Pattern Analysis and Machine Vision*, vol. 22, no. 8, pp. 809–830, August 2000.
- [2] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers, "Wallflower: principles and practice of background maintenance," in *IEEE Conference on Computer Vision*, Kerkyra, Greece, 1999, pp. 255–261.
- [3] Christopher Richard Wren, Ali Azarbayejani, Trevor Darrell, and Alex Pentland, "Pfinder: Real-time tracking of the human body," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 780–785, 1997.
- [4] A.P. Dempster, N.M. Laird, and D.B. Rubin, "Maximum likelihood from incomplete data via the EM-algorithm," *Journal of the Royal Statistical Society*, vol. B 39, no. 1, pp. 1–38, 1977.
- [5] C. Priebe, "Adaptive mixtures," *Journal of the American Statistical Association*, vol. 89, no. 427, pp. 796–806, 1994.
- [6] R. Kalman, "A new approach to linear filtering and prediction problems," *Transaction of the ASME-Journal of Basic Engineering*, pp. 35–45, March 1960.
- [7] M.J. Swain and D.H. Ballard, "Indexing via color histograms," in *DARPA90*, 1990, pp. 623–630.
- [8] Kenneth R. Castleman, *Digital Image Processing*, Prentice Hall, Englewood Cliffs, NJ, USA, 1996.
- [9] G. Stijnman and R. van den Boomgaard, "Background estimation in video sequences," Tech. Rep. 10, Intelligent Sensory Information Systems Group, University of Amsterdam, (<http://www.science.uva.nl/research/reports-isis/ISISreport.html>), January 2000.
- [10] C. Stauffer and W.E.L. Grimson, "Adaptive background mixture models for real-time tracking," in *IEEE Conference on Computer Vision and Pattern Recognition*, Fort Collins, CO, 1999, vol. 2, pp. 246–252.
- [11] Frank Zwarthoed, "Real-time object detectie in video mat gebruik van mixture modellen voor achtergrond-schatting (in dutch)," M.S. thesis, IAS group, University of Amsterdam, April 2002.