Bayesian Pixel Classification for Human Tracking

Daniel Roth, Petr Doubek, and Luc Van Gool
Computer Vision Laboratory, ETH Zürich, Switzerland
{*roth,doubek,vangool*}*@vision.ee.ethz.ch*

Abstract

We present a monocular object tracker, able to detect and track multiple objects in non-controlled environments. Bayesian per-pixel classification is used to build a tracking framework that segments an image into foreground and background objects, based on observations of object appearances and motions. Gaussian mixtures are used to build the color appearance models. The system adapts to changing lighting conditions, handles occlusions, and works in real-time.

1. Introduction

In this paper we focus on the monocular detection and tracking of people in non-controlled environments. The proposed method adapts to changing lighting conditions, handles occlusions and newly appearing objects, and works in real-time. It uses a Bayesian approach, assigning pixels to objects by exploiting learned expectations about both motion and appearance of objects.

Human tracking has a rich tradition and we can therefore only describe the work in relation to the most relevant prior art. Mittal and Davis [1] developed a multi-camera system which also uses Bayesian classification. It calculates 3D positions of humans from segmented blobs and then updates the segmentation using the 3D position. The approach owes its robustness to the use of multiple cameras and more sophisticated calculations, that stand in the way of real-time implementation. Capellades *et al.* [2] implemented an appearance based tracker which uses color correlograms and histograms for modeling foreground regions. A correlogram is a cooccurrence matrix, thereby including joint probabilities of colors at specific relative positions. Also in our case, taking account of how colors are distributed over the tracked objects is useful, but we prefer a sliced color model instead, as will be explained. Senior *et al.* [3] use an appearance model based on pixel RGB colors combined with shape probabilities.

Majority of trackers are not based on segmentation. They are applicable even for moving camera however, the absence of segmentation typically requires manual initialization of tracked target, such as in the tracker by Comaniciu *et al.* [4] and Nummiaro *et al.* [5]. Okuma *et al.* [6] overcome this drawback by using trained detector, but only a special class of targets is tracked which allows to keep the detector fast and accurate.

Publications [1, 2] are most closely related to our work since they use Bayesian classification and appearance models in three and two dimensions respectively. In comparison to [1], we developed a modular real-time tracker that could serve as a basic building block for a multi-camera setup. The original tracker needs a fixed calibrated setup and does not scale well due to the pairwise stereo calculations and iterative segmentation. [2] applies Bayesian classification only on pre-segmented foreground pixels, while we let all the models – object models, background model and model explaining newly appeared objects – compete for all pixels, thus maintaining a consistent probabilistic approach throughout the whole algorithm.

The outline of this paper is as follows. Section 2 presents the overall strategy underlying our tracker, which is based on both appearance and motion models. The appearance models are explained in Section 3 and the motion models in Section 4. Section 5 describes how occlusions are handled. Results are discussed in Section 6 and Section 7 concludes the paper.

2. Bayesian Per-Pixel Classification

The proposed method performs a per-pixel classification to assign the pixels to different objects that have been identified, including a background. The probability of belonging to one of the objects is determined on the basis of two components. On the one hand, the appearance of the different objects is learned and updated, and yields indications of how compatible observed pixel colors are with these models. On the other hand, a motion model makes predictions of where to expect the different objects, based on their previous position. Combined, these two aspects yield a probability that, given its specific color and position, this pixel belongs to one of the objects. The approach is akin to similar Bayesian filtering approaches, but has been slimmed down to strike a good balance between robustness and speed.

As said, each object, including the background, is addressed by a pair of appearance and motion models. Fig-

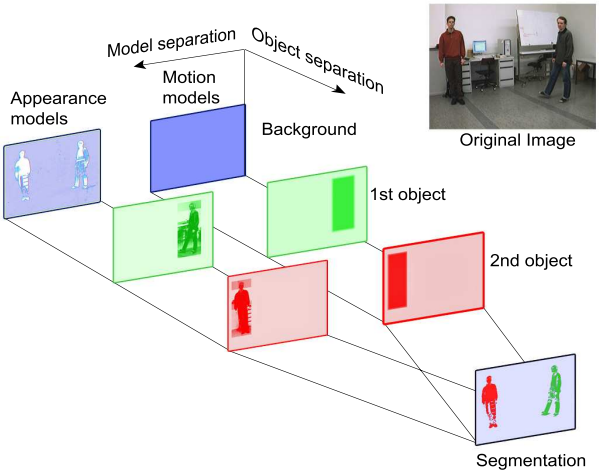


Figure 1: Tracking framework

ure 1 sketches this tracking framework. It incorporates different characteristics of objects such as their appearance and their motion via separate and specialized models, each updated by observing them over time. This method has several advantages for object tracking over simple foreground / background segmentation, especially in cases when an object stands still for a while and must not 'disappear' into the background, or when objects overlap.

Formally, the classification is described by (1) and (2). They describe how, for different objects, their probabilities to occupy a specific pixel are calculated and compared. These probabilities are calculated as the product of a prior probability $P_{prior}(object)$ to find the object there, resulting from its motion model and hence the object's motion history, and the conditional probability $P_{posterior}(object|pixel)$ that, if the object covers the pixel, its specific color would be observed.

$$P_{posterior}(object|pixel) \propto P(pixel|object)P_{prior}(object) \quad (1)$$

$$segmentation = \max_{object}(P_{posterior}(object|pixel)). \quad (2)$$

The tracker executes the steps in Table 1 for every frame. First, the prior probabilities of all objects are computed. Then they are used in the second step to segment the image with the Bayesian per-pixel classification. Each pixel is assigned to the object with the highest probability at this position. In a third step, the object positions are found by applying a connected components algorithm [7] to the segmentation image, which groups pixels assigned to the same object. The fourth step handles several special situations detected by the connected components algorithm. Missing objects are deleted. Objects are split if they have multiple new object positions. New objects are initialized from the regions claimed by a special new object model (to be described in Section 3.3). The fifth step handles objects which are partially or completely occluded and infers their spatial extent according to the motion model. Occlusion handling is the subject of Section 5. Finally, all object model pairs are updated according to the newly found positions and the pixels assigned to them in the segmentation image.

1.	compute prior probability of all models
2.	segment image (Bayesian per-pixel classification)
3.	find connected components and group them to objects
4.	add, delete, split objects
5.	handle occlusion
6.	update all models

Table 1: Tracking framework algorithm

3 Appearance Models

In this section we introduce the color-based appearance models, responsible for providing the $P(pixel|object)$ probabilities in (1). Different models are used for the background $\mathcal{B}$, for newly detected objects $\mathcal{N}$, and for the objects $\mathcal{O}_i$ that are already being tracked. The differences in the models reflect the different expectations as to how these objects change their appearances over time. Each model provides a probability image, where pixels which match the model will have a high probability and the others a low probability. The appearance models are updated after pixels have been assigned to the different objects, based on (2).

3.1 Color Modeling with Mixtures of Gaussians

The appearance models $\mathcal{B}$ of the background and $\mathcal{O}_i$ of all tracked objects are based on Gaussian mixtures in RGB color space. Methods employing **t**ime-**a**daptive **p**er-**p**ixel **m**ixtures **o**f **G**aussians (TAPPMOGs) have become a popular choice for modeling scene backgrounds at the pixel level, and were proposed by Stauffer and Grimson [8]. Other color spaces than simple RGB could be considered, as e.g. described by Collins and Liu [9].

The probability of observing the current pixel value $\overrightarrow{X}_t = [R_t \quad G_t \quad B_t]$ at time t, given the mixture model built from previous observations is

$$P(\overrightarrow{X}_t|\overrightarrow{X}_1, ..., \overrightarrow{X}_{t-1}) = \sum_{k=1}^{K} w_{t-1,k} * \eta(\overrightarrow{X}_t, \overrightarrow{\mu}_{t-1,k}, \Sigma_{t-1,k}) \quad (3)$$

where $w_{t-1,k}$ is the weight of the k^{th} Gaussian at time $t-1$, where $\overrightarrow{\mu}_{t-1,k}$ is the vector of RGB mean values, where the $\Sigma_{t-1,k}$ is the covariance matrix of the k^{th} Gaussian and where η is the Gaussian probability density function.

3.2 Background Model

The algorithm by Stauffer was originally designed to combine foreground and background Gaussians together into one model, where the foreground Gaussians have lower weights. Due to the separation of foreground and background pixels into different models in our approach, there is no more need to model foreground colors within $\mathcal{B}$. Experiments have showed that a single Gaussian suffices to model the background colors at a single pixel in the background, assuming the camera is static. Such simplification is beneficial for the number of parameters to be determined and kept updated. Going one step further, our background model uses one Gaussian per pixel with evolving $\overrightarrow{\mu}_t$ but with the same and fixed diagonal covariance matrix everywhere. This further simplification speeds up the computations without sacrificing too much accuracy.

The covariance matrix is set beforehand and the mean vectors are initialized based on the values in the first image. Thereafter, the pixels $\overrightarrow{X}_t$ segmented as background are used for updating the corresponding color model, where α is the learning rate $\overrightarrow{\mu}_{t,k} = (1-\alpha)\overrightarrow{\mu}_{t-1,k} + \alpha\overrightarrow{X}_t$. The background is not updated if not visible.

3.3 Finding New Object Models

The tracker also caters for newly appearing objects. Their creation is based on a generic 'new object model' $\mathcal{N}$. This appearance model has a uniform, low probability $p_{\mathcal{N}}$. Thus, when the probabilities of the background and all objects drop below $p_{\mathcal{N}}$, the pixel is assigned to $\mathcal{N}$. Typically this is due to the following reasons:

- A new object appears in front of the background and the background probability drops.
- The pixel is on the "edge" of an object and its value is a mixture of the background and foreground color.
- The foreground model does not contain all colors of the object.

A new foreground model $\mathcal{O}_i$ is initialized as soon as a region of connected pixels has a minimal size. Some rules should avoid erroneous initializations:

- Objects entering the image from the sides are not initialized until they are fully visible, i.e. until they are no longer connected with the image border. This prevents objects from being split into multiple parts.
- Pixels assigned to $\mathcal{N}$ which are connected with a foreground object $\mathcal{O}_i$ are not taken for a new object. Instead, it is assumed that $\mathcal{O}_i$ is not properly segmented. If the number of these pixels exceed 20% of $\mathcal{O}_i$, they are added to $\mathcal{O}_i$. The threshold is due to the assumption that a smaller number of pixels are a result of pixels on the "edge" of the object.

$\mathcal{N}$ also has a mechanism for cleaning up itself by reducing the probability even below $p_{\mathcal{N}}$ for those pixels which are assigned to $\mathcal{N}$ for a longer period of time. This mechanism is essential to keep the new object model clean from noise blobs and it increases the accuracy of the background model. $\mathcal{N}$ is initialized at startup with $p_{\mathcal{N}}$.

3.4 Foreground Models

Foreground models $\mathcal{O}_i$ are created from regions detected by $\mathcal{N}$ during the runtime of the tracker. Each foreground object has its own $\mathcal{O}_i$, which is independent from that of other objects. We use a 'sliced object model', as it divides the object into a fixed number of horizontal slices of equal height. For each slice the expectation-maximization (EM) algorithm is used to generate a Gaussian mixture to optimally represent the colors of all the pixels in the slice, as visualized by Fig. 2. The number K of Gaussian mixtures is the same for all slices and typically ranges from 2 to 5. More Gaussians are able to model objects with more diverse colors, but at the cost of slowing down the algorithm.

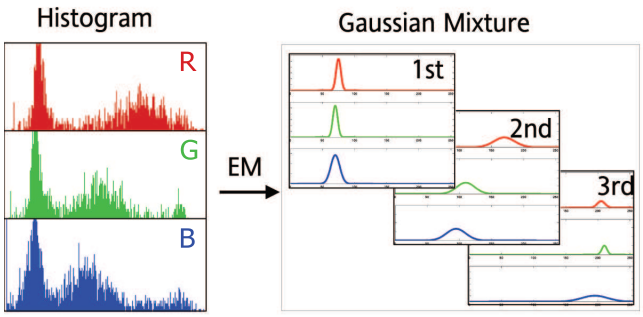
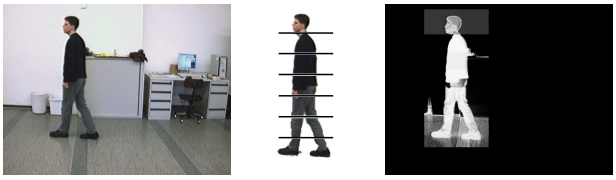


Figure 2: Constructing Gaussian mixture for a whole slice

The approach of slicing a foreground object was previously proposed by Mittal and Davis [1] for 3D multi-camera tracking. Figure 3 shows an example of a sliced object model. Dividing an object into horizontal regions is an effective way of modeling approximately cylindrical objects like humans. It assumes that the color characteristics within a given slice do not change with object rotation around the vertical axis. This said, our model does adapt to such occasional changes, but horizontal slicing tends to at least limit them and therefore to keep the model more discriminative. As long as the object does not rotate around other axes and if the number of Gaussians K is big enough to represent all colors at a particular height, the sliced color model can adapt. Updating the color model for one slice is done by replacing the old mixture with the new one, with equal K. This is a fast solution for updating the mixture without a costly iterative EM algorithm. The classification prevents



(a) original image (b) 7 slices (c) probability image

Figure 3: Sliced object model

the model from changing too fast as only colors similar to the previous model are segmented. The adaption to new colors is provided by the 20% rule of $\mathcal{N}$, as described in 3.3.

4 Motion Model

In this Section we describe the motion model which is responsible for the $P_{prior}(object)$ probability in (1). Just as was the case for the appearance model, each object has its own motion model. The probability of the object position is high in an area where the object is expected to appear in the current frame. For $\mathcal{B}$ and $\mathcal{N}$ we use a simple model with uniform probability and without any tracking capabilities.

For all objects $\mathcal{O}_i$ we use a linear Kalman filter with a constant velocity model for tracking the 2D image position. An object is represented by its bounding box, given the box center x, y and size H_x, H_y. The probability that the object may be present at the different pixels is shown in Fig. 4(b). The probability is high inside the area of the bounding box and decreases as one passes through a border region, linearly dropping off. This border region increases the robustness. The border width is the same for all objects in all directions. This is important in the case of occlusion, where the joining and separation works best if both objects have the same probability border size.

(a) bounding box coordinates (b) Probability image

Figure 4: Object position priors, from the original image in Figure 3(a)

After per-pixel classification, updated object positions are obtained from the minimal enclosing rectangles of the pixels assigned to the objects. The box center x, y is used for updating the Kalman filter. H_x, H_y are not incorporated directly, but are obtained as a weighted average of the new ones and the previous ones at time step $t-1$: $V_{x,t} = (1-\alpha)V_{x,t-1} + \alpha H_{x,t}$ and $V_{y,t} = (1-\alpha)V_{y,t-1} + \alpha H_{y,t}$, where α is the learning rate.

5 Occlusion Handling

The Bayesian classification tracker handles occlusion by comparing the estimated object positions with the actual observations. The lack of image depth information (z coordinate) is partially compensated by assuming a horizontal and upright camera orientation, as well as a planar floor. In this case, objects closer to the camera have higher $bottom\_line = y + |H_y|$ value. In the special case, when the $bottom\_line$ values of two objects are equal, the taller object is assumed to be in front of the smaller one. This special case occurs when two objects are behind an obstacle or when the lower part of the objects is outside the field of view.

The occlusion handling works in two steps, occlusion analysis and occlusion interpretation. During the analysis, each predicted object position is checked against each object in the current image. If two objects are overlapping, the type of occlusions is classified as one of eight possible cases of occlusion (Fig. 5). Depending on the case, some or all of the edges of the new object position box are affected by the occlusion. Those edges which are occluded are not used for updating the object position.

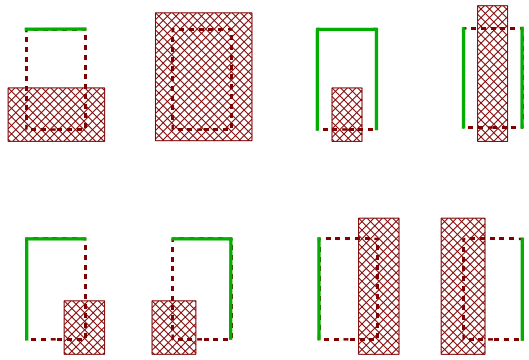


Figure 5: Eight cases of occlusion. The hatched rectangle represents the object in front. The bounding box of the object behind is represented by solid (=valid) and dashed (=invalid) lines.

The occlusion interpretation then updates the object position by applying only the valid edges of the current observation. Missing edges are reconstructed by using the object width and height from previous images without occlusion. The position of completely occluded objects is only based on the position predicted by the constant velocity assumption. The accuracy of predicting completely occluded objects could be increased by refining this model.

6 Results and Discussion

The presented tracking framework has the following advantages:

- The separation of foreground and background into different and discrete models is able to track moving and motionless objects in front of an adaptive background.
- The use of adaptive color modeling with mixtures of Gaussians enables the background and foreground models to adapt individually to changing lighting conditions.
- Occlusion handling can be implemented at the high level of object positions. Objects under occlusion can be tracked with the help of accurate foreground models and per-pixel classification.
- The current implementation works in real-time on QVGA (320x240) resolution.

This section demonstrates several of these features using the dataset provided by the *Second IEEE International Workshop on Performance Evaluation of Tracking and Surveillance* 2001. The test dataset 3 with 5336 frames is used, which is a challenging sequence in terms of multiple targets and significant lighting variation. The sequence is down-scaled to a resolution of 320x240 which allows to run the tracker in real-time.

No objects are entering the scene in the first 1200 frames. As expected, the tracker adapts to the changing lighting conditions from bright sunshine to dull weather conditions and no false objects are detected, showing the strength of the adapting background model.

In Fig. 6(a) a group of two people enter the field of view in the lower right corner. The whole group is detected as one object and is tracked on their way through the image. The tracker is not designed to separate grouped objects as long as the group is not visually disconnected, due to the use of connected components for building objects. Furthermore, the object detected by the tracker includes the shadow cast by the object on the ground. This is an expected result as the background model has no special shadow removal capabilities.

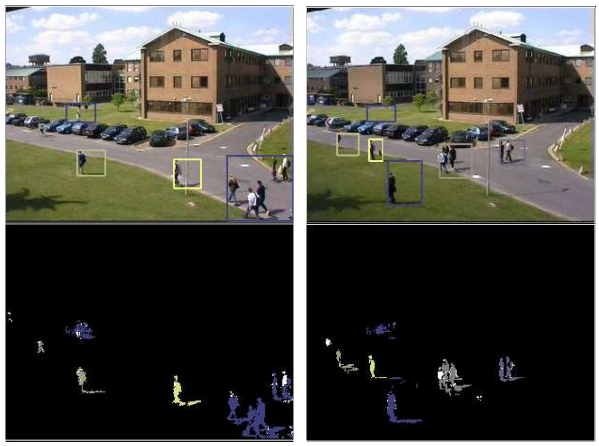
At Fig. 6(b) a single person enters the scene in the lower left corner in the shadow of a tree. The person is correctly detected even if it is eclipsed in the shadow. The object model adapts to the brighter colors of the person when it steps into the sunlight, showing the adaptiveness of the foreground models. The person is correctly tracked until it leaves the image. Fig. 6(a) and 6(b) show the high accuracy and the fast detection of new object.

In Fig. 6(c) four people walk into the scene individually. A total number of five people are tracked correctly during

(a) Frame 1382 (b) Frame 1415 (c) Frame 2401

Figure 6: PETS 2001 test dataset 3, part 1

this phase of the sequence. The tracker is able to follow the group of two people which walks further apart between the cars until they shrink to little dots, where the tracker reaches its resolution limits. The low number of pixels for updating the color models is insufficient and the object is gradually adapted to the background where the foreground model will stay, even if the two people have disappeared.



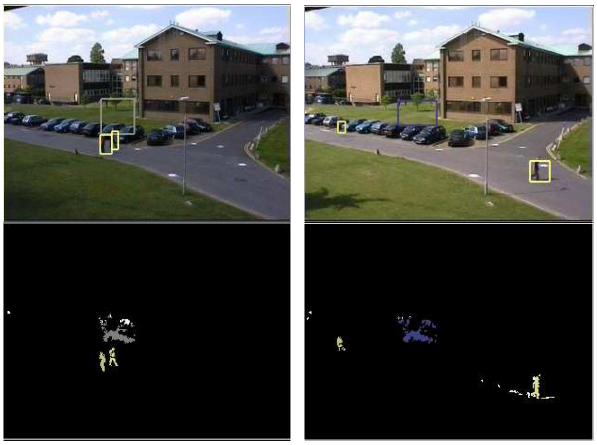
(a) Frame 2510 (b) Frame 2701

Figure 7: PETS 2001 test dataset 3, part 2

Five people enter in Fig. 7(a) from the right side. Due to the shadows and the close distance between the people the group is detected as one big object. In the following frames, when the distances between some of the people of this group increase, the group is split into three objects, containing one single person and two couples. After the split six objects are tracking eight people in Fig. 7(b). The computational time reaches its maximum with 55 milliseconds per frame on a 3GHz P4.

The frames up to number 3301 are sometimes a bit chaotic due to multiple occlusions of whole groups of objects. During this phase some object tracks are mislead by other objects and new objects are initialized on the old ob-

ject positions. But the tracker correctly recognizes that three objects have left the image on the left border, a group of four people is walking between the cars to the back and that a cyclist passed through the scene. The changing lighting condition of the disappearing sun has no negative effect on the tracker.



(a) Frame 4000 (b) Frame 4201

Figure 8: PETS 2001 test dataset 3, part 3

Fig. 8(a) shows that crossing people are correctly tracked by the occlusion handling if only two objects are undergoing occlusion. The tracker distinguishes between the two objects, even under occlusion, and the object positions are correctly determined.

A fast illumination change in Fig. 8(b) shows the limitations of the adaptive background for those parts of the image, which are covered by an object. While visible background pixels adapt, the background color models of pixels behind the objects are not updated. This results in an outdated background model when the object leaves its position. The outdated background model leads to the initialization of new foreground object containing only background and no real objects.

7 Conclusions and Future Work

We have proposed a framework for the fast tracking of multiple objects or people in monocular video sequences. The tracker can handle changing object and background appearances, as well as newly appearing objects and occlusions.

Of course, the tracker can still be improved, and this in several ways:

- The time-adaptiveness of the models is only ensured in visible parts of the image. Occluded objects are not updated. Especially the background model is sensitive to outdated color models in areas where it was occluded by a static foreground object. In such case, changes observed in the visible part could be used to infer changes at occluded pixels.
- The background is still assumed to be static. Background pixels should be given more sophisticated appearance models to deal with changes (e.g. waving tree branches, flickering lights, ...)
- The occlusion handling performs well for two objects, but fails in crowded scenes with multiple objects occluding each other. The performance is directly linked to the uniqueness of each foreground model, which is more doubtful in case of multiple overlaps.

As future work we would like to improve the current appearance and motion models to overcome the current limitations. We also plan to implement this tracking method for multiple, but non-overlapping cameras, like the system presented by Javed *et al.* [10].

Acknowledgments

The authors gratefully acknowledge support by the Swiss SNF NCCR project IM2 and ETH Zürich project blue-c-II.

References

[1] A. Mittal and L. S. Davis, M2tracker: A multi-view approach to segmenting and tracking people in a cluttered scene using region-based stereo, in *ECCV*, 2002.

[2] M. B. Capellades, D. Doermann, D. DeMenthon, and R. Chellappa, An appearance based approach for human and object tracking, in *ICIP*, 2003.

[3] A. Senior, A. Hampapur, Y.-L. Tian, L. Brown, S. Pankanti, and R. Bolle, Appearance models for occlusion handling, in *PETS*, 2001.

[4] D. Comaniciu, V. Ramesh, and P. Meer, Real-time tracking of non-rigid objects using mean shift, in *CVPR*, 2000.

[5] K. Nummiaro, E. Koller-Meier, and L. Van Gool, Journal of Image and Vision Computing **21(1)** (2003).

[6] K. Okuma, A. Taleghani, N. d. Freitas, J. J. Littlei, and D. G. Lowe, A boosted particle filter: Multitarget detection and tracking, in *ECCV*, 2004.

[7] Ali Rahimi, Fast Connected Components on Images, http://xenia.media.mit.edu/ rahimi/connected/.

[8] C. Stauffer and W. Grimson, Adaptive background mixture models for real-time tracking, in *CVPR*, 1999.

[9] R. T. Collins and Y. Liu, On-line selection of discriminative tracking features, in *ICCV*, 2003.

[10] O. Javed, Z. Rasheed, K. Shafique, and M. Shah, Tracking across multiple cameras with disjoint views, in *ICCV*, 2003.