Multiple Hypotheses Multiple Levels Object Tracking

Ronan Sicre[1,2] and Henri Nicolas[1]
[1]*LaBRI, University of Bordeaux, 351 Cours de la Liberation, 33405 Talence Cedex, France*
[2]*MIRANE SAS, 16 Rue du 8 mai 1945, 33150 Cenon, France*

Keywords: Object Tracking, Motion Detection.

Abstract: This paper presents an object tracking system. Our goal is to create a real-time object tracker that can handle occlusions, track multiple objects that are rigid or deformable, and on indoor or outdoor sequences. This system is composed of two main modules: motion detection and object tracking. Motion detection is achieved using an improved Gaussian mixture model. Based on multiple hypothesis of object appearance, tracking is achieved on various levels. The core of this module uses regions local and global information to match these regions over the frame sequence. Then higher level instances are used to handle uncertainty, such as miss-matches, objects disappearance, and occlusions. Finally, merges and splits are detected for further occlusions detection.

1 INTRODUCTION

Object tracking is an important task in the computer vision field. There are two main steps in object tracking: interest moving object detection and tracking these objects from frame to frame. Then analysis can determine the objects behaviors. Thus, object tracking is used in various applications, such as: motion-based recognition, automated surveillance, traffic monitoring, human computer interaction, etc.

Tracking can be defined as estimating the trajectory of an object in the image, i.e. assigning consistent labels to each tracked objects in the frames of a video. The tracking process often provides object size, orientation, area, or shape.

The main difficulties of tracking are: loss of information due to the 2D nature of the data, noises, complex object motion, non-rigid objects, occlusions, illumination changes, and real-time requirements.

To select a relevant method, we have to answer various questions: What type of object do we track? What model can efficiently detect these objects? What representation should be used for tracking? What information do we require for further analysis?

We are interested in tracking any object: deformable or non-deformable objects. We can use a pixel-based background model to detect motion. Tracking can be achieved by matching regions features that are likely to remain stable from one frame to the next, such as color, size, surface area, etc. We want to precisely detect objects shape and contours, for further behavior analysis in a shopping setting (Sicre and Nicolas, 2010) for example.

2 PREVIOUS WORK

This section presents motion detection and object tracking. For an overview of the field the reader can refer to (Hu et al., 2004), (Yilmaz et al., 2006), and (Moeslund et al., 2006).

2.1 Motion Detection

The aim in this phase is to distinguish the moving objects from the background. Most motion detection techniques use a background model. Depending on the type of model used, we can classify methods. The model can be pixel based, local, or global.

Pixel based models associate to each pixel of an image a value or an intensity function that gives the appearance of the background. Local models use the neighborhood of a pixel instead of the pixel itself to calculate the similarity measurement. Global methods use the entire image at each moment to build a model of the entire background.

In our study, we chose a pixel based model that offers a good compromise between quality and speed.

2.2 Object Tracking

Once moving regions are detected, the following step is to track these regions from one frame to another. Tracking can be based on regions, contours, features or a model (Yilmaz et al., 2006).

Region based tracking identifies connected regions corresponding to each object in the scene, describes and matches them. Active contour uses the shape of the detected regions to match them from one frame to another. Feature-based tracking does not aim at tracking an object as one entity. We do look here for distinctive features that can be local or global. Model based tracking can be done in various ways: articulated skeleton, 2-D contours, 3-D volumes. For each new image, detected regions are compared to models previously built.

In this paper, we propose an object tracking method based on regions and regions features. More recent tracking system are presented in (Zhang et al., 2012), (Yang and Nevatia, 2012), and (Pinho and Tavares, 2009).

3 MOTION DETECTION

Motion detection uses a pixel based model of the background. We use a method based on the Gaussian mixture model (GMM) first introduced in (Stauffer and Grimson, 2002). The GMM is composed of a mixture of weighted Gaussian densities, which allows the color distribution of a given pixel to be multimodal. Such a model is robust against illumination changes.

Weight ω, mean μ, and covariance Σ are the parameters of the GMM that are updated dynamically over time. The following equation defines the probability density function P of occurrence of a color u at the pixel coordinate s, at time t, in the image sequence I.

$$P(I(s,t)=u)=\sum_{i=1}^{k}\omega_{i,s,t}N(I(s,t),\mu_{i,s,t},\Sigma_{i,s,t}) \quad (1)$$

Where $N(I(s,t),\mu_{i,s,t},\Sigma_{i,s,t})$ is the i-th Gaussian model and $\omega_{i,s,t}$ its weight. The covariance matrix $\Sigma_{i,s,t}$ is assumed to be diagonal, with $\sigma^2_{i,s,t}$ as its diagonal elements. k is the number of Gaussian distributions.

For each pixel value, $I(s,t)$, the first step is to calculate the closest Gaussian. If the pixel value is within T_σ deviation of the Gaussian mean, then parameters of the matched distribution are updated. Otherwise, a new Gaussian with mean $I(s,t)$, a large initial variance, and a small initial weight is created to replace the existing Gaussian with the lower weight. Once Gaussians are updated, weights are normalized and distributions are ordered based on the value $\omega_{i,s,t}/\sigma_{i,s,t}$.

As proposed in (Zivkovic and van der Heijden, 2006), we improve the GMM by adapting the number of selected Gaussian densities. To select the most reliable densities, we modify the calculation of their weights. The weight is decreased when a density is not observed for a certain amount of time.

$$\omega_{i,t}=\omega_{i,t-1}+\alpha(M_{i,t}-\omega_{i,t-1})-\alpha\, c_T \quad (2)$$

Where α is the learning rate and $M_{i,t}$ is equal to 1 for the matched distribution and 0 for the others. c_T is a scalar representing the prior evidence.

Pixels that are matched with any of the selected distributions are labeled as foreground. Otherwise, pixels belong to the background. We note that the model is updated at every frame.

This method remains sensible to shadows. Thus, we use a shadow detection algorithm. Shadows detection requires a model that can separate chromatic and brightness components. We use a model that is compatible with the mixture model (KaewTraKulPong and Bowden, 2001). We compare foreground pixels against current background model. If the differences in chromatic and brightness are within some thresholds, pixels are considered as shadows. We calculate the brightness distortion a and color distortion c as follow:

$$a=argmin_z(I(s,t)-zE)^2 \; and \; c=||I(s,t)-aE|| \quad (3)$$

Where E is a position vector at the RGB mean of the pixel background and $I(s,t)$ is the pixel value at position s and time t. A foreground pixel is considered as a shadow if a is within T_σ standard deviations and $\tau < c < 1$. Where τ is the brightness threshold.

Finally, we modify the updating process to better handle objects stopping in the scene. With the current model, stopped people starts disappearing, because they become part of the background. We modify the updating process for the distributions parameters, i.e. we do not update the model on areas that are considered as belonging to a tracked object. Tracked objects are defined in the next section.

We introduce $F_{s,t}$ that is a binary image representing these tracked objects. $F_{s,t}$ is a filtered foreground image where regions that were tracked for several frames, or objects, are displayed. Pixels covered by an object have value 1 while the others have value 0. We modify the distribution parameters updating equations:

$$\omega_{i,t} = \omega_{i,t-1} + (1-F_{s,t})(\alpha(M_{i,t} - \omega_{i,t-1}) - \alpha\, c_T)$$
$$\mu_t = \mu_{t-1} + (1-F_{s,t})(\rho(I(s,t) - \mu_{t-1})$$
$$\sigma_t^2 = \sigma_{t-1}^2 + (1-F_{s,t})\rho((I(s,t)-\mu_t)^T(I(s,t)-\mu_t) - \sigma_{t-1}^2) \tag{4}$$

Where $\rho = \alpha\,\eta(I_{s,t}|\mu_k, \sigma_k)$. Once shadows are detected and erased, morphological filters are finally applied on this result to reduce noises, fill holes, and improve regions shape.

4 OBJECT TRACKING

Based on the motion detection, we want to match the detected connected regions, or blobs, over the frame sequence.

After presenting our problematic, the first step is to merge regions, so these regions better match actual persons. Then we match detected regions from two consecutive frames. These matched regions are then used to build and maintained an object list. Objects are higher level instances that correspond to regions that are tracked for several frames. In our application, one object should correspond to one person or more than one when an occlusion occurs. Finally, object merge and split are detected to solve occlusions. Figure 1 shows the functional diagram of the system.

4.1 Multiple Hypotheses

In practice, a detected object, or person, can be covered by several disconnected regions, because the algorithm misses part of the person, see figure 2. Thus, we assume that a detected region can be:

- a part of a person
- an entire person
- a group of people

Therefore our system is complex and has to cope with many cases.

4.2 Merging Regions

After filtering out small regions, there are two separated part in the merging process. First, relevant merges are made. These merges are detected when two regions bounding boxes overlap with a surface area greater than a given value. After this process, regions better match actual persons.

Then potential merges are considered. These merges are less reliable and are detected when two regions bounding boxes a slightly overlapping, when regions are closed one to another, or when regions are located in the same vertical axis. In fact, we assume that a person is significantly taller than wide. We note that this ratio depends on each person and on the camera view point. Therefore, when several regions cover a person, they should be closed one to another and relatively in the same vertical axis.

The two types of merge have different effects on the matching process. When two regions are reliably merged, the two original regions become one merged region. However, when two regions are potentially merged, the two original regions are kept in the region list and a new merged region is generated. Since we are not sure about the reliability of these merges, we use the following matching process to decide whether merging is relevant or not. Figure 2 shows an example of reliable and potential merge.

4.3 Frame to Frame Matching

In order to match regions, we first build a descriptor for each of them. The descriptor is composed of the region gravity centre position, size, position of the bounding box centre, surface area, and first and second order color moments.

We note that all these measurements allow us to match regions of different size and shape. Therefore, the selection of such feature is consistent with our hypotheses.

The regions' matching is achieved by using a descriptor matching algorithm, similar to (Matas et al., 2004). We define two sets, or list, of regions descriptors S_1 and S_2. S_1 corresponds to the previous frame and S_2 to the current one. Two regions with descriptors $x \in S_1$ and $y \in S_2$ are matched if and only if x is the most similar descriptor to y and vice-versa, i.e.

$$\begin{gathered}\forall y' \in S_2 \backslash y : sim(x,y) > sim(x,y')\ and \\ \forall x' \in S_1 \backslash x : sim(y,x) > sim(y,x')\end{gathered} \tag{5}$$

Where *sim* is the asymmetric similarity measure defined below. To calculate *sim*, each component of the descriptor is treated independently. The similarity between the *i*-th component of x and y is equal to 1 if y *i*-th component is the closest measurement to x *i*-th component. Otherwise, the similarity is equal to 0. Closest measurements have smaller Euclidean distance.

$$\begin{aligned} sim^i(x,y) = {} & 1\ if\ \forall y' \in S_2, sim^i(x,y) \geq sim^i(x,y') \\ & 0\ otherwise \end{aligned} \tag{6}$$

The overall similarity measure is defined as follows

$$sim(x,y) = \sum_{i=1}^{n} \omega^i\, sim^i(x,y) \tag{7}$$

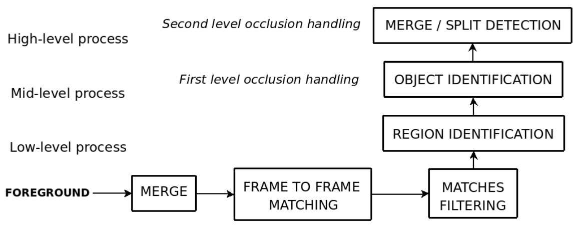

Figure 1: Diagram of the proposed object tracking method.

Where n is the dimension of the descriptor and ω^i the weight of the i-th measurement. We choose to give the same weight $\omega^0 = 1$ to each measurement of the descriptor. The calculation of $sim(y,x)$ is analogous with the roles of S_1 and S_2 interchanged.

An interesting property of this calculation is that the influence of any single measurement is limited to 1. Another major property of this algorithm is that measurements of different orders of magnitude can fit together in the descriptor and are easily handled.

4.4 Matches Filtering and Regions Identification

Once the matching process is achieved, we have couples of matched regions. We first filter these matches: we remove under-regions, i.e. regions that are a part of other matched regions. Then, we test matched regions for relevant merging.

The next step is to identify regions. Regions receive the identification of the region they are matched with, in the previous frame. If this region is not identified, we create an identity for the matched region.

4.5 Objects Identification

However, we need to achieve matching on several levels to handle the uncertainty. We use objects to represent tracked regions. These identified objects use extra temporal information. We compare each matched region with the list of tracked objects. There are two main cases:

- A matched region corresponds to an object and this object is corresponding to only one region. The region is used to update objects information, such as its location, size, surface, color, etc.

- No object is corresponding to a matched region; this region can be a new object entering the scene or an old object that was lost, due to an occlusion for example. To retrieve an object after a miss-detection or an occlusion, we reiterate the matching process. However, we modify the descriptor by only keeping the measurements that are invariant to displacement. If the region is matched to an inactive object, we may

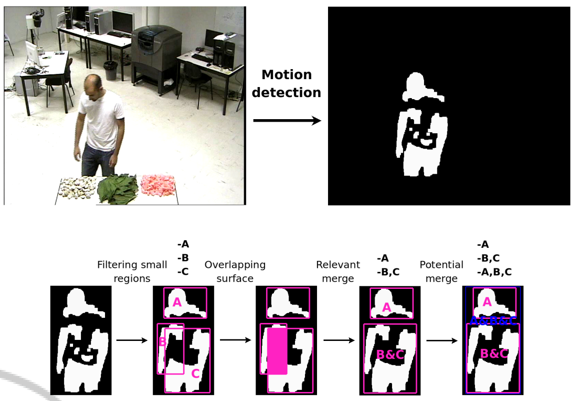

Figure 2: Diagram representing the merging process. After filtering, we have three regions A, B, and C. B and C are reliably merged. Then, A and BC are potentially merged (blue bounding box).

have encountered an occlusion. Otherwise, a new object is created and filled with the region's information.

4.6 Merges - Splits Detection

We note that when an object disappears, during an occlusion for example, as soon as this object reappears, our method can not find a match to the detected region. Therefore, the algorithm tries to find a match with an old object, as presented in the previous section. This process already solves most occlusions.

However, some cases can be more complex and then splits and merges offer us another clue to identify occlusions.

Merge Detection. Several regions are merging when these regions are considered as different identified regions in the previous frames and then become one single region at the current frame. For example, two regions are tracked A, B and a region $C = A,B$ representing the potential merge of these two regions is matched at the current frame. Then a merge just occur.

Split Detection. A region is splitting into several regions when a tracked region is not matched at the current frame and only its under-regions are matched. For example, one potentially merged region $C = A,B$ is not tracked anymore and two under-regions are matched at the current frame: A and B.

Occlusion Detection. We use several measurements to define the consistency of these splits and merges. First, we filter out small objects that can not correspond to an entire person. Then, when a merge, or split, occur we calculate if the concerned object(s) was (were) tracked for a certain amount of time.

In fact, these events do not last for a long pe-

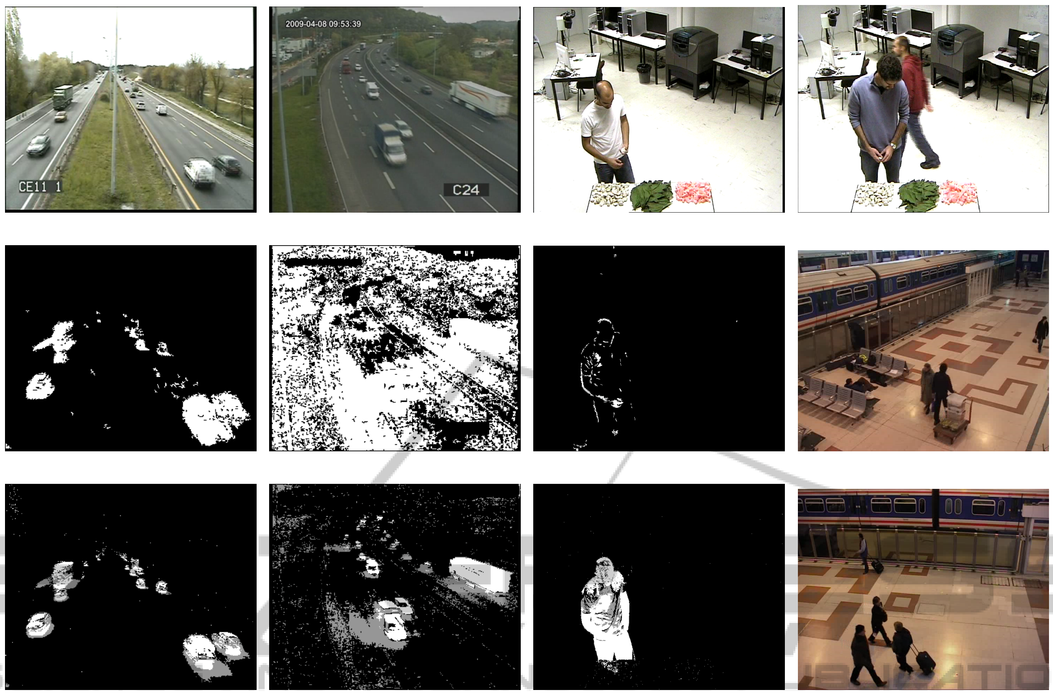

Figure 3: The three left columns present motion detection results. The original frame is on the first row, the GMM on the second, and the improved GMM (with shadow detection in grey) on the third. These results are obtained from V1, V2, and LAB videos. The last column shows occlusion sequences from LAB and PETS datasets.

riod of time when they occur on a single person that splits into several parts and merge back into one piece. Moreover, when two tracked people meet, they are usually tracked for a certain time before the encounter. Finally, we calculate the amount of time between the merge and the split. Based on these measurements we can detect occlusions.

5 RESULTS

We first present some motion detection results. Figure 3 shows three images from three different sequences and the detection results for GMM and iGMM. The two first columns show that shadows detection improves the detection results. The third column shows the detection of a stopped person. iGMM detects properly stopped objects where GMM fails to detect them. Further evaluation of this method is presented in (Sicre and Nicolas, 2011).

Then, we compare our method with four tracking algorithms from OpenCV. These algorithm are commonly used and cover the main current tracking methods: a connected component with Kalmann Filtering (CC) method, a mean-shift tracker initialized using motion detection (MS), a particle filtering method using mean-shift weight (MSPF) (Korhonen et al., 2005), and a combined CC tracker with particle filtering to solve collisions (CCMSPF).

We first compare these trackers on partial and complete occlusion sequences, see table 1. We use our dataset showing shopping scenarios and videos from PETS 2006 (Pet, 2013). The number of occlusions correctly handled on the total number of occlusions are presented in table 1. We note that tests are achieved on more occlusions sequences for our technique. In fact, the other methods require a longer initialization phase. It is therefore not possible to compare results on several videos because the other methods do not track the objects fast enough.

We finally achieve the task of counting cars for traffic monitoring purposes, see table 1. We count only objects that are tracked for at least 30 frames. Based on the same detection, we test the various tracking algorithms. Our method outperforms the other methods on this task and is the second fastest method, see table 1.

6 CONCLUSIONS

This paper presents our tracking algorithm. The method is based on several hypotheses of the moving objects appearances. We match moving regions

Table 1: Table relating the evaluation of several tracking algorithm on the task of counting vehicles and handling occlusions.

	Occlusions		# of vehicules		Execution time (s)	
video - frames	**LAB**	**PETS 2006**	**V1**	**V2**	**V1-3305**	**V2-2257**
Ground truth	N.A.	N.A.	213	133	N.A.	N.A.
Our method	**4|4**	**2|2**	**198**	**116**	128.8	86.5
CC	0|2	0|1	173	107	**87.2**	**60.4**
MS	**2|2**	**1|1**	162	90	190.8	134.5
CC-MSPF	**2|2**	**1|1**	175	102	743.3	180.7
MSPF	2|2	0|1	106	65	1773.2	1306.1

on several levels to cope with uncertainty and we detect splits and merges to detect occlusions.

We compare our method with other tracking method such as connected components, mean-shift, particle filtering, and a combination of connected components and particle filtering to manage occlusions.

The proposed method can track more than a dozen objects simultaneously. We track various types of objects: deformable, non-deformable, with different sizes. Tracking works indoors or outdoors and handles various occlusions sequences. Finally, our system can be used for real-time applications.

ACKNOWLEDGEMENTS

ADACIS sarl and CETE sud-ouest provided the traffic sequences.

REFERENCES

(2013). *PETS: Performance Evaluation of Tracking and Surveillance*.

Hu, W., Tan, T., Wang, L., and Maybank, S. (2004). A survey on visual surveillance of object motion and behaviors. *Systems, Man and Cybernetics, Part C, IEEE Transactions on*, 34(3):334–352.

KaewTraKulPong, P. and Bowden, R. (2001). An improved adaptive background mixture model for real-time tracking with shadow detection. In *Proc. European Workshop Advanced Video Based Surveillance Systems*, volume 1. Citeseer.

Korhonen, T., Pertil, P., and Visa, A. (2005). Particle filtering in high clutter environment. In *Proceedings of the 2005 Finnish Signal Processing Symposium. FINSIG*.

Matas, J., Chum, O., Urban, M., and Pajdla, T. (2004). Robust wide-baseline stereo from maximally stable extremal regions. *Image and Vision Computing*, 22(10):761–767.

Moeslund, T. B., Hilton, A., and Krger, V. (2006). A survey of advances in vision-based human motion capture and analysis. *Computer Vision and Image Understanding*, 104(2-3):90 – 126.

Pinho, R. and Tavares, J. (2009). Tracking features in image sequences with kalman filtering, global optimization, mahalanobis distance and a management model.

Sicre, R. and Nicolas, H. (2010). Human behaviour analysis and event recognition at a point of sale. In IEEE, editor, *Proceedings of PSIVT PSIVT*.

Sicre, R. and Nicolas, H. (2011). Improved gaussian mixture model for the task of object tracking. In *Computer Analysis of Images and Patterns*, pages 389–396. Springer.

Stauffer, C. and Grimson, W. (2002). Adaptive background mixture models for real-time tracking. In *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, volume 2.

Yang, B. and Nevatia, R. (2012). An online learned crf model for multi-target tracking. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2034–2041. IEEE.

Yilmaz, A., Javed, O., and Shah, M. (2006). Object tracking: A survey. *Acm Computing Surveys (CSUR)*, 38(4):13.

Zhang, T., Ghanem, B., Liu, S., and Ahuja, N. (2012). Robust visual tracking via multi-task sparse learning. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2042–2049. IEEE.

Zivkovic, Z. and van der Heijden, F. (2006). Efficient adaptive density estimation per image pixel for the task of background subtraction. *Pattern recognition letters*, 27(7):773–780.