

Multi-Object Tracking Using an Adaptive Transition Model Particle Filter with Region Covariance Data Association

Hélio Palaio and Jorge Batista*
ISR-Institute of Systems and Robotics
DEEC-FCTUC, University of Coimbra, Portugal

Abstract

We present an approach for detection, labelling and tracking multiple objects through both temporally and spatially significant occlusions. The proposed method builds on the idea of multiple objects scenario where grouping and occlusions are a reality. To this end, the objects are represented by covariance matrices and particle filters perform the object tracking. We propose a different measurement for the particles weights and a new update for the objects descriptor in a Riemannian framework. The results show the effectiveness of the approach hereby proposed in very clutter scenes.

1. Introduction

Visual Surveillance Systems (VSS) have the purpose of searching possible targets to be followed in a given scene. To achieve this goal, the system needs to detect the objects, represent and discriminate those objects and track them. This task can be very complicated in realistic scenes, which may contain cluttered backgrounds, unknown number of objects, multiple interactions and occluded objects.

Our goal was to develop a robust solution able to track and label the targets in a very clutter scene with both temporally and spatially significant occlusions.

To this end, tracking is performed at both the region level and the object level. At the region level, a particle filter with an adaptive transition model is used to search for optimal region tracks. This limits the scope of object trajectories. At the object level, each object is located based on adaptive appearance models, spatial distributions and inter-occlusion relationships. Region covariance matrices are used to model objects appearance and the dissimilarity between region covariance matrices is used as a new measurement for the particle weights. The covariance matrices are updated using a novel approach in a Riemannian space. The proposed architecture is capable of tracking multiple objects even in the presence of periods of full occlusions using a simple and efficient solution for group handling and occlusion reasoning.

The results show the effectiveness of the approach hereby proposed.

2. Foreground Detection

In a surveillance system, the first step is to have a good process for detection of foreground regions. To accomplish this task we adopted the method proposed by [5] which is based on intrinsic images, where a scene is decomposed as a product between the background and foreground images. The background image is associated to the static constituent of the scene and the foreground image is associated to the dynamic constituent of the scene. So, the foreground objects, F_j , are the moving objects on the scene. Since our first purpose is to detect the moving objects, this approach fits perfectly.

We refer the readers to [5] for a detailed discussion on this method.

3. Object Descriptor

Generally, the most desirable property of a visual feature is its uniqueness so that the objects can be easily distinguished in the feature space. Feature selection is closely related to the object representation.

Let I be a three dimensional color image, and F a $w \times h \times d$ dimensional feature image, extracted from I : $F(x, y) = \Phi(I, x, y)$ where Φ represents the mapping such as intensity, color, gradients, etc. If we define a region R in image F , so that, $R \subset F$ then this region could be represented by a $d \times d$ covariance matrix \mathbf{C}_R . Defining Φ as $[x \ y \ I_r \ I_g \ I_b \ I_x \ I_y \ I_{xy}]$, where x and y are the pixel location in R ; I_r , I_g and I_b are the red, green and blue color components of I ; I_x and I_y are first derivatives of the grey image of I ; I_{xy} is the laplacian of the the grey image of I . In this way any region R is mapped into a 8×8 covariance matrix.

3.1 Riemannian Manifolds

An important issue is how to measure the dissimilarity between two covariance matrices and how to update the covariance matrix in the next time slot. The answer to both problems is reached in an Riemannian framework, more precisely on a Riemannian Manifold, which is a Manifold with a Riemannian metric. In the present work we use a metric proposed in [6] and [7] which is an invariant metric

*This was funded by FCT Project POSC/EEA-SRI/61150/2004

Figure 1. Object representation.

for the tangent space of symmetric positive definite matrices (e.g. covariance matrices) and is given by $\langle \mathbf{y}, \mathbf{z} \rangle_{\mathbf{X}} = \text{tr}(\mathbf{X}^{-\frac{1}{2}} \mathbf{y} \mathbf{X}^{-1} \mathbf{z} \mathbf{X}^{-\frac{1}{2}})$, where capital letters denote the points on the Manifold M and small letters correspond to vectors on the tangent space.

Let $\mathbf{y} \in T_{\mathbf{X}}M$, where $T_{\mathbf{X}}M$ is the tangent space at point $\mathbf{X} \in M$. There is a unique geodesic starting at \mathbf{X} with tangent vector \mathbf{y} . The exponential map, $\text{exp}_{\mathbf{X}} : T_{\mathbf{X}}M \mapsto M$, maps the vector \mathbf{y} to a point \mathbf{Y} belonging to the previous geodesic. We denote by $\text{log}_{\mathbf{X}}$ its inverse. The distance between \mathbf{X} and \mathbf{Y} is given by $d^2(\mathbf{X}, \mathbf{Y}) = \|\mathbf{y}\|_{\mathbf{X}}^2$ and the exponential map is given by,

$$\text{exp}_{\mathbf{X}}(\mathbf{y}) = \mathbf{X}^{\frac{1}{2}} \text{exp}(\mathbf{X}^{-\frac{1}{2}} \mathbf{y} \mathbf{X}^{-\frac{1}{2}}) \mathbf{X}^{\frac{1}{2}} \quad (1)$$

$$\text{log}_{\mathbf{X}}(\mathbf{Y}) = \mathbf{X}^{\frac{1}{2}} \text{log}(\mathbf{X}^{-\frac{1}{2}} \mathbf{Y} \mathbf{X}^{-\frac{1}{2}}) \mathbf{X}^{\frac{1}{2}} \quad (2)$$

If we use the definition of the geodesic distance and substituting (2) into our metric we have, $d^2(\mathbf{X}, \mathbf{Y}) = \text{tr}(\text{log}^2(\mathbf{X}^{-\frac{1}{2}} \mathbf{Y} \mathbf{X}^{-\frac{1}{2}}))$.

We refer the readers to [4] for a detailed discussion on the Riemannian Geometry with the used metric.

We propose a different method to update the covariance matrix which consists in a mean of the new covariance matrix and the last covariance updated. If \mathbf{y} is the velocity that takes us from X to Y , $\mathbf{y}/2$ will take us half the distance to point $\bar{\mathbf{C}}$. Using equations (1), (2) and the new velocity $\mathbf{y}/2$ the new mean covariance matrix is equal to

$$\bar{\mathbf{C}} = (\mathbf{X}^{\frac{1}{2}} \mathbf{Y} \mathbf{X}^{\frac{1}{2}})^{\frac{1}{2}} \quad (3)$$

where $\bar{\mathbf{C}}$ is the average distance between two points on a Riemannian Manifold. This update means that the last covariance is more important than the previous covariances. Since we are tracking objects that could change over time, the last information about it is more reliable.

3.2 Improvement to Occlusion

To improve the matching capacity of the system to occlusions we adopted a method proposed in [7] which consists in representing the object with five covariances, see figure 1. In this way we obtain a reasonable match even if one half of the image is occluded because the others regions will get good matches. The dissimilarity between two objects is obtained using

$$\rho(O_1, O_2) = \sum_{i=1}^5 d^2(\mathbf{C}_i^{O_1}, \mathbf{C}_i^{O_2}) \quad (4)$$

where $\mathbf{C}_i^{O_1}$ and $\mathbf{C}_i^{O_2}$ are the five covariance matrices of object 1 and object 2, respectively.

4. Particle Filter

The particle filter (PF) is a method widely used in the *Filtering and Data Association* class of tracking algorithms [1]. Our approach is based on the bootstrap filter [2].

Let $\{\mathbf{x}_t^{(1)}, \dots, \mathbf{x}_t^{(N_s)}\}$ and $\{\mathbf{z}_t, \dots, \mathbf{z}_t\}$ be, respectively, the samples and the observations up to time. The particle filter approximates the posterior distribution $p(\mathbf{x}_t | \mathbf{z}_{1:t})$ by a set of weighted samples $\{\mathbf{x}^{(i)}, w^{(i)}\}_1^{N_s}$. In our solution, the state estimate, $\hat{\mathbf{x}}_t$, can be approximate by the sample with the higher weight,

$$\hat{\mathbf{x}} = \mathbf{x}^j |_{w^j = \max(w)}. \quad (5)$$

The weights are computed based on the dissimilarity equation (4),

$$w^{(i)} = \text{exp}(-\rho(C(\mathbf{x}^{(i)}), \bar{\mathbf{C}})). \quad (6)$$

where $C(\mathbf{x}^{(i)})$ represents the five covariance matrices in region centred at $\mathbf{x}^{(i)}$. To simplify the notation we will refer to the five covariance matrices only as \mathbf{C} . Since the object may change rapidly, it isn't known the true size of the region in the present instant, so it is performed a search in five different scales based in the last size known. Therefore it is necessary to change equation 6, that will be given by

$$w^{(i)} = \max_j (\text{exp}(-\rho(\mathbf{C}^j(\mathbf{x}^{(i)}), \bar{\mathbf{C}}))). \quad (7)$$

where \mathbf{C}^j is the covariance matrices at scale j which is the index of scale, $Scale = \{100\%, 109\%, 118\%, 92\%, 86\%\}$.

The motion between two instants is described by the transition model. The ideal model has the exact kinematics of the object movement. However, that is not possible in practice, so approximated models are used. The most common is a fixed constant-velocity model with a fixed noise variance: $\mathbf{x}_t^{(i)} = \mathbf{x}_{t-1}^{(i)} + \mathbf{v} + \xi$, where \mathbf{v} is the velocity of the system and $\xi : \mathbf{N}(0, \Sigma)$ is a Gaussian noise. This kind of model presents some problems: the difficulty of a correct first velocity estimate; the dynamics usually don't follow a constant velocity model; and the trajectory changes during time.

In [8] it is proposed an adaptive model for a particle filter. Our approach is based on a adaptive velocity model. As time runs it is possible to compute a more accurate estimate for the true velocity of the system as an average velocity of the last k instants, $\mathbf{v}_t = \frac{1}{k} \sum_{n=t-k}^t |\mathbf{x}_n - \mathbf{x}_{n-1}|$.

5. Tracking Manager

At this stage it is important to clearly define the concept of an object and how it is integrated with the particle filter. An object can be of a single nature or a group object and represents an image tracked target. It is represented by the descriptor $O_n = [X, Y, S_n, N, L, Id]$, where X and Y are the target centre of mass coordinates, S_n is the PF filter parameters, N the number of targets associated to the object n , L is a list of pointers to the N object descriptors that form the group object ($N > 1$) and Id is the target label.

The descriptor $S_n = [Sz, \bar{\mathbf{C}}, \{w^i\}_1^{N_s}, \{\mathbf{x}^i\}_1^{N_s}, N_s, \mathbf{v}]$ is composed by: Sz the dimension of the target bounding box; $\bar{\mathbf{C}}$ the five covariance matrices of the target; w^i the weights of particle i ; \mathbf{x}^i , which is the vector with the coordinates of the particle i ; the number of particles N_s ; and the velocity of the model, \mathbf{v} .

5.1 Single Object Tracking

Let us consider the simple situation of one object tracked with the object descriptor $O_1(t)$ at time frame t . The estimate of the object mass centre coordinates in the next instant is given by $[\hat{X}, \hat{Y}]^T = \mathbf{x}^j|_{w^j = \max(w)}$. Then the object is associated to the detected blob given by the segmentation process. If the estimated coordinates are inside that blob the object descriptor is updated and $O_1(t+1) = \hat{O}_1(t)$.

5.2 Grouping and Occlusion Manager

When we have multiple objects to track, it is possible that there exist various candidates for a single blob, which results in a merge, or group objects that split. Our solution to handle this problem is based in a event manager but without constraint regarding to the number of objects, unlike the method proposed in [3].

Let assume that we have N objects \hat{O}_i with the position estimate given by equation (5) and M detected blobs F_j which are the foreground images. The detected blobs will also be called as targets. At a frame time t , to disambiguate the problem, the first step is to build a correspondence matrix between \hat{O}_i and F_j . The correspondence matrix (CM) is a $N \times M$ matrix, defined as follows

$$CM(i, j) = Blgs(\hat{O}_i, F_j), \quad \forall i \in 1 \dots N, j \in 1 \dots M \quad (8)$$

where $Blgs$ returns 1 if $[\hat{X}_i, \hat{Y}_i]^T \subset F_j$ and returns 0 otherwise. Defining $CM_l(i) = \sum_{j=1}^M CM(i, j)$, $CM_c(j) = \sum_{i=1}^N CM(i, j)$ our CM will be in the form

	F_1	F_2	...	F_M	CM_l
\hat{O}_1	1	0	0	1	2
\hat{O}_2	0	1	0	0	1
...
\hat{O}_N	0	1	0	0	1
CM_c	1	2	0	0	

Signal	Event
$CM_l > 1$	split
$CM_l = 0$	lost
$CM_c > 1$	merge
$CM_c = 0$	new
$CM_l = CM_c = 1$	stable

Table 1. Left- Correspondence Matrix; Right- Event table

Now, we define events based on the cardinality of CM_l and CM_c . Table 1 shows how the event appears. As far as the merging is concerned, the algorithm has to deal with occlusion and grouping. Its goal is to manage the paths of group objects and single objects, applying the dynamic model if occluded or updating otherwise. For splitting events, the algorithm has to disambiguate which objects are associated to different targets.

Based on the CM, four managers, running in cascade, were used to handle the image objects: split manager, merge manager, new/lost manager and update manager.

Three lists of objects were considered, the active list, occluded list and lost list corresponding, respectively, to the visibly objects, occluded objects and objects that were not detected for a while.

5.2.1 The Split manager

When a split event is detected, there are two possible situations that may occur: a group object split or a single object split.

The first case is the most common one. The single objects that compound the group object are detected in more than one target (F_j). Then the split manager pass the group object to an inactive state (lost list) and those single objects passes to the active list. The CM is rebuilt.

To handle the combined event split/merge it is necessary to ensure that for a group object \hat{O}_i that was associated to F_j , the single objects referenced in L , must be inside the target F_j . Otherwise, it occurs a split and merge. If it is the case, the group object passes to the lost list and the single objects referenced in L return to the active list. The CM is rebuilt.

Regarding a single object split, new objects are created and added the active list.

5.2.2 The Merge manager

When a merge situation is detected, a group object is created in the active list. In this situation the single objects that make the group object will go to the occluded list, the parameter L is filled with the Id of the single objects and the value N is set up.

5.2.3 New/Lost manager

When a new event is detected, it may take place one of two situations: there is a new object or it is a miss split. In the second case, it is performed a search for a group object near the new possibility. If a group object is detected, it is performed a match test between the single objects that compound it and the region of the new possibility. If a single object matches, it is associated to that region and the CM is rebuilt. Otherwise it is considered a new single object.

If it is detected a lost event, it means that an object could have really disappeared or it may be a single object that belongs to a group with a bad position estimation. If so, it means that the position estimation was made based only on the dynamic model of the filter. When that happens the position is set equal to the last known position and the group object passes to the active list. The CM is rebuilt and the split and merge manager are called again. If this is not the case, the object descriptor goes to the lost list.

5.2.4 Updating manager

The update manager just updates the state of the objects. The single objects are updated directly with the estimation

Figure 2. Results of a scene with five people that occludes each others

Figure 3. Object trajectories

\hat{O}_i . The group objects, before doing their own update, search for all single objects that compose them and, if not occluded, perform the single object update based on the position estimate, otherwise the update is performed based only on the equation of the dynamic model. The object is considered occluded when $\max(w) < th$, where w are the particles weight of the PF associated to the object and th is a defined threshold.

6. Results

The algorithm hereby presented was tested in different scenarios. We started by testing it in a two people crossing scenario, with excellent results, followed by a four people interacting scenario, and finishing with a five people scenario with multiple crosses. Figure 2 shows the results of the five people scenario. In this case the human figures pass by one another several times occluding each other. It is a very challenging scenario. Nevertheless, the method here proposed is ultimately able to correctly track and label all persons.

In figure 3 it is shown the object's trajectories of the five people scenario. Each object is represented by a color where the dashed lines mean a labelling when the object was grouped, the dots means that it was occluded (trajectories are based only in the dynamic model) and the continuous line means a labelling when the object was alone. In the black object trajectory it is possible to verify that when grouped it had some bad labellings.

Table 2 shows the accuracy of this method and we can observe that when the object is alone this method always does a correct labelling in this scenario. However when grouped only 75 % of correct labelling was achieved.

7. Conclusions

We proposed a complete system for detection, labelling and tracking multiple objects. A particle filter was also introduced with the dissimilarity between covariance matrices as a new measurement. In order to update those matrices, a novel solution was proposed in a Riemannian space. Furthermore, we presented a different manager for the merge/split problem. The system proved its effectiveness, even in a very clutter scene with multiple occlusions. It also has the advantage of tracking all kind of moving objects in scene. The update hereby proposed allows the newer object data to be more important (usually the most representative one).

	Labelling when Single	Labelling when Grouped	Correct Grouping	Correct Splitting
Miss Labeling	- -	12 %		
Bad Labeling	- -	13 %	9 / 10	8 / 9
Correct Labeling	100 %	75 %		

Table 2. Result of the tracking system

References

- [1] S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for on-line non-linear/non-gaussian bayesian tracking. In *IEEE Transactions of Signal Processing*.
- [2] A. Doucet, N. De Freitas, and N. Gordon, editors. *Sequential Monte Carlo methods in practice*. 2001.
- [3] P. Nillius, J. Sullivan, and S. Carlsson. Multi-target tracking - linking identities using bayesian network inference. In *CVPR*, 2006.
- [4] X. Pennec, P. Fillard, and N. Ayache. A riemannian framework for tensor computing. In *IJCV*, 2006.
- [5] F. Porikli. Multiplicative background-foreground estimation under uncontrolled illumination using intrinsic images. In *IEEE Computer Society Workshop on Motion and Video Computing*, 2005.
- [6] F. Porikli, O. Tuzel, and P. Meer. Covariance tracking using model update based on means on riemannian. In *CVPR*, 2006.
- [7] O. Tuzel, F. Porikli, and P. Meer. Region covariance: A fast descriptor for detection and classification. In *ECCV*, 2006.
- [8] S. K. Zhou, R. Chellappa, and B. Moghaddam. Visual tracking and recognition using appearance-adaptive models in particle filters. In *Image Processing, IEEE Transactions*, 2004.