Computer Vision and Pattern Recognition 2010

A Probabilistic Framework for Joint Segmentation and Tracking

Chad Aeschliman, Johnny Park, and Avinash C. Kak
Purdue University
http://rvl.ecn.purdue.edu/

Abstract

Most tracking algorithms implicitly apply a coarse segmentation of each target object using a simple mask such as a rectangle or an ellipse. Although convenient, such coarse segmentation results in several problems in tracking—drift, switching of targets, poor target localization, to name a few—since it inherently includes extra non-target pixels if the mask is larger than the target or excludes some portion of target pixels if the mask is smaller than the target. In this paper, we propose a novel probabilistic framework for jointly solving segmentation and tracking. Starting from a joint Gaussian distribution over all the pixels, candidate target locations are evaluated by first computing a pixel-level segmentation and then explicitly including this segmentation in the probability model. The segmentation is also used to incrementally update the probability model based on a modified probabilistic principal component analysis (PPCA). Our experimental results show that the proposed method of explicitly considering pixel-level segmentation as a part of solving the tracking problem significantly improves the robustness and performance of tracking compared to other state-of-the-art trackers, particularly for tracking multiple overlapping targets.

1. Introduction

This paper addresses two closely related problems in computer vision, tracking and segmentation. The tracking problem consists of determining the location of all objects of interest in each frame of a video sequence. This is a core problem in the area, often serving as a preprocessing step for applications such as automated video surveillance. The segmentation problem is to determine which pixels in an image frame were generated by each target of interest. This is also an important problem with applications such as object identification and activity recognition.

While these two problems may seem to be independent, they are in fact inseparable and, in practice, any approach to solving one of the problems typically involves solving the other either implicitly or explicitly. It is clear that by solving the segmentation problem, we can easily obtain a solution to the tracking problem. Although the segmentation algorithm may not explicitly contain a tracking component, we can, for example, find the centroid of the pixels returned by the algorithm and use that as a measure of the target's location.

While not as obvious, it is also true that to solve the tracking problem we must provide at least a crude solution to the segmentation problem. During tracking it is the location of each target's projection onto the image that is tracked. We assume that in every frame the appearance of some subset of the image pixels was generated by the target and that it is these pixels we use for tracking. However, we *cannot* say a priori which pixels in a new frame will be generated by a particular target and which will be generated by another target or the background. Hence, some method for determining which pixels are generated by the target must be included as part of every tracking algorithm. This is the segmentation problem.

Many tracking algorithms implicitly or explicitly use a coarse segmentation of each target object using a simple mask such as a rectangle or an ellipse. We begin by showing that this coarse segmentation results in several problems in tracking. We will then present a probabilistic framework for jointly solving tracking and fine, pixel-level segmentation . Finally, we will present experimental results that demonstrate the performance improvement achieved by explicitly considering the segmentation compared to other state-of-the-art tracking algorithm that employ implicit segmentation.

2. Background

Numerous algorithms have been developed to address the tracking problem [19], and we will not attempt an exhaustive review. Tracking methods may be broadly categorized by the types of features used for tracking, e. g., brightness, edges, colors, etc. and whether the method supports multiple targets. We now consider approaches relevant to the current work, focusing on the implicit or explicit segmentation employed by the method and how this segmentation affects the tracking results.

Early multi-target tracking algorithms such as multiple hypothesis tracking [15] and the joint probabilistic data association filter [9] were developed in the context of radar and sonar tracking. In this context, there is no segmentation problem to solve since each target is in a sense one "pixel" in size.

More recently, particle filters have been widely used in tracking algorithms [10]. Khan et al. [11] give an interesting extension to the multi-target case, which is particularly relevant to the current discussion. The tracking uses a multivariate Gaussian over the brightness of the pixels contained in the target. The segmentation is a rectangle of fixed size. As the authors show in their results, the tracking breaks down when the targets overlap. This is to be expected since, under fixed segmentation, the same pixels are in a sense claimed by multiple targets when they overlap. The resulting ambiguity may cause track failure or switching of targets.

Another recent multi-target tracking algorithm based on Bayesian inference is given in [14]. Again a fixed rectangular segmentation is used. The appearance model is a multivariate Gaussian which is updated using probabilistic principal components analysis (PPCA). Our algorithm uses a similar probabilistic framework, but with a critical difference of incorporating fine segmentation into the tracking. A direct comparison of results between the method of [14] and ours is given in section 6.

The segmentation problem has also been extensively studied. A comparison of many methods is given in [13]. One common method is background subtraction. Under this framework, a model for the background is learned a priori and used to identify pixels that do not belong to the background and hence belong to one of the targets. An effective technique for doing this is based on training a mixture of Gaussians for each pixel [17]. A complement to background subtraction is appearance based methods, which learn an appearance model for each target and use this model to decide which pixels come from the target. Comaniciu et al. [8] use a color histogram as the model for the target. The method proposed in section 4.1 takes advantage of the appearance model already learned as part of tracking for computational efficiency.

Some effective tracking algorithms have used segmentation as the primary vehicle for tracking. In the CAMSHIFT [5] algorithm, a probability model is learned over various local image features (e. g. hue, gradient, etc.) for target pixels. This model is then used in the next frame to identify which pixels have a high probability of being part of the target. These probabilities are in turn used to simultaneously find both the center of the target and its approximate size. The ensemble tracker [3] is based on a similar idea but uses an AdaBoost classifier instead of a probability model to determine which pixels make up the target. Both the CAMSHIFT and the ensemble tracker use a single model across all pixels for determining which pixels make up the target, thus they work best when the appearance is fairly uniform within each target but at the same time distinctive against other targets. For some applications, e. g. face tracking, a single model for all pixels can work quite well. However, in a general tracking context, these methods may not be effective. Some recent work has used a bag of pixels approach to help compensate for this deficiency [4].

While outside of the scope of this work, a related technique for combined tracking and segmentation is based on contour tracking [7, 16]. While these approaches seem best suited for single target tracking, Bugeau and Peréz give a multi-target version in [6]. However, it is difficult to determine the robustness of their technique for splitting merged targets—a major concern for all contour based algorithms.

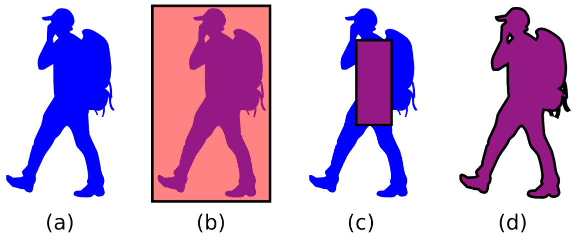

Figure 1. Comparison of coarse and fine segmentation. Blue pixels represent the target with a semi-transparent red overlay indicating the segmentation (best viewed in color). (a) target; (b) coarse segmentation with extra pixels; (c) coarse segmentation with missing pixels; (d) fine segmentation.

3. Coarse Segmentation and Its Problems

Recall that the segmentation problem consists of determining which pixels were generated by a particular target in each frame. One easy way to handle the segmentation problem is to perform a very coarse segmentation based on a mask with a fixed shape. To further simplify the problem, a simple geometric shape such as a rectangle or an ellipse can be used. In each frame the mask is applied (often implicitly) around a candidate target location, and the resulting pixels are used to evaluate if the location is suitable for tracking. A variation on this method is to support scaling of the mask in the x and y directions as is done in CAMSHIFT [5]. Often the same mask is used to determine which pixels should be included when updating the model of the target.

Unfortunately, using a fixed shape mask for segmentation results in several tracking problems. Allowing scaling of the mask mitigates these problems to some extent, but does not address the fundamental issues. Figure 1 highlights some of the difficulties in using a fixed shape mask with complex targets, such as the walking person shown in (a). First, the mask will, under any realistic scenario, include some extra pixels that are not part of the target as shown in Figure 1(b). As the tracking algorithm attempts to track all of the pixels indicated by the segmentation, these extra pixels will introduce drift since the background and target do not move together. Also, if another moving object passes in front of the target, then the combination of occlusion of the target being tracked and a significant number of extra pixels from the occluding object can cause the tracker to switch targets.

Another difficulty in using a coarse segmentation in tracking arises from the fact that the segmentation, in general, will miss some pixels that belong to the target as shown in Figure 1(c). The result is less robust tracking since some of the useful information available is being ignored. This can be particularly important in the case of occlusions since a partial occlusion of the target could be a full occlusion of the pixels within the tracker window. Furthermore, the pixels in the center of the target in general have a similar appearance. Therefore, a tracker that includes also the boundary pixels of the target will achieve more precise lo-

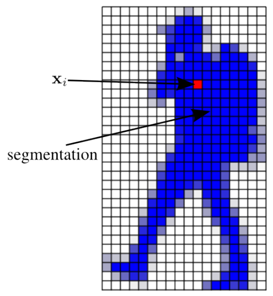

Figure 2. Target representation. The red pixel labelled $\mathbf{x}_i$ is an arbitrary reference pixel. The opacity of the blue pixels represents the segmentation information (best viewed in color).

calization.

It should be noted that both of the problems associated with coarse segmentation have serious effects on tracking , particularly for multiple nearby or overlapping targets as we will show in Section 6. For this reason we propose a method which combines both tracking and fine segmentation of the targets, as illustrated in Figure 1(d).

4. Joint Segmentation and Tracking

As shown in Figure 2, each target is represented by two pieces of information: the location $\mathbf{x}_i = [x_i, y_i]^T$ of a reference pixel on the target and the segmentation of the target (the probability for each pixel in the image that it was generated by the target). The reference pixel could be initialized by a user click or an initial object detection algorithm. The absolute location of the reference pixel is not particularly important; however, the shift in $\mathbf{x}_i$ from frame to frame is indicative of the general shift of all pixels making up the target. Also, $\mathbf{x}_i$ provides a point of reference on the target for defining the coordinates of pixels relative to the target. The segmentation encodes the shape and size of the target. Furthermore, the centroid of the segmentation is taken to indicate the center of the target and is used for comparison against ground truth target centers in Section 6.

We address both tracking (updating the reference location for all targets) and segmentation jointly as follows:

1. Propose a candidate solution for the target locations $\mathrm{X} = [\mathbf{x}_1\, \mathbf{x}_2\, \ldots\, \mathbf{x}_M]$ where M is the number of targets
2. Using the proposed solution X, compute the segmentation for each target and the background using the method described in Section 4.1
3. Compute the probability of X given the input image $\mathbf{z}$, $P\{\mathrm{X}|\mathbf{z}\}$, as explained in Section 4.2
4. Repeat the process from step 1, keeping the locations X which maximize $P\{\mathrm{X}|\mathbf{z}\}$

Mathematically, our estimate for the target locations X in the kth frame is given by

$$\mathrm{X}^{(k)} = \arg\max_{\mathrm{X}} P\{\mathrm{X}|\mathbf{z}\} \tag{1}$$

Note that throughout this paper a superscript will be used, when necessary, to denote the frame number. This formulation is referred to as a maximum a posteriori (MAP) estimator.

Alternatively, instead of choosing the value of X which maximizes $P\{\mathrm{X}|\mathbf{z}\}$, we could update the full probability distribution for each frame using, for example, a particle filter. However, $P\{\mathrm{X}|\mathbf{z}\}$ is a joint distribution over the location of multiple targets and hence lies in a high dimensional space. Consequently a large number of particles may be required for an accurate characterization of the distribution. Therefore we follow the method of a similar multi-target tracker in Nguyen et al. [14] and use a MAP estimator for simplicity and computational efficiency.

4.1. Segmentation

The goal of segmentation is to determine for each pixel the probability that it was generated by each of the targets and the background, given a set of target locations X. This can be done using Bayesian inference. Let $g_i(\mathbf{p})$ be the event that the pixel located at $\mathbf{p}$ was generated by the ith target. Taking the pixel brightness z as the evidence and conditioning on X throughout, the probability distribution for the parameter $g_i(\mathbf{p})$ is given by

$$P\{g_i(\mathbf{p})|z, \mathrm{X}\} = \frac{P\{z|g_i(\mathbf{p}), \mathrm{X}\} P\{g_i(\mathbf{p})|\mathrm{X}\}}{\sum_{j=0}^{M} P\{z|g_j(\mathbf{p}), \mathrm{X}\} P\{g_j(\mathbf{p})|\mathrm{X}\}} \tag{2}$$

where target 0 represents the background.

The prior distribution $P\{g_i(\mathbf{p})|\mathrm{X}\}$ is defined to be

$$P\{g_i(\mathbf{p})|\mathrm{X}\} \propto \begin{cases} k_{background} & i = 0 \\ s_i^{(k-1)}(\mathbf{p} - \Delta\mathbf{x}_i) + k_{target}, & else \end{cases} \tag{3}$$

where $s_i^{(k-1)}(\mathbf{p} - \Delta\mathbf{x}_i)$ is the posterior probability from the preceding frame with $\Delta\mathbf{x}_i = \mathbf{x}_i - \mathbf{x}_i^{(k-1)}$ accounting for the shift in the target location. The constants $k_{background}$ and k_{target} are included in the prior as regularization terms to control how easy it is for a pixel to switch from the target to the background and vice versa.

The likelihood distribution $P\{z|g_i(\mathbf{p}), \mathrm{X}\}$ is defined to be a Gaussian distribution

$$P\{z|g_i(\mathbf{p}), \mathrm{X}\} \triangleq \mathcal{N}(\hat{\mu}, \hat{\sigma}^2) \tag{4}$$

where $\hat{\mu}$ is the mean and $\hat{\sigma}^2$ is the variance. The values of $\hat{\mu}$ and $\hat{\sigma}^2$ are obtained by marginalizing the full multivariate Gaussian distribution over the appearance of the target which is computed using the techniques described in Section 5.

Although at first glance the need to recompute $P\{g_i(\mathbf{p})|z, \mathrm{X}\}$ whenever one of the targets changes position may seem to be an excessive computational burden, in practice the computation can be made more efficient by enforcing a simple constraint. Note that both the likelihood distribution (Eq. (4)) and the prior distribution (Eq. (3)) depend only on the location of the associated target, i. e. they are independent of the other targets. Hence by considering a change in the location of only one target at a time, the likelihood and prior for the other targets remain unchanged and only the denominator of Eq. (2)—which is common between all targets—will change for these targets.

It is important to note that the sum of $P\{g_i(\mathbf{p})|z, \mathrm{X}\}$ over all targets and the background is unity for each pixel. This has two important effects. First, every pixel must be accounted for by one or more targets. Because of this, tracking failures in which multiple trackers jump to the same target are naturally avoided. If multiple trackers jump to the same target then the background is left to account for the pixels of the remaining targets. Since it is very unlikely that the background model will match the appearance of the targets, such an arrangement will not maximize $P\{\mathrm{X}|\mathbf{z}\}$. Second, multiple targets cannot fully claim the same pixel. As will be seen in the next section, this means that disputed pixels have a less of an impact in determining the target locations, thus improving robustness.

4.2. Tracking

Recall that the tracking solution is obtained by maximizing $P\{\mathrm{X}|\mathbf{z}\}$. By applying Bayes' theorem we obtain

$$P\{\mathrm{X}|\mathbf{z}\} = \frac{P\{\mathbf{z}|\mathrm{X}\}P\{\mathrm{X}\}}{P\{\mathbf{z}\}}. \tag{5}$$

Noting that $P\{\mathbf{z}\}$ is independent of X, we replace Eq (1) by the equivalent expression

$$\mathrm{X}^{(k)} = \arg\max_{\mathrm{X}} P\{\mathbf{z}|\mathrm{X}\}P\{\mathrm{X}\}. \tag{6}$$

Eq. (6) depends on two probability distributions, the likelihood distribution $P\{\mathbf{z}|\mathrm{X}\}$ and the prior distribution $P\{\mathrm{X}\}$. Note that because we are using a MAP approach, we do not need to normalize Eq. (6). This simplifies the definition of the likelihood and prior distributions.

4.2.1 Prior Distribution

The prior distribution $P\{\mathrm{X}\}$ in Eq. (6) has two roles in the optimization problem. First, it allows us to incorporate prior knowledge about where the targets may be located. Second, it allows us to constrain the space of possible positions X, reducing the computational load on the tracker and improving robustness. Under the assumption that the target has a limited maximum speed but can change direction arbitrarily, a suitable prior for each target is a uniform distribution over a fixed radius from the location of the target in the preceding frame. Specifically the prior is defined as

$$P\{\mathrm{X}\} \triangleq \prod_{i=1}^{M} P\{\mathbf{x}_i|\mathbf{x}_i^{(k-1)}\} \tag{7}$$

with

$$P\{\mathbf{x}_i|\mathbf{x}_i^{(k-1)}\} \propto \begin{cases} 1, & d(\mathbf{x}_i, \mathbf{x}_i^{(k-1)}) \le r \\ 0, & \texttt{otherwise} \end{cases} \tag{8}$$

where $d()$ is the Euclidean distance and r is a fixed parameter which must be specified. Note that for the sake of defining the prior, we have considered the location of each target to be independent of the others.

4.2.2 Likelihood Distribution

The likelihood distribution $P\{\mathbf{z}|\mathrm{X}\}$ specifies the probability of observing a particular set of intensity values in the image conditioned on the positions of the targets. Assuming that the appearance distribution is proportional to a product of multivariate Gaussian distributions, we have

$$P\{\mathbf{z}|\mathrm{X}\} \propto \prod_{i=0}^{M} \mathcal{N}(\mu_i, \hat{\Sigma}_i). \tag{9}$$

Using the techniques outlined in Section 5, we can compute a multivariate Gaussian distribution over pixel intensities for each target. with an associated mean vector μ_i and covariance matrix Σ_i. The mean vector is used directly in Eq. (9), but the covariance matrix is modified to account for the segmentation information. We wish to modify the covariance matrix for each Gaussian so that the following two conditions are met:

1. With respect to the ith target, if the segmentation for a particular pixel is close to 0, then this pixel should have no effect on the estimated location of the ith target;

2. With respect to the ith target, as the segmentation for a particular pixel goes to 0, this pixel becomes uncorrelated from all other pixels.

These conditions can be met by weighting the diagonal entries of the covariance matrix. Specifically if we have $\Sigma_i = [\sigma_{jk}]$ and $\hat{\Sigma}_i = [\hat{\sigma}_{jk}]$ then

$$\hat{\sigma}_{jk} = \begin{cases} w_j \sigma_{jk}, & j = k \\ \sigma_{jk}, & \text{else} \end{cases} \tag{10}$$

with

$$w_j = \frac{1}{\left(P\{g_i(\mathbf{p}_j)|z_j, \mathrm{X}\}\right)^2} \tag{11}$$

where we assume that the probability the ith target generated any given pixel is always greater than 0, which prevents w_j from being unbounded. This choice for $\hat{\Sigma}_i$ is very efficient to compute and ensures that the conditions specified above are satisfied.

We now show that $\hat{\Sigma}_i$ satisfies the two conditions given above. For the first condition, note that as the segmentation for a pixel goes to 0, the corresponding diagonal entry in $\hat{\Sigma}_i$ goes to ∞. Also, it can be shown that the corresponding row and column of $\hat{\Sigma}_i^{-1}$ go to 0. Hence the value of the quadratic form in the exponent of $\mathcal{N}(\mu_i, \hat{\Sigma}_i)$ will be independent of the intensity of the pixel.

For the second condition, we simply need to write out the expression for the correlation coefficient between two pixels. The correlation coefficient between the jth and kth pixels is given by

$$\begin{aligned}\rho_{jk} &= \frac{\hat{\sigma}_{jk}}{\sqrt{\hat{\sigma}_{jj}\hat{\sigma}_{kk}}} \qquad (12)\\ &= P\{g_i(\mathbf{p}_j)|z_j, \mathrm{X}\}P\{g_i(\mathbf{p}_k)|z_k, \mathrm{X}\}\frac{\sigma_{jk}}{\sqrt{\sigma_{jj}\sigma_{kk}}}.\end{aligned}$$

Thus, as either $P\{g_i(\mathbf{p}_j)|z_j, \mathrm{X}\}$ or $P\{g_i(\mathbf{p}_k)|z_k, \mathrm{X}\}$ goes to 0, the correlation coefficient ρ_{jk} goes to 0, and hence the jth and kth pixels become uncorrelated.

Note that the individual target probability distributions are typically only defined over a small set of pixels around the target location $\mathbf{x}_i$. Theoretically we can imagine simply adding extra pixels to the distribution with the corresponding segmentation set to 0 in order to account for all of the pixels in $\mathbf{z}$. In reality, because these pixels do not contribute to tracking, they can be ignored in the implementation, and only a subset of the pixels in $\mathbf{z}$ are considered when evaluating the target probabilities.

The product in Eq. (9) includes a target 0 which represents the background. We can use any background training and update method which gives a multivariate Gaussian distribution over the pixel intensities. For ease of implementation, we imposed two constraints although these constraints are not a requirement of the proposed tracking method. First, the covariance matrix is assumed to be a scaled identity matrix. Second, the model is determined a priori and is not updated during tracking.

5. On-line Update of the Probability Model

As explained in section 4.2.2, our model for the likelihood distribution of each target used in tracking is given by a modified multivariate Gaussian over the intensity of the pixels generated by the target with parameters μ and Σ (which should not be confused with the modified covariance matrix $\hat{\Sigma}$ presented in Section 4.2). These parameters must be updated in an on-line manner in order to take into account all of the available information for each target. Furthermore, the update should take into account the probability that each pixel within the image was generated by the target, i. e. the segmentation information. If a pixel has a low probability of being generated by the target, then the statistics associated with it should not change significantly.

5.1. Updating μ

The mean vector μ is easily updated in a way consistent with the segmentation information using a simple modification of the sample mean. Note that the sample mean can be expressed as a weighted average with all samples given the same weight. To take into account the segmentation information, we perform a weighted average using the probability it was generated by the target. Specifically, we define the update for the ith entry of μ as follows

$$\mu_i^{(k)} \triangleq \frac{1}{\sum_{j=1}^{k} s_i^{(j)}}\left(\mu_i^{(k-1)}\sum_{j=1}^{k-1} s_i^{(j)} + b_i s_i^{(k)}\right) \qquad (13)$$

where $s_i^{(j)} \in [0, 1]$ is the probability the pixel was generated by the target in the jth frame, and b_i is the intensity of the pixel in the current (kth) frame. Note that if a pixel has no probability of being generated by the target, i. e. $s_i^{(k)} = 0$, then $\mu_i^{(k)} = \mu_i^{(k-1)}$. Furthermore, if the segmentation $s_i^{(j)}$ is the same for all j, then Eq. (13) reduces to the sample mean.

5.2. Updating Σ

To reduce the computational burden of determining and using the covariance matrix required by this distribution, we use probabilistic principal component analysis (PPCA) to approximate the covariance [18]. In PPCA the covariance matrix is approximated by

$$\Sigma^{(k)} = \sigma^2 \mathrm{I}_n + \mathrm{W}^{(k)}\left(\mathrm{W}^{(k)}\right)^T \qquad (14)$$

with $\mathrm{W}^{(k)} \in \mathbb{R}^{n\times d}$ where n is the number of pixels being considered and d is small. The user specified parameter σ^2 represents the variance which is not captured in $\mathrm{W}^{(k)}\left(\mathrm{W}^{(k)}\right)^T$. In practice, we set σ^2 to the variance observed in the intensity of background pixels, reasoning that a similar noise level should apply to both the targets and the background. Decomposing the covariance matrix according to Eq. (14) has several important advantages. First, because we typically have $d \ll n$, we can use the Woodbury formula to efficiently compute the inverse of Σ, which is required to evaluate the multivariate Gaussian distribution. Second, there are fast ways to incrementally update W which can be modified to take into account the segmentation information.

Lin et al. [12] provide a method for on-line updating of the W matrix in Eq. (14). With a modification to their method, we can also incorporate the segmentation information into the update.

We begin by noting that if $\mathrm{V}^{(k-1)} \in \mathbb{R}^{n\times d}$ is a matrix whose columns are eigenvectors of $\Sigma^{(k-1)}$ and $\Lambda^{(k-1)}$ is a diagonal matrix of the corresponding eigenvalues, then

$$\mathrm{W}^{(k-1)} = \mathrm{V}^{(k-1)}\left(\Lambda^{(k-1)} - \sigma^2 \mathrm{I}_d\right)^{1/2}. \qquad (15)$$

Because of this, we can update W and hence Σ by simply updating the eigenvalues and eigenvectors of Σ to account for the new data $\mathbf{z}^{(k)}$. We first form

$$\begin{aligned}\tilde{\mathrm{W}} &= (\mathrm{I}_n - \mathrm{Q})^{1/2}\mathrm{W}^{(k-1)} \qquad (16)\\ \tilde{\mathbf{y}} &= ((\mathrm{I}_n - \mathrm{Q})\mathrm{Q})^{1/2}\mathbf{y} \qquad (17)\end{aligned}$$

where Q is a diagonal matrix with entries given by

$$q_{ii} = \frac{s_i^{(k)}}{\sum_{j=1}^{k} s_i^{(j)}} \tag{18}$$

and

$$\mathbf{y} = \mathbf{z}^{(k)} - \mu^{(k-1)} \tag{19}$$

Intuitively we may think of $\tilde{\mathrm{W}}$ as a weighted version of the information in the current covariance matrix and $\tilde{\mathbf{y}}$ as a weighted version of the new data. We next compute the SVD of $\mathrm{E} = [\tilde{\mathrm{W}}\, \tilde{\mathbf{y}}]$ s. t.

$$\mathrm{E} = \mathrm{UDT}^T. \tag{20}$$

where U is orthonormal and D is a diagonal matrix. The first d columns of U are the updated eigenvectors of $\Sigma^{(k)}$. The corresponding eigenvalues λ_i can be obtained from D as follows:

$$\lambda_i = d_i^2 + \sigma^2 \tag{21}$$

where d_i is the ith diagonal entry of D.

5.3. Adding and Removing Pixels

To account for the dynamic size and shape of the target, we need to add or remove pixels as part of the model update. To identify new pixels to be added to the target, we first compute the probability that pixels near the target were generated by the background. If this probability is below a threshold, the pixel is added to the target. Adding a new pixel to the probability model can be accomplished as follows. First, we append the mean vector with the brightness of the pixel. Second, we append a 0 to each of the eigenvectors (since we do not yet know how this pixel may be correlated with the other pixels in the target). Using this technique, we are able to perform single click target initialization by starting with a small square of pixels around the selected point and adding pixels until the target is covered.

If the target shrinks in size over time, then there will be many extra pixels included in the probability model. An extra pixel is defined as one which is consistently unlikely to be generated by the target. While these extra pixels have almost no impact on the tracking performance, they do add to the computational requirements so it is convenient to remove them. A pixel can be removed from the model by simply removing its entry from the mean vector and from each of the eigenvectors of the covariance matrix.

6. Results

We present results for the tracker on one sequence from the PETS2001[2] data set and two video sequences from the CAVIAR [1] data set. The chosen sequences have multiple interacting targets in order to highlight the advantages of combining tracking with fine segmentation. It is important to note that, unlike many recent tracking algorithms, the method proposed in this paper is a generic technique not tailored towards people tracking. Despite this fact, as the results show, the method works well for this kind of application. Furthermore, because the proposed method is a generic tracking algorithm we also show results for an outdoor environment with a person and a car interacting. Complete results for these example videos can be seen at the following URL: http://rvl.ecn.purdue.edu/RVL/Research/JointSegTrack/

Table 1 gives quantitative results for the proposed method and for the method of [14], which we will refer to as the fixed segmentation method. The proposed method successfully tracks all targets with a high degree of accuracy for the full length of each video sequence while achieving a reasonable average frame rate for each sequence of 3.5-5.5 frames per second on a 2.7GHz processor. The fixed segmentation method was not able to track the targets for the full length of each sequence due to occlusions and drift. Therefore, the quantitative results reflect up until the first track failure. Even when considering only the frames for which the tracker was successfully tracking the targets, the fixed segmentation method gives a mean pixel error that is 2.5 times larger than the proposed method. By explicitly considering segmentation, we are able to significantly improve the localization of the targets.

Figure 3 shows four frames from the Meet_WalkSplit video sequence. The top row shows the center of each target from the published ground truth. The second row gives results of the fixed segmentation method using a 15×10 fixed segmentation, which includes only the center pixels of the target. Notice that as a result of ignoring many of the target pixels, there is some ambiguity when the targets are close to each other, causing both trackers to jump to a single target. The third row shows the results when using a larger segmentation window in which some pixels that do not belong to the target are included in the tracking. Again there are difficulties with the target ambiguity when the targets interact. Furthermore, because of drift, by the end of the sequence neither target is being tracked. The fourth row gives results for the ensemble tracker [3], which performs similarly as the fixed segmentation method. In particular, there is significant ambiguity between the targets when they overlap, resulting in both trackers jumping to a single target. Since the ensemble tracker does not include any global information about the targets, it is unlikely the method could be modified to avoid this problem.

The bottom row of Figure 3 shows the results for the proposed method, with the intensity of each pixel indicating the probability that pixel was generated by the target. The problems we encountered when using the fixed segmentation and ensemble tracking methods have been avoided. As can be seen in the second and third columns, even when one of the targets is partially occluding the other, correct tracking and segmentation is maintained, eliminating the target ambiguity. Comparing the first and last columns, the size of both targets changes dramatically, which shows the robustness of the proposed method against changes in scale. These results do show one of the weaknesses of the current approach to segmentation in that shadows and encoding artifacts are included as part of the target since they have low probability of being generated by the background.

Table 1. Quantitative evaluation against published ground truth data. The mean pixel error was computed only for the frames for which the fixed segmentation method was able to track all the targets. Note that the proposed method was able to track all targets for the full length of each video sequence. Average frame rate is for the proposed method and includes only the frames in which tracking was occurring.

Video Sequence	# Targets	Image Size	Average Frame Rate	Mean Pixel Error	
				Fixed Seg. [14]	Proposed Method
Meet_WalkSplit	2	384×288	3.81fps	6.01 pixels	2.25 pixels
Meet_Split_3rdGuy	3	384×288	4.16fps	8.55 pixels	2.94 pixels
PETS2001	2	768×576	5.47fps	13.51 pixels	4.94 pixels

Figure 4 shows results for the proposed method on the Meet_Split_3rdGuy video sequence. Of particular interest is the tracking of the person who enters from the bottom of the video sequence. As he passes through the area of bright sunlight, his appearance changes dramatically. However, since only a portion of the pixels change in any particular frame the remaining pixels can still be tracked resulting in accurate tracking of the target. The fixed segmentation and ensemble tracker methods were not able to maintain tracking through the appearance change.

As mentioned, the proposed method is not tailored to people tracking. Because of this it also works effectively on outdoor scenes with vehicles and people interacting. Figure 5 shows results from the PET2001 data set in which a vehicle passes in front of a person. This sequences demonstrates the flexibility of the approach to tracking a variety of targets.

7. Conclusions and Future Work

By jointly considering both tracking and segmentation, we are able to robustly track a variety of targets in difficult scenarios. In particular we show improved performance compared to other techniques when there are multiple overlapping targets. There are also some weaknesses in the current approach which we hope to address in the future. First, the probability model could be extended to include more than just intensity, e. g. adding features like color and local gradients. Also, we would like to move away from a static background model to increase the range of applications in which this tracking method can be applied.

Acknowledgement

The authors want to thank Sierra Nevada Corporation for their support with special thanks to Glenn Boudreaux and Trei Gilbert for many useful discussions related to object tracking.

References

[1] http://homepages.inf.ed.ac.uk/rbf/CAVIAR/. 6

[2] http://www.cvg.cs.rdg.ac.uk/PETS2001/. 6

[3] S. Avidan. Ensemble tracking. *IEEE transactions on pattern analysis and machine intelligence*, 29(2):261, 2007. 2, 6

[4] C. Bibby and I. Reid. Robust real-time visual tracking using pixel-wise posteriors. In *ECCV*, pages 831–844. Springer, 2008. 2

[5] G. Bradski. Computer vision face tracking for use in a perceptual user interface. *Intel Technology Journal*, 2(2):12–21, 1998. 2

[6] A. Bugeau and P. Pérez. Track and cut: simultaneous tracking and segmentation of multiple objects with graph cuts. In *Proceedings of the 3rd International Conference on Computer Vision Theory and Applications (VISAPP'08)*, pages 1–8, 2008. 2

[7] P. Chockalingam, N. Pradeep, and S. Birchfield. Adaptive fragments-based tracking of non-rigid objects using level sets. In *Proceedings of the International Conference on Computer Vision*, 2009. 2

[8] D. Comaniciu, V. Ramesh, and P. Meer. Kernel-based object tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(5):564–577, 2003. 2

[9] T. Fortmann, Y. Bar-Shalom, and M. Scheffe. Sonar tracking of multiple targets using joint probabilistic data association. *IEEE Journal of Oceanic Engineering*, 8(3):173–184, 1983. 1

[10] M. Isard and A. Blake. Contour tracking by stochastic propagation of conditional density. *Lecture Notes in Computer Science*, 1064:343–356, 1996. 1

[11] Z. Khan, T. Balch, and F. Dellaert. An MCMC-based particle filter for tracking multiple interacting targets. *Lecture Notes in Computer Science*, pages 279–290, 2004. 1

[12] R. Lin, D. Ross, J. Lim, and M. Yang. Adaptive discriminative generative model and its applications. *Advances in neural information processing systems*, pages 801–808, 2004. 5

[13] T. Moeslund, A. Hilton, and V. Krüger. A survey of advances in vision-based human motion capture and analysis. *Computer vision and image understanding*, 104(2-3):90–126, 2006. 2

[14] H. Nguyen, Q. Ji, and A. Smeulders. Spatio-temporal context for robust multitarget tracking. *IEEE transactions on pattern analysis and machine intelligence*, 29(1):52, 2007. 2, 3, 6, 7

[15] D. Reid. An algorithm for tracking multiple targets. *IEEE Transactions on Automatic Control*, 24(6):843–854, 1979. 1

[16] Y. Shi and W. Karl. Real-time tracking using level sets. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, page 34. Citeseer, 2005. 2

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

[18] M. Tipping and C. Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society. Series B, Statistical Methodology*, pages 611–622, 1999. 5

[19] A. Yilmaz, O. Javed, and M. Shah. Object tracking: A survey. *ACM Computing Surveys (CSUR)*, 38(4), 2006. 1

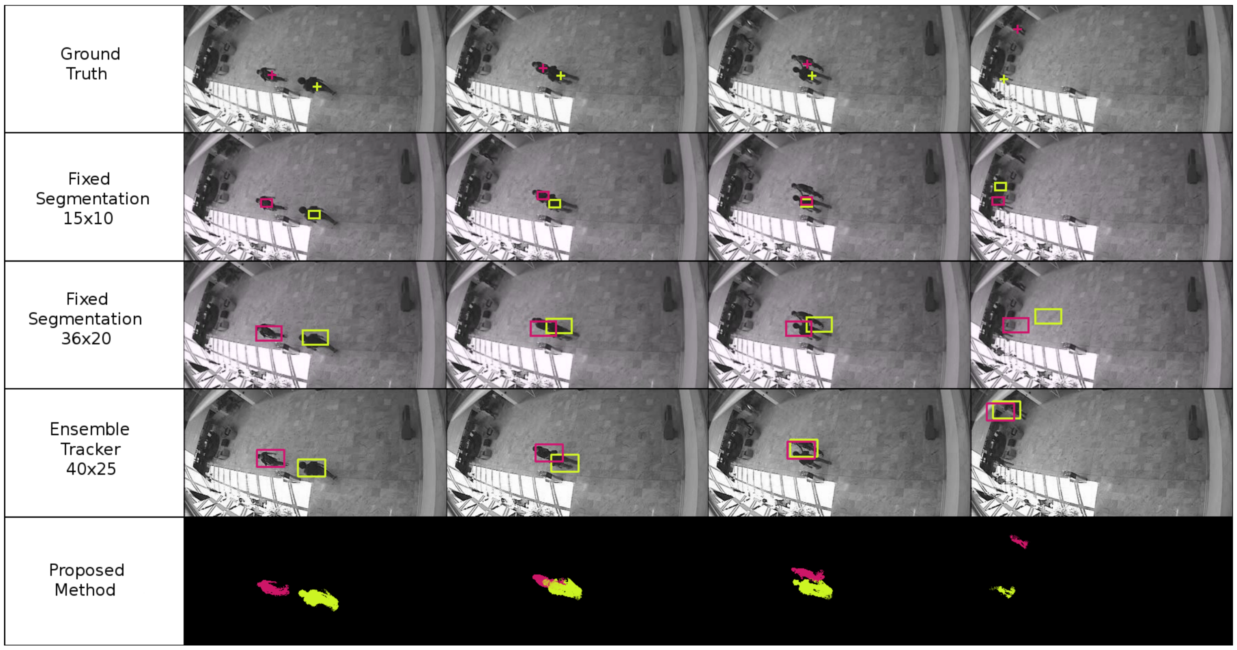

Figure 3. Results for Meet_WalkSplit from the CAVIAR data set (best viewed in color). The frames have been cropped to show detail.

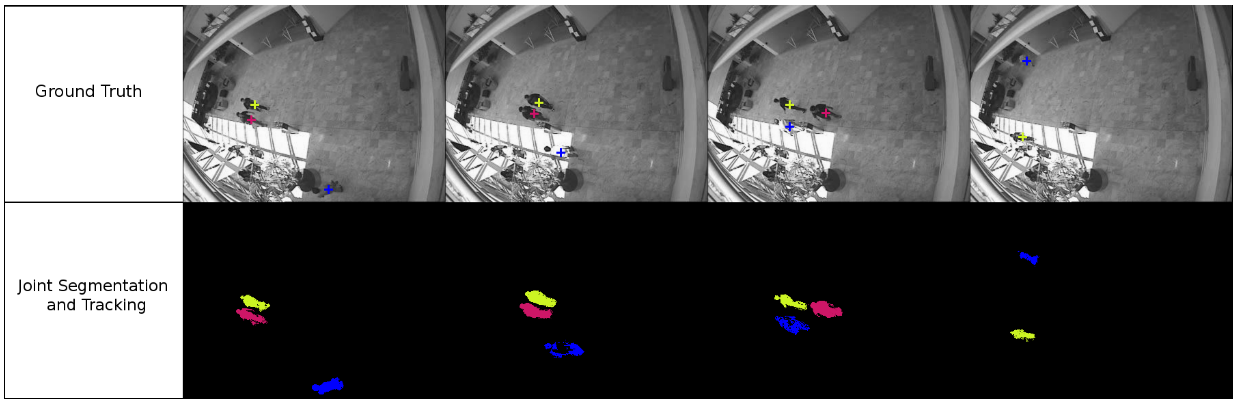

Figure 4. Results for Meet_Split_3rdGuy from the CAVIAR data set (best viewed in color).

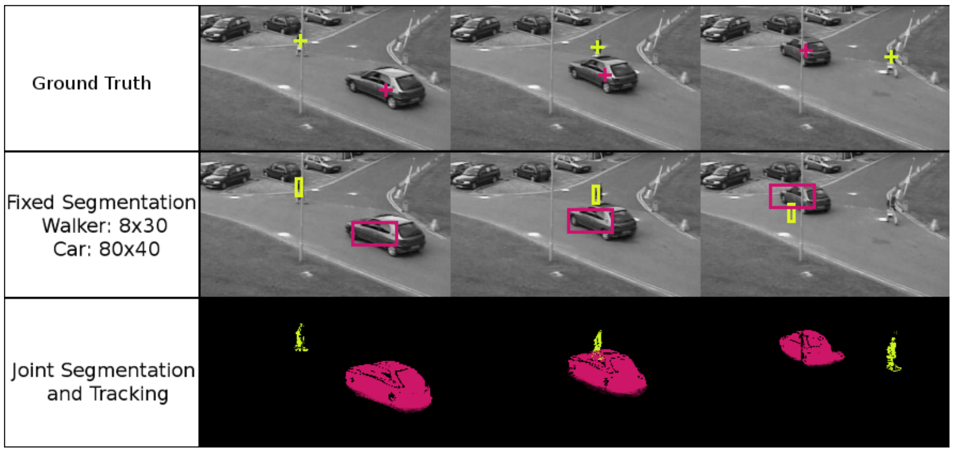
Ground Truth
Fixed Segmentation Walker: 8x30 Car: 80x40
Joint Segmentation and Tracking

Figure 5. Results for a video sequence from the PETS2001 data set (best viewed in color). The frames have been cropped to show detail.