International Conference on Computer Vision (ICCV) Kyoto, Japan, September 2009

Adaptive Fragments-Based Tracking of Non-Rigid Objects Using Level Sets

Prakash Chockalingam Nalin Pradeep Stan Birchfield
Electrical and Computer Engineering Department
Clemson University, Clemson, SC 29634
{cchocka, nsentha, stb}@clemson.edu

Abstract

We present an approach to visual tracking based on dividing a target into multiple regions, or fragments. The target is represented by a Gaussian mixture model in a joint feature-spatial space, with each ellipsoid corresponding to a different fragment. The fragments are automatically adapted to the image data, being selected by an efficient region-growing procedure and updated according to a weighted average of the past and present image statistics. Modeling of target and background are performed in a Chan-Vese manner, using the framework of level sets to preserve accurate boundaries of the target. The extracted target boundaries are used to learn the dynamic shape of the target over time, enabling tracking to continue under total occlusion. Experimental results on a number of challenging sequences demonstrate the effectiveness of the technique.

1. Introduction

Recent interest in visual tracking has centered around on-line learning of multiple cues to adaptively select the most discriminative ones. With this focus, significant progress has been achieved by algorithms such as those of Avidan [2], Collins *et al*. [6], and Grabner *et al*. [10]. In these approaches, tracking is formulated as a classification problem in which the probability of each pixel belonging to the target is computed. While the results have been promising, several limitations remain:

- Important but secondary cues are often ignored because of the employment of linear classifiers. As a result, even though the object may be tracked, many pixels that do not correspond to the dominant cue are misclassified when the data are not linearly separable. This limitation prevents an accurate determination of the target object's contour.
- Occlusion of the target can cause the learner to adapt to occluding surfaces, thus causing the model to drift from the target. A more accurate contour representation would enable such errors to be prevented.
- Spatial information that captures the joint probability of pixels is often ignored. While many tracking approaches use local spatial information in the form of texture measures or spatial means [2, 10], such methods do not take advantage of the wealth of information available in the global spatial arrangement of the pixels in the target which have proved useful in classic template-based and recent techniques [12, 14].

In this paper we present a technique that overcomes these limitations. Like Adam *et al*. [1], we split the target into a number of fragments to preserve the spatial relationships of the pixels. Unlike their work, however, our fragments are adaptively chosen according to the image data, by clustering pixels with similar appearance, rather than using a fixed arrangement of rectangles. This adaptive fragmentation captures all the secondary cues and also ensures that each fragment captures a single mode of the distribution. We classify individual pixels, as in [2, 6, 10], but by incorporating multiple fragments we are better able to preserve the shape of multi-modal targets. The boundary is represented by a level set using a Chan-Vese [5] model that enables level set tracking to be formulated in a Bayesian manner and leads to more stable convergence of the algorithm. This work extends the variational work of [21] by allowing multimodal backgrounds, extreme shape changes, and unpredictable motion. To address the problem of drastically moving targets with untextured regions, the recently proposed approach of [3] is employed to impose a global smoothness term in order to produce accurate sparse motion flow vectors for each fragment. The fragment models are updated automatically using the estimated contour and the image data, and the previous shapes are used to track the object through occlusion.

2. Approach

To represent the target being tracked, we use the formulation of level sets due to their numerical stability and their

©*2009 IEEE. Personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution to servers or lists, or to reuse any copyrighted component of this work in other works, must be obtained from the IEEE.*

ability to accurately represent a generic contour [15, 4]. Let $\Gamma(s) = [\,x(s) \quad y(s)\,]^T$, $s \in [0,1]$, be a closed curve in $\mathbb{R}^2$, and define an implicit function $\phi(x,y)$ such that the zeroth level set of ϕ is Γ, i.e., $\phi(x,y) = 0$ if and only if $\Gamma(s) = [x,y]^T$ for some $s \in [0,1]$. Let R^- be the region inside the curve (where $\phi > 0$) and R^+ the region outside the curve (where $\phi < 0$).

Our goal is to estimate the contour from a sequence of images. Let $I_t : \mathbf{x} \rightarrow \mathbb{R}^m$ be the image at time t that maps a pixel $\mathbf{x} = [\,x \quad y\,]^T \in \mathbb{R}^2$ to a value, where the value is a scalar in the case of a grayscale image ($m = 1$) or a three-element vector for an RGB image ($m = 3$). The value could also be a larger vector resulting from applying a bank of texture filters to the neighborhood surrounding the pixel, or some combination of these raw and/or preprocessed quantities. Similar to [21], we use Bayes' rule and an assumption that the measurements are independent of each other and of the dynamical process to model the probability of the contour Γ at time t given the previous contours $\Gamma_{0:t-1}$ and all the measurements $I_{0:t}$ of the causal system as

$$p(\Gamma_t|I_{0:t},\Gamma_{0:t-1}) \propto \underbrace{p(I_t^+|\Gamma_t)}_{\text{target}} \underbrace{p(I_t^-|\Gamma_t)}_{\text{background}} \underbrace{p(\Gamma_t|\Gamma_{0:t-1})}_{\text{shape}}, \tag{1}$$

where $I_t^+ = \{\xi_I(\mathbf{x}) : \mathbf{x} \in R^+\}$ captures the pixels inside Γ_t, $I_t^- = \{\xi_I(\mathbf{x}) : \mathbf{x} \in R^-\}$ captures the pixels outside Γ_t, and $\xi_I(\mathbf{x}) = [\,\mathbf{x}^T \quad I(\mathbf{x})^T\,]^T$ is a vector containing the pixel coordinates coupled with their image measurements.

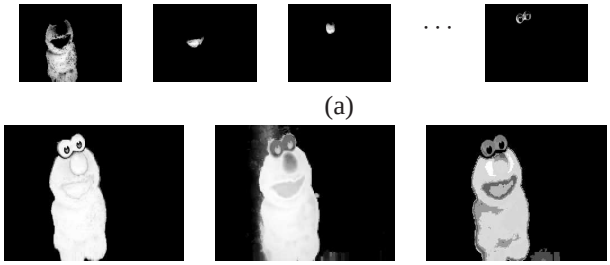
2.1. Fragment modeling

Assuming conditional independence among the pixels, the joint probability of the pixels in a region is given by

$$p(I_t^\star|\Gamma_t) = \prod_{\mathbf{x}\in R^\star} p_\star(\xi_I(\mathbf{x})|\Gamma_t), \tag{2}$$

where $\star \in \{-,+\}$. One way to represent the probability of a pixel $\xi_I(\mathbf{x})$ is to measure its signed distance to a separating hyperplane in $\mathbb{R}^n$, where $n = m + 2$, as in [2, 6], or using a single covariance matrix, as in [16]. A slightly more general approach would be to measure its Mahalanobis distance to a pair of Gaussian ellipsoids representing the target and background. None of these approaches, however, is able to capture the subtle complexities of multi-modal regions. As a result, we instead represent both the target and background appearance using a set of *fragments* in the joint feature-spatial space, where each fragment is a separate Gaussian ellipsoid, similar to [11]. Letting $\mathbf{y} = \xi_I(\mathbf{x})$ for brevity, the likelihood of an individual pixel is then given by a Gaussian mixture model (GMM):

$$p_\star(\mathbf{y}|\Gamma_t) = \sum_{j=1}^{k_\star} \pi_j p_\star(\mathbf{y}|\Gamma_t, j), \tag{3}$$



(a)

(b) (c) (d)

Figure 1. (a) Probabilities determined by individual fragments are combined to compute (b) our strength image. For comparison, the strength image computed using (c) a single Gaussian [16] and (d) a linear separation over a linear combination of multiple color spaces [6] are also shown. Our fragment-based GMM representation more effectively represents the multi-colored target.

where $\pi_j = p(j|\Gamma_t)$ is the probability that the pixel was drawn from the jth fragment, $k_\star$ is the number of fragments in the target or background, $\sum_{j=1}^{k_\star} \pi_j = 1$, and

$$p_\star(\mathbf{y}|\Gamma_t, j) = \eta \exp\left\{-\frac{1}{2}(\mathbf{y}-\mu_j^\star)^T \left(\Sigma_j^\star\right)^{-1} (\mathbf{y}-\mu_j^\star)\right\}, \tag{4}$$

where $\mu_j^\star \in \mathbb{R}^n$ is the mean and $\Sigma_j^\star$ the $n \times n$ covariance matrix of the jth fragment in the target or background model (depending upon $\star$), and η is the Gaussian normalization constant.

2.2. Computing the strength image

We follow the recent approach of formulating the object tracking problem as one of binary classification between target and background pixels [2, 10]. In this approach, a strength image is produced indicating the probability of each pixel belonging to the target being tracked. The strength image is computed using the log ratio of the probabilities:

$$S(\mathbf{x}) = \log\left(\frac{p_+(\mathbf{x})}{p_-(\mathbf{x})}\right) = \Psi^-(\mathbf{x}) - \Psi^+(\mathbf{x}), \tag{5}$$

where $\Psi^\star(\mathbf{x}) = -\log p_\star(\mathbf{x})$. Positive values in the strength image indicate pixels that are more likely to belong to the target than to the background, and vice versa for negative values. An example strength image is shown in Figure 1, illustrating the improvement achieved by considering spatial information. The strength image is used to update the implicit function, which enables the level set machinery to enforce smoothness on the resulting object shape.

2.3. Segmentation

Our fragment-based representation of the target is similar to that of Adam *et al*. [1] but with two significant differences. First, we use fragments to model the

background as well as the target, and secondly, our fragments are automatically determined and adapted by the image data rather than being fixed and hardcoded. The challenge is to compute the model parameters $\mu_1^+, \ldots, \mu_{k_+}^+, \Sigma_1^+, \ldots, \Sigma_{k_+}^+, \mu_1^-, \ldots, \mu_{k_-}^-, \Sigma_1^-, \ldots, \Sigma_{k_-}^-$ from the current contour Γ_t. This is essentially a problem of segmentation. We tried the graph-based algorithm of [9] but found it to unacceptably merge regions with distinct colors. We also experimented with mean-shift segmentation [7], but it was not only too slow for a tracking application but it also tended to oversegment the image. In addition, we considered the greedy Expectation-Maximization approach of Vlassis *et al*. [20], but its estimate of the number of components was too unreliable for our purposes.

Instead, we devised a region-growing algorithm, inspired by work on Spatially Variant Finite Mixture Models (SVFMM) [17, 18]. Initially a pixel in the image is selected at random, and a single fragment is created to hold the pixel. Neighboring pixels are added to the segment if they are within τ standard deviations of the Gaussian model of the fragment, with an appropriate relaxing of the threshold for small regions that do not yet have enough pixels for their model to be reliable. The mean $\mu_j^\star$ and covariance $\Sigma_j^\star$ are updated efficiently using a running accumulation of first- and second-order statistics. Once the fragment has finished growing, a new pixel is selected at random, and the procedure is repeated for a new fragment. This process continues until all pixels have been added to a fragment, at which point small fragments are discarded and the remaining fragments are labeled as target or background depending upon whether the majority of pixels are within or without a manually drawn initial contour Γ_0, respectively. Any fragment for which the pixels are roughly evenly distributed is split along Γ_0 to form two fragments, one labeled foreground and the other labeled background. Finally, we choose π_j based on the size of the fragments.

This efficient, simple procedure is quite effective at dividing the target and background into multiple fragments, as shown in Figure 2, and it is much faster than time-consuming EM [11]. For comparison, we also show the output of graph-based and mean-shift segmentations in Figure 3.

2.4. Level set formulation

Maximizing the probability of (1) is equivalent to minimizing the following energy functional over the level set function [5]:

$$E(\phi) = \int_{R^+} \Psi^+(\mathbf{x}) d\mathbf{x} + \int_{R^-} \Psi^-(\mathbf{x}) d\mathbf{x} + \mu\ell(\Gamma), \quad (6)$$

where μ is a scalar that weights the relative importance of the shape term, which is assumed for the moment to consist only in measuring $\ell(\Gamma)$, the length of the curve. At

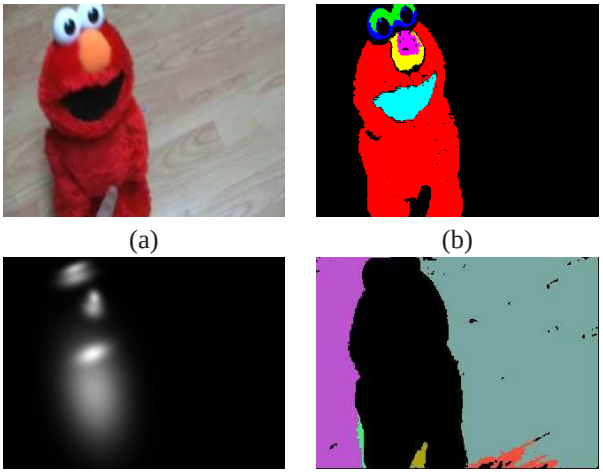


Figure 2. (a) Image of Elmo. (b) Foreground regions and (d) background regions found by our segmentation algorithm. (c) The six foreground spatial ellipsoids overlaid.

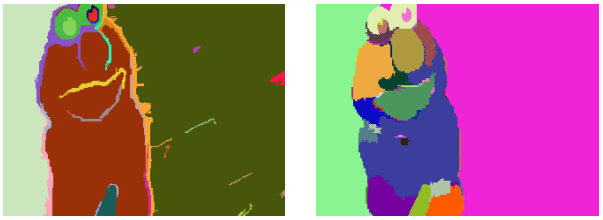


Figure 3. The output of competing algorithms on the Elmo image, for comparison. LEFT: Graph-based segmentation [9] accidentally merges regions with distinct colors. RIGHT: Mean-shift segmentation [7], even with a large scale parameter, oversegments the image.

this point we introduce the regularized Heaviside function $H(z) = \frac{1}{1+e^{-z}}$ as a differentiable threshold operator to rewrite the above as

$$E(\phi) = \int_\Omega H(\phi)\Psi^+(\mathbf{x}) + (1-H(\phi))\Psi^-(\mathbf{x}) + \mu|\nabla H(\phi)| d\mathbf{x}, \quad (7)$$

where $\ell(\Gamma) = \int_\Omega |\nabla H(\phi)| d\mathbf{x}$, and $\Omega = R^+ \cup R^-$ is the image domain. With $E = \int_\Omega F(x, y, \phi, \phi_x, \phi_y) d\mathbf{x}$, the associated Euler-Lagrange equation is given by

$$\begin{aligned} 0 &= \frac{\partial F}{\partial \phi} - \frac{\partial}{\partial x}\left[\frac{\partial F}{\partial \phi_x}\right] - \frac{\partial}{\partial y}\left[\frac{\partial F}{\partial \phi_y}\right] \\ &= h(\phi)\left(\Psi^+(\mathbf{x}) - \Psi^-(\mathbf{x}) - \mu \text{div}\left(\frac{\nabla\phi}{|\nabla\phi|}\right)\right), \end{aligned}$$

where $\phi_x = \partial\phi/\partial x$, $\phi_y = \partial\phi/\partial y$, $h(\phi) = \partial H/\partial\phi$, $\nabla\phi = [\,\phi_x \quad \phi_y\,]^T$ is the gradient of ϕ, and div is the divergence operator. To avoid the difficulty of solving this PDE explicitly for ϕ, we instead take the value on the right-hand side as an indication of the error, and apply gradient

descent iterations [5] with

$$\phi^{(k+1)} = \phi^{(k)} + |\nabla\phi| \left(\Psi^-(\mathbf{x}) - \Psi^+(\mathbf{x}) + \mu \text{div}\left(\frac{\nabla\phi}{|\nabla\phi|} \right) \right), \quad (8)$$

where k is the iteration number, and we have used the approximation $h(\phi) \approx |\nabla\phi|$, which is accurate as long as the level set function is smooth away from the boundary. The sign in the equation comes from the convention that $\phi > 0$ inside the boundary.

Note that unlike the traditional level set formulation, ours is not based upon intensity edges. Rather, we have adopted the Chan-Vese approach [5] of modeling the foreground and background regions explicitly. This approach results in a large basin of attraction, so that the iterations above will converge to the target from a wide variety of initial curves, without being significantly distracted by local noise in the data. Since the curve evolution is not required to be monotonic, the initial curve may be inside the target, outside the target, or some combination of the two. Note that our multi-modal spatial-feature models are able to capture much more complex targets than [5], in which the foreground and background regions are modeled simply by their average grayscale values.

2.5. Fragment motion

While the minimization above is not extremely sensitive to the initial contour, nevertheless it is beneficial for the coordinate systems of the target and the model fragments to be approximately aligned. Such alignment increases the accuracy of the strength image, due to the use of spatial information in the joint spatial-feature vectors. As a result we seek to recover, *prior* to computing the strength image, approximate motion vectors between the previous and current image frame for each fragment: $\mathbf{u}_i^\star = (u_i^\star, v_i^\star), i = 1, \ldots, k^\star$.

One way to solve this alignment problem would be to compute the motion of the target using traditional motion estimation techniques. However, existing dense motion algorithms do not perform well on complex imagery in which highly non-rigid, untextured objects undergo drastic motion changes from frame to frame, such as the videos considered in this work. Moreover, dense motion computation wastes precious resources for this application, since we only need approximate alignment between the fragments. In a similar manner, traditional sparse feature tracking algorithms are not suitable for recovering the motions of the individual fragments. Due to their independent handling of the features, such algorithms often yield some percentage of unreliable estimates.

To solve this dilemma, we utilize the recent joint feature tracking approach of [3]. Starting with the well-known *optic flow constraint equation*

$$f(u, v; I) = I_x u + I_y v + I_t = 0, \quad (9)$$

the traditional Lucas-Kanade and Horn-Schunck formulations are combined into a single differential framework. The functional to be minimized is given by

$$E_{JLK} = \sum_{i=1}^{N} (E_D(i) + \lambda_i E_S(i)), \quad (10)$$

where N is the number of feature points, and the data and smoothness terms are

$$E_D(i) = K_\rho * \left((f(u_i, v_i; I))^2 \right) \quad (11)$$

$$E_S(i) = \left((u_i - \hat{u}_i)^2 + (v_i - \hat{v}_i)^2 \right). \quad (12)$$

In these equations, the energy of feature i is determined by how well its motion $(u_i, v_i)^T$ matches the local image data, and by how far the motion deviates from the expected value $(\hat{u}_i, \hat{v}_i)^T$. The latter is computed by fitting an affine motion model to the neighboring features, where the connections between features are computed by a Delaunay triangulation.

Differentiating E_{JLK} with respect to the motion vectors $(u_i, v_i)^T$, $i = 1, \ldots, N$, and setting the derivatives to zero, yields a $2N \times 2N$ sparse matrix equation, whose $(2i-1)$th and $(2i)$th rows are given by

$$Z_i \mathbf{u}_i = \mathbf{e}_i, \quad (13)$$

where

$$Z_i = \begin{bmatrix} \lambda_i + K_\rho * (I_x I_x) & K_\rho * (I_x I_y) \\ K_\rho * (I_x I_y) & \lambda_i + K_\rho * (I_y I_y) \end{bmatrix}$$

$$\mathbf{e}_i = \begin{bmatrix} \lambda_i \hat{u}_i - K_\rho * (I_x I_t) \\ \lambda_i \hat{v}_i - K_\rho * (I_y I_t) \end{bmatrix}.$$

This sparse system of equations can be solved using Jacobi iterations of the form

$$\tilde{u}_i^{(k+1)} = \hat{u}_i^{(k)} - \frac{J_{xx}\hat{u}_i^{(k)} + J_{xy}\hat{v}_i^{(k)} + J_{xt}}{\lambda_i + J_{xx} + J_{yy}} \quad (14)$$

$$\tilde{v}_i^{(k+1)} = \hat{v}_i^{(k)} - \frac{J_{xy}\hat{u}_i^{(k)} + J_{yy}\hat{v}_i^{(k)} + J_{yt}}{\lambda_i + J_{xx} + J_{yy}}, \quad (15)$$

where $J_{xx} = K_\rho * (I_x^2)$, $J_{xy} = K_\rho * (I_x I_y)$, $J_{xt} = K_\rho * (I_x I_t)$, $J_{yy} = K_\rho * (I_y^2)$, and $J_{yt} = K_\rho * (I_y I_t)$. In practice, Gauss-Seidel iterations with successive overrelaxation yield increased convergence. An example output is shown in Figure 4.

Once the N features have been tracked, the mean motion vector of each fragment $\mathbf{u}_i^\star$ is computed using the motions of the features within the fragment. Note that there is little risk to this averaging, since outliers are avoided by the smoothness term incorporated by the joint Lucas-Kanade approach, which enables features to be tracked even in untextured areas, as shown in [3]. Feature selection is determined by those image locations for which $\max(e_{\min}, \eta e_{\max})$ is above a threshold, where $e_{\min}$ and $e_{\max}$ are the two eigenvalues of the 2×2 gradient covariance matrix, and $\eta < 1$ is a scaling factor.

Figure 4. Joint Lucas-Kanade (right) produces smoother motion vectors than standard Lucas-Kanade (left). The vectors are colored by the fragment in which they are contained.

2.6. Updating fragment models

This paper proposes *adaptive* fragments, i.e., fragments that are determined by the image data rather than being hardcoded. Once the target has been tracked to the current image frame I_t, the GMMs representing the target and background must be updated. We accomplish this objective in the following manner. First, for each pixel, we find the fragment that contributed most to its likelihood:

$$\zeta(\mathbf{x}) = \arg \max_{j=1,\ldots,k^\star} p_\star(\xi_{I_t}(\mathbf{x})|\Gamma_{t-1}, j). \quad (16)$$

Then the statistics of each fragment are computed using its associated pixels:

$$\mu^\star_{j,t} = \frac{1}{|\mathcal{Z}^\star_j|} \sum_{\mathbf{x}\in\mathcal{Z}^\star_j} \xi_{I_t}(\mathbf{x}) \quad (17)$$

$$\Sigma^\star_{j,t} = \frac{1}{|\mathcal{Z}^\star_j|} \sum_{\mathbf{x}\in\mathcal{Z}^\star_j} \xi_{I_t}(\mathbf{x})\xi_I(\mathbf{x})^T, \quad (18)$$

where $\mathcal{Z}^\star_j = \{\mathbf{x} : \zeta(\mathbf{x}) = j, \text{sgn}(\phi(\mathbf{x})) = b(\star)\}$, $b(+) = 1$, $b(-) = -1$, and $\mu^\star_{j,t}$ is $\mu^\star_j$ at time t. The appearances are then updated using a weighted average of the initial values and a function of the recent values:

$$\mu^\star_{j,t} = \alpha^\star_j \bar{\mu}^\star_{j,0:t} + (1-\alpha^\star_j)\mu^\star_{j,0} \quad (19)$$

$$\Sigma^\star_{j,t} = \alpha^\star_j \bar{\Sigma}^\star_{j,0:t} + (1-\alpha^\star_j)\Sigma^\star_{j,0}, \quad (20)$$

where $\bar{\mu}^\star_{j,0:t}$ is a function of the past and present statistics, e.g.,

$$\bar{\mu}^\star_{j,0:t} = \frac{\sum_{\tau=0}^t e^{-\lambda(t-\tau)}\mu^\star_{j,\tau}}{\sum_{\tau=0}^t e^{-\lambda(t-\tau)}} \quad (21)$$

$$\bar{\Sigma}^\star_{j,0:t} = \frac{\sum_{\tau=0}^t e^{-\lambda(t-\tau)}\Sigma^\star_{j,\tau}}{\sum_{\tau=0}^t e^{-\lambda(t-\tau)}}, \quad (22)$$

where λ is a constant ($\lambda = 0.1$). The weights are computed by comparing the Mahalanobis distance to the two models: $\alpha^\star_j = \beta^\star_{j,0}/(\beta^\star_{j,0} + \bar{\beta}^\star_{j,0:t})$, where

$$\beta^\star_{j,0} = \sum_{\mathbf{x}\in\mathcal{Z}^\star_j} (\xi_{I_t}(\mathbf{x}) - \mu^\star_{j,0})^T (\Sigma^\star_{j,0})^{-1} (\xi_{I_t}(\mathbf{x}) - \mu^\star_{j,0})$$

$$\bar{\beta}^\star_{j,0:t} = \sum_{\mathbf{x}\in\mathcal{Z}^\star_j} (\xi_{I_t}(\mathbf{x}) - \bar{\mu}^\star_{j,0:t})^T (\bar{\Sigma}^\star_{j,0:t})^{-1} (\xi_{I_t}(\mathbf{x}) - \bar{\mu}^\star_{j,0:t}).$$

A fragment is declared as occluded if the cardinality of $\mathcal{Z}^\star_j$ is less than a constant (0.2% of the image size in our implementation). The updated mechanism is overrided for occluded fragments, whose spatial model is adapted to that of the target as a whole and whose appearance model remains unchanged throughout the occlusion. Finding such occluded fragments can serve as a good cue for handling partial occlusion, however we do not handle cases of partial occlusion. The number of fragments is fixed throughout a sequence and only its statistics are modified using the update strategy.

3. Experimental Results

The algorithm was implemented in Visual C++ and runs at 6-10 frames per second, depending upon the size of the object and motion. The algorithm was tested on a number of challenging sequences captured by a moving camera viewing complex scenery. Most of the sequences presented here were chosen so that the tracker could be evaluated for objects undergoing significant scale changes, extreme shape deformation, and unpredictable motion. The first row of Figure 5 shows the results of the algorithm on a sequence of a Tickle Me Elmo doll.[1] The benefit of using a multi-modal framework is clearly shown, with accurate contours (green outlines) being computed despite the complexity in both the target and background as Elmo stands tall, falls down, and sits up.

The second row shows the output on a sequence in which a monkey undergoes rapid motion and drastic shape changes. For example, as the monkey swings around the tree, its shape changes substantially in just a few image frames, yet the algorithm is able to remain locked onto the target as well as compute an accurate outline of the animal. Additional results involving occlusion are displayed in the third and fourth rows of Figure 5. In our approach, the shape of the object contour is learned over time by retaining the output of the tracker in each image frame. To detect occlusion, the rate of decrease in the object size is determined over the previous few frames. Once the object is determined to be occluded, a search is performed in the learned database to find the contour that most closely matches the one just prior to the occlusion using a Hausdorff distance. Then as long as the target is not visible, the subsequent sequence of contours occurring after the match is used to hallucinate the contour. Once the target reappears, tracking resumes. This approach prevents tracker failure during complete occlusion and predicts contours when the motion is periodic. The third row in the figure shows a sequence in which a person is completely occluded by a tree. Our approach predicts

[1] http://www.ces.clemson.edu/~stb/research/adafrag

both the shape and the location of the object and displays the contour accordingly. The fourth row shows a more complex scenario where a girl, moving quickly in a circular path (a complete revolution occurs in just 35 frames), is occluded frequently by a boy. Our approach is able to handle this difficult scenario as well.

The final row of Figure 5 shows the results of tracking multiple fish in a tank. The fish are multicolored and swim in front of a complex, textured, multicolored background. Note that the fish are tracked successfully despite their changing shape. Moreover, note that the small blue fish near the bottom of the tank is camouflaged and yet is recovered correctly due to the effective representation of the object and the background using multiple GMMs.

To provide quantitative evaluation of our approach, we generated ground truth for the experiments by manually labeling the object pixels in some of the intermediate frames (every 5 frames for the monkey and tree sequences, every 10 frames for Elmo, and every 4-6 frames for the girl sequence, avoiding occluded frames in the latter). We computed the error of each algorithm on an image of the sequence as the number of pixels in the image misclassified as foreground or background, normalized by the image size.

We compared our algorithm with two approaches. In one, the strength image was computed using the linear RGB histogram representation of Collins *et al*. [6]. In the other, the strength image was computed using a standard color histogram, similar to [21, 22, 13, 19]. In both cases the contours were extracted using the level set framework, but the fragment motion was not used. To evaluate the importance of using fragment motion, we also ran our algorithm without this component. Note that both versions of our algorithm were automatic, whereas the linear RGB histogram and the RGB histogram were manually restarted after every occlusion to simulate what they would be capable of achieving even with a perfect module for handling full occlusion.

Figure 6 plots the average normalized error for the four sequences. Our algorithm, with or without motion, performs better than the two alternatives on the Elmo, tree, and girl sequences. While the motion does not help significantly in the first two sequences since the motion of the target is not large from frame to frame, there is a noticeable improvement in the latter sequence. The difference is even more pronounced in the monkey sequence, where the rapid motion of the monkey causes all of the techniques except for the proposed algorithm to fail. We have also compared our technique against a color-based version of FragTrack [1] which also loses the monkey due to its quick movement. We omit these results here due to space constraints, and because FragTrack does not compute a pixelwise classification.

4. Conclusion

We have presented a tracking algorithm based upon modeling the foreground and background regions with a mixture of Gaussians. A simple and efficient region growing procedure to initialize the models is proposed, and comparison with state-of-the-art segmentation algorithms show improved results with regard to over- and under-segmentation. The GMMs are used to compute a strength image indicating the probability of any given pixel belonging to the foreground. This strength image computation is embedded into a level set tracking framework in which the target location is estimated by updating a level set function. Joint feature tracking and model updating are both incorporated to improve performance. Extensive experimental results show that the resulting algorithm is able to compute accurate boundaries of multi-colored objects undergoing drastic shape changes, unpredictable motions, and complete occlusion on complex backgrounds. Future work will involve utilizing the extracted shapes to learn more robust priors (e.g., [8]), and automating the initialization.

References

[1] A. Adam, E. Rivlin, and I. Shimshoni. Robust fragments-based tracking using the integral histogram. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2006.

[2] S. Avidan. Ensemble tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005.

[3] S. T. Birchfield and S. J. Pundlik. Joint tracking of features and edges. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2008.

[4] T. Brox, A. Bruhn, and J. Weickert. Variational motion segmentation with level sets. In *Proceedings of the European Conference on Computer Vision*, pages 471–483, May 2006.

[5] T. F. Chan and L. A. Vese. Active contours without edges. *IEEE Transactions on Image Processing*, 10(2):266–277, Feb. 2001.

[6] R. Collins, Y. Liu, and M. Leordeanu. On-line selection of discriminative tracking features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(10):1631 – 1643, Oct. 2005.

[7] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):603–619, May 2002.

[8] D. Cremers, F. R. Schmidt, and F. Barthel. Shape priors in variational image segmentation: Convexity, Lipschitz continuity and globally optimal solutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2008.

[9] P. Felzenszwalb and D. Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2):167–181, 2004.

Figure 5. The top two rows shows the results of our algorithm on the Elmo and Monkey sequences, in which target undergoes shape deformation and large unpredictable motion. The next two rows shows results on sequences in which a person walks behind a tree and a girl runs in circles around a room; the hallucinated contour is shown when the target is completely occluded, in frames 137 and 106, respectively. The fifth row shows the results of a sequence in which multiple fish swim in a tank and are all tracked successfully by the algorithm. Note especially the camouflaged small blue fish (magenta outline) at the bottom of frames 017 and 045. The last row shows the comparison of our results (red contour) with Linear RGB Histogram [6] (yellow) and standard color histogram [21, 22, 13, 19] (blue) on the girl sequence.

[10] H. Grabner and H. Bischof. On-line boosting and vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 260–267, June 2006.

[11] H. Greenspan, J. Goldberger, and A. Mayer. Probabilistic space-time video modeling via piecewise GMM. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(3):384–396, Mar. 2004.

[12] J. Ho, K.-C. Lee, M.-H. Yang, and D. Kriegman. Visual tracking using learned subspaces. In *Proceedings of the*

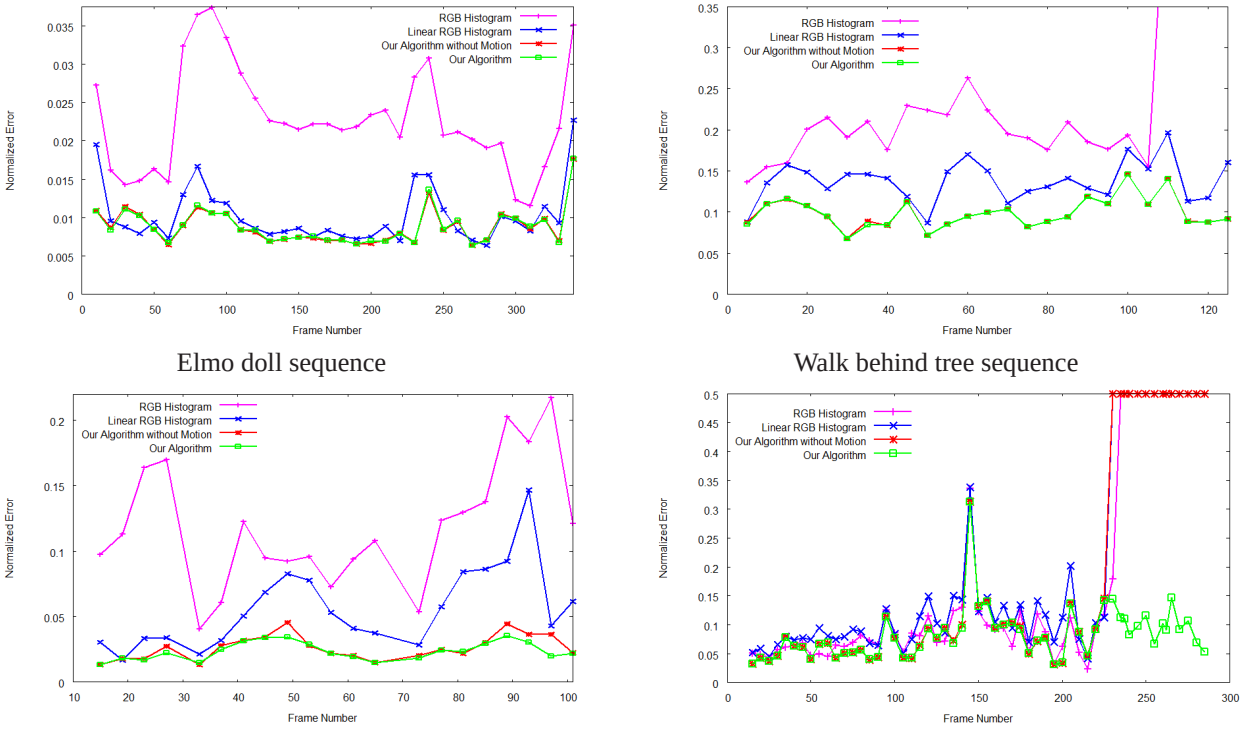


Figure 6. Normalized pixel classification error for the four sequences. Our algorithm outperforms implementations based upon [6] and [21, 22, 13, 19], showing the importance of spatial information for capturing accurate target representation. Motion marginally assists our algorithm, except when the drastic movement of the target (Monkey) causes the tracker to fail without it.

IEEE Conference on Computer Vision and Pattern Recognition (CVPR), volume 1, pages 782–789, 2004.

[13] S. Jehan-Besson, M. Barlaud, G. Aubert, and O. Faugeras. Shape gradients for histogram segmentation using active contours. In *Proceedings of the International Conference on Computer Vision*, volume 1, pages 408–415, 2003.

[14] A. D. Jepson, D. J. Fleet, and T. F. El-Maraghi. Robust online appearance models for visual tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(10):1296–1311, 2003.

[15] N. K. Paragios and R. Deriche. A PDE-based level-set approach for detection and tracking of moving objects. In *Proceedings of the 6th International Conference on Computer Vision*, pages 1139–1145, 1998.

[16] F. Porikli, O. Tuzel, and P. Meer. Covariance tracking using model update based on Lie algebra. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 728–735, June 2006.

[17] S. Sanjay-Gopal and T. J. Hebert. Bayesian pixel classification using spatially variant finite mixtures and the generalized EM algorithm. *IEEE Transactions on Image Processing*, 7(7):1014–1028, July 1998.

[18] G. Sfikas, C. Nikou, and N. Galatsanos. Edge preserving spatially varying mixtures for image segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2008.

[19] Y. Shi and W. C. Karl. Real-time tracking using level sets. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 34–41, 2005.

[20] N. Vlassis and A. Likas. A greedy EM algorithm for Gaussian mixture learning. *Neural Processing Letters*, 15(1):77–87, 2002.

[21] A. Yilmaz, X. Li, and M. Shah. Contour-based object tracking with occlusion handling in video acquired using mobile cameras. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(11):1531–1536, Nov. 2004.

[22] T. Zhang and D. Freedman. Tracking objects using density matching and shape priors. In *Proceedings of the International Conference on Computer Vision*, volume 2, pages 1056–1062, 2003.