Tracking Through Clutter Using Graph Cuts

James Malcolm Yogesh Rathi Allen Tannenbaum
School of Electrical and Computer Engineering
Georgia Institute of Technology, Atlanta, Georgia 30332-0250
{malcolm,yogesh.rathi,tannenba}@bme.gatech.edu

Abstract

The standard graph cut technique is a robust method for globally optimal image segmentation. However, because of its global nature, it is prone to capture outlying areas similar to the object of interest. This paper proposes a novel method to constrain the standard graph cut technique for tracking objects in a region of interest. By introducing an additional penalty on pixels based upon their distance from a region of interest, segmentation is biased to remain in this area. We employ a filter which predicts the location of the object. The distance penalty is then centered at this location and adaptively scaled based on prediction confidence. This method tracks at real-time rates and easily generalizes to tracking multiple noninteracting objects.

1 Introduction

Tracking rigid objects has been the focus of much research, and the problems accompanying this key task are well-known. For example, the object might have weak edges causing the segmentation to leak out into the surrounding area, or the object may move suddenly outside the algorithm's region of detection, or the object may be near other objects of similar intensity causing unintended objects to be tracked.

Various methods have been proposed to overcome these difficulties. To keep segmentations from spilling over object boundaries, learned shape priors constrain segmentation to a set of possible shapes [8, 9, 14]. To account for object movement, motion models can predict the likely location of the object in subsequent frames [7, 11]. When adjacent regions are similar to the object of interest, multiple hypothesis trackers can keep track of each region while determining the most likely in each frame based on some criteria [1, 12, 15, 18].

1.1 Graph cut techniques

Graph cut techniques have received considerable attention as robust methods for image segmentation. Despite their widespread use for computer vision problems such as image segmentation and stereo disparity, graph cuts have received little attention with respect to tracking. This is largely due to the global segmentations they produce which tend to catch unintended regions that are similar to the object of interest. For example, the standard graph cut technique for image segmentation [4] finds regions with high likelihood given intensity priors. Figure 1 shows an example where there are multiple regions of similar

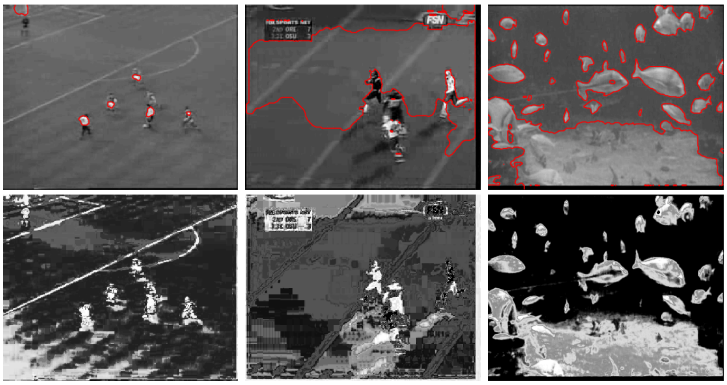

Figure 1: Standard graph cut segmentation *(top)* and normalized likelihood of object intensity used in graph edge weights *(bottom)*. Likely regions throughout the image are captured with the standard method making it unsuitable for tracking.

intensity to the object. The standard graph cut algorithm captures such regions. Post-processing must be performed to filter out those regions that are not part of the object. However, this same feature, that of grabbing such regions anywhere in the image, naturally solves the problem of large object movements. The graph cut will find the object even if it moved far relative to its location in the previous frame. The problem is now one of constraining the graph cut to capture only the object of interest, even if it made a large movement yet ignoring other regions of similar intensity. Hence a spatial constraint is needed.

Several techniques have used graph cuts for segmentation in visual tracking applications. In [22] the segmentation is constrained to a narrow band. For each frame, successive graph cut segmentations converge on a final segmentation, each pass constrained to a narrow band around the cut boundary resulting from the previous pass. This method is dependent upon initial contour placement and requires repeated cuts on this reduced domain. In [10] the authors use one graph cut for each frame to both estimate the optical flow and object position based on that flow despite changes in illumination. However, since optical flow requires the multi-label graph cut technique [6] and the graph proposed has such dense neighborhoods, the authors' current approach requires about a minute per frame. Also, due to the local nature of optical flow, the technique cannot handle large movements.

Besides tracking, work has been done to constrain segmentations based on a user selected region. The work of [19] begins with a rectangle bounding the object, while the work of [2] uses a narrow band to constrain segmentation. Both perform successive graph cut segmentations incorporating additional user interaction with each pass. Neither method is targeted towards tracking *per se*, but instead seeks a perfect segmentation. In these works, hard constraints confine the segmentation within a user-selected region and multiple graph cuts are performed. In our work, the object may be found a distance from the predicted centroid depending on the scale of the distance penalty, and segmentation is performed only once per frame.

1.2 Our contributions

The method presented here makes several important contributions to the field of visual tracking. First, we incorporate a distance penalty into the graph cut algorithm to bias segmentations to a region likely to contain the object. Second, we present a simple filter to predict the object location based on the centroid of the previous segmentation and a moving average of the object's velocity. The distance penalty is then centered at the predicted object centroid and extends outward forming a basin of attraction. Third, to further integrate the filter with the distance penalty, the scale of this distance penalty, and hence the slope of its surface, is adaptively set based on the prediction error. Finally, since the segmentation is performed in one cut using the standard binary label graph cut method, the unoptimized system tracks at up to 15 Hz on 240x320 images using a Pentium IV 3.6 GHz workstation. The method generalizes to multiple noninteracting objects.

The rest of the paper is organized as follows. Section 2 outlines the standard graph cut segmentation framework. Section 3 describes the distance penalty constraining segmentation. Section 4 defines the filter used to predict the object centroid. Section 5 integrates the filter prediction error with the distance penalty. Next, in Sections 6 and 7, we present our algorithm and results on several video sequences tracking single and multiple objects. Finally, in Section 8 we summarize our work and describe some possible future research directions.

2 Graph cuts

In this section, we briefly outline the graph cut methodology; for more details see [2, 3, 4, 19] and the references therein. Taking advantage of efficient algorithms for global min-cut solutions, we cast the energy-based image segmentation problem in a graph structure of which the min-cut corresponds to a globally optimal segmentation.

Evaluated for a pixel object/background assignment A, such energies are designed as a data dependent term and a smoothness term. The data dependent term evaluates the penalty for assigning a particular pixel to a given region. The smoothness term evaluates the penalty for assigning two neighboring pixels to different regions, i.e. a boundary discontinuity. These two terms may be thought of as a region-based term and a boundary term, often weighted by $\lambda \geq 0$ for relative influence:

$$E(A) = \sum_{p \in I} R_p(A_p) + \lambda \sum_{\substack{(p,q) \in \mathcal{N} \\ A_p \neq A_q}} B_{(p,q)} \tag{1}$$

where I represents all image pixels, $\mathcal{N}$ all unordered neighborhood pixel pairs. The choice of neighborhood size and structure has a large influence on the solution as smaller neighborhoods tend to introduce metrication artifacts [5].

To construct the graph representing this energy, each pixel is considered as a graph node in addition to two nodes representing object and background. The data dependent term is implemented by connecting each pixel to both the object and background nodes with non-negative edge weights $R_p(O)$ and $R_p(\mathcal{B})$ representing the penalty for assigning pixel p to the object or background region, respectively. Lastly, the smoothness term is implemented by connecting each pairwise combination of neighboring pixels (p,q) with a non-negative edge weight $B_{(p,q)}$ representing the penalty for separating pixels p and q.

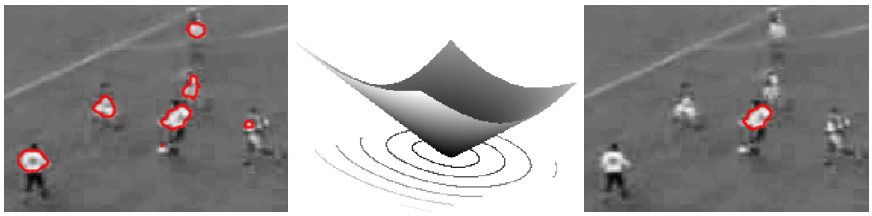

Figure 2: Mean intensity tracking of a soccer player among others of similar intensity: no distance penalty, distance penalty ϕ with isocontours, applying distance penalty *(left to right)*. Without the distance penalty, multiple non-intended objects were captured.

Notice that, since the min-cut sums only along the boundary, the boundary condition of $A_p \neq A_q$ in (1) may be ignored and every pair of neighboring pixels may be connected with edge weight $B_{(p,q)}$. The min-cut of the weighted graph represents the segmentation that best separates the object from its background. See [4] for more details.

Typical applications of graph cuts to image segmentation differ only in the definitions of R_p and $B_{(p,q)}$. For example, the authors of [4] use the negative log-likelihood of a pixel's intensity to compute the regional weights while intensity contrast is used in the boundary term:

$$R_p(O) = -\ln P(I_p|O), \quad R_p(\mathcal{B}) = -\ln P(I_p|\mathcal{B}), \quad B_{(p,q)} = \exp\left(\frac{-\|I_p - I_q\|^2}{2\sigma^2}\right)\frac{1}{\|p-q\|} \tag{2}$$

where $\|p-q\|$ is the standard L_2 Euclidean norm yielding pixel distance in the image and σ^2 is often set to the average squared norm: $\sigma^2 = \frac{1}{|\mathcal{N}|}\sum_{(p,q)\in\mathcal{N}}\|I_p - I_q\|^2$. In [4] the user marks regions of object and background that are then used to generate the intensity histograms for calculating $P(I_p|O)$ and $P(I_p|\mathcal{B})$ (see Figure 6).

The authors of [24] demonstrate the use of the mean intensity of the two regions to classify image pixels into two piecewise constant regions. They propose the following definitions:

$$R_p(O) = (I_p - \mu_O)^2, \qquad R_p(\mathcal{B}) = (I_p - \mu_{\mathcal{B}})^2, \qquad B_{(p,q)} = \frac{c_{\mathcal{N}}}{\|p-q\|} \tag{3}$$

where μ_O and $\mu_{\mathcal{B}}$ are the mean intensities of the regions marked by the user as object and background and $c_{\mathcal{N}}$ is a constant based on the chosen neighborhood size.

3 Distance penalty

The standard graph cut technique is capable of finding regions matching the object intensity located anywhere in the image. By penalizing pixels based on their distance from the expected location, a potential well is formed biasing segmentation to a region of interest. Figure 2 shows segmentation with and without such a penalty in the presence of multiple similar objects.

The distance penalty ϕ is formed from the user segmented shape of the object in the first frame. Centering that mask M at the predicted object location and assigning it zero penalty, each pixel x outside the mask is assigned its distance from the nearest masked pixel $m_x \in M$, i.e. $\phi(x) = \|x - m_x\|$ or zero if $x \in M$. Such a construction can be quickly computed with the Fast Marching algorithm [20, 23]. More deformable shape priors may be used for the base patch [10, 17, 21].

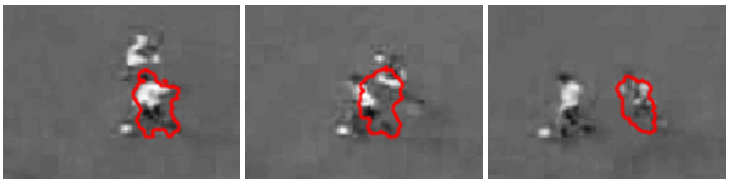

Figure 3: Without location prediction, tracking can fail when the target makes sudden movements. Here the tracker catches a defender as the target passes *(left to right)*.

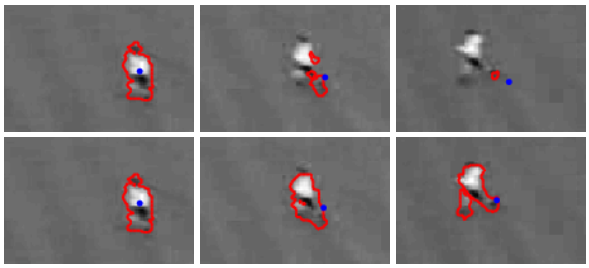

Figure 4: Effect of adaptive α on full intensity tracking: non adaptive alpha (assume zero error) *(top, left to right)*, alpha with prediction error *(bottom, left to right)*. Tracking fails without using error feedback to scale distance penalty.

4 Location prediction

It is often the case that the object makes a large movement, large enough at times to place it in an area of high distance penalty. To overcome this problem, we predict the location of the object in each frame based on its previous location and center the distance penalty at this predicted location.

To demonstrate the need for some form of prediction, we experimented with the assumption that the object has not moved: the distance penalty is centered at the last known object position. Figure 3 shows the failure to track after the object has made a sudden move, despite the use of adaptive α scaling described in Section 5. The movement placed the object too far outside of the basin of attraction.

Introducing actual prediction, we assume the object is traveling with continuous velocity, hence we predict the next object location $\tilde{c}_{t+1}$ based on projecting forward by the average displacement in the past few frames. A simple filter that projects the centroid c_t forward in time based on a moving average of the past N displacements is defined as:

$$\tilde{c}_{t+1} = c_t + \frac{1}{N}\sum_{j=0}^{N}(c_{t-j} - c_{t-j-1}). \tag{4}$$

5 Error feedback

We now have the distance penalty constraining segmentation and the filter predicting where to center this distance penalty, but what if the filter is wrong? Figure 4 shows such a case. The object has made a sudden move outside the predicted basin of attraction.

What is needed is a way of adaptively scaling the distance penalty based on the prediction error. In this work, we take the error in prediction to be the distance between the

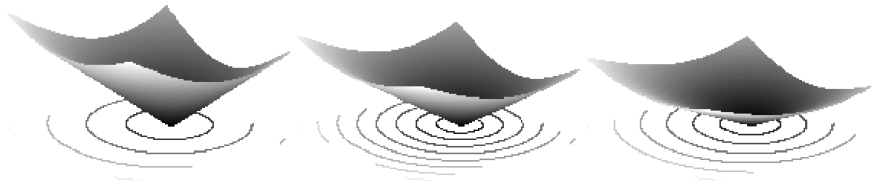

Figure 5: Distance penalty surface and isocontours are shown scaled by α for increasing prediction error $\|\tilde{c}-c\|$. Notice the basin of attraction widening as the error increases *(left to right)*.

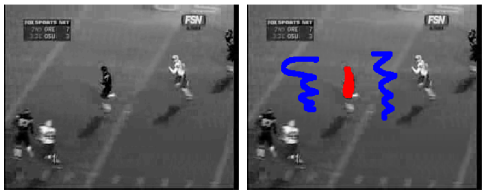

Figure 6: User initialization of object *(red)* and background *(blue)* regions: original and initialization scribbles *(left to right)*.

predicted $\tilde{c}$ and actual c centroids. The distance map is then scaled by $\alpha(\|\tilde{c}-c\|)$ taken from an exponential distribution of the prediction error, $\alpha(x) = \exp\left(-x^2/\rho^2\right)$, where ρ is user specified based on empirical motion. The effect is that when the filter is off in its predictions of the object centroid, the distance penalty is lowered to hopefully still capture the object. After locking back onto the object, the α automatically raises the distance penalty back up to tighten around the object as the error decreases. See Figure 5 for a visual of this distance penalty as it is scaled by α for increasing prediction error. Figure 4 shows how, despite incorrectly predicted centroids, the system is able to recover by adaptively widening the distance penalty.

6 Proposed algorithm

In an observer-type framework, at each frame the algorithm predicts the object location, determines the distance penalty scaling based on prediction error, computes edge weights for the graph, and performs a graph cut segmentation. For initialization, the user is required to roughly mark in the first frame the object and background as in Figure 6. This initialization defines the intensity priors used in constructing the priors used in regional edge weights (2) and (3).

In the prediction step, the centroid from the previous frame's segmentation is used as a measurement c. The filter predicts the object centroid location in this new frame $\tilde{c}$ from a moving average of displacements as in (4).

The $\alpha(\cdot)$ scaling function for the distance penalty is calculated from an exponential distribution of error $\|\tilde{c}-c\|$. Since the proposed simple filter is unstable against large displacements, we found the need to limit this distance in practice to a user-defined γ so that the distance penalty is not driven completely to zero. The $\alpha(\cdot)$ used is then:

$$\alpha(x) = \exp\left(\frac{-\min(x,\gamma)^2}{\rho^2}\right). \tag{5}$$

We propose a new regional edge weight to augment the standard weights in (2) and

(3). Our goal is to determine $P(O|I)$ for each pixel, and Bayes rule tells us that $P(O|I) \propto P(I|O)P(O)$. If we were to assume $P(O)$ and $P(\mathcal{B})$ are uniform, then their negative log-likelihoods are zero, and so they fall out of the expression as in (2). Here, we assume a non-uniform object prior $P(O)$ and claim: $-\ln P(O) \propto \alpha(\|\tilde{c}-c\|)\phi$. We assume the background to still be uniformly distributed $P(\mathcal{B})$. Introducing a weight $\beta \geq 0$ for relative distance penalty influence, we have a new regional term:

$$R_p(O) = -\ln P(I_p|O) - \beta \ln P(O_p) = -\ln P(I_p|O) + \beta\, \alpha(\|\tilde{c}-c\|)\phi(p) \quad (6)$$

$$R_p(\mathcal{B}) = -\ln P(I_p|\mathcal{B}) - \beta \ln P(\mathcal{B}_p) = -\ln P(I_p|\mathcal{B}) \quad (7)$$

Similarly, this additional weight may be added to the regional mean intensity term (3):

$$R_p(O) = (I_p - \mu_O)^2 + \beta\, \alpha(\|\tilde{c}-c\|)\phi(p) \quad (8)$$

$$R_p(\mathcal{B}) = (I_p - \mu_{\mathcal{B}})^2. \quad (9)$$

We use the standard intensity contrast smoothness term (2) for all experiments. Finally, we take the min-cut of this graph to yield a binary segmentation.

To track multiple similar objects, the same distance penalty may be used if the objects do not interact. If the objects do not touch, then their respective potential wells separate the segmentation into blobs. The centroid of each object is predicted independently. Since the segmentation is a binary mask of indistinguishable blobs, the identity of each blob is assigned to the object of nearest centroid in the previous frame. See Figure 11 for an example of simultaneously tracking two soccer players. If the objects were to touch, the segmentation will likely merge blobs and the unique identity of such blobs would be undefined for determining object centroids.

7 Results

Tracking was performed on three natural image sets and representative frames chosen to exhibit clutter with objects of similar intensity. Full videos are included in the supplementary material. The system is a combination of Matlab and C/C++ operating on a Pentium IV 3.6 GHz processor with 2GB RAM and tracks at roughly 5-15 Hz depending upon the graph neighborhood used[1]. The image size in the fish sequence is 360x480 while both the soccer and football sequences have frames of size 240x320.

Parameters are defined as follows. For all experiments, objects are assumed to not move more than 5 pixels between frames so $\gamma = 5$ in (5) and in practice $\rho = \frac{1}{2}\gamma$ is quite robust. For all full intensity experiments, $\lambda = 6$ in (1) and $\beta = 8$ in (6). For all mean intensity experiments both $\lambda = 10000$ and $\beta = 10000$.

The choice of neighborhood directly influenced the speed of computing the graph cut since larger neighborhoods induced denser graphs. Using a neighborhood of size 4 enabled tracking at 15 Hz, size 8 at 9 Hz, and size 16 at 5 Hz. The choice of neighborhood also affects the smoothness of the segmentation. Smaller neighborhoods tend to introduce irregular segmentations [5]. It is important to note that, since the segmentations for sizes 4 and 8 were not as smooth, they introduced larger variations in the calculated centroid and hence larger prediction errors. Increased smoothing (λ) was required to maintain track

[1]The min-cut is computed using the publicly available software of Vladimir Kolmogorov (`http://www.adastral.ucl.ac.uk/ vladkolm/`)

Figure 7: Several frames from the soccer sequence using full intensity capturing more of the multimodal object. Target object makes contact with another player yet the filter breaks them free. Full image *(left)* and selected cropped frames *(right)*. Yellow dot represents predicted centroid.

Figure 8: Several cropped frames from the soccer sequence using mean intensity. Blue dot represents predicted centroid.

with smaller neighborhoods. Tracking with size 4 or 8 was therefore not as robust as size 16. Unless otherwise noted, results are shown with a neighborhood of size 16.

The first video sequence involves several soccer players of similar intensity. Figure 7 shows full intensity tracking grabbing much of the object while Figure 8 shows mean intensity tracking grabbing the bright jersey, the optimal piecewise constant segmentation.

The second video sequence involves two dark football players touching. Figure 9 shows that despite this, the filter is able to track the intended player.

The third video sequence involves a fish crossing the screen among many other fish of identical intensity distributions. The high frame rate of the video sequence results in the fish moving slowly resulting in extended contact with the other fish. Figure 10 shows several such frames where the distance penalty correctly contains the segmentation.

In Figure 11 we demonstrate mean intensity tracking of multiple similar noninteracting objects.

8 Conclusion

This paper demonstrates a distance penalty to constrain the standard graph cut segmentation to a region of interest. An observer is proposed to predict object location while the prediction error is used to scale the distance penalty forming a basin of attraction that is adaptively sized. The binary graph cut algorithm is then used to find the object in one pass. The method operates at real-time rates and generalizes to multiple noninteracting targets.

There are several future directions of research. The multi-label graph cut method [6] may naturally allow segmentation of multiple dissimilar objects with interaction penalties. Anisotropic distance penalties may be used to bias certain directions based on expected

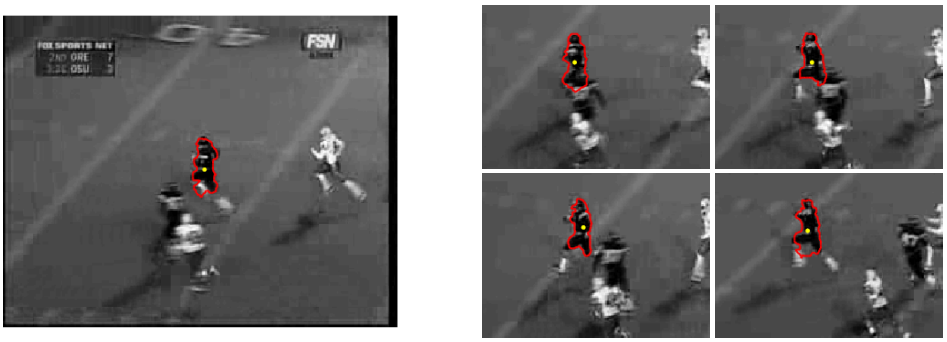

Figure 9: Several frames from the football sequence showing the target touching a teammate yet maintaining track *(yellow dot)*. Full image *(left)* and selected cropped frames *(right)*.

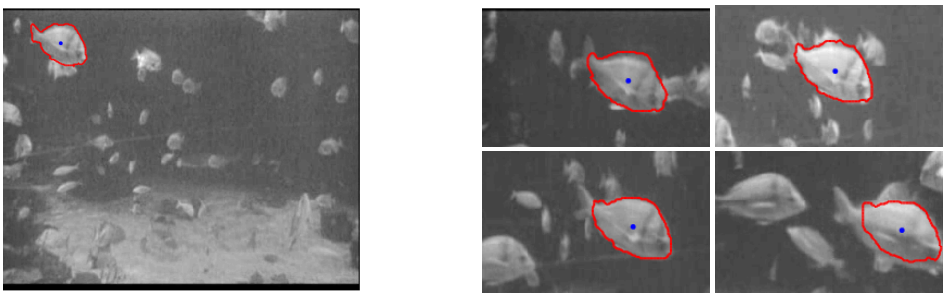

Figure 10: Selected frames from the fish sequence where the segmentation is correctly contained despite prolonged contact with other fish of similar intensity. The fish accelerates toward the end of the sequence yet the filter manages to maintain track *(blue dot)*. Full image *(left)* and selected cropped frames *(right)*.

object trajectory. Instead of rebuilding the graph from scratch for each frame as in the current system, speed can be enhanced by updating the graph in place from frame to frame [13]. Furthermore, segmentation may be made more robust for a larger class of imagery by tracking in a feature space with more information than simple intensity [16].

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. *IEEE Trans. Signal Processing*, 55(2):174–188, 2002.

[2] A. Blake, C. Rother, M. Brown, and P. Torr. Interactive image segmentation using an adaptive GMMRF model. In *ECCV*, volume 3021, pages 428–441, 2004.

[3] Y. Boykov and G. Funka-Lea. Graph cuts and efficient N-D image segmentation. *IJCV*, 70:109–131, 2006.

[4] Y. Boykov and M. Jolly. Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images. In *ICIP*, pages 105–112, 2001.

[5] Y. Boykov and V. Kolmogorov. Computing geodesics and minimal surfaces via graph cuts. In *ICCV*, pages 26–33, 2003.

[6] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *PAMI*, 23:1222–1239, 2001.

Figure 11: Uniquely tracking multiple similar, noninteracting players. Each object has its own centroid prediction *(blue dot)* and initialization mask but share the same distance penalty. *(Cropped frames shown.)*

[7] D. Cremers. Dynamical statistical shape priors for level set based tracking. *PAMI*, 28(8):1262–1273, 2006.

[8] D. Cremers, T. Kohlberger, and C. Schnorr. Shape statistics in kernel space for variational image segmentation. *Pattern Recognition*, 36:1929–1943, 2003.

[9] S. Dambreville, Y. Rathi, and A. Tannenbaum. Shape-based approach to robust image segmentation using kernel PCA. In *CVPR*, pages 977–984, 2006.

[10] D. Freedman and T. Zhang. Interactive graph cut based segmentation with shape priors. In *CVPR*, pages 755–762, 2005.

[11] R. Frezza, G. Picci, and S. Soatto. *A Lagrangian formulation of nonholonomic path following*, pages 118–133. The Confluence of Vision and Control. Springer Verlag, 1998.

[12] M. Isard and A. Blake. Condensation – conditional density propagation for visual tracking. *IJCV*, 29(1):5–28, 1998.

[13] P. Kohli and P. Torr. Effciently solving dynamic markov random fields using graph cuts. In *ICCV*, pages 922–929, 2005.

[14] M. Leventon, E. Grimson, and O. Faugeras. Statistical shape influence in geodesic active contours. In *CVPR*, pages 1316–1324. IEEE, 2000.

[15] E. Maggio and A. Cavallaro. Hybrid particle filter and mean shift tracker with adaptive transition model. In *ICASSP*, pages 221–224, 2005.

[16] J. Malcolm, Y. Rathi, and A. Tannenbaum. A graph cut approach to image segmentation in tensor space. In *Workshop on Component Analysis (CVPR)*, pages 18–25, 2007.

[17] J. Malcolm, Y. Rathi, and A. Tannenbaum. Graph cut segmentation with nonlinear shape priors. In *ICIP*, 2007. (to appear).

[18] Y. Rathi, N. Vaswani, A. Tannenbaum, and A. Yezzi. Particle filtering for geometric active contours with application to tracking moving and deforming objects. In *CVPR*, pages 2–9, 1997.

[19] C. Rother, V. Kolmogorov, and A. Blake. GrabCut: Interactive foreground extraction using iterated graph cuts. In *ACM Trans. on Graphics (SIGGRAPH)*, 2004.

[20] J. Sethian. A fast marching level set method for monotonically advancing fronts. In *Proc. Nat. Acad. Sci.*, volume 93, pages 1591–1595, 1996.

[21] G. Slabaugh and G. Unal. Graph cuts segmentation using an elliptical shape prior. In *ICIP*, pages 1222–5, 2005.

[22] N. Xu, R. Bansal, and N. Ahuja. Object segmentation using graph cuts based active contours. In *CVPR*, pages 46–53, 2003.

[23] L. Yatziv, A. Bartesaghi, and G. Sapiro. O(N) implementation of the fast marching algorithm. *J. of Computational Physics*, 212:393–399, 2006.

[24] X. Zeng, W. Chen, and Q. Peng. Efficiently solving the piecewise constant mumford-shah model using graph cuts. Technical report, Dept. of Computer Science, Zhejiang University, P.R. China, 2006.