Non-Rigid Object Tracking using Performance Evaluation Measures as Feedback

Çiğdem Eroğlu Erdem, Bülent Sankur *
Dept. Electrical and Electronics Engineering
Boğaziçi University
İstanbul, 80815, Turkey
erogluc,sankur@boun.edu.tr

A. Murat Tekalp
Dept. Electrical and Computer Engineering
University of Rochester
Rochester, NY 14627, USA
tekalp@ece.rochester.edu

Abstract

We present a scalable object tracking framework, which is capable of tracking the contour of rigid and non-rigid objects in the presence of occlusion. The method adaptively divides the object contour into subcontours, and employs several low-level features such as color edge, color segmentation, motion models, motion segmentation, and shape continuity information in a feedback loop to track each subcontour. We also introduce some novel performance evaluation measures to evaluate the goodness of the segmentation and tracking. The results of these performance measures are utilized in a feedback loop to adjust the weights assigned to each of these low-level features for each sub-contour at each frame. The framework is scalable because it can be adapted to roughly track simple objects in real-time as well as pixel-accurate tracking of more complex objects in off-line mode. The proposed method does not depend on any single motion or shape model, and does not need training. Experimental results demonstrate that the algorithm is able to track the object boundaries accurately under significant occlusion and background clutter.

1 Introduction

The problem of 2-D object tracking has attracted significant attention due to many applications in computer vision and video processing, including surveillance, content-based indexing and retrieval, object-based video coding, and video post-production. Some of these applications such as surveillance, require automatic real-time processing while tolerating some performance inaccuracy. In other applications, such as video-post processing and object-based coding, accuracy is very important while the processing may not need to be done real-time and a reasonable amount of user interaction is allowed (in fact desired). Thus, it is of interest to develop a generic scalable video object tracking framework that can address all of these diverse requirements.

The simplest approach for fully automatic object segmentation is to use blue screening (chroma keying), which requires special video capture apparatus. Other approaches for automatic segmentation assume that the background is stationary (or has global motion that can be compensated using a parametric model) and the object of interest is moving independently of background motion. Then, semantic object segmentation can be achieved by change detection or motion segmentation [16]. Other methods impose constraints on the shape of the tracked object by using 2D [13, 18] or 3D shape [9, 24] models. Some algorithms acquire the 2D shape space information through training [7, 5], and use projections onto the shape space to estimate the most likely object boundary at a certain frame. The CONDENSATION algorithm [17], which is a state-space sampling approach, needs the shape space to be known beforehand. Another application of learned motion models for tracking is presented in [26]. Pfinder [25] is a blob tracking method, that runs in real time assuming that the background is relatively stationary. Another blob tracking approach that uses color histogram of the tracked object is presented in [14]. Semi-automatic segmentation/tracking approaches incorporate minimal user interaction to resolve the ambiguity of semantic meaning. A recent such technique [15] tracks the visible boundary of the object under occlusion. While some of these methods emphasize real-time tracking of approximate object boundaries, others ad-

*This work was supported by Scientific and Research Council of Turkey (TÜBİTAK-BAYG) and Boğaziçi University Research Fund under project 99A203.

0-7695-1272-0/01 $10.00 © 2001 IEEE

dress pixel accurate object tracking in off-line mode. However, none is scalable to address both requirements in a single generic framework.

In this paper, we present a scalable video object tracking framework, which employs performance evaluation measures in a feedback loop. In the real-time mode, the method employs a region tracking paradigm in open loop. In the pixel accurate tracking mode, it adaptively divides the object contour into sub-contours, and employs several low-level features such as color edge, color segmentation, motion models, motion segmentation, and shape continuity information in a feedback loop to track each subcontour. In Section 2, an overview of the algorithm is given. In Section 3, the metrics used to evaluate the performance of the algorithm are introduced. Section 4 discusses the proposed closed-loop pixel-accurate video object tracking framework. In Section 6, experimental results are presented. Finally, in Section 7, concluding remarks are provided.

2 Overview of The Framework

We propose a scalable framework for 2-D video object tracking as shown in Figure 1. The block depicted by the dashed lines in Figure 1 performs a coarse tracking of the location of the object from frame $t-1$ to t in real-time or near real-time. The global motion compensation box is activated in the presence of fast camera motion, such as zoom or pan. However, the output of the coarse tracker does not yield pixel accurate contour localization. Thus, the remainder of the proposed framework deals with enhancing the accuracy of the tracked contour (for non real-time applications) by using a closed-loop energy-minimization scheme. The goodness of the tracking results after each iteration are evaluated using a set of novel performance metrics (without ground-truth) to fine tune the weights assigned to each low-level cue, such as color edges, color segments, motion models, etc.

We adopted a modified version of the feature point tracker of Shi et. al. [23] in order to track the coarse location of the object. The contour of the desired object in the initial frame is interactively marked. Then, the algorithm is allowed to select and track "good feature points" only within the boundaries of the selected video object. We then update each segment of the boundary, using the affine motion parameters estimated from a set of features that are closest to the segment. If a feature point cannot be tracked due to occlusion, temporal and spatial prediction schemes are employed for that point until (and if) it becomes uncovered again. We note that this coarse boundary tracking block may be replaced by other real-time region (blob) tracking

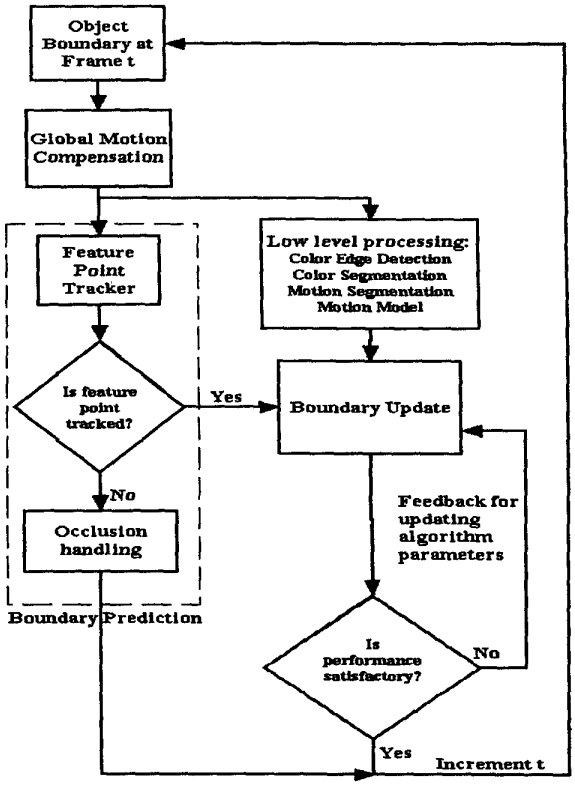

Figure 1. The tracking framework.

scheme without affecting the rest of the framework.

Next, the accuracy of the contour is fine-tuned using a closed-loop energy minimization scheme. The boundary is modeled as a union of contour segments to minimize a set of energy terms derived from color edges [20], color segmentation boundaries [11] and motion segmentation boundaries [1]. The proposed closed-loop scheme allows adaptive adjustment of the weights assigned to each term at each sub-contour segment. That is, if one type of information is more reliable at a certain sub-contour segment (deduced from new performance metrics), then that term is given more weight.

3 Performance Evaluation Metrics

The proposed performance evaluation metrics are presented in three groups as color, motion, and shape metrics [12].

3.1 Color Metrics

We present two color metrics, which are based on the following assumptions: 1) Object boundaries coincide with color boundaries. 2) The color histogram of the

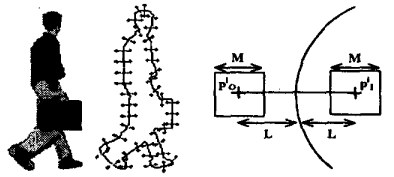

Figure 2. (a) A video object in frame 32 of "Hall monitor." (b) Boundary of the video object with the normals. (c) Close-up of a boundary normal. The points 'just inside' and 'just outside' of the boundary are denoted as p_I^i and p_O^i, respectively.

object is stationary from frame to frame. 3) The color histogram of the background is different from the color histogram of the object. Note that the background and its color histogram are not restricted to be stationary from frame to frame. There are also no restrictions on the shape and rigidity of the segmented/tracked object.

3.1.1 Intra-frame Boundary Color Differences

The first metric is based on the first assumption above, and it compares the color of the pixels 'just inside' and 'just outside' of the estimated object boundary. In order to define 'just inside' and 'just outside', we draw short normal lines of length L to the estimated object boundary at equal intervals towards the inside and outside of the object as illustrated in Figure 2(b). The points at the ends of these normal lines are marked with plus signs. A closer look at one of these normal lines is given in Figure 2(c). We define the color difference metric calculated along the boundary of the object in frame t as:

$$0 \le d_{CB}(t) = 1 - \frac{1}{K_t}\sum_{i=1}^{K_t} d_{CB}(t;i) \le 1, \quad (1)$$

$$d_{CB}(t;i) = \frac{\|C_O^i(t) - C_I^i(t)\|}{\sqrt{3 \times 255^2}} \quad (2)$$

where K_t is the total number of normal lines in frame t, and $C_O^i(t)$ is the average color calculated in the $M \times M$ neighborhood of the pixel $p_O^i(x,y;t)$ using Y-Cb-Cr color space. The average color inside $C_I^i(t)$ is defined similarly. Instead of average color, the α-trimmed mean [6] can also be used in Eq. 1.

3.1.2 Inter-frame Color Histogram Differences

The second metric is based on the second and third assumptions above, and it evaluates the stationarity of the color histogram of the segmented video object planes (VOP). This can be achieved by calculating the color histogram difference of the VOP at frames t and $t-1$. A better approach is to calculate the difference between the color histogram of the VOP at frame t and a smoothed reference color histogram, that can be computed by simple averaging or median filtering of the corresponding bins in the histograms of the VOP at frames $\{t-i,\ldots,t\}$. This makes histogram differences more robust to self-occlusions and mild intensity variations.

Let H_t denote the color histogram of the VOP in the YCbCr color space at frame t. The locally smoothed color histogram is calculated using the formula:

$$H_{t,av}(j) = Med\{H_{t-i}(j),\ldots,H_{t+i}(j)\}, \quad j = 1,\ldots,B, \quad (3)$$

where B denotes the total number of bins in the color histogram. The color histogram is represented as a 1-D vector obtained by concatenating the histograms for Y, Cb and Cr components.

The discrepancy between the color histograms H_t and $H_{t,av}$ is estimated using the χ^2 metric [22], which has been shown to be more sensitive to histogram differences compared to other metrics [12]. In the following formulae, the scaling parameters R_1 and R_2 are used to normalize the data when the total number of elements in the two histograms are different:

$$R_1 = \sqrt{\frac{N_{H_{t,av}}}{N_{H_t}}}, \quad R_2 = \frac{1}{R_1},$$

$$N_{H_t} = \sum_{j=1}^{B} H_t(j), \quad N_{H_{t,av}} = \sum_{j=1}^{B} H_{t,av}(j),$$

The χ^2 metric is used to compare two binned data sets, and to determine if they are drawn from the same distribution function [22]. The histogram difference is defined and normalized to the range $[0,1]$ as follows:

$$0 \le d_{CH}(t) = \frac{\sum_{j=1}^{B} \frac{[R_1 H_t(j) - R_2 H_{t,av}(j)]^2}{H_t(j)+H_{t,av}(j)}}{N_{H_t} + N_{H_{t,av}}} \le 1. \quad (4)$$

3.2 Motion Metric

The assumptions that we make to define a motion metric are as follows: 1)The motion vectors of the object that are 'just inside' of the object boundary and the background motion vectors that are 'just outside' of the object boundary are different. In other words, object boundaries coincide with the motion boundaries. 2) Background is either stationary or has global motion which shall be compensated for.

In order to quantify how well the estimated object boundaries coincide with actual motion boundaries,

we draw short normal lines to the boundary at regular intervals as shown in Figure 2(b), and we look at the difference of the motion vectors around the points $p_O^i(x,y;t)$ and $p_I^i(x,y;t)$. The motion metric estimated following this approach for frame t can be expressed as:

$$0 \le d_M(t) = 1 - \frac{\sum_{i=1}^{K_t} d_M(t;i)}{\sum_{i=1}^{K_t} w_i} \le 1, \quad (5)$$

$$d_M(t;i) = d(v_O^i(t), v_I^i(t)) \cdot w_i \quad (6)$$

$$0 \le w_i = R(v_O^i(t)) \cdot R(v_I^i(t)) \le 1, \quad (7)$$

where $v_O^i(t)$ and $v_I^i(t)$ denote the average motion vectors calculated in a $M \times M$ square around the points $p_O^i(x,y;t)$ and $p_I^i(x,y;t)$, respectively, and $d(v_O^i(t), v_I^i(t))$ denotes the distance between the two average motion vectors which is calculated as:

$$0 \le d(v_O^i(t), v_I^i(t)) = \frac{0.5 \parallel v_O^i(t) - v_I^i(t) \parallel}{\parallel v_O^i(t) \parallel + \parallel v_I^i(t) \parallel} + A \le 1,$$

$$A = \frac{\mid \angle v_O^i(t) - \angle v_I^i(t) \mid}{4\pi}, \quad (8)$$

where $\angle v_O^i(t)$ is the angle of the motion vector $v_O^i(t)$. In Eq. (7), $R(.)$ denotes the reliability of the motion vector $f^i(t)$ at point p^i [15]:

$$R(v^i(t)) = e^{\left(-\frac{\|v^i(t) - b^i(t+1)\|^2}{2\sigma_m^2}\right)} \cdot e^{\left(-\frac{\|c(p^i;t) - c(p^i+v^i;t+1)\|^2}{2\sigma_c^2}\right)},$$

where $b^i(t+1)$ denotes the backward motion vector at location $p^i + v^i$ in frame $t+1$; $c(p^i;t)$ denotes the color intensity and the parameters σ_m, σ_c are chosen freely.

3.3 Inter-frame Shape Differences

When a region of the object is occluded by another object, the above color and motion metrics may not be helpful. In occluded regions, we have no clue about the actual boundary, unless the object is known to be rigid. However, we can penalize the deviation between the shapes of the previous object boundary and the current boundary (or boundary segments).

We define the shape difference metric as:

$$0 \le d_S(t) = \frac{\sum_{s=1}^{P} \|\Theta^{t-1}(s) - \Theta^t(s)\|}{P \cdot 2\pi} \le 1, \quad (9)$$

where $\Theta^{t-1}(.)$ and $\Theta^t(.)$ denote the 1D turning angle functions (TAF) of the object boundary at frame $t-1$ and t, respectively. The parameter P denote the number of contour elements. We extend the definition of the turning angle function [3] from polygons to arbitrarily shaped objects by fitting a continuous curve to the boundary points at frame t and $t-1$ and sampling at uniform intervals [10].

3.4 Overall Performance at Each Frame

The overall tracking performance at frame t can be evaluated by checking the combined color, motion and shape metrics:

$$d(t) = \lambda_1 d_{CB}(t) + \lambda_2 d_{CH}(t) + \lambda_3 d_M(t) + \lambda_4 d_S(t) \quad (10)$$

against a threshold. The parameters $\lambda_1, \lambda_2, \lambda_3$ and λ_4 can be adjusted depending on the relative importance of each feature.

4 Boundary Update

The predicted boundary obtained using the prediction block, depicted in Figure 1, may not exactly match object boundaries due to inaccuracies in feature point tracking and motion estimation. However, the predicted boundary is generally close enough to the actual object boundary to allow a snake-based boundary updating. This section presents the details of the boundary update block, also depicted in Figure 1.

The active contour model was first introduced in [19]. A recent paper [15] presents a new approach of associating the energy terms with boundary segments rather than node points in a dynamic programming energy minimization framework [2].

If only edge-based energy terms are used, the active contour is easily distracted by background clutter or inside texture and snaps to the wrong edges. The novel energy terms introduced in this section fuse the information from the color segmentation boundaries, color edges and the motion segmentation boundaries to prevent such distractions. We associate the energy terms with boundary segments as in [15].

Let $w_i^t = (x_i^t, y_i^t), i = 1, \ldots, N$ denote the selected node points along the predicted object boundary at frame t. In the rest of the paper, the superscript t will be dropped. The node points are chosen as high curvature points along the boundary. The snake energy of the boundary is expressed as:

$$E_{snake} = \sum_{i=1}^{N} E_{int,i} + E_{ext,i} \quad (11)$$

$$E_{int,i} = \beta_{cu,i} E_{curv,i} \quad (12)$$

$$E_{ext,i} = \beta_{col,i} E_{col,i} + \beta_{e,i} E_{edge,i} + \beta_{m,i} E_{mot,i} \quad (13)$$

where $E_{col,i}, E_{edge,i}$ and $E_{mot,i}$ are external energy terms which are calculated from color segmentation boundaries, edges and motion segmentation boundaries, respectively. $E_{curv,i}$ is an internal energy term associated with the curvature of the boundary segment between nodes $i-2$ and i. The energy is minimized using the dynamic programming method [2], where the

search locations for each node are selected along the normal lines drawn to the node points. In the following we discuss the external and internal energy terms in detail.

4.1 External Energy Terms

A valid assumption in object segmentation and tracking is that object boundaries coincide with color boundaries. Color segmentation boundaries and edges contain different information as can be observed in Figure 3. If the object is also moving, object boundaries are also expected to coincide with motion boundaries. Based on these assumptions, we present new external energy terms below.

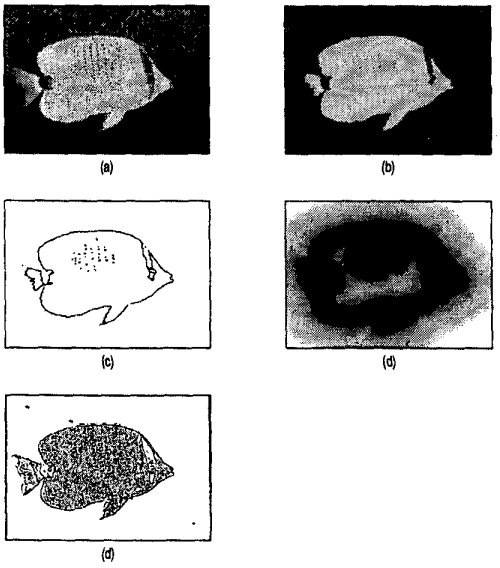

Figure 3. (a) The first frame of "Bream." (b) Two class color segmentation. (c) Color segmentation boundaries. (d) The Chamfer 3-4 transform of color segmentation boundaries. (e) The edges.

4.1.1 Color Boundary Energy

The color segmentation of each frame is estimated using an illumination invariant algorithm based on the fuzzy c-planes approach [11]. Then, the color boundaries are extracted from the segmented frame to obtain the contours (see Figure 3(c)). This contour map should be transformed such that it will be able to guide the snake nodes to the correct locations, even from distant places. This is achieved by using the Chamfer 3-4 distance transformation [8]. This transform approximates the Euclidean distance of a pixel location to the nearest contour point by transforming the binary contour image using the mask:

$$\begin{bmatrix} 4 & 3 & 4 \\ 3 & 0 & 3 \\ 4 & 3 & 4 \end{bmatrix}. \tag{14}$$

For example, a pixel location that is in the 4-neighborhood of a boundary pixel will have a value of 3, and a pixel which is a diagonal neighbor will have a value of 4. In Figure 3(d), the Chamfer distance transformation of the color segmentation boundaries can be seen which is varying smoothly. Let the Chamfer distance transform of the color segmentation boundaries be $CC(u)$.

The color boundary energy term is defined as follows:

$$E_{col,i} = \frac{\sum_{u \in \overline{w_i(l)w_{i-1}(k)}} CC(u)}{\|w_i(l) - w_{i-1}(k)\|}. \tag{15}$$

In Eq. (15), $\overline{w_i(l)w_{i-1}(k)}$ denotes the line segment that is connecting the current search nodes $w_{i-1}(k)$ and $w_i(l)$ (see Figure 4) during energy minimization using dynamic programming [2]. The indices k and l denote the search locations along the normal lines drawn at nodes $i-1$ and i, respectively.

As illustrated in Figure 4, the search direction (towards the inside of the segmentation map or towards outside) is pre-selected at each node. This is done by comparing the color of the search node with the color set of the object which is known from the previous frame. If the color of a search node is an element of the set of object colors, then we only search towards outside of the estimated boundary, otherwise we search towards the inside on the normal line. This approach not only adds robustness to the algorithm but also reduces the search area. The assumption behind it is that background colors are different from the object colors, at least around the object of interest. If the set of object colors and background colors have common elements, then bi-directional search is done.

4.1.2 Edge Boundary Energy

Edges are found using the edge detector [20] as seen in Figure 3(e). Note that the edge map is quite noisy compared to the color segmentation boundaries. Rather than finding the image gradient values we use the Chamfer distance transformation approach to obtain the terrain that the snake will slide on. This approach gives us a smoothly varying field that enables the snake to move towards the edges even if the image gradient is zero at its current location. Let the Chamfer distance transform of the edge map be denoted as $CE(u)$.

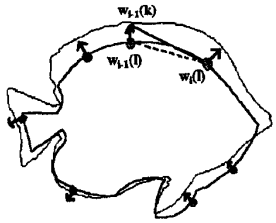

Figure 4. Illustration of energy minimization using dynamic programming. The correct and estimated (to be updated) object boundaries are shown by the thin and thick lines, respectively. Points $w_i(l)$ and $w_{i-1}(k)$ denote the current and previous search nodes.

The edge energy of a searched line segment is found as follows:

$$E_{edge,i} = \frac{\sum_{u \in \overline{w_i(l)w_{i-1}(k)}} CE(u)}{\|w_i(l) - w_{i-1}(k)\|}. \quad (16)$$

4.1.3 Motion Boundary Energy

When the object is moving with a motion pattern different from the background, the motion boundaries provide an important cue about the object boundaries. However, most dense motion estimation approaches such as block matching and Lucas-Kanade method [4], give poor motion boundaries since they tend to smooth the motion field by using motion constancy assumptions inside blocks. Therefore, methods to refine the motion segmentation boundaries using the more reliable color boundaries has been suggested [1] to perform reliable motion segmentation. In this section, we introduce a new snake energy term that utilizes this important cue: color refined motion segmentation boundaries.

The dense motion estimation is performed by the hierarchical version of Lucas-Kanade motion estimation algorithm [21]. Then following the approach of [1], the dense motion vectors are clustered into affine motion classes using the k-means algorithm. Then, color patches of the current frame obtained by color segmentation are assigned to one of these motion classes by comparing the dense motion field of the color patch with the motion cluster centroids.

The motion segmentation boundaries are transformed using the Chamfer 3-4 transform as discussed in the previous sections. Let this transform be $CM(u)$. Then the external energy term of motion segmentation boundaries is:

$$E_{mot,i} = \frac{\sum_{u \in \overline{w_i(l)w_{i-1}(k)}} CM(u)}{\|w_i(l) - w_{i-1}(k)\|}. \quad (17)$$

4.2 Internal Energy Term

The curvature energy term is estimated using the following equation [15]:

$$E_{curv,i} = 2 + 2\cos(\angle \overline{w_{i-1}(k)w_{i-2}(j)}, \overline{w_{i-1}(k)w_i(l)}), \quad (18)$$

where the indices j, k and l denote the search locations along the normal lines drawn at nodes $i-2$, $i-1$ and i, respectively.

5 Adaptation of the Energy Weights

In the literature there are no systematic ways to select the coefficients of the energy terms given in Eq. (12) and (13). However, these parameters are critical to the performance of the energy minimization. In this section, we present approaches to adjust the coefficients of the energy terms in a locally adaptive way using the performance evaluation metrics defined in the previous section.

After the minimization of the overall energy (11), some segments of the estimated boundary of the object may not coincide with color boundaries. This may indicate that the weight of the color energy term was not high enough. Similarly, if a segment of the boundary after energy minimization does not coincide with edges or motion boundaries we may conclude that the weight of that term was low.

Following this line of thought, the coefficients of the energy terms may be adjusted as follows:

$$\beta_{col,i} = g_1(d_{CB}(t;i)), \quad (19)$$

$$\beta_{m,i} = g_2(d_M(t;i)) \quad (20)$$

where $d_{CB}(t;i)$ and $d_M(t;i)$ were defined in Eq. (2) and Eq. (7), respectively. The functions $g_1(.)$ and $g_2(.)$ should be chosen such that, when $d_{CB}(t;i)$ or $d_M(t;i)$ are low at a certain segment, the corresponding weight should increase.

A possible choice for the above functions is:

$$g_1(d_{CB}(t;i)) = \frac{1}{d_{CB}(t;i) + \mu_1} \quad (21)$$

$$g_2(d_M(t;i)) = \frac{1}{d_M(t;i) + \mu_2}, \quad (22)$$

where the parameters μ_1 and μ_2 can be adjusted to set the minimum and maximum values of the coefficients $\beta_{col,i}$ and $\beta_{m,i}$. Note that $d_{CB}(t;i)$ and $d_M(t;i)$ take values between 0 and 1. During the experiments the parameters μ_1 and μ_2 were chosen around 0.25. Another method may be to increment $\beta_{col,i}$ and $\beta_{m,i}$

with certain step sizes at each iteration starting from a value.

The numerical range of the energy terms $E_{col,i}$, $E_{edge,i}$ and $E_{mot,i}$ are comparable since they are all calculated through the Chamfer distance transformation. If maximum search length is r for a certain node, then, the maximum value that each of the energy terms can take is approximately $4r$. The maximum value that the curvature energy term E_{curv} can take is 4 as given in Eq. (18). Therefore, the coefficient of the curvature term should be normalized by the search range.

After the coefficients are updated, the energy minimization is repeated forming the closed-loop. The algorithm exits from the loop when the performance of the tracking is satisfactory.

6 Experimental Results

The algorithm is tested on sequences containing rigid and non-rigid objects. In Figure 5, the tracking results for the standard MPEG test sequence "Coast Guard" is shown. The parameter K_t is chosen such that the distance between two successive perpendicular lines as shown in Figure 2(b) is 5 pixels. Feature points which have been successfully tracked at each frame are shown with yellow boxes and the predicted feature points are shown with red boxes. The algorithm tracks the boat successfully despite the heavy blurring due to camera movement in frame 72.

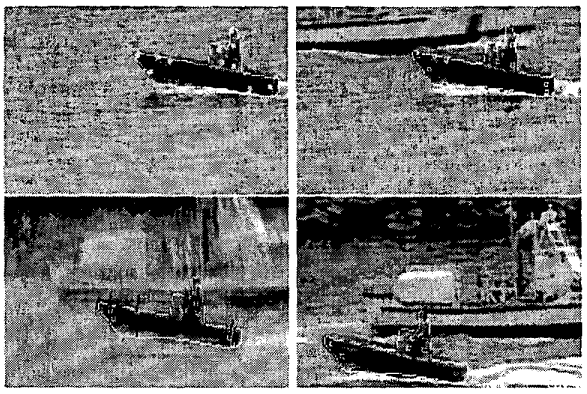

Figure 5. Open-loop tracking results of the boat in "Coast Guard." Frames 1, 45, 72, and 86 are shown. Tracked and predicted feature points are shown by yellow and red boxes, respectively. Motion vectors of the feature points are shown by blue lines.

The tracking results of the "Bream" sequence are given in Figure 6. In Figure 6, the fish is shown while it turns from right to left. The algorithm gives satisfactory results under severe self-occlusion of the fish.

The third sequence that we tested the tracking framewrok with is the "Dancer" sequence as shown in Figure 7. The tracking results are successfull despite the self-occlusion as the dancer raises her arms.

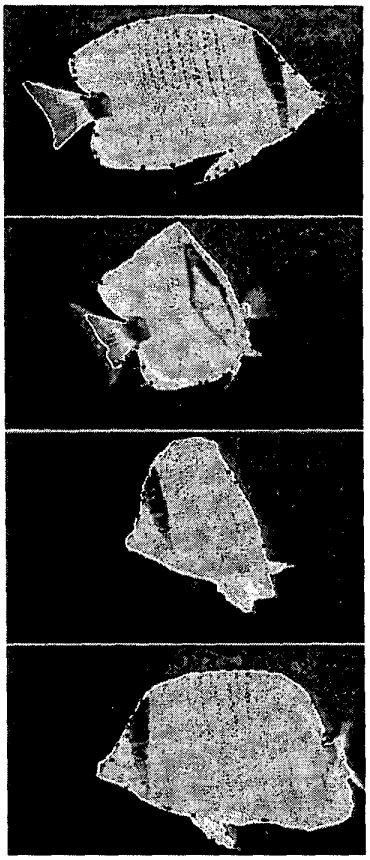

Figure 6. Closed-loop tracking results of "Bream." Frames 100, 115, 120, and 127 are shown. Tracked feature points are shown by black boxes and the motion vectors of the feature points are shown by yellow lines.

7 Conclusions

We have presented a novel scalable framework for tracking rigid and non-rigid objects. The approach consists of a rough boundary prediction and fine-tuning steps. We also presented new energy terms for improving the accuracy of the estimated boundary. The weights of the energy terms are updated using the introduced performance metrics. The experimental results demonstrate that the presented framework is quite successful in tracking the boundary of various

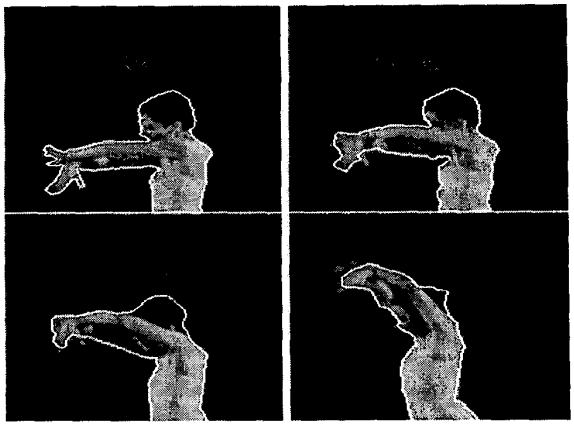

Figure 7. Closed-loop tracking results of the "Dancer." Frames 310, 312, 315, and 319 are shown.

rigid and non-rigid objects accurately, without using object-specific models or training.

References

[1] Y. Altunbaşak, P. E. Eren, and A. M. Tekalp. Region-based parametric motion segmentation using color information. *Graphical Models and Image Processing*, 60(1):13–23, 1998.

[2] A. A. Amini, T. E. Weymouth, and R. C. Jain. Using dynamic programming for solving variational problems in vision. *IEEE Trans. Pattern Analysis, and Machine Intelligence*, 12(9):885–867, 1990.

[3] E. M. Arkin, L. P. Chew, D. P. Huttenlocker, K. Kedem, and J. S. B. Mitchell. An efficient computable metric for comparing polygonal shapes. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 13:209–215, 1991.

[4] J. L. Barron, D. J. Fleet, and S. S. Beauchemin. Performance of optical flow techniques. *International Journal of Computer Vision*, 12(1):43–77, 1994.

[5] A. Baumberg and D. Hogg. Generating spatiotemporal models from examples. *Image and Vision Computing*, 14:525–532, 1996.

[6] J. Bednar and T. L. Watt. Alpha-trimmed means and their relationship to median filters. *IEEE Trans. Acoust., Speech, and Signal Processing*, 32(1):145–153, 1984.

[7] A. Blake and M. Isard. *Active Contours*. Springer-Verlag, 1998.

[8] G. Borgefors. Distance transformations in digital images. *Computer Vision, Graphics, and Image Processing*, 34:344–371, 1986.

[9] C. Bregler and J. Malik. Tracking people with twists and exponential maps. In *Proc. IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pages 239–245, Santa Barbara, CA, June 1998.

[10] Ç. E. Erdem and B. Sankur. Performance evaluation metrics for object-based video segmentation. In *Proc. X European Signal Processing Conference*, volume 2, pages 917–920, September 2000.

[11] Ç. E. Erdem and B. Sankur. Illumination invariant fuzzy image segmentation. In *Proc. IEEE Balkan Conf. Signal Processing, Communications, Circuits and Systems*, Turkey, 2000.

[12] Ç. E. Erdem, A. M. Tekalp, and B. Sankur. Metrics for performance evaluation of video object segmentation and tracking without ground-truth. In *Proc. IEEE Int. Conf. on Image Processing (ICIP)*, Oct.7-11, Greece, 2001.

[13] T. Cham and J. M. Regh. A multiple hypothesis approach to figure tracking. In *Proc. IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pages 239–245, 1999.

[14] D. Comaniciu, V. Ramesh, and P. Meer. Real-time tracking of non-rigid objects using mean shift. In *Proc. IEEE Int. Conf. on Computer Vision and Pattern Recognition*, 2000.

[15] Y. Fu, A. T. Erdem, and A. M. Tekalp. Tracking visible boundary of objects using occlusion adaptive motion snake. *IEEE Trans. Image Processing*, 9(12):2051–2060, December, 2000.

[16] M. Irani and S. Peleg. Motion analysis for image enhancement: Resolution, occlusion, and transparency. *J. Visual Commun. Image Repr.*, 4(4):324–335, 1993.

[17] M. Isard and A. Blake. Condensation - conditional density propagation for visual tracking. *Int. Journal of Computer Vision*, 29(1):5–28, 1998.

[18] S. X. Ju, M. J. Black, and Y. Yacoob. Cardboard people: A parametrized model of articulated image motion. In *Proc. Int. Conf. Face and Gesture Recognition*, pages 561–567, 1996.

[19] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *Int. Journal of Computer Vision*, 1(4):321–331, 1988.

[20] H. C. Lee and D. R. Cok. Detecting boundaries in vector field. *IEEE Trans. Signal Processing*, 39:1181–1194, 1991.

[21] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. DARPA Image Understanding Workshop*, pages 121–130, 1981.

[22] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flanney. *Numerical Recipes in C*. Cambridge Univeristy Press, 1992.

[23] J. Shi and C. Tomasi. Good features to track. In *Proc. IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pages 593–600, 1994.

[24] S. Wachter and H. H. Nagel. Tracking persons in monocular image sequences. *Computer Vision and Image Understanding*, 74(3):174–192, June 1999.

[25] C. R. Wren, A. Azerbayejani, T. Darrell, and A. P. Pentland. Pfinder: Real-time tracking of the human body. *IEEE Trans. Pattern Analysis, and Machine Intelligence*, 19(7):780–785, July 1997.

[26] Y. Yacoob and L. S. Davis. Learned models for estimation of rigid and articulated human motion from stationary or moving camera. *Int. Journal of Computer Vision*, 12(1):5–30, 2000.