Tracking of Multiple Objects Using Optical Flow Based Multiscale Elastic Matching

Xingzhi Luo And Suchendra M. Bhandarkar
Department of Computer Science, The University of Georgia
Athens, Georgia 30602-7404, USA

Abstract

A novel hybrid region-based and contour-based multiple object tracking model using optical flow based elastic matching is proposed. The proposed elastic matching model is general in two significant ways. First, it is suitable for tracking of both, rigid and deformable objects. Second, it is suitable for tracking using both, fixed cameras and moving cameras since the model does not rely on background subtraction. The elastic matching algorithm exploits both, the spectral features and contour-based features of the tracked objects, making it more robust and general in the context of object tracking. The proposed elastic matching algorithm uses a multiscale optical flow technique to compute the velocity field. This prevents the multiscale elastic matching algorithm from being trapped in a local optimum unlike conventional elastic matching algorithms that use a heuristic search procedure in the matching process. The proposed elastic matching based tracking framework is combined with Kalman filter in our current experiments. The multiscale elastic matching algorithm is used to compute the velocity field which is then approximated using B-spline surfaces. The control points of the B-spline surfaces are used directly as the tracking variables in a Kalman filtering model. The B-spline approximation of the velocity field is used to update the spectral features of the tracked objects in the Kalman filter model. The dynamic nature of these spectral features are subsequently used to reason about occlusion. Experimental results on tracking of multiple objects in real-time video are presented.

1 Introduction and Background

Multiple object tracking in dynamic scenes is challenging in several aspects. The first challenge arises from occlusions, which includes mutual occlusion between foreground objects and occlusion caused by background objects. When occlusion occurs, some objects are partially or totally invisible. This makes it difficult to accurately localize the occluded object and track it continuously over several image frames. The second challenge is the formulation of an object model that is capable of handling object deformation. The object model should be able to capture the most important and relevant information about the object and facilitate fast and reliable tracking. The ability to deal with occlusion depends, to a great extent, on the object model. The third challenge is to meet the real time constraints of most tracking applications in the real world. Fast and accurate object localization over time is the ultimate objective of a tracking system.

Generally speaking, there exist three broad categories of object models in the context of tracking: contour-based models [1], [5], [7], [8], [23], region-based models [2], [3], [4], and feature point-based models [9], [10], [22]. The contour-based model does not encode any color or edge information within the interior of the object. The contour information by itself is not enough to handle general instances of occlusion. In the absence of any spectral information, feature point-based tracking methods are easily distracted by noisy feature points in the background and are, by their very nature, limited to objects rich in feature points. A region-based object model is more suitable when occlusion is present since it encodes spectral information.

Occlusion handling is another important issue that arises in multiple object tracking systems and is closely intertwined with the choice of the object model. In the case of contour-based models, the robustness of the occlusion reasoning is highly dependent on the quality of object segmentation and typically, only simple cases are well handled. Koller *et al.* [7] propose a depth-based occlusion reasoning scheme based on a geometric model which computes the projection of the moving object in the 3-D world onto the image plane. However, their method is only applicable in situations where the object moves in the vertical direction in the image plane and hence not valid for more general tracking scenarios. Also, region-based object models that rely primarily on color/gray level histograms of the moving regions are not well suited to handle occlusion since no object shape information is available. McKenna *et al.* [3] use simple correspondence analysis to do tracking when the tracked objects are independent of each other. In the event of occlusion, their technique can only compute a statistical probability that a pixel of a given color belongs to a specific object which is not useful for the purpose of accurate object localization.

Elastic matching has been used widely for deformable object recognition [6] and tracking [15]. Elastic matching is potentially well suited for tracking of deformable objects. Since elastic matching exploits both, the spectral information within and the spatial coherence constraint amongst the object pixels, it can be easily integrated with a region-based object model. Another advantage of elastic matching is that the tracking results are not dependent on the accuracy of the background subtraction used to extract the moving objects. This makes it possible to track moving objects with a moving camera. However, most existing elastic matching methods use a heuristic search algorithm within the matching procedure [6, 15, 16, 17] and are hence prone to get trapped in a local optimum. Optical flow methods exploit the image gradient information to compute the velocity field, making it possible to avoid a local optimum within the elastic matching procedure. However, most optical flow methods use only gray scale images and the velocities at each image pixel are computed independently, thus resulting in a noisy velocity field. In a homogeneous image region, the image gradient value is a constant (close to zero) resulting in ambiguous values for the velocity field. Also, most optical flow algorithms ensure only local consistency of the velocity field since the image gradient is computed within a local window.

Very few optical flow algorithms using multiple-channel (multi-spectral) images have been reported in the research literature. Markandey *et al.* [19] and Golland *et al.* [18] have proposed optical flow algorithms that use images with 2 and 3 channels respectively. In this paper, the optical flow algorithm is generalized to use images with an arbitrary number of channels. The generalized optical flow computation procedure is used within a hybrid region- and contour-based elastic matching scheme. Since the spatial coherence constraint is imposed in the elastic matching procedure, the ambiguities in the computation of the velocity field in a locally homogenous region of the image is resolved. In order to increase the range of consistency of the optical flow algorithm, two schemes are adopted. The first scheme employs an iterative multiscale Lucas-Kanade algorithm for optical flow computation using a pyramidal image structure. The second scheme uses a Kalman filtering algorithm to predict the velocity field which is subsequently refined by performing a search in the neighborhood of the predicted location. The incorporation of multiscale elastic matching within the Kalman filtering framework has two advantages: (a) the inherent limitation of the linear Kalman filter which assumes that the tracking variables have Gaussian distributions is addresed, and (b) changes in object size in the image (scale changes) are effectively dealt with.

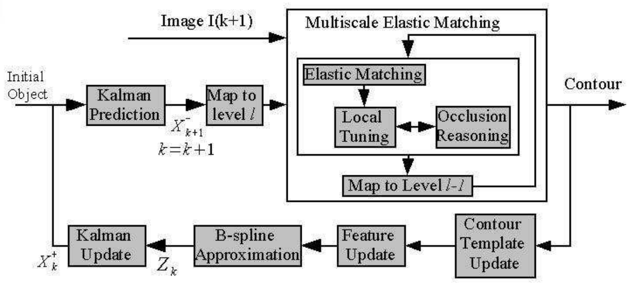


Figure 1. The tracking scheme

The overall system is depicted in Figure 1. Given an initial object position and its velocity field, the Kalman filtering algorithm predicts the new velocity field for the next stage. An iterative elastic matching algorithm uses the predicted initial velocity and the computed optical flow to determine the new velocity field for the object. The iterative elastic matching algorithm first maps the predicted velocity field to a manually chosen level l in the pyramid, and determines the actual velocity at level l using elastic matching. Then the velocity computed at level l is mapped to level $l - 1$ and used as the initial velocity at level $l - 1$. This procedure is performed iteratively until the velocity at level 0 is obtained. An occlusion reasoning and local tuning procedure is performed at each level to generate and verify the occlusion hypothesis at each pixel location and refine the velocity field. The new velocity field is then used to update the contour template and object model. The new velocity field is approximated using B-spline surfaces, which are then used to update the status of the Kalman filter. The updated Kalman filter is used to predict the velocity field for the new stage.

2 The Multiscale and Multiresolution Object Model

In order to compute a multiscale representation of the moving objects, each of the original images is encoded as L levels of Gaussian pyramid and Laplacian pyramid. The image at level l in the Gaussian pyramid is denoted by I^l. The Gaussian pyramid image at level $l + 1$ is computed as $I^{l+1}(x, y) = G(x, y; \sigma) \circ I^l(2x, 2y)$, where $G(x, y; \sigma)$ is the Gaussian smoothing operator, $\circ$ is the convolution operator and σ is the scale parameter ($\sigma = 1$ is used in our case). This procedure, termed as the REDUCE function [21], smoothly samples the original image with a sampling interval of length 2 along the x axis and y axis. Its inverse function, defined as $\hat{I}^l(x, y) = G(x, y; \sigma) \circ I^{l+1}(x/2, y/2)$, samples the image at level $l + 1$ back to l. The Laplacian pyramid image at level l is given by $P^l = I^l - \hat{I}^l$. The pyramid image at the first level $(l = 0)$ has the same size as the original image. If the size of the original image is (W, H), where W is the image width and H the image height, the image size at level l in the pyramid is $(W/2^l, H/2^l)$. An RGB color image is encoded as a six-channel image at each level, where three of the images (R,G and B) are obtained from the Gaussian pyramid and the other three images (R,G and B) are obtained from the Laplacian pyramid.

Object $O^l(i)$ at level l is represented by a network of points $O^l(i) = (\{X_j^l\}, K^l, L^l, \{T_j^l\})$, where $1 \leq j \leq N^l$ and N^l is the number of points used to represent the object. K^l is the connectivity matrix with dimension $N^l \times N^l$ such that $k_{ij}^l = 1$ if points X_i^l and X_j^l are connected, and $k_{ij}^l = 0$ otherwise. In practice, $k_{ij}^l = 1$ if point X_i^l is one of the neighboring points of point X_j^l. The matrix K^l is symmetric and $k_{ii}^l \triangleq 0$. L^l is the connectivity matrix for the boundary points. If X_i^l and X_j^l are contour points and are connected to each other within a window of predetermined size, then $l_{ij}^l = 1$, else $l_{ij}^l = 0$. The matrix L^l is also symmetric. $\{T_j^l\}$ is the object contour template at level l. Each contour point in $\{X_j^l\}$ has one and only one corresponding point in contour template $\{T_j^l\}$, and T_j^l is not valid if X_j^l is not a contour point. In our current work, the boundary template model is obtained using B-spline contour fitting as described in Section 3.

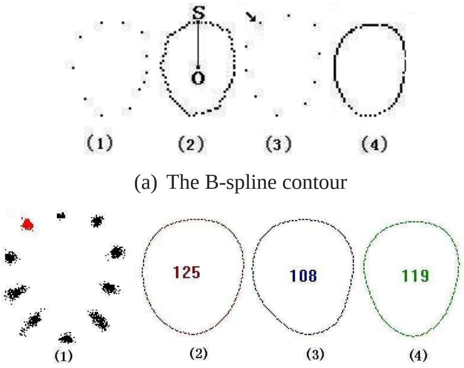
For each point X_j^l, $\mathbf{c}_j^l = \mathbf{c}(X_j^l)$ represents the feature vector associated with point X_j^l and $(\Sigma_j^2)^l = \Sigma^2(X_j^l)$ represents the covariance matrix of the feature vector at point X_j^l. Since $\mathbf{c}_j^l$ is a vector comprising of the values from each of the six channels and it is 6-dimensional. Both $\mathbf{c}_j^l$ and $(\Sigma_j^2)^l$ are temporally varying values and are updated online. When there is insufficient temporal information for a point, $(\Sigma_j^2)^l$ is initialized to a default value $(\Sigma_0^2)^l$. An online updating scheme for $\mathbf{c}_j^l$ and $(\Sigma_j^2)^l$ is given in Section 4.4. $V^l = \{v_j^l\}$ is the velocity field associated with the object where $v_j^l = (v_x(X_j^l), v_y(X_j^l))$ is the velocity at point X_j^l.

Instead of computing the velocity for each point of the object, the designed object model dynamically adjusts the sampling intervals of the points used for velocity computation in order to balance the computational load. Given N_0, the desired number of points used for computation, and N, the total number of points belonging to the object, the sampling interval used to obtain the points for velocity computation is determined as $\max(\sqrt{N/N_0}, 1)$. Simple interpolation is used to determine the velocity for the points which are not sampled. The contour template T_j^l is used to guide the search process.

3 Contour Template

The contour template is pre-trained for the purpose of object tracking. Note that the object contour is implicitly included in the object model. Given the contour training set $\{\mathbf{C}_i\}$, each contour $\mathbf{C}_i$ is first normalized to a specified size (eg. 100×100 for face tracking) as shown in Figure 2(a)(1). The contours in the training set are obtained manually from the training videos. The spatial distribution of contour points is equalized using interpolation such that the resulting contour points are uniformly distributed along the contour. The starting point on the contour is chosen to be aligned with centroid point of the contour, as shown in Figure 2(a)(2) where O is the centroid point and S is the chosen staring point. Let $\mathbf{C}_i' = (c_{i,0}, ..., c_{i,N-1})$ be the contour obtained from $\mathbf{C}_i$ after normalization, equalization and alignment. A vector of B-spline control points $\{\mathbf{T}_i\} = (t_{i,0}, ..., t_{i,M-1})$ is obtained for each contour $\mathbf{C}_i'$ in the training set as shown in Figure 2(a)(3) where the point marked in red represents the first control point. $M = 10$ is used in the face tracking experiment. Figure 2(a)(4) depicts the contour restored using the B-spline control points. The control point vectors $\{\mathbf{T}_i\}$ are modeled as a mixture of K Gaussian distributions for a predetermined value of K. The corresponding K clusters of control points are determined using the K-means algorithm. The distance between the control point vectors $\mathbf{T}_i$ and $\mathbf{T}_j$ is measured using the Euclidean metric $d(\mathbf{T}_i, \mathbf{T}_j) = \|\mathbf{T}_i - \mathbf{T}_j\|$. As shown in Figure 2(b), 352 faces are randomly extracted from the video data for training in the face tracking experiment. The control points obtained from the normalized faces are shown in Figure 2(b)(1). With the K-means algorithm, 3 clusters are obtained from the 352 control point vectors. The 3 contours generated from the 3 cluster centers of the control point vectors are shown in Figure 2(b) (2) (3) and (4) where the number within each contour represents the cluster cardinality. Each cluster is represented as a Gaussian distribution $\mathcal{N}(\overline{\mathbf{T}_k}, \Sigma_k)$.

Given an object contour $\mathbf{C}$ obtained via elastic matching, its contour template is determined using the following procedure. The contour $\mathbf{C}$ is first normalized, equalized and aligned, and its B-spline control point vector $\mathbf{T}$ is determined using the training procedure described above. Q control point vectors are randomly generated from each of the K clusters resulting in a total of KQ control point vectors. The random generation function uses the Gaussian distribution of each cluster. Among the KQ control point vectors, the one at minimum distance to $\mathbf{T}$ is chosen as the contour template. The control value for each point in $\mathbf{C}$ is computed and used to compute its corresponding points with the contour template. These points are scaled back to their original size and used in the elastic matching algorithm in the next step.



(a) The B-spline contour

(b) Contour clustering

Figure 2. Contour template

4 Pyramidal Elastic Matching Model

Inspired by the pyramidal implementation of Lucas-Kanade image registration algorithm [20], a multiple scale elastic matching model is designed for region-based tracking. However, in the original algorithm [20] only grayscale images are used, each feature point is tracked independently and image interpolation is used for velocity computation, which incurs high computation load. In the proposed scheme, the original algorithm is generalized to use images with any number of channels. In particular, 6-channel images are used as described in the object model. The imposition of the spatial coherence constraint suppresses the random noise in the velocity field computation, and the use of multiple scales ensures that an optimal velocity field can be obtained. Instead of using image interpolation, a local tuning algorithm is used to obtain sub-pixel accuracy.

The pyramidal elastic matching algorithm can be generalized as follows. Given object $O^l(t)$ at time t and level l, its initial velocity field $(\mathbf{V}^0)^l$, and the new image $I^l(x, y; t+1)$, the objective of elastic matching algorithm is to refine the velocity field $\mathbf{V}^l$ of the tracked object(s), such that the energy function defined in equation (1) is minimized.

$$E^l = \sum_{i=1}^{N^l} \epsilon^l(v_i^l) \tag{1}$$

in which

$$\begin{aligned} \epsilon^l(v_i^l) &= \underbrace{g_i^l \sum_{X_j^l \in O(X_i^l)} [c_j^l - I^l(X_j^l + (v_i^0)^l + v_i^l)]^2}_{feature\,matching} \\ &+ \underbrace{\gamma \sum_{j=1}^{N^l} l_{ij}^l \|(X_i^l + (v_i^0)^l + v_i^l - T_i^l) - (X_j^l + (v_j^0)^l + v_j^l - T_j^l)\|^2}_{contour\,constraints} \\ &+ \underbrace{\beta \sum_{j=1}^{N^l} k_{ij}^l \|(v_i^0)^l + v_i^l - (v_j^0)^l - v_j^l\|^2}_{velocity\,constraints} \end{aligned} \tag{2}$$

where $g_i^l = g(X_i^l)$ is the occlusion hypothesis for point X_i^l of the given object such that $g_i^l = 0$ if X_i^l is occluded, and $g_i^l = 1$ otherwise, and $(v_j^0)^l = (v_{x_j}^0, v_{y_j}^0)^l$ is the initial velocity at point X_j^l. In order to avoid the need for image interpolation, each value in the initial velocity $(\mathbf{V}^0)^l$ is actually its nearest integer value. $O(x_i^l)$ is the set of pixels in a window centered at point X_i^l and v_j^l is the incremental velocity. The first part in equation (2) measures the feature match of each point of the object with the new image, which requires that the corresponding image point has a similar feature vector in order to minimize the energy function. The second part in equation (2) defines the contour constraint, which requires that the neighboring contour points have similar displacement values from the contour template in order to minimize the energy function. In cases where the contour template is not available, T_j^l can be simply set to $(0, 0)$ for all j, which is equivalent to a smoothness constraint imposed on the object contour. The third part in equation (2) imposes spatial coherence on the velocity field, which requires the velocity of neighboring pixels to be close to each other in order to minimize the energy function. The parameter β controls the elasticity of the object. Equation (1) is minimized when $\partial E^l / \partial(\mathbf{V}^l)^\tau = 0$, which is equivalent to:

$$\begin{aligned} \frac{\partial E^l}{\partial(v_i^l)^\tau} &= -2g_i^l \sum_{X_j^l \in O(X_i^l)} [c_j^l - I^l(X_j^l + (v_i^0)^l + v_i^l)] \frac{\partial I^l(X_j^l + (v_i^0)^l + v_i^l)}{\partial(v_i^l)^\tau} \\ &+ 4\gamma \sum_{j=1}^{N^l} l_{ij}^l (X_i^l - X_j^l + T_i^l - T_j^l + (v_i^0)^l - (v_j^0)^l + v_i^l - v_j^l)^\tau \\ &+ 4\beta \sum_{j=1}^{N^l} k_{ij}^l (v_i^l - v_j^l + (v_i^0)^l - (v_j^0)^l)^\tau \end{aligned} \tag{3}$$

Note that the above equation is obtained under the assumption that K^l and L^l are symmetric. By using Taylor expansion,

$$I^l(X_j^l + (v_i^0)^l + v^l) \approx I^l(X_j^l + (v_i^0)^l) + I_{v_j}^l (v^l)^\tau \tag{4}$$

in which, $I_{v_j}^l = (I_{x_j}^l, I_{y_j}^l)$ is the gradient vector at location $X_j^l + (v_i^0)^l$. $I_x^l = [I^l(x+1, y) - I^l(x-1, y)]/2$ and $I_y^l = [I^l(x, y+1) - I^l(x, y-1)]/2$. Based on these gradient equations, the Taylor expansion in equation (4) is valid when $|v_x^l| \leq 1$ and $|v_y^l| \leq 1$.

For convenience, let $\delta I_j^l = c_j^l - I^l(X_j^l + (v_i^0)^l)$. Equation (3) can be rewritten as:

$$\begin{aligned} \frac{\partial E^l}{\partial(v_i^l)^\tau} &= g_i^l \sum_{X_j^l \in O(X_i^l)} (I_{v_j}^l)^\tau I_{v_j}^l (v_i^l)^\tau - g_i^l \sum_{X_j^l \in O(X_i^l)} \delta I_j^l (I_{v_j}^l)^\tau \\ &+ 2\gamma \sum_{j=1}^{N^l} l_{ij}^l (X_i^l - X_j^l + T_i^l - T_j^l + ((v_i^0)^l - (v_j^0)^l + v_i^l - v_j^l)^\tau \\ &+ 2\beta \sum_{j=1}^{N^l} k_{ij}^l ((v_i^l)^\tau - (v_j^l)^\tau + ((v_i^0)^l)^\tau - ((v_j^0)^l)^\tau) \end{aligned} \tag{5}$$

The derivation of equation (5) takes advantage of the fact that $\partial I^l(X_j + v_i^0 + v_i)/\partial(v_i^l)^\tau = (I_{v_j}^l)^\tau$ and

$I^l_{v_j}(v^l_i)^\tau(I^l_{v_j})^\tau = (I^l_{v_j})^\tau I^l_{v_j}(v^l_i)^\tau$. Equation (5) is equivalent to equation (6) and equation (7) given below:

$$\begin{aligned}
\frac{-\partial E^l}{2\partial v^l_{x_i}} &= [2k^l_i\beta + 2l^l_i\gamma + g^l_i \sum_{X^l_j \in O(X^l_i)} I^2_{x_j}]v^l_{x_i} \\
&+ g^l_i \sum_{X^l_j \in O(X^l_i)} I^l_{x_j} I^l_{y_j} v^l_{y_i} - 2\beta \sum_{j=1}^{N^l} k^l_{ij} v^l_{x_j} \\
&- g^l_i \sum_{X^l_j \in O(X^l_i)} \delta I^l_j I^l_{x_j} + 2\beta \sum_{j=1}^{N^l} k^l_{ij}((v^0_{x_i})^l - (v^0_{x_j})^l) \\
&+ 2\gamma \sum_{j=1}^{N^l} l^l_{ij}(x^l_i - x^l_j + {T_x}^l_i - {T_x}^l_j + (v^0_{x+_i})^l - (v^0_{x_j})^l) \qquad (6)
\end{aligned}$$

$$\begin{aligned}
\frac{-\partial E^l}{2\partial v^l_{y_i}} &= [2k^l_i\beta + 2l^l_i\gamma + g^l_i \sum_{X^l_j \in O(X^l_i)} I^2_{y_j}]v^l_{y_i} \\
&+ g^l_i \sum_{X^l_j \in O(X^l_i)} I^l_{x_j} I^l_{y_j} v^l_{x_i} - 2\beta \sum_{j=1}^{N^l} k^l_{ij} v^l_{y_j} \\
&- g_i \sum_{X^l_j \in O(X^l_i)} \delta I^l_j I^l_{y_j} + 2\beta \sum_{j=1}^{N^l} k^l_{ij}((v^0_{y_i})^l - (v^0_{y_j})^l) \\
&+ 2\gamma \sum_{j=1}^{N^l} l^l_{ij}(y^l_i - y^l_j + {T_y}^l_i - {T_y}^l_j + (v^0_{y+_i})^l - (v^0_{y_j})^l) \qquad (7)
\end{aligned}$$

By letting all $\partial E^l/\partial v^l_{x_i} = 0$ and $\partial E^l/\partial v^l_{y_i} = 0$, a system of linear equations describing the incremental velocity field $\mathbf{V}^l$ can be obtained in the form of $\mathbf{A}(\mathbf{V}^l)^\tau = \mathbf{b}$, which can be solved with the LU decomposition method. The matrix **A** is given by:

$$\begin{pmatrix}
(I^l_{v_1})^\tau(I^l_{v_1}) + 2(\beta k^l_1 + \gamma l^l_1)I_0 & \cdots & -2(\beta k^l_{1,N} + \gamma k^l_{1,N})I_0 \\
-2(\beta k^l_{21} + \gamma k^l_{21})I_0 & \cdots & -2(\beta k^l_{2,N} + \gamma k^l_{2,N})I_0 \\
\cdots & \cdots & \cdots \\
-2(\beta k^l_{N,1} + \gamma k^l_{N,1})I_0 & \cdots & (I^l_{v_N})^\tau(I^l_{v_N}) + 2(\beta k^l_N + \gamma l^l_N)I_0
\end{pmatrix} \qquad (8)$$

where $I_0 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, $k^l_i = \sum_j k_{ij}$ and $l^l_i = \sum_j l^l_{ij}$. Vector **b** is given by:

$$\begin{aligned}
b_i &= g^l_i \sum_{X^l_j \in O(X^l_i)} \delta I^l_j (I^l_{v_j})^\tau - 2\sum_{j=1}^{N^l} \beta k^l_{ij}(((v^0_i)^l)^\tau - ((v^0_j)^l)^\tau) \\
&- 2\sum_{j=1}^{N^l} \gamma l^l_{ij}(X^l_i - X^l_j + T^l_i - T^l_j + ((v^0_i)^l - (v^0_j)^l)^\tau) \qquad (9)
\end{aligned}$$

The velocity field $\mathbf{V}^l_0$ is initialized under one of the following situations: when the object tracking procedure is initialized, the velocity field is assumed to be all 0. The pyramidal elastic matching algorithm starts at a given level (eg. $l = 2$) in the pyramid to compute the velocity field. When the velocity level at level l is known, it can be mapped to level $l - 1$. The velocity v at point (x, y) maps to point $(2x, 2y)$ and its value at level $l - 1$ is $2v$. The third situation is when the velocity field in previous frames ($t \leq t_0$) is known, in which case the Kalman filter is used to predict the velocity field at time $t_0 + 1$.

4.1 Using Images with Any Number of Channels

The aforementioned pyramidal elastic matching algorithm assumes that a single channel image is used. It is easy to show that it can be extended to input images with any number of channels. Suppose the multiple channel image is given by $\mathbf{I}^l(x, y) = (i_1(x, y), ..., i_D(x, y))$, where D is the number of channels. For a simple grayscale image, $D = 1$. In the case of an RGB image, $D = 3$. As mentioned in the description of the proposed object model, we use feature images with $D = 6$ in our experiments. For multiple channel images, equation (2) can be rewritten as:

$$\begin{aligned}
\epsilon^l(v_i) &= g^l_i \sum_{X^l_j \in O(X^l_i)} \sum_{d=1}^{D} \alpha_d [c^l_{j,d} - I^l_d(X_j + v_i)]^2 + \beta \sum_{j=1}^{N^l} k^l_{ij} \|v^l_i - v^l_j\|^2 \\
&+ \gamma \sum_{j=1}^{N^l} l^l_{ij} \|X^l_i + (v^0_i)^l + v^l_i - T^l_i - (X^l_j + (v^0_j)^l + v^l_j - T^l_j)\|^2 \qquad (10)
\end{aligned}$$

in which, α_d is the coefficient of dimension d. Its differential equation is given by:

$$\begin{aligned}
\frac{\partial E^l}{2\partial(v^l_i)^\tau} &= g^l_i \sum_{d=1}^{D} \alpha_d \sum_{X^l_j \in O(X^l_i)} ((I^d_{v_j})^l (v^l_i)^\tau - \delta(I^d_j)^l)((I^d_{v_j})^l)^\tau \\
&+ 2\gamma \sum_{j=1}^{N^l} l^l_{ij}(X^l_i - X^l_j + T^l_i - T^l_j + ((v^0_i)^l - (v^0_j)^l + v^l_i - v^l_j)^\tau) \\
&+ 2\beta \sum_{j=1}^{N^l} k^l_{ij}(v^l_i - v^l_j + (v^0_i)^l - (v^0_j)^l))^\tau \qquad (11)
\end{aligned}$$

4.2 Local Tuning

Note that equation (5) is obtained by assuming the Taylor expansion given in equation (4) is valid. That is, the incremental velocity v should be small at each point of the object (eg. $|v_x| < 1$ and $|v_y| < 1$). In other words, the initial velocity needs to be close to the actual velocity. By using a pyramidal image structure, the above condition is met if the elastic matching algorithm starts at a higher level in the pyramid where the velocities along the X axis and Y axis are small enough, or if the given initial velocity is close enough to the actual velocity. In addition to these methods, an iterative local tuning algorithm is designed to adjust the velocity locally, such that the final incremental velocity at each point is small enough. This local tuning algorithm is especially useful when only some of the object points have velocity values larger than 1 along the X axis or the Y axis. The local tuning algorithm is described below.

(1) For point X^l_i, assume every other v^l_j is fixed for $j \neq i$, then v^l_i can be obtained by solving the binary linear equations (6) and (7).

(2) If the incremental value $|v_x| > 1$, then the new value $v^0_x = v^0_x + sign(v_x)$. If the incremental value $|v_y| > 1$, then the new value $v^0_y = v^0_y + sign(v_y)$ where $sign(x) = 1$ if $x > 0$, and $sign(x) = -1$ if $x < 0$.

(3) Repeat steps (1) and (2) until all the incremental values $|v_x| < 1$ and $|v_y| < 1$, or the maximum number of iterations is met. The final velocity for each point is $v^0 + v$.

When the occlusion hypothesis changes at some of the points, the local tuning algorithm is also used to recompute the velocity at those points.

4.3 Occlusion Reasoning

When occlusion exists, it is necessary to update the occlusion hypothesis for each point when its velocity is available. A reasonable assumption about occlusion is that an object is occluded gradually instead of suddenly. It is also assumed that an occluded object becomes unoccluded gradually instead of suddenly. Therefore, only those points near the object boundary or the boundary of the occluded area are chosen as candidates to be updated. In order to decide whether a point is occluded or not, the Mahalanobis distance between the feature vector associated with the countour point $\mathbf{c}(x, y)$ and the feature vector associated with the corresponding image point $\mathbf{I}(x+v_x, y+v_y)$ is computed as $d = [\mathbf{c}(x,y)-\mathbf{I}(x+v_x, y+v_y)]\Sigma^{-1}[\mathbf{c}(x,y)-\mathbf{I}(x+v_x, y+v_y)]^\tau$. If d is above a certain threshold, this point is classified as occluded, otherwise it is classified as unoccluded. When multiple objects correspond to the same point in the image, only one object can be visible at this image point. The object with the minimum distance d to this point in the image is classified as the visible object at this image point.

After the occlusion reasoning is performed, the resulting information is fed back to the elastic matching algorithm, and the tuning algorithm is used to adjust the velocity field locally. At most two iterations are needed for the occlusion hypothesis to be updated and the resulting information fed back to elastic matching algorithm.

4.4 Adaptation of the Object Template

As an object moves, its shape and color features change dynamically. Thus, the object shape model and the color features of the points on the object need to be updated at each iteration. For a point which is not occluded, its feature point is updated as: $\mathbf{c}(X_i, t+1) = \mathbf{c}(X_i, t) + \rho[\mathbf{I}(X_i + v_i; t+1) - \mathbf{c}(X_i, t)]$, $\Sigma^2(X_i; t+1) = \Sigma^2(X_i; t) + \rho[(\mathbf{I}_k(X_i + v_i; t+1) - \mathbf{c}_k(X_i; t))(\mathbf{I}_k(X_i + v_i; t+1) - \mathbf{c}_k(X_i; t))^\tau - \Sigma^2(X_i, t)]$, where ρ is a given learning rate.

5 Velocity Field Approximation Using B-spline Surfaces

The elastic matching algorithm yields a mapping which minimizes an energy function that takes into account both, feature similarity and shape distortion during tracking. The computed mapping determines the displacement of each point along the X and Y axes, i.e. the velocity of each point. However, this mapping may also yield some noisy and false matches that do not reflect the actual motion of the object. Hence B-spline surfaces are used to smooth the velocity field and suppress the effect of the noisy and false matches. Provided the velocity in $V = \{v_k\} = \{(v_{k,x}, v_{k,y})\}$ is known, the following procedure is used to determine the $N_X \times N_Y$ B-spline control points in order to approximate V. For each point location $X_k = (x_k, y_k)$ of the object, the corresponding B-spline control parameter $(\hat{u}_k, \hat{v}_k)$ is estimated as:

$$(\hat{u}_k, \hat{v}_k) = ((N_X - m + 1) * x_k/W, (N_Y - n + 1) * y_k/H) \qquad (12)$$

where W and H are the width and height of the object respectively. The estimated velocity component along the X axis $\hat{v}_{k,x}$ is expressed in terms of $(\hat{u}_k, \hat{v}_k)$ as follows:

$$\hat{v}_{k,x}(\hat{u}_k, \hat{v}_k) = \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} d^x_{i'_k, j'_k} N_i^m(u_k^1) N_j^n(v_k^1) \qquad (13)$$

where the $d_{i,j}$'s are the control points which determine the association of a given point on the B-spline surface with the control parameters (u, v), $N_i^m(u)$ and $N_j^n(v)$ are the basis functions along the u and v axes respectively, and n and m are the orders of the B-spline ($m = n = 4$ in our case). Here $i'_k = i + u_k^0$, $j'_k = j + v_k^0$, $(u_k^0, v_k^0) = (\lfloor \hat{u}_k \rfloor, \lfloor \hat{v}_k \rfloor)$, and $(u_k^1, v_k^1) = (\hat{u}_k - u_k^0, \hat{v}_k - v_k^0)$. Equation (13) can be further generalized as $\hat{v}_{k,x}(\hat{u}_k, \hat{v}_k) = \sum_{i=0}^{N_X} \sum_{j=0}^{N_Y} d^x_{i,j} B_k(i,j)$, where $B_k(i,j) = N^{m-1}_{i-\hat{u}_k^0}(\hat{u}_k^1) N^{n-1}_{j-\hat{u}_k^0}(\hat{v}_k^1)$, if $\lfloor \hat{u}_k \rfloor \le i < \lfloor \hat{u}_k \rfloor + m - 1$ and $\lfloor \hat{v}_k \rfloor \le j < \lfloor \hat{v}_k \rfloor + n - 1$; $B_k(i,j) = 0$ otherwise.

For each point associated with an object, minimization of the following objective function is used to determine the values of the $L = N_X \times N_Y$ control points:

$$E = \sum_{k=1}^{N} \| v_{k,x} - \hat{v}_{k,x}(\hat{u}_k, \hat{v}_k) \|^2 \qquad (14)$$

where $(\hat{u}_k, \hat{v}_k)$ are the estimated control parameters computed using equation (12), and N is the total number of points of the object. The minimization entails solving a system of equations given by $\partial E / \partial \mathbf{d}^x = 0$, which, in turn, can be represented by equation $\mathbf{A}\mathbf{d}^x = \mathbf{b}^x$, where $\mathbf{A}$ is given by:

$$\begin{pmatrix} \sum_{k=1}^N B_k(0,0)B_k(0,0) & \dots & \sum_{k=1}^N B_k(0,0)B_k(N_X, N_Y) \\ \sum_{k=1}^N B_k(0,1)B_k(0,0) & \dots & \sum_{k=1}^N B_k(0,1)B_k(N_X, N_Y) \\ \dots & \dots & \dots \\ \sum_{k=1}^N B_k(N_X, N_Y)B_k(0,0) & \dots & \sum_{k=1}^N B_k(N_X, N_Y)B_k(N_X, N_Y) \end{pmatrix}$$

and $\mathbf{b}^x = (\sum_{k=1}^N B_k(0,0)v_{k,x}, \sum_{k=1}^N B_k(0,1)v_{k,x}, ..., \sum_{k=1}^N B_k(N_X, N_Y)v_{k,x})^\tau$.

The system of equation can be solved using LU decomposition. The value of N_X and N_Y is usually small for rigid object tracking. In our experiments, 4×4 control points can approximate the velocity field in the image plane resulting from 3-D movement of a planar object (translation, rotation or a combination of the two) with negligibly small mean squared error (MSE). Although we have not examined the MSE resulting from the approximation, using 4×4 control points, of the velocity field in the image plane resulting from 3-D movement of a 3-D object, our experiments on human tracking yield very good results. Since an elastic object with restricted deformation can be usually approximated by a rigid object with articulated motion, the resulting velocity field can still be approximated using 4×4 control points. More control points are necessary only for modeling complex and/or abrupt motion and deformation of a highly elastic object. To further reduce the computational complexity, the values of N_i^m and N_i^n can be precomputed and stored in a lookup table.

6 Velocity Estimation Model

A Kalman filter is used to estimate/predict the velocity field of an object. Note that the term velocity field, in the Kalman filter algorithm, actually denotes the control point values resulting from the B-spline approximation of the velocity field. The canonical Kalman filter used in this paper can be described using the following equations:

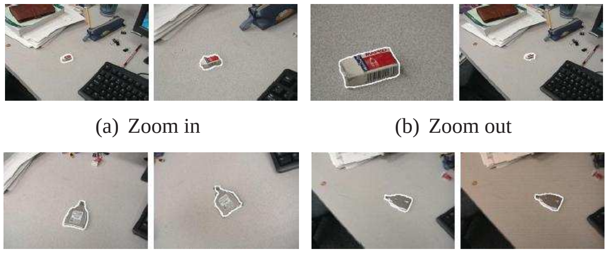
$$\hat{\mathbf{V}}^{-}_{d,k+1} = \hat{\mathbf{V}}^{+}_{d,k} + \mathbf{q}_k \qquad (15)$$

$$\mathbf{Z}_k = \hat{\mathbf{V}}^{-}_{d,k} + \mathbf{v}_k \qquad (16)$$

where $\mathbf{V}_d$ is the estimated/predicted velocity field, and $\mathbf{Z}_k$ is the actually measured velocity field. Equation (15) represents the prior estimation of $\mathbf{V}_d$ whereas equation (16) describes the linear relation between the estimated $\mathbf{V}_d$ and the actually measured velocity field $\mathbf{Z}_k$. Variables $\mathbf{q}_k$ and $\mathbf{v}_k$ represent random noise in the prior estimation and actual measurement of the velocity field respectively. Both $\mathbf{q}_k$ and $\mathbf{v}_k$ are modeled as Gaussian white noise with distributions $\mathcal{N}(0, \mathbf{Q})$ and $\mathcal{N}(0, \mathbf{R})$ respectively.

7 Experimental Results

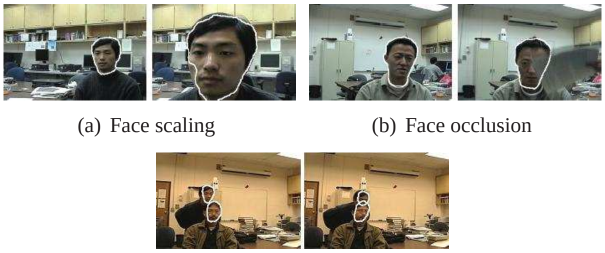
The proposed tracking algorithm has been applied to various tracking scenarios. In our current experiment, the objects are initialized manually by labeling their contours over the first image. Figure 3(a) shows the snapshots of the tracking of an eraser in a video while the camera is zooming in. Figure 3(b) shows the snapshots of the tracking of an eraser in a video while the camera is zooming out. Figure 3(c) shows the snapshots while the tracked object is rotating in the image plane and Figure 3(d) shows the snapshots while the scene is subject to global change in illumination. Figure 3 shows that the proposed tracking algorithm can handle large changes in object size (i.e., significant scale changes) and handle object rotation and scene illumination change due to the adaptive nature of the object template and the robustness of the features used. Experiments are also conducted on face tracking. Figure 4(a) shows the tracking result when the tracked face exhibits large scale changes in the video. Figure 4(b) shows the tracking result in the presence of occlusion thus demonstrating that the occlusion reasoning is very robust in handling occlusions. Figure 4(c) shows the tracking results on two faces where one face occludes another. The various tracking results can be viewed in the video accompanying this paper.



(a) Zoom in (b) Zoom out

(c) Rotation (d) Illumination change

Figure 3. Zooming, rotation, illumination change



(a) Face scaling (b) Face occlusion

(c) Two face tracking

Figure 4. Face Tracking

8 Conclusions

A novel hybrid region-based and contour-based multiple object tracking model using elastic matching is proposed. The elastic matching algorithm exploits both, the spectral features and contour-based features of the tracked objects, making it more robust and general in the context of object tracking. The proposed elastic matching algorithm uses a multiscale optical flow computation algorithm to compute the velocity field. This prevents the multiscale elastic matching algorithm from being trapped in a local optimum unlike conventional elastic matching algorithms that use a heuristic search procedure in the matching process. The proposed tracking framework can be viewed as a generalization of the traditional linear Kalman filter where the multiscale elastic matching algorithm is used to compute the velocity field which is then approximated using B-spline surfaces. The control points of the B-spline surfaces are directly used as the tracking variables in a Kalman filtering model. The B-spline approximation of the velocity field is used to update the spectral features of the tracked objects in the Kalman filter model. The dynamic nature of these spectral features are subsequently used to reason about occlusion. Experimental results on tracking of multiple objects in real-time video are presented.

Experimental results show that the proposed algorithm is very efficient in handling occlusions and changes in scale and illumination. However, it is observed that a single object hypothesis is not sufficient to handle all possible tracking scenarios. Without a suitable foreground or background model, this scheme is not suitable for long term tracking. Future work will integrate the optical flow based elastic matching model with a foreground object detection algorithm and particle filtering algorithm to achieve robust and consistent tracking.

References

[1] P. Li, T. Zhang, and A.E.C. Pece, Visual contour tracking based on particle filters, *Img. Vis. Comput.*, Vol. 21, 2003, pp. 111-123.

[2] M. Isard and J. MacCormick, BraMBLe: A Bayesian Multiple-Blob Tracker, *Proc. ICCV*, Vancouver, Canada, Vol. 2, July 2001, pp. 34-41.

[3] S.J. McKenna, S. Jabri, and Z. Duric, A. Rosenfeld, H. Wechsler, Tracking Groups of People, *CVIU*, Vol. 80, 2000, pp. 42-56.

[4] K. Nummiaro, E. Koller-Meier, and L.V. Gool, An adaptive color-based particle filter, *Img. Vis. Comput.*, Vol. 21, No. 1, 2003, pp. 99-110.

[5] A. Blake, R. Curwen, and A. Zisserman, A framework for spatio-temporal control in the tracking of visual contours, *IJCV*, Vol. 11, No. 2, 1993, pp. 127-145.

[6] M. Lades, J. Vorbruggen, J. Buhmann, J. Lange, C.V.D. Malburg, and R. Wurtz, Distortion Invariant Object Recognition in the Dynamic Link Architecture, *IEEE Trans. Comp.*, Vol. 42, No. 3, 1992, pp. 300-311.

[7] D. Koller, J.W. Weber and J. Malik, Robust Multiple Car Tracking with Occlusion Reasoning. *Proc. ECCV* Stockholm, Sweden, 1994, pp. 189-196.

[8] D. Terzopoulos and R. Szeliski, Tracking with Kalman Snakes, in *Active Vision*, MIT Press, Cambridge, MA, USA, 1993, pp. 3-20.

[9] W.J. Rucklidge, Locating Objects Using the Hausdorff Distance, *Proc. ICCV*, Massachusetts USA. June, 1995, pp. 457-464.

[10] S. Malik, G. Roth, and C. McDonald, Robust Corner Tracking for Real-Time Augmented Reality, *Proc. Vision Interface*, Calgary, Alberta, Cananda, May, 2002, pp. 399-406.

[11] V. Lepetit, J. Pilet, and P. Fua, Point Matching as a Classification Problem for Fast and Robust Object Pose Estimation. *Proc. IEEE Conf. CVPR*, Washington DC, USA, June 2004. Vol. 2, pp. 244-250.

[12] D. Chung, W.J. MacLean, and S. Dickinson, Integrating Region and Boundary Information for Improved Spatial Coherence in Object Tracking, *Proc. IEEE Conf. CVPR*, Washington DC, USA, June 2004.

[13] D. Comaniciu, V. Ramesh, and P. Meer, Kernel-Based Object Tracking, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, May 2003, Vol. 25, No. 5, pp.564-567.

[14] J. Sullivan, A. Blake, M. Isard, and J. MacCormick, Object Localization by Bayesian Correlation, *Proc. ICCV*, Corfu, Greece, September 1999, Vol.2, pp. 1068-1075.

[15] R. Bajcsy, and S. Kovacic, Multiresolution Elastic Matching, *Computer Vision, Graphics, and Image Processing*, April 1989, Vol. 46 , Issue 1, pp.1-21.

[16] C. Chuang, and C. Kuo, Wavelet Deformable Model for Shape Description and Multiscale Elastic Matching, *SPIE's Symposium on Visual Communications and Image Processing*, Cambridge, MA, Nov 1993.

[17] M. Tang, and S. Ma, A Fast Algorithm of Multiresolution Elastic Matching, *The 10th Scandinavian Conference on Image Analysis*, Lappeenranta, Finland, June 1997.

[18] P. Golland, and A.M. Bruckstein. Motion from color. Center for Intelligent Systems TR #9513, Computer Science Department, Technion, Haifa, August 1997.

[19] V. Markandey, and B.E. Flinchbaugh. Multispectral constraints for Optical Flow Computation. Proc. 3rd Intl. Conf. on Computer Vision. Osaka, Japan, December 1990. pp.38-41.

[20] J. Bouguet, Pyramidal Implementation of the Lucas-Kanade Feature Tracker: Description of the algorithm, *OpenCV Documentation*, http://www.intel.com/research/mrl/research/opencv/

[21] P.J. Burt, and E.H. Adelson, The Laplacian Pyramid as a Compact Image Code, *IEEE. Trans. on Communications*, April 1983, Vol. COM-31, No.4, pp.532-540.

[22] K. Shafique, and M. Shah, A Noniterative Greedy Algorithm for Multiframe Point Correspondence, *IEEE. Trans. PAMI*, Januray 2005, Vol. 27, No. 1, pp 51-65.

[23] A. Yilmaz, X. Li, and M. Shah, Contour-Based Object Tracking with Occlusion Handling in Video Acquired Using Mobile Cameras, *IEEE. Trans. PAMI*, Novemebr 2004, Vol. 26, No. 11, pp. 1531-1536.