2007 IEEE International Conference on
Robotics and Automation
Roma, Italy, 10-14 April 2007

Combining Texture and Edge Planar Trackers based on a local Quality Metric

A. H. Abdul Hafez [1,2], Visesh Chari [1], and C.V. Jawahar [1]

***Abstract*— A new probabilistic tracking framework for integrating information available from various visual cues is presented in this paper. The framework allows selection of "good" features for each cue, along with factors of their "goodness" to select the best combination form. Two particle filter based trackers, which use edge and texture features, run independently. The output of the master tracker is computed using democratic integration using the "goodness" weights. The final output is used as apriori for both tracker in the next iteration. Finally, particle filters are used to deal with non-Gaussian errors in feature extraction / prior computation. Results are shown for planar object tracking.**

I. INTRODUCTION

Object tracking is an important task in robotic vision, particularly for visual servoing [1]. The tracking problem in the robotic literature has been modeled as a motion estimation problem. Thus 3D model based tracking is considered as a pose estimation problem and 2D planar object tracking as a homography estimation problem. This class of trackers differs from the popular class of algorithms which aim at drawing a bounding box for an object of interest, for every successive frame. There are two major sources of visual features that are used in marker-less visual tracking: edges and texture. Both visual features have advantages and disadvantages that make them suitable/unsuitable in many scenarios.

For scenes with sharp edges and high spatial gradients, contour features are very informative. Active contours can be used to track complex shapes in 2D tracking systems as in [2] and [3]. Straight lines are more suitable for model-based 3D tracking applications [4]. When the scene is very cluttered or textured, contours may be absent. In addition, shadows may considerably affect the edge detector performance. The tracker may deviate towards the edge of the shadow itself. Texture-based methods are required in such situations.

Interest points are useful for tracking of textured scenes. The Shi-Tomasi algorithm [5] for detecting "good" feature points is considered as an effective algorithm for identifying texture points that are tractable. Unfortunately, tracking of interest points is sensitive to the quality of the image. In the presence of poor and noisy images, the tracking process fails. Another inconvenience is the effect of the size of the tracking window. Errors in the tracking process may also occur as a result of large displacements or changes in illumination.

[1] Center for Visual Information technology, International Institute of information Technology, Gachibowli, Hyderabad-500032, India {ukvisesh@students,jawahar@}iiit.ac.in

[2] Dept. of Computer Science and Engineering, University college of Engineering, Osmania University, Hyderabad-500007, India hafezsyr@ieee.org

To perform accurate edge tracking, an accurate prior of the motion model is needed. Points are robust to large motion and the dynamic model of their distribution can be learned easily. Indeed, whenever we have a better prior of feature (point or edge), it is more useful for the tracker. In general, weights corresponding to feature goodness can be assigned to features. Each feature can contribute to the distribution proportionally to its goodness factor value.

Statistically, the posterior of line or edge measurements is non-Gaussian; so we need a nonlinear filter like particle filter to model the posterior. In addition, we need a robust technique to withstand large aspect changes. For example, large changes in illumination can cause changes in the intensity of interest points, making them inappropriate for tracking. Another example is the large rotation motion which may result in an incorrect edge correspondence.

Considering the complementary advantages and inconvenience of each of the two methods, it is matured to consider both edges and textures together [1],[6], [7]. In most of the integration cases between contour and texture, the integration is done sequentially. For example, [8] uses the result from the texture point tracker to provide better positioning of the edge location.

In this paper, we propose a robust integration framework using both edge and texture features. This framework probabilistically integrates the visual information collected from contour and texture. The integration is based on probabilistic goodness weights for each type of feature. The weighting functions have been developed starting from the dissimilarity of point features. In fact, we also use this measurement to identify good edge features. The motion posteriori is then the weighted sum of the posteriors computed from each feature likelihood separately. The likelihood models of texture and contour are defined in a robust fashion to meet the large aspect changes. It is the point wise mth smallest value of the classical likelihood model. Integration of cues results in good performance even in situations with widely varying scene illumination. The integrated tracker shows impressive results due to the robust likelihood function and the integrated trackers.

There have been many recent attempts at 2D tracking using integration of multiple cues. Integration of 2D cues like color, motion, and edges using fuzzy-based voting technique was considered by Kragic and Christensen in [9]. Li and Chaumette [10] probabilistically integrated the visual cues like color, structure, and contour. They use particle filter for maximizing a likelihood function that fuses probabilistically the mentioned cues. Other probabilistic methods for integra-

1-4244-0602-1/07/$20.00 ©2007 IEEE.

tion of edges and texture are those presented in [11], and [6], they are based on some priority criteria for each of contour and point features. In the context of minimizing a deterministic cost function, the work presented in [1], [12] is the most recent one. They proposed a first-order optimization process to minimize the error between the image measurements and the reprojected one from the reference frame onto the current frame.

II. MOTION ESTIMATION FOR PLANAR OBJECT BAYESIAN TRACKING

A. Motion Model

Given an initial image I^0 and an image I^t of a planar object at time instant t, there exists a homography H_t relating these images. If the vector $x_0 = [u_0, v_0, 1]^T$ represents the homogeneous coordinates of a point in the first image and the vector $x_t = [u_t, v_t, 1]^T$ represents a point in the second image, the relation between these two points is written as $x_t = H_t x_0$ or

$$x_t \sim \begin{bmatrix} h_t^1 & h_t^2 & h_t^3 \\ h_t^4 & h_t^5 & h_t^6 \\ h_t^7 & h_t^8 & h_t^9 \end{bmatrix} x_0. \quad (1)$$

Estimating the homography can be posed as estimating the parameters of H_t, represented as a vector

$$h_t = \begin{bmatrix} h_t^1 & h_t^2 & h_t^3 & h_t^4 & h_t^5 & h_t^6 & h_t^7 & h_t^8 & h_t^9 \end{bmatrix}^T. \quad (2)$$

If we assume that the camera/object motion is smooth, then h_t can be written as an increment over the homography computed in the previous frame. Thus if $\hat{h}_t$ is the current homography estimate, change in this vector owing to inter frame displacement can be written as

$$\hat{h}_t = \hat{h}_{t-1} + \Delta\hat{h}_t. \quad (3)$$

Now, let us consider a set F_0 of M visual features $F_0 = \{f_0^1, \ldots, f_0^M\}$ in the reference image I^0. This set of features is mapped by the transformation $\hat{h}_t$ to the estimated set of features $F_{h_t} = \{f_{\hat{h}_t}^1, \ldots, f_{\hat{h}_t}^M\}$ in the current image. The true estimate of the vector $\hat{h}_t$ can be computed by minimizing an error function of the form

$$\mathcal{G}(\hat{h}_t) = \mathcal{F}(F_{h_t} - F_t), \quad (4)$$

where F_t is the measured visual feature vector and $\mathcal{F}$ is the distance measure. The optimal value $\hat{h}_t$ is given as

$$\hat{h}_t = \arg\min_{h_t} \mathcal{F}(F_{h_t} - F_t), \quad (5)$$

When the errors in feature correspondence are non-Gaussian, there exists no direct linear analytical method that can minimize this error function. Particle filter algorithm or what is called Condensation algorithm[13] is preferred here because it provides an efficient probabilistic framework to take care of such uncertainties.

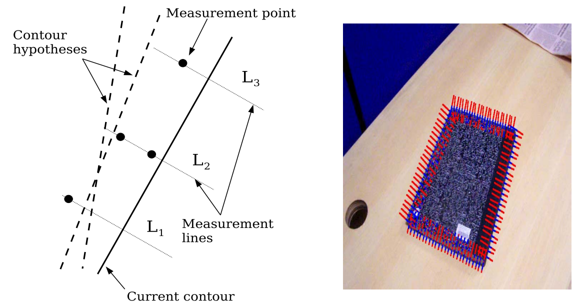


Fig. 1. Left: Contour features in the current frame are obtained by searching along lines perpendicular to the contour estimated in the previous frame. Right: Contour extraction shown in practice.

B. Particle Filter based Bayesian Tracking

We now present the Bayesian filter [10] formulation for computing homography h_t. Let $\pi(h_t)$ be the belief of the random vector h_t at time t represented by posterior probability $p(h_t \mid F_{1,\ldots,t})$ based on features $F_{1,\ldots,t}$. Expanding using Bayes rule

$$p(h_t \mid F_{1,\ldots,t}) = \frac{p(F_t \mid h_t)p(h_t \mid F_{1,\ldots,t-1})}{p(F_T \mid F_{1,\ldots,t-1})}. \quad (6)$$

Considering that $p(F_T \mid F_{1,\ldots,t-1})$ is a constant we marginalize the probability $p(h_t \mid F_{1,\ldots,t-1})$ and apply Bayes' rule again to obtain the Bayesian estimation $p(h_t \mid F_{1,\ldots,t})$ as

$$\alpha\, p(F_t \mid h_t) \int p(h_t \mid h_{t-1})p(h_{t-1} \mid F_{1,\ldots,t-1})dh_{t-1}. \quad (7)$$

Equation (5) can be understood as the *maximum posteriori* (MAP) of (7). Thus, equation (4) is the likelihood $p(F_t \mid h_t)$ and equation (3) represents the motion model prior $p(h_t \mid h_{t-1})$. while $p(h_{t-1} \mid F_{1,\ldots,t-1})$ is the posterior estimate in the the previous iteration.

The basic idea of particle filters is to approximate posterior density $p(h_t \mid F_{1,\ldots,t})$ by a set of samples (particles) h_t^i with associated weights or importance factors w_t^i. The M particle-weight pairs $\{h_{t-1}^i, w_{t-1}^i\}_{i=1}^M$,chosen to approximate density $p(h_{t-1} \mid F_{1,\ldots,t-1})$, are propagated to pairs $\{h_t^i, w_t^i\}_{i=1}^M$ using the motion model prior $p(h_t \mid h_{t-1})$. A detailed explanation of particle filters and their use to represent probability distributions can be found in [14]. The weights $\{w_t^i\}_{i=1}^M$ associated to the particles h_t^i are computed proportional to the likelihood function in case of using the bootstrap filter as

$$w_t^i = \alpha\, p(F_t \mid h_t^i). \quad (8)$$

III. THE OVERALL ALGORITHM

In our approach, the visual feature F can be an edge of contour feature F_C or texture point feature F_T. Thus, the functions $p(F_C \mid h_i)$ and $p(F_T \mid h_i)$ are the likelihood functions of the contour and texture point features respectively.

The overall algorithm using particle filters is explained in Figure 2 and Algorithm 1. The input to the algorithm is a sequence of images $I_1, \ldots, I_t$, initial features $F_1 =$

$f_i^1 : i \in 1, \ldots, n$ and particle filter's output *i.e.* the motion estimate is initialized to identity (here the reference frame is the current one). The algorithm undergoes one iteration of particle filters for each frame of the sequence, in order to produce a homography estimate between the current and reference frames, using features belonging to the current frame, and previous frames.

At each time instance t, M particles from the priori distribution $h_{t-1|t-1}$ are drawn where 0 is the reference frame. Two sets of particles are drawn for evaluation. These particles are propagated to the current frame using the *a priori* motion model $h_{t|t-1}$. This motion model may be calculated from edge or texture features. Here, one set of particles is propagated using texture based motion priors, the other set is propagated using edge based ones. In both cases, this motion model is calculated using a least squares minimization. In case of edge features, the motion apriori is approximated using an affine motion model. Once the particles are propagated to the current frame, edge and texture likelihoods, (see (17), (19) and (20) in Section V), are used to evaluate the particles and select the appropriate homography (Algorithm 1). Finally, edge-based and texture-based homographies are combined using democratic integration. The combination weights W_T and W_C are calculated adaptively as in Section IV.

Algorithm 1 Visual Tracking based on Goodness Weight

Input: $I_0, \ldots, I_t$, $F_0 = \{f_i^0 : i \in \{1, \ldots, n\}\}$.
Output: $h_t = \{h_t^1, \ldots, h_t^9\}$
NumParticles: $M = \{M_C + M_T\}$ {Set the number of particles needed to sample the space effectively}
for $k \in \{1, \ldots, t\}$ **do**
  $F = \mathrm{ExtractFeatures}(I_k)$
  $F_k = \mathrm{TrackFeatures}(I_k, F, F_{k-1})$
  $\{\mathrm{h}_{\mathrm{k-1}}^{\mathrm{i}}\}_{i=1}^{M_C} = \mathrm{DrawSamples}(h_{k-1}, M_C)$
  $\{\mathrm{h}_{\mathrm{k-1}}^{\mathrm{i}}\}_{i=1}^{M_T} = \mathrm{DrawSamples}(h_{k-1}, M_T)$
  $\{\mathrm{h}_{\mathrm{k-1}}^{\mathrm{i}}\}_{i=1}^{M} = \{\{\mathrm{h}_{\mathrm{k-1}}^{\mathrm{i}}\}_{i=1}^{M_C}, \{\mathrm{h}_{\mathrm{k-1}}^{\mathrm{i}}\}_{i=1}^{M_T}\}$
  $h_{k|k-1} = \mathrm{MotionPrior}(F_k, F_{k-1})$
  $\{\mathrm{h}_{\mathrm{k}}^{\mathrm{i}}\}_{i=1}^{M} = \mathrm{PropogateParticles}(\{\mathrm{h}_{\mathrm{k-1}}^{\mathrm{i}}\}_{i=1}^{M}, h_{k|k-1})$
  $\{Q_C\}_{i=1}^{M_C} = \log(\mathrm{CLikelihood}(F_k, \{\mathrm{h}_{\mathrm{k}}^{\mathrm{i}}\}_{i=1}^{M_C}))$
  $h_k^C = \arg\min_i \{Q_C\}_{i=1}^{M_C}$
  $\{Q_T\}_{i=1}^{M_T} = \log(\mathrm{TLikelihood}(F_k, \{\mathrm{h}_{\mathrm{k}}^{\mathrm{i}}\}_{i=1}^{M_T}))$
  $h_k^T = \arg\min_i \{Q_T\}_{i=1}^{M_T}$
  $W_C = \mathrm{GoodEdges}(F_k, F_{k-1})$
  $W_T = \mathrm{GoodTextures}(F_k, F_{k-1})$
  $h_k = W_C * h_k^C + W_T * h_k^T$ {Democratic Integration}
end for

IV. THE WEIGHTED GOOD FEATURE

Shi and Tomasi [5] developed a method that measures the dissimilarity between image point features. They found out a measurement matrix whose eigenvalues are large when the dissimilarity is less. In other words, if the measurement matrix has a large enough eigenvalue, the feature is considered

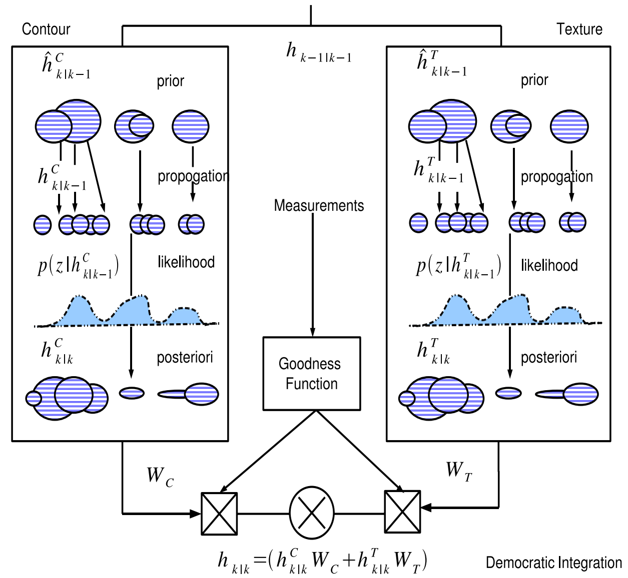


Fig. 2. Steps in one iteration of our algorithm. The motion posterior $h_{k-1|k-1}$ computed in the previous frame/iteration is propagated separately using texture and contour/edge information to get the initial priors $h_{k|k-1}^C$ and $h_{k|k-1}^T$. Particles are then drawn from these two distributions separately and the features observed in the current frame are evaluated on these particles using robust likelihood functions. The resulting posteriors $h_{k|k}^C$ and $h_{k|k}^T$ are then combined using democratic integration. The weights W_C and W_T are computed based on the "goodness" of features.

as a good feature to track. We generalized this concept about points, given by Shi and Tomasi, to the case of edge features located on a measurements line. Thus, we start from these goodness measurements to define a function that associate a probabilistic weights for the edge (contour) and texture features.

A. Good Features to Track

Similar to [5], the affine image motion between successive frames is computed as one that minimizes the intensity *dissimilarity*

$$\epsilon = \int \int_W [J(A\mathbf{x} + \mathbf{d}) - I(\mathrm{x})]^2 w(\mathbf{x}) d\mathbf{x} \qquad (9)$$

where W represents the feature window around a Harris corner and $w(\mathbf{x})$ is a weighting function. Using Taylor expansion and after a few simplifications, we arrive at the following equation for determining a "good" feature.

$$Z\mathbf{d} = e$$

where Z represents the covariance of the image derivative

$$Z = \begin{bmatrix} g_x^2 & g_x g_y \\ g_x g_y & g_y^2 \end{bmatrix}$$

For a tractable feature, it is required that the matrix Z has large eigenvalues.

1) Good Texture Features:: Texture features essentially represent a pattern in which intensity change within a feature window is present along both x and y-directions. Thus, the two eigenvalues are both large and comparable in magnitude. Thus ensuring that the minimum eigenvalue is above a threshold suffices to find “good” texture points. This is expressed by the equation

$$\min(\lambda_1, \lambda_2) > \lambda_p \tag{10}$$

where (λ_1, λ_2) are the eigenvalues and λ is a chosen constant.

2) Good Edge Features:: In case of edge features, the intensity pattern in the feature window is unidirectional. Since the feature window needs large eigenvalues to be resilient to noise, the above condition for texture applies here as well. However, due to the uni-directional nature of edges, one eigenvalue is significantly smaller than the other. Thus, we may impose the following additional constraint to acquire “good” edge features

$$\max(\lambda_1, \lambda_2) > \lambda_e. \tag{11}$$

B. Assigning weights to features

Given a set of good texture and edge features, we may define the goodness of each type of feature by measuring the amount of features that are tracked along the sequence of frames. Let us remember that we consider here two types of visual features. They are contour feature F_C and texture feature F_T. Texture feature is essentially an image point; while contour feature is a gradient peak along a measurement line. We develop a goodness function for texture point features and the one for contour features is analogous.

Assume we select N_0 texture point features in the initial frame. Only N_t features in the current frame have been selected as good features and matched to its corresponding points in the initial frame. Let the set $\mathcal{N}_t = \{n_i \mid i = 1, \cdots, N_t\}$ be the set good features tracked in the current frame. The probability that a point feature n_i is in this set can be given as a function of the dissimilarity measurement given in (9) as

$$W_T = p(\mathcal{N}_t \in \mathcal{N}_0) = p(\mathcal{N}_t \in \mathcal{N}_{t-1})\ p(\mathcal{N}_{t-1} \in \mathcal{N}_0) \tag{12}$$

$$p(\mathcal{N}_t \in \mathcal{N}_{t-1}) = \frac{1}{N_{t-1}} \sum_{i=1}^{N_{t-1}} \frac{1}{2\pi\sigma^2} \exp\left[-\frac{\epsilon_i^T \epsilon_i}{2\pi\sigma^2}\right] \tag{13}$$

To simplify the computation, let $p(\mathcal{N}_{t-1} \in \mathcal{N}_0) \approx \frac{N_{t-1}}{N_0}$ and finally we write

$$W_T = \frac{1}{N_0} \sum_{i=1}^{N_{t-1}} \frac{1}{2\pi\sigma^2} \exp\left[-\frac{\epsilon_i^T \epsilon_i}{2\pi\sigma^2}\right]. \tag{14}$$

For edges, assume that there are N_0 measurement lines on the object contour in the initial frame and N_t matched good feature measurement lines in the current frame. Analogous to texture point features, the weighting function for edge features can be written as

$$W_C = \frac{1}{N_0} \sum_{i=1}^{N_{t-1}} \frac{1}{2\pi\sigma^2} \exp\left[-\frac{\epsilon_i^T \epsilon_i}{2\pi\sigma^2}\right]. \tag{15}$$

This allows us to get a quantitative evaluation of the feature’s reliability. The higher the weight, the more reliable a feature is. By comparing the weights, we may decide upon the most reliable feature.

V. FEATURE LIKELIHOOD

A. Contour likelihoods

Let C_{t-1} be the contour representing the object in the previous frame at instant $(t-1)$. This contour, for simplicity, has been assumed to be an edge. Let h_{t-1} be the vector representing the homography that relates C_{t-1} to a reference contour C_0. A suitable discretization of the both contours C_{t-1} and C_0 is represented by the set of image points $\{p^m\}_{m=1}^{M}$. Points belonging to the current contour C_t can be searched along lines l_m perpendicular to the previous contour C_{t-1} centered at points $\{p_{t-1}^m\}_{m=1}^{M}$. Points $\{p_{t-1}^m\}_{m=1}^{M}$ are called principle points and lines l_m are called measurement lines.

Fig. 1 demonstrates the measurement process to estimate the contours in the current frame. The object contour represents the object edge at the previous frame. The measurement lines are drawn normal to, and centered around the previous edge. In the figure, there are three normal measurement lines with one or more edge points detected on each line. Two edge proposals are shown and only the nearest edge point measurement to the edge proposal is considered. In case the measurement line does not intersect the edge proposal, as the line L_3, the data from this line will be nullified and will not be used in the likelihood function. Note here that by using short measurement lines, we reduce the effect of spurious contour points in cluttered environments.

A generic model of the contour likelihood was proposed in [3]. We start from this model to develop our robust contour likelihood model. Let the hypothesis h_t^i be a proposal of the state of the contour C_t that intersects the measurement line l_m at a distance d_m from the same principle point (Fig. 1). Starting from the generic likelihood model after some development and enforcing certain assumptions [3], we write

$$p(F_C \mid h^i) = \prod_{m=1}^{\bar{M}} \left(\frac{1}{\sqrt{2\pi}\ \sigma} \sum_{k=1}^{n_m} \exp(-\frac{(D_m)^2}{2\sigma^2}) \right) = \prod_{m=1}^{\bar{M}} Q_m, \tag{16}$$

where $D_m = \min\{v_k - d_m\}_{n_m}^{k=1}$ represents the sum along each line approximated by its large value to speed the process. The number of measurement lines that intersect the contour C_t is $\bar{M}$. Taking log to simplify multiplication, we can write

$$Q_C = \log(p(F_C \mid h^i)) = \sum_{m=1}^{\bar{M}} \log(Q_m). \tag{17}$$

B. Texture likelihoods

Harris detector is a method that detects interest point in scale-space based on the Laplacian. A popular tracker of Harris points is Shi-Tomasi-Kanade tracker [5]. In fact, the point locations of the features in the initial frame are selected as those points that show more tractability than others. The higher singular values are, the more interesting the point feature is.

One can note that the most detected features are the corners and similar entities where its high spatial gradient gives robust information about its 2D properties.

The model used for the texture likelihood is the point-wise re-projection error, most suitable for Harris [15] points. Let a set of texture features F_T be extracted from the image at time t using Harris' detector. These features are matched to the corresponding features F_0 in the initial frame. Given a proposed motion model h^i, the probability density function of the likelihood is given as

$$p(F_T \mid h^i) = \prod_{k=1}^{N_p} \left(\frac{1}{\sqrt{2\pi}\ \sigma} \exp(-\frac{(D_k)^2}{2\sigma^2}) \right) = \prod_{k=1}^{N_p} Q_k, \tag{18}$$

Here, N_p is the number of texture points under consideration, the error D_k is the Euclidean distance $D_k = F_{kh_t} - F_{kT}$ between the kth measured point F_T and the projection of the kth point F_0 from the initial frame to the current frame(F_{h_t}). Again taking log, Q_T can be written as

$$Q_T = \log(p(F_T \mid h^i)) = \sum_{k=1}^{N_p} \log(Q_k). \tag{19}$$

C. Robust likelihood models

The likelihood functions Q_C and Q_T presented in Eq.(17) and Eq.(19) are robust to Gaussian noise but are sensitive to outliers. To handle outliers, we use Q_C^m and Q_T^n. These error functions are the mth and nth smallest values of the error vectors Q_C and Q_T respectively.

$$Q_C^m = m\text{th}_i\ \{Q_C\}_i \quad Q_T^n = n\text{th}_i\ \{Q_T\}_i \tag{20}$$

The objective functions that belongs to this class are highly robust to outliers. For example, when $m = M/2$ and $n = N_p/2$, these are the median operator and the minimization process leads to least median optimization [16], which can handle noisy measurements with upto 50% outliers.

VI. EXPERIMENTAL RESULTS AND ANALYSIS

Our tracker has been tested using different video sequences to show the performance over different qualities of texture and edges. The tracker has been aligned with edges of the object in the initial frame. Objective is to follow the object borders along the sequence. The proposed integration tracker is compared to texture-based one and edge-based one using the same video sequences. The edge tracker is based on moving edge algorithm and the texture tracker is based on Harris points. The state vector is estimated and tracked probabilistically using particle filter. The tracker works approximately on 15-18 frame per second speed using laptop system with 1.6 GH AMD processor and 256 MB RAM memory.

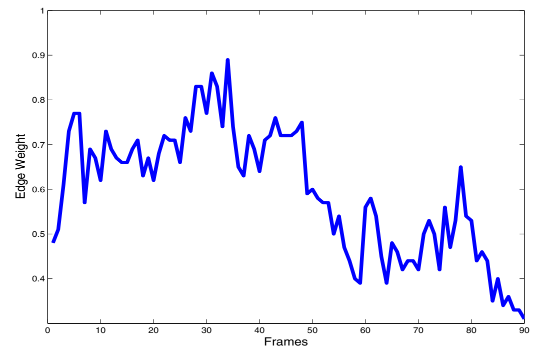


Fig. 6. Goodness weights for edges in the "Book-newspaper" video sequence. The decrease in weights explain the failure of the edge tracker.

The *"Book-newspaper"* sequence is a video sequence which includes a book with highly textured front face. The book is also surrounded by a textured background. The sequence also contains a considerable amount of motion. In addition, changes in the illumination was introduced with a good amount of shadow. Fig. 3 shows six sample images from a total 94 images. The first row (a) shows the results from the edge tracker. The second row (b) contains the texture tracker. The results of the proposed integration tracker are shown in the last row (c). It can be seen that the edge tracker has lost the object due to the large aspects changes like motion, shadow and illumination. The texture tracker gives good results. However, the integration tracker gives more perfect and precise performance. Figure 6 shows the edge goodness weight along frames.

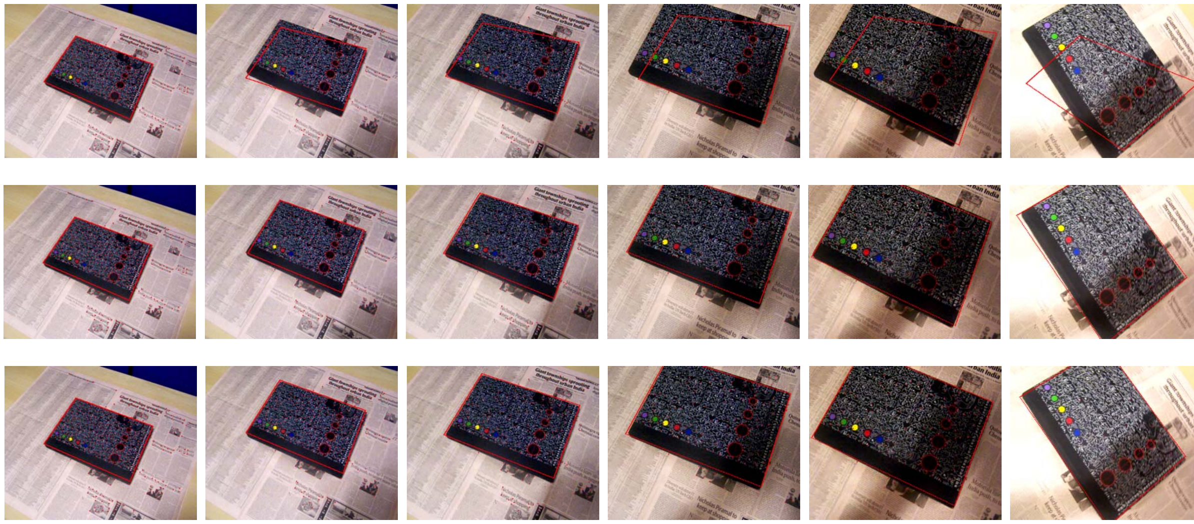
Figures 4 and 5 show the results of testing the integration based tracker using *"Not-book"* and *"Panoramic-book"* sequences respectively. In the former the tracker works properly due to the texture feature availability. The later one skewed a little due to the absence of the texture and the presence of considerable amount of edge shadow.

VII. CONCLUSIONS

In this paper, we have presented an integration framework that integrates edge and texture points for planar object visual tracking. The integration process is done between two sub-trackers using democratic integration based on goodness weights. The weights are computed with respect to each type of feature adaptively. Selection of "good" features ensures reliability of the tracker under a variety of conditions. On the other hand, robust likelihood models ensure accurate computation of motion. Application to 2D planar tracking is shown.

REFERENCES

[1] M. Pressigout and E. Marchand, "Real-time planar structure tracking for visual servoing: a contour and texture approach," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS'05*, vol. 2, Edmonton, Canada, August 2005, pp. 1701–1706.



(a)

(b)

(c)

Fig. 3. Image samples of testing the three edge-based (a), texture-based (b), and integration-based (c) trackers on the "Book-newspaper" Sequence. A tracker is aligned to the object boundaries (red color). The edge tracker failed to follow the object due to shadows and changes in illumination. Texture tracker gives better alignment in case of changes in illumination and motion. The integration-based tracker outperforms the other two.

Fig. 4. Image samples of testing the integration-based tracker on the "Notebook" Sequence. A tracker is aligned to the object boundaries (red color).

Fig. 5. Image samples of testing the integration-based tracker on the "Panoramic-book" Sequence. A tracker is aligned to the object boundaries (red color).

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

[3] M. MacCormick, "Probabilistic models and stochastic algorithms of visual tracking," Ph.D. dissertation, University of Oxford, U.K., 2000.

[4] A. Comport, E. Marchand, and F. Chaumette, "Robust model-based tracking for robot vision," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS'04*, vol. 1, Sendai, Japan, September 2004, pp. 692–697.

[5] J. Shi and C. Tomasi, "Good features to track," in *IEEE International Conference on Computer Vision and Pattern Recognition, CVPR'94*, Seattle, Washington, June 1994, pp. 593–600.

[6] E. Rosten and T. Drummond, "Fusing points and lines for high performance tracking." in *IEEE International Conference on Computer Vision*, vol. 2, October 2005, pp. 1508–1511.

[7] L. Vacchetti, V. Lepetit, and P. Fua, "Combining edge and texture information for real-time accurate 3d camera tracking," in *International Symposium on Mixed and Augmented Reality, Arlington, VA,*, November 2004.

[8] E. Marchand, P. Bouthemy, F. Chaumette, and V. Moreau, "Robust real-time visual tracking using a 2d-3d model-based approach," in *IEEE Int. Conf. on Computer Vision, ICCV'99*, vol. 1, Kerkira, Greece, September 1999, pp. 262–268.

[9] D. Kragic and H. Christensen, "Cue integration for visual servoing," *IEEE Transactions on Robotics and Automation*, vol. 17, no. 1, pp. 19–26, February 2001.

[10] P. Li and F. Chaumette, "Image cues fusion for contour tracking based on particle filter," in *Int. Workshop on articulated motion and deformable objects, AMDO'04*, ser. Lecture Notes in Computer Science, J. Perales and B. Draper, Eds., vol. 3179. Palma de Mallorca, Spain: Springer-Verlag, September 2004, pp. 99–107.

[11] Spengler and B. M., Schiele, "Towards robust multi-cue integration for visual tracking," *Machine Vision and Applications*, vol. 14, no. 1, pp. 50–58, April, 2003.

[12] M. Pressigout and E. Marchand, "Real-time 3d model-based tracking: Combining edge and texture information," in *IEEE Int. Conf. on Robotics and Automation, ICRA'06*, Orlando, Florida, May 2006, pp. 2726–2731.

[13] M. Isard and A. Blake, "Condensation - conditional density propagation for visual tracking," *International Journal of Computer Vision*, vol. 29, no. 1, pp. 5–28, August 1998.

[14] A. Doucet, N. de Freitas, and N. Gordon, *Sequential Monte Carlo in Practice*. Springer-Verlag, 2001.

[15] C. Harris and M. Stephens, "A combined corner and edge detector," in *Alvey Conference*, . 1988, pp. 189–192.

[16] Z. Zhang, "Determining the epipolar geometry and its uncertainty: A review," *International Journal of Computer Vision, IJCV*, vol. 27, no. 2, pp. 161–195, 1998.