Reliable 2D Tracking using good texture and edge features for Robotic Vision

Abdulhafez ABDULHAFEZ * **Visesh Chari** **,[1]

* *Dept. of Computer Engineering, Hasan Kalyoncu University, Gaziantep, Turkey (e-mail: abdul.hafez@hku.edu.tr).*
** *WILLOW, 23 avenue d'Italie, CS 81321, 75214 PARIS Cedex 13 France (e-mail: visesh@gmail.com)*

Abstract: We present an algorithm for highly reliable tracking of planar objects using visual cues like texture and contour in presence of feature correspondence errors. These two cues are integrated using a probabilistic formulation. The integration is based on quality goodness factors. The goodness criterion is a generalization of the well known "good features to track" concept to the both point and edge cases. The motion model of the object is computed as a homography between reference and current frames. A probabilistic formulation of the problem is proposed and implemented using particle filters. Tracking for geometric computation is useful in applications like object grasping, 3D reconstruction, augmented reality, etc. The algorithm combines contour and texture information in a novel manner to achieve robustness that outperforms the state of the art methods, which is justified by the results of experiments.

Keywords: Visual tracking, Texture tracker, Contour tracker, Probabilistic integration.

1. INTRODUCTION

Object tracking problem attracts a great interest by the robotic vision. It finds applications in tasks like manipulation, grasping, servoing, and assembly. The key factor to success in these tasks is the robust estimation of the target object motion against clutters, and changes in illumination. The different approaches to object tracking in the literature can be classified into two major categories.

The first category pertains to the conventional tracking algorithms as in Chen et al. (2017), Liwicki et al. (2016), Saqui et al. (2013), and Le and Kosecka (2017). In these works, the objective is to place and maintain a "window" on the object of interest. The position of the object in the current frame is computed based on features extracted in the frame and information about the objects' past positions. Features used as cues include optical flow, Harris points Shi and Tomasi (1994), edges as in Isard and Blake (1998), color distributions and lightness Chen et al. (2017) and Liwicki et al. (2016), etc. Past information in the form of motion models may be used to propagate the target window across frames, or to register image features. Morphological operations are used to track the object in Saqui et al. (2013). Maintaining the object in the camera field of view is the main goal in such cases. Applications of such systems are found in surveillance, human tracking for pedestrian detection, automatic vehicle driving, etc.

The second category deals with the computation of geometric information Benhimane and Malis (2004), Pressigout and Marchand (2007), Conte et al. (2013), Borum et al. (2014) and Petit et al. (2014). The objective of these works is to accurately compute the geometric transformation that finds the relation between the reference and current frames of the object. In case of planar objects, this transformation is represented as a homography matrix like Benhimane and Malis (2004), Pressigout and Marchand (2007), Kobayashi et al. (2016) and Petit et al. (2014). Such computations are useful in object grasping, 3D reconstruction, augmented reality, etc. Typically, correspondence of visual cues like texture and contour might be already established in such cases. The challenge is to accurately compute geometric transformation even in the presence of erroneous visual cues. The need for the tracker is not to lose the target in spite of changes in illumination, clutter, occlusion, large camera motion, and perspectively, where feature correspondence might not be reliable.

The focus of this paper is to present a tracker in the second category which estimates a motion model to precisely estimate the 2D object position in the image. Two very common image features, contour and texture, and a powerful probabilistic framework based on Bayesian filter are used here. Our emphasis is on the robustness and reliability of the tracking in presence of feature correspondence errors. The terms "contour" and "texture" are used here in the following context. Contour features are assumed to mean points on the boundary of an object that can be propagated to subsequent frames using edge detection operations. Texture features are assumed to be interest points like Harris points used in KLT tracker presented in Shi and Tomasi (1994) that can be extracted from local gray level distributions.

Contour features are reliable when scenes have sharp edges, high spatial gradients or contrast at object boundaries. Active contours or straight lines are generally used for 2D tracking as in (Haines and Calway (2015)) and Isard and Blake (1998), model-based 3D tracking Vacchetti et al. (2004), or dense point tracking Le and Kosecka (2017).

[1] Visesh Chari was Postdoc at Willow group during this work.

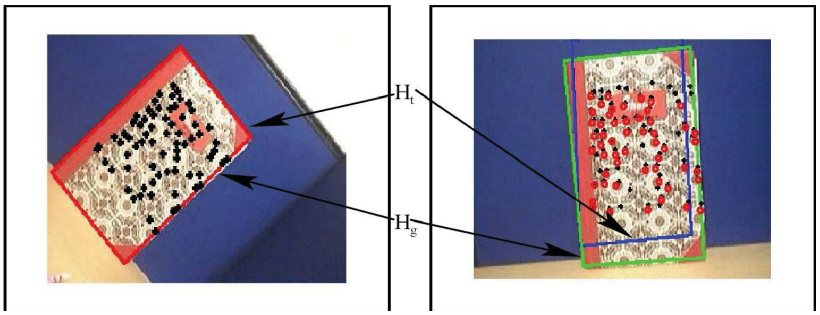

Fig. 1. Two planar images are related by homography H. Small errors in interest point estimation might result in large error in Homography estimation. Homography can be computed from contour or texture points. H_g (bottom) shows the true Homography and H_t (top) shows the homography computed using KLT (Shi and Tomasi (1994)) correspondences. In addition, we also show on the right, computed KLT correspondences(squares) and correct re-projected ones(circles). Although KLT gives reasonable correspondence, estimation of homography suffers largely even due to small errors.

Factors affecting reliability include clutter, shadow and occlusion. In such cases, texture features are reliable. But, such texture features are sensitive to large camera/object motion where contour features can perform well. Observing complementary advantages of both cues, it is matured to consider a combination of these cues Pressigout and Marchand (2007), and Petit et al. (2014). Past integration schemes include sequential processing and cost function minimization like Pressigout and Marchand (2007) and Petit et al. (2014) based methods.

The closest works to our work presented in this paper are those presented in Pressigout and Marchand (2007) and Petit et al. (2014). The framework in Pressigout and Marchand (2007) is deterministic and fuses the edges and texture points features in a single non-linear objective function which is minimized. In other words, it is non-linear optimization. The variables under minimization consist the 2D transformation (homography) between the current object position and the reference one. The objective function in this case is the error between the current position of the object and the transformed reference one, this error is minimized by regulating it to zero. The work presented in Petit et al. (2014) is an improvement to the previous works presented in Pressigout and Marchand (2007). The color cue is added to edge and point of interest features to minimize an objective function containing three of them together. Similarity between our work and Pressigout and Marchand (2007) is that we do use the same state variables, i.e. the homography components. However, our framework is probabilistic framework that utilizes Bayesian filtering techniques. This leads to tracking with higher accuracy as shown in Section 4.

The aim of this work is to address the problem of 2D homography estimation problem motivated by the fact that errors in feature correspondence can cause gross errors in homography estimation. Figure 1 shows typical errors in homography that occur due to inaccurate texture tracking. This work presents an algorithm that robustly computes homographies based on a probabilistic framework implemented using particle filters, by combining contour and texture information in a novel manner. In this framework, multiple samples of homography are drawn and their likelihood is estimated using tracked texture and contour features. A *maximum aposteriori* (MAP) approach allows us to deal with uncertainties in feature correspondence. Further, robust models are used to reject outliers in the features. Texture features are interest points computed and tracked using the KLT tracker Shi and Tomasi (1994). Contour features are tracked and proposed for the first time in MacCormick (2000). Samples can be drawn from the space of all homographies or the space of all poses, i.e. rotation,translation. It is shown that the latter is more effective. Forth, our problem is shown as the computation of incremental homography resulting from inter-frame motion. The reference frame is taken as the first frame of a video sequence.

The contribution of this paper is summarized as a hybrid tracking framework that probabilistically integrates contour and texture point features. The integration is based on goodness factors computed based on the quality of the selected measurement points. The probabilistic tracking is realized using the particle filter. The good edge and point features are extracted by generalizing Harris good points to both good points and good edges.

The remaining of this paper is organized as follows. The probabilistic hybrid tracking framework is presented and discussed in Sec 2. The process of feature detection, good point and good edges extraction, weighting factor calculation, and features likelihood functions are presented in Sec 3. Section 4 presents our experiments and the analysis of the results.

2. PLANAR OBJECT TRACKING BASED ON PROBABILISTIC 2D MOTION ESTIMATION

Given an initial image I^0 and an image I^t of a planar object at time instant t, there is a homography H_t that relates the image points belonging to the object. 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 $x_t = [u_t, v_t, 1]^T$ represent a point in the second image, the relation between these two points is written as $x_t = H_t x_0$ or

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

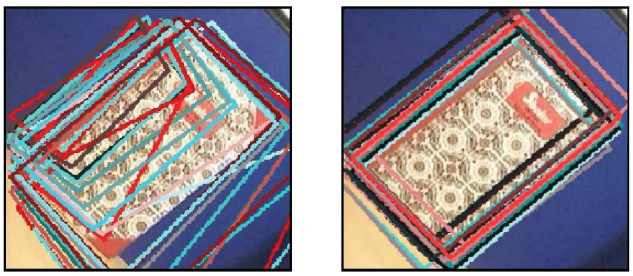

Fig. 2. Verifying claim of non-linear pose-homography relationship. Left: Homography samples drawn from the homography space from a Gaussian distribution of small covariance. Right: Homography samples resulting from pose space points drawn from a Gaussian.

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

$$h_t = \left[\, h_t^1 \;\; h_t^2 \;\; h_t^3 \;\; h_t^4 \;\; h_t^5 \;\; h_t^6 \;\; h_t^7 \;\; h_t^8 \;\; h_t^9 \,\right]^T. \tag{2}$$

By assuming 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

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

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

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

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

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

When the errors in the feature correspondences are non-Gaussian, there exists no analytical method that can minimize this error function. Particle filter algorithm or what is called Condensation algorithm Isard and Blake (1998) is preferred here because it provides an efficient probabilistic framework to take care such uncertainties.

2.1 Pose space sampling

Using particle filters for estimating posterior probabilities represents a generative model. Various homography (hypotheses) are generated as 9×1 vectors and are tested for re-projection errors. One important aspect to such a method is the sampling of the posterior distribution. The space of homographies is a nonlinear function of the pose space. Figure 2 shows homographies plotted by sampling from homography space and pose space. Due to the non-linearity, a small change in homography parameters represents a large change in pose, which is undesirable since motion between frames is small.

Hence the solution is to sample in pose space. Decomposing any set of 2 homographies induced by the same plane in 2 views, to obtain unique values of pose and the plane normal $n^\top$ is well known Hartley and Zisserman (2003).

$$H = K(R + t\frac{n^\top}{d})K^{-1} \tag{6}$$

This allows us to sample in the pose space, when we have the additional knowledge of camera internal parameters. Further, since many systems for geometric computations assume the knowledge of internal parameters, sampling from pose space seems reasonable.

2.2 Particle Filter for feature integration

Bayesian filter formulation for computing homography h_t is presented now.

Bayesian Tracking: 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,

$$p(h_t \mid F_{1,\ldots,t}) = \frac{p(F_t \mid h_t)p(h_t \mid F_{1,\ldots,t-1})}{p(F_T \mid F_{1,\ldots,t-1})}. \tag{7}$$

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 formula as follows

$$p(h_t \mid F_{1,\ldots,t}) = \alpha\; p(F_t \mid h_t) \int p(h_t \mid h_{t-1}) p(h_{t-1} \mid F_{1,\ldots,t-1}) dh_{t-1}. \tag{8}$$

Equation (5) can be modeled as the *maximum aposteriori* (MAP) of (8). Thus, equation (4) becomes the likelihood $p(F_t \mid h_t)$ and equation (3) represents the motion model $p(h_t \mid h_{t-1})$. Now, $p(h_{t-1} \mid F_{1,\ldots,t-1})$ represents the previous iteration.

Once features are detected in the current image, likelihoods for different homography hypotheses need to be calculated in order to select a hypotheses that represents the current homography. The likelihoods of visual features, $p(F_C \mid h)$ and $p(F_T \mid h)$, are assumed to be independent. Our approach combines the log of these likelihoods to give a robust estimate of the validity of a homography in presence of feature correspondence errors.

Particle Filters: 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 N_p particle-weight pairs $\{h_{t-1}^i, w_{t-1}^i\}_{i=1}^{N_p}$, chosen to approximate density $p(h_{t-1} \mid F_{1,\ldots,t-1})$, are propagated to pairs $\{h_t^i, w_t^i\}_{i=1}^{N_p}$ using the motion model $p(h_t \mid h_{t-1})$. A detailed explanation of particle filters and their use to represent probability functions can be found in Doucet et al. (2001). The weights $\{w_t^i\}_{i=1}^{N_p}$ associated to the particles h_t^i are computed proportional to the likelihood function in case of using the bootstrap filter as

$$w_t^i = \alpha\; p(F_t \mid h_t^i). \tag{9}$$

In the next section, we present the methods to compute likelihood for the two visual cues texture $p(F_T \mid h)$ and contour $p(F_C \mid h)$. The subscript T of F_T will be used to denote texture rather than time henceforth.

Algorithm 1 Reliable visual tracking using texture and contour features

Input: Sequence of images $I\{1,\ldots,m\}$, initial features $F_1 = \{f_i^1 : i \in \{1,\ldots,n\}\}$.
Output: Homographies $H = \{h_1,\ldots,h_m\}$ that satisfy the observations in presence of noise.
ParamVector = $[h_t^1,\ldots,h_t^9]$(Eq 2) initialized to identity.
NumParticles = N {Set the number of particles needed to sample the space effectively}
for $i \in \{2,\ldots,m\}$ **do**
  F_i = ExtractFeatures(ParamVector, F_{i-1})
  Particles = DrawSamples(ParamVector, NumParticles)
  for $j \in \{1,\ldots,\text{NumParticles}\}$ **do**
    Q_j = log(TLikelihood($F_i, Particles_j$)) + log(CLikelihood($F_i, Particles_j$)) {Taking particle minimizing log likelihood of the errors corresponding to texture and contour features.}
  end for
  ParamVector = $Particle(\min_j Q_j)$
end for

Integrating features: Combining $p(F_C \mid h_i)$ and $p(F_T \mid h_i)$ in one framework we get

$$p(F \mid h^i) = p(F_C \mid h^i)p(F_T \mid h^i). \tag{10}$$

However, we consider the logarithm of the above likelihood function, Equations (23),(25), such that the multiplication becomes summation

$$\begin{aligned} Q &= \log[p(F \mid h^i)] = \log[p(F_C \mid h^i)] + \log[p(F_T \mid h^i)] \\ &= Q_C + Q_T. \end{aligned} \tag{11}$$

It can be noticed that the contribution of each cue will be nullified in case of the failure or the absence of this cue, due to the way we define our likelihoods. Also, if one of the cues is erroneous, the other cue can compensate for this error. This gives robustness to our approach. Using likelihoods defined above, we can now give an outline of our algorithm.

3. FEATURE DETECTION, GOODNESS AND LIKELIHOODS

3.1 Feature Detection

Contour features of the object are represented at the time instant $(t-1)$ by C_{t-1} in the previous frame. The vector h_{t-1} represents the homography matrix that maps the current contour in the previous frame C_{t-1} to the contour in the initial frame C_0.

The contour estimation process using normal measurement lines is depicted in Figure 3. The contour is mainly representing the object edge in the previous frame. A set of normal measurement lines are considered with respect to the edge in the previous frame. Three normal line examples are shown in Figure 3. Two edge proposals are shown and only the nearest edge point measurement to the edge proposal is considered. If there is no features detected on the normal line like the line L_3, this line will be discarded and not considered in the likelihood function. It is worth to note here that using short normal measurement will reduce the effect cluttered scenes.

Textures features are considered here as Harris points. Since it is based on a Laplacian formula of a similarity criterion, it is also known as a differential tracker. Harris points are tracked by the very well common tracker by Shi-Tomasi-Kanade tracker Shi and Tomasi (1994). The concept of Harris point is used to initiate the tracker with highly tractable points. This quality of the point features are measured using the singular values of the matrix C_p. The feature points are selected with the highest singular values. However, such points are the corners and similar entities. In contract, points located in a uniform area will not be detected.

3.2 The Weighted Good Features

To estimate the goodness of the features we build on the method developed by Shi and Tomasi to measure the dissimilarity of point features Shi and Tomasi (1994). The goodness in other words means, it is a good feature to track . They established a (2×2) measurement matrix around the point feature. The two eigenvalues of this matrix are large when the feature point is good to detect and track. If there is only one large eigenvalue and another respectively small one then it is a point that belongs to an edge. We propose to generalize the goodness concept about points to the case of edge features located on a measurements line. In the sequel, we use the goodness of an image point to define probabilistic weights to both contour and texture points.

Good Features to Track: Similar to Shi and Tomasi (1994), the affine image motion between successive frames is computed as one that minimizes the intensity *dissimilarity*

$$\epsilon = \int\int_W [J(A\mathbf{x} + \mathbf{d}) - I(\mathrm{x})]^2 w(\mathbf{x}) d\mathbf{x} \tag{12}$$

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 feature to be tracked, it is required that the matrix Z has large eigenvalues.

Good Texture point Features: Good point features are considered if the two eigenvalues are large enough. In other words, it must ensure that the minimum of them is larger than a predefined value λ_{max}. This means that the intensity around the point changes in both x and y-directions. This is expressed by the equation

$$\min(\lambda_1, \lambda_2) > \lambda_{max} \tag{13}$$

where (λ_1, λ_2) are the eigenvalues and λ_{max} is a predefined constant value.

Good Edge Features: The goodness of edge features can be defined similarly, where one eigenvalue is significantly

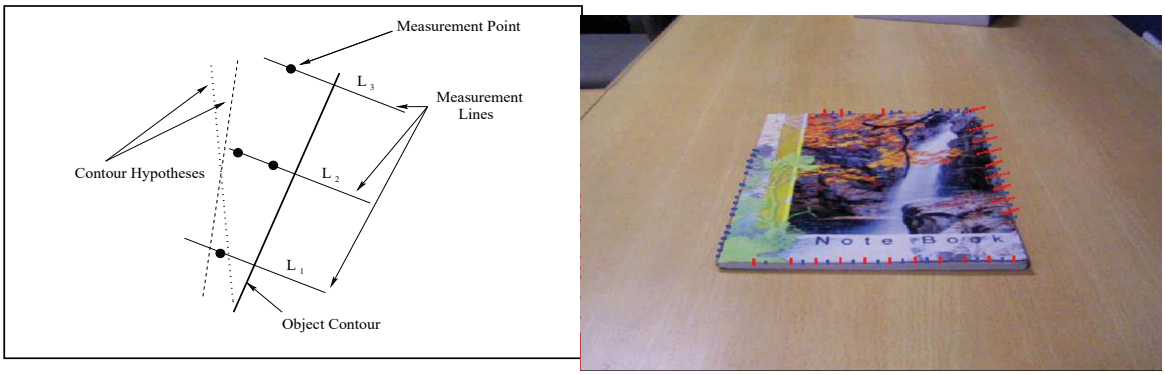

Fig. 3. The normal measurement lines are used to extract contour features in the current frame is depicted in the left image. Example on the extracted contour is shown in the right image.

(a) (b)

Fig. 4. Features detected (red dots) using "goodness" criteria for texture and edge in (a) and (b) respectively. The results in (a) are from Shi and Tomasi (1994) paper. The adaptation to edge features is shown in (b).

smaller than the other value. Thus, we suggest here a good contour feature. This is expressed by the equation

$$\max(\lambda_1, \lambda_2) > \lambda_{max}. \tag{14}$$

and

$$\min(\lambda_1, \lambda_2) < \lambda_{min}. \tag{15}$$

Figure 4 represents the features extracted by examining these two conditions for an example image.

Assigning weights to features: Let us define the goodness of texture features and edge features as the number of points that are tracked along the sequence of considered frames. One may note here two types of visual features are considered here, they are contour feature F_C and texture feature F_T. Texture feature is essentially considered as an image Harris point; while contour feature is considered as an image point that represent a gradient peak along a measurement line. We show the development of the goodness function for texture point features in the sequel, while the one for contour features is dropped since it is analogue to the texture's one.

Let us assume that we select N_0 texture point feature in the refernce frame. Only N_t features in the current frame have been selected as good features and have its corresponding points in the initial frame. Let the set $\mathcal{N}_t = \{n_i \mid i = 1, \cdots, N_t\}$ be the set of 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 (12) as

$$W_T = \; p(\mathcal{N}_t \in \mathcal{N}_0) = p(\mathcal{N}_t \in \mathcal{N}_{t-1})\; p(\mathcal{N}_{t-1} \in \mathcal{N}_0) \tag{16}$$

$$p(\mathcal{N}_t \in \mathcal{N}_{t-1}) = \frac{1}{N_{t-1}} \sum_{i=1}^{N_{t-1}} \frac{1}{2\pi\sigma^2} \exp\left[-\frac{\epsilon_i^T \epsilon_i}{2\pi\sigma^2}\right] \tag{17}$$

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

$$W_T = \frac{1}{N_0} \sum_{i=1}^{N_{t-1}} \frac{1}{2\pi\sigma^2} \exp\left[-\frac{\epsilon_i^T \epsilon_i}{2\pi\sigma^2}\right]. \tag{18}$$

For edges, we 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

$$W_C = \frac{1}{N_0} \sum_{i=1}^{N_{t-1}} \frac{1}{2\pi\sigma^2} \exp\left[-\frac{\epsilon_i^T \epsilon_i}{2\pi\sigma^2}\right] \tag{19}$$

To reduce the complexity of the computation, we replace the Gaussian function in (18) and (19) by a rectangle

function with maximum value of one. Equations (18) and (19) can now be rewritten as

$$W_T = \frac{N_{T_t}}{N_{T_0}}, \qquad W_C = \frac{N_{C_t}}{N_{C_0}}. \tag{20}$$

Here, N_{T_t} and N_{T_0} are the number of good matched point feature in the current frame and reference frame respectively. Similarly, N_{C_t} and N_{C_0} are the number of contour measurement lines matched with a good features in the current frame and reference frame respectively. This allows us to get a quantitative evaluation of the feature's reliability. Indeed the higher the weight, the more reliable a feature is. By comparing the weights, we may decide upon the most reliable feature. One more thing top notice is that these two weights are normalized to one.

3.3 Defining feature likelihoods

Contour likelihoods: We start from the generic model of the contour likelihood which was presented in MacCormick (2000) to develop our contour likelihood model. Let the vector h_t^i be a hypothesis of the state of the contour C_t that intersects the normal measurement line l_m at a distance d_m from the same contour point (Figure 3). The p.d.f. of the generic likelihood model is given as

$$p(F_C \mid h^i) = \prod_{m=1}^{M} b(n_m - 1) \sum_{k=1}^{n_m} \frac{\exp(-\frac{(v_k - d_m)^2}{2\sigma^2})}{n_m L^{n_m - 1}}. \tag{21}$$

Here, $b(n)$ is probability of obtaining n background features along the measurement line with length L. After some development and enforcing certain assumptions MacCormick (2000), we write

$$p(F_C \mid h^i) = \prod_{m=1}^{\bar{M}} \left(\frac{1}{\sqrt{2\pi}\ \sigma} \sum_{k=1}^{n_m} \exp(-\frac{(D_m)^2}{2\sigma^2}) \right) = \prod_{m=1}^{\bar{M}} Q_m, \tag{22}$$

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 probability $b(n_m)$ is assumed as Poisson with density $\lambda = 1$, and the probability of not detecting an edge feature along the mth line is zero. The number of measurement lines that intersect the contour C_t is $\bar{M}$. Taking the logarithm to simplify multiplications and introducing the goodness weight W_C, we can write

$$Q_C = W_C \log(p(F_C \mid h^i)) = W_C \sum_{m=1}^{\bar{M}} \log(Q_m). \tag{23}$$

Texture likelihoods: We start from the point-wise reprojection error model to define the texture likelihood function, which is the most suitable for Harris points detector Harris and Stephens (1988). Let us have a set of texture features (Harris points) F_T that are extracted from the image at the time moment t. These features are matched and mapped to the corresponding features F_0 in the initial frame. Using the hypothesised motion model h^i, the p.d.f. of the likelihood is given as

$$p(F_T \mid h^i) = \prod_{k=1}^{N_p} \left(\frac{1}{\sqrt{2\pi}\ \sigma} \exp(-\frac{(D_k)^2}{2\sigma^2}) \right) = \prod_{k=1}^{N_p} Q_k, \tag{24}$$

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 the logarithm to simplify multiplications and introducing the goodness weight W_T, we can write, Q_T can be written as

$$Q_T = W_T \log(p(F_T \mid h^i)) = W_T \sum_{k=1}^{N_p} \log(Q_k). \tag{25}$$

4. EXPERIMENTAL ANALYSIS AND RESULTS

In this section, we present results of tracking a planar object in different conditions and situations. We present a quantitative evaluation of our results along with qualitative comparison with a recently reported tracker. The proposed method shows a good amount of robustness toward changes in illumination, large motion, particularly rotation and zooming, textured background. In fact, these conditions are the cases where texture and edge cues individually either fail completely or are inaccurate. This is in addition to motion blur occuring due to large camera motion. Outdoor environment exhibites highly textured background, strong changes in illumination, and a lot of clutter. To show robustness to these factors, we also conducted experiments on a video taken from the outdoor environment.

The videos we use in the experiments were captured either using a Kodak DX7950 digital camera for indoor scenes or a Logitech web cam for outdoor scenes. Videos were captured at 15 fps. Texture points are tracked using the KLT tracker Shi and Tomasi (1994). Typically, 400 points are selected and tracked throughout the video sequences, without replacement of lost features. In implementations, particles were drawn from both uniform and Gaussian distributions centered around the previous frame's results. We found uniform distributions to work better. The tracker works approximately on 12 frame per second speed using a laptop system with 1.5 GH AMD processor and 256 MB RAM memory.

In order to highlight the effects of intensity changes, we allow the camera to automatically adjust to the ambient light, and switch off/on lights during the video. In the process of tracking, we do not replace the lost features since the homography is computed with respect to the reference (first) frame of the video sequence. The particle that maximizes the a posteriori (MAP) function is selected as the current motion model. The tracking process is initialised by a manually selecting a rectangular marker that defines the position of the planar object in the first frame. This marker will be propagated to every frame using the estimated motion model.

Two video sequences are considered to show the robustness and efficiency of the proposed tracker in different bad conditions of both object's edge and texture. In the following subsections, we analyse the situations of the contour and texture presented in our selected videos and the advantages of combining them in one tracker. After that, the results of each of edge and texture individual trackers in addition to our integration proposed tracker are presented. The comparison with the state of the art work is also presented using the outdoor videos. Before that, we

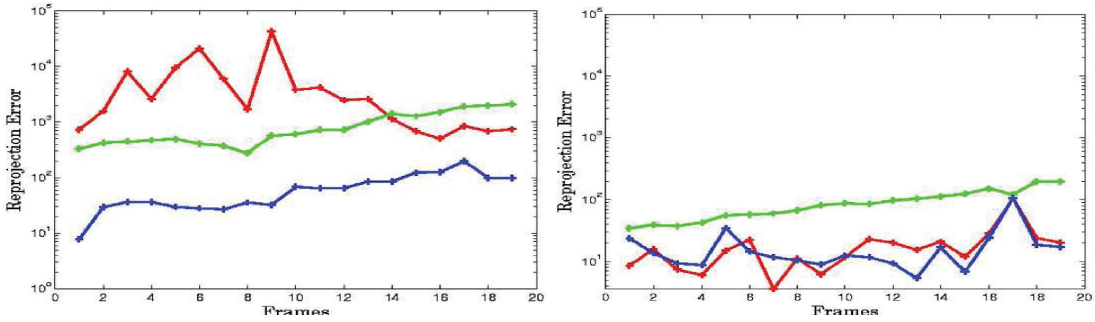

Fig. 5. Left: Homography based sampling. Right: Pose based sampling. Clearly, pose based sampling produces results with less jitter, more smoothness and lesser error that corresponding sampling in the homography space. Frames are sampled at regular intervals of 5 frames from 100 frame videos. (Light,Green) Contour tracker. (Dark,Red) Texture tracker. (Blue,Darkest) Integrated tracker.

show comparison between the two sampling cases, from the homography space and from the pose space.

4.1 Sampling from Pose space or from Homography space

In this experiment we compare the re-projection error resulted from the case of using samples from the homography space H and the one resulted from using samples from the pose space. Of course, this situation assumes that the camera is internally calibrated. In other word the matrix K of the intrinsic camera parameters are available in advance.

We sample from the space $\{\rho, \theta, \gamma, t_x, t_y, t_z\}$ representing rotation in roll, pitch and yaw and translation in the x, y and z directions. As expected, results show significant improvement in such a case, as depicted in Figure 5.

4.2 Experiments on the Texture, Edge, and hybrid trackers in the Video Sequences

Experiment 1: The video contains a textured object (book) behind a plain background. The camera, initially positioned about 2 meters on top of the book, moves sideways and zooms simultaneously. Sideways movement is accompanied by large rotation, of about 2 degrees per frame. In between the video, the lights of the room are switched off to simulate illumination change. This results in the KLT tracker loosing around 200 points within a span of 5 frames. Finally, the camera ends up close to the book performing as large as 90 degrees of rotation during the motion. As shown in the Figure 6(a), the texture tracker loses the target as soon as the zoom and rotational motion of the camera commence. On the other hand, due to small measurement lines, the edge tracker is able to sustain its position on the object. Finally, the texture tracker, combining information from both sources gives as low an error as 8-10 pixels per interest point, which is acceptable considering the high motion, zooming and illumination change. Inter-frame motion of the camera is as large as 40-50 pixels owing to the small fps rate of the camera. KLT correspondences typically are off by 4-5 pixels in random directions. Samples from the video during the tracking processes using the three different trackers, texture, contour, and hybrid trackers are presented in Figure 7.

Experiment 2: To analyze the effect of cluttered backgrounds, another experiment was conducted. Here, the same book is placed behind a newspaper background, which will result in spurious edges. This time, the camera starts from one side of the book and converges onto its center, leading to mild occlusion and high zooming. The operator of the camera comes between the camera and the light source to induce shadow effects. As high as 50 pixel inter-frame motion is observed. KLT features are computed not only on the book, but are also even spread around the newspaper to simulate erroneous interest points. Results are shown in Figure 6(b). As expected, the edge tracker succumbs to the spurious edges present all around the object contour. Although the texture tracker is much more stable, it is initially affected by the large rotation which again amounts to a full 90 degree turn. On the other hand, by combining edge and texture information, the integrated tracker is able to reliably track even in presence of large clutter, shadow and large motion. The number of KLT correspondences reduces from 400 to 16 over the course of the video. Samples from the video during the tracking processes using the three different trackers, texture, contour, and hybrid trackers are presented in Figure 8.

Experiment 3: The third video is of a poster on the wall. The specialty of this video is the high occlusion. Almost 40-50% of the poster is occluded in the first 10-15 frames and continues to be so throughout the video. As a result, the individual trackers do not have enough information (interest points and edges) to stabilize. This results in a high error rate for both. The integrated trackers takes the advantage of the complementary information provided by texture and contour points. This results in highly stable tracking, see figure 6(c).

Analysis of texture and edge cues: Texture points detector fails when there is an intensity change or when the camera performs large motion or zooming. In case of intensity change, texture points are lost. We typically start with 200-400 feature points and end up with the likes of 10-15 feature points in the end. When the camera performs large motion (translation or rotation), then either the texture points are lost or they are tracked but not with sub-pixel accuracy. Zooming presents the same problem. When texture points are not accurate, motion model that minimizes the reprojection error are very different from the

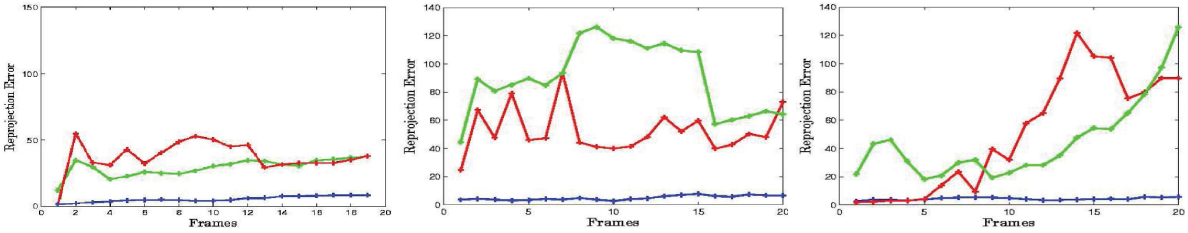

Fig. 6. Plots of the performance of our tracker in presence of large motion, high clutter, and occlusion. Although contour and texture trackers fail miserably, combining both information provides high stability to the integrated tracker. Frames are sampled at regular intervals of 5 frames from 100 frame videos. (Light,Green) Contour tracker. (Dark,Red) Texture tracker. (Blue,Darkest) Integrated tracker.

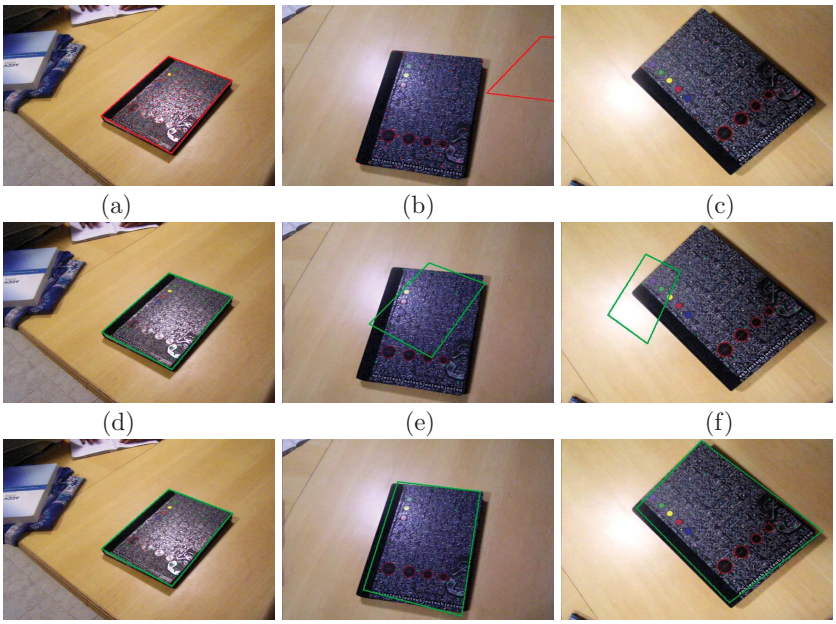

Fig. 7. *Experiment 1.* Texture does not work well with intensity change (b) or with large motion (c). Edge tracking fails due the motion blur induced by intensity change (e), and large motion (f). As mentioned earlier, integration is able to track stably intensity changes (h), or large motion (i)

actual one. This, accompanied by the recursive estimation problem, leads to a very inaccurate tracking as shown in the figures.

There are many factors that affect the accuracy of the edges detection process. First one is when the considered object is sufficiently textured, and so the edge detector finds many points in the object interior. Second factor is the motion blur artefact induced by large camera motion. These decrease the information extracted from the edge. In addition, the length of the measurement line has to be quite large to compensate for the large camera motion, this introduces a lot of points which do not belong to the edge of the object, but either to its interior or background. Finally, shadows that appear in the video and the lighting variation also reduce the amount of edge information. All of these lead to inaccurate tracking.

The framework of integration, unlike the individual methods presented above, takes advantage of both edge and texture information. By minimizing the sum of the log-likelihoods of the inidividual errors, we ensure that robustness is achieved. For example, texture cues on one side of the book and edge cues on the other side in the videos that show in the next subsection may be accurate. By combining them, we ensure that there are cues on all parts of the book that helps the sampling algorithm pick the correct particle. Thus, in the results, we can see that the tracker clearly is stable and placed over the object, and is not drifting away as should be expected.

4.3 Comparison

Qualitative results are presented on an outdoor scene captured with a web cam. As can be seen in the frames, this scene has considerable clutter (Figure 9). The use of a web cam for taking this video adds to the problem by introducing abrupt motion of the camera in the video. As mentioned earlier that the closest work to ours is the one presented in Pressigout and Marchand (2007) and in Petit et al. (2014). Since the work in Petit et al.

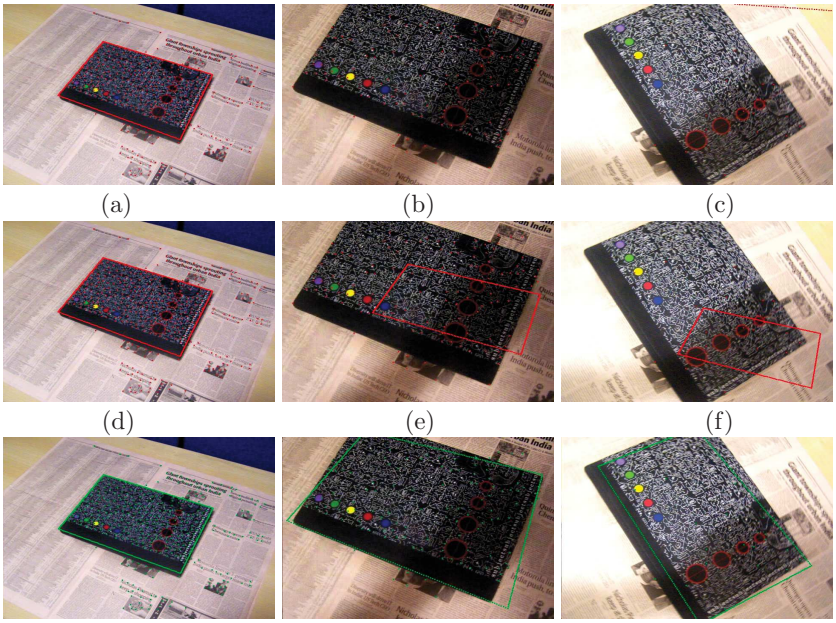

Fig. 8. Samples from *Experiment 2*. Texture tracker is in (a, b, c), Contour tracker is in (d, e, f). Edge detection faces tough problems due to a lot of surrounding texture. Hybrid tracker is in (g, h, i). As is shown in (b) and (c), the tracker seems to handle even occlusion very well

(2014) is an improvement to previous works Pressigout and Marchand (2007), where the color cue is added to edge and point of interest features. We do not use color cues in our formulation consequently we compare with work in Pressigout and Marchand (2007), in order to be more fair.

We present the initial and final frames of a 120 frame video, with results from our approach and the one presented in Pressigout and Marchand (2007). In order to introduce more challenge, we compute as much as 50% KLT interest points on the background of the video as well. Since Pressigout and Marchand (2007) minimizes a cost function based on re-projection error, the tracker settles on the background (Figure 9). Our tracker, on the other hand, remains on the object due to the high robustness we provide to texture and contour feature usage.

5. CONCLUSIONS

In this paper, we have presented a new tracking formulation suited for robotic vision applications. The tracker offers robustness to various kinds of situations like cluttered environments, shadow and illumination changes and occlusion. We show that by employing particle filters, the tracker can also be made robust to non-Gaussian errors induced in the feature extraction process. We also show that sampling in the pose space is an efficient method to produce accurate results.

ACKNOWLEDGEMENTS

The authors acknowledge efforts by Mr. Baraa Sabe in proofreading this article.

REFERENCES

Benhimane, S. and Malis, E. (2004). Real-time image-based tracking of planes using efficient second-order minimization. *IEEE/RSJ Intelligent Robots and Systems*, 943–948.

Borum, A., Matthews, D., and Bretl, T. (2014). State estimation and tracking of deforming planar elastic rods. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, 4127–4132. IEEE.

Chen, L., Zhou, F., Shen, Y., Tian, X., Ling, H., and Chen, Y. (2017). Illumination insensitive efficient second-order minimization for planar object tracking. ICRA.

Conte, F., Cusimano, V., and Germani, A. (2013). Robust planar tracking via a virtual measurement approach. *European Journal of Control*, 19(2), 146–156.

Doucet, A., de Freitas, N., and Gordon, N. (2001). *Sequential Monte Carlo in Practice*. Springer-Verlag.

Haines, O. and Calway, A. (2015). Recognising planes in a single image. *IEEE transactions on pattern analysis and machine intelligence*, 37(9), 1849–1861.

Harris, C. and Stephens, M. (1988). A combined corner and edge detector. In *Alvey Conference*, 189–192.

Hartley, R. and Zisserman, A. (2003). *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition.

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

Kobayashi, T., Kato, H., and Sugano, M. (2016). Planar markerless augmented reality using online orientation estimation. In *Asian Conference on Computer Vision*, 424–439. Springer.

Le, P.H. and Kosecka, J. (2017). Dense piecewise planar rgb-d slam for indoor environments. *arXiv preprint arXiv:1708.00514*.

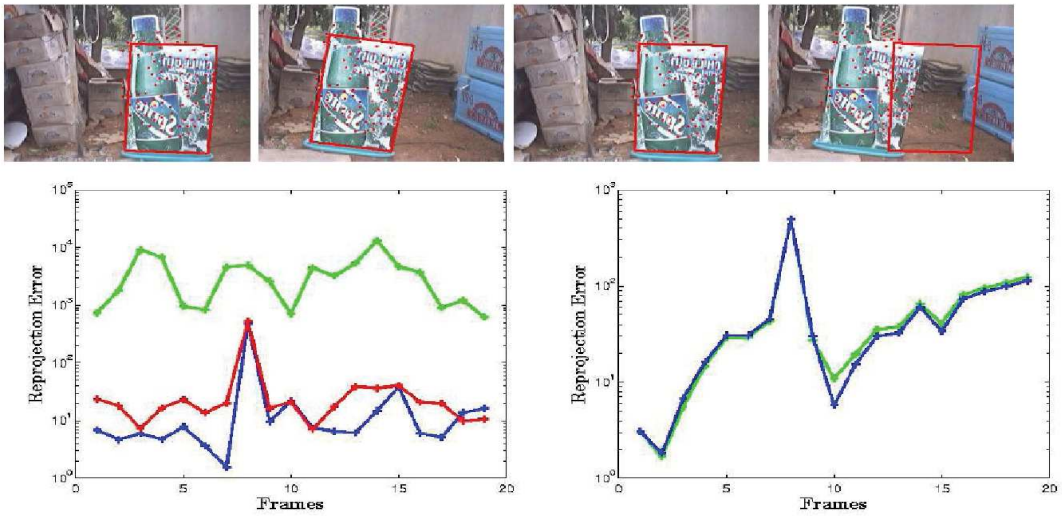

Fig. 9. Cluttered scene, with many spurious KLT interest points. Frames are sampled at regular intervals of 5 frames from 100 frame videos. (Light,Green) Contour tracker. (Dark,Red) Texture tracker. (Blue,Darkest) Integrated tracker. Left: Our result. Right: Texture/Edge tracker of Pressigout and Marchand (2005). The hybrid tracker does not converge in their case.

Liwicki, S., Zach, C., Miksik, O., and Torr, P.H. (2016). Coarse-to-fine planar regularization for dense monocular depth estimation. In *European Conference on Computer Vision*, 458–474. Springer.

MacCormick, M. (2000). *Probabilistic Models and Stochastic Algorithms of Visual Tracking*. Ph.D. thesis, University of Oxford, U.K.

Petit, A., Marchand, E., and Kanani, A. (2014). Combining complementary edge, point and color cues in model-based tracking for highly dynamic scenes. In *IEEE Int. Conf. on Robotics and Automation, ICRA'14*, 4115–4120. Hong Kong, China.

Pressigout, M. and Marchand, E. (2005). 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*, volume 2, 1701–1706. Edmonton, Canada.

Pressigout, M. and Marchand, E. (2007). Real-time hybrid tracking using edge and texture information. *Int. Journal of Robotics Research, IJRR*, 26(7), 689–713.

Saqui, D., Sato, F.C., Kato, E.R.R., Pedrino, E.C., and Tsunaki, R.H. (2013). Mathematical morphology applied in object tracking on position-based visual servoing. In *IEEE International Conference on Systems, Man, and Cybernetics, Manchester, SMC 2013, United Kingdom, October 13-16, 2013*, 4030–4035.

Shi, J. and Tomasi, C. (1994). Good features to track. In *IEEE International Conference on Computer Vision and Pattern Recognition, CVPR'94*, 593–600. Seattle, Washington.

Vacchetti, L., Lepetit, V., and Fua, P. (2004). Combining edge and texture information for real-time accurate 3d camera tracking. In *International Symposium on Mixed and Augmented Reality, Arlington, VA,*.