ROBUST CONTOUR LINE EXTRACTION USING CONTEXT

Feng Guo, Gang Qian

Department of Electrical Engineering and
Arts, Media and Engineering Program
Arizona State University,
Tempe, AZ, 85287, USA

ABSTRACT

In this paper, a robust approach to contour line extraction in cluttered images using context is presented. Object contours are often used as key features in model-based 2D and 3D tracking systems. In many applications, object contours can be approximated using line segments, e.g. in human limb tracking. In cluttered images, undesired edges other than the true contour lines often present severe disturbance for reliable object tracking. In our approach, we reduce the effect of unwanted edges by exploring context information, including edge orientation and previous contour line position and orientation. Experimental results have shown that by using context, the number of false contour lines can be reduced significantly, which will greatly improve the tracking performance.

1. INTRODUCTION

Object contours are important image features for tracking. It has attracted tremendous attention in the computer vision literature (see [1]). The tracking framework deployed in many tracking systems which use contours can be roughly summarized into two steps: first a prediction of contour is obtained from the previous estimate and the template (either 2D or 3D) using the knowledge of dynamic models; then according to the predicted contour, local optimization techniques are used to fit the prediction to the measurement in the feature map which is obtained from feature detection filters such as edge/ridge/valley filters. Usually, during the optimization procedure, the displacement between the predicted positions of contour and the features in the feature map along norm direction of the predicted contour are used to evaluate the distance between the predicted contour and the one found in the feature map. The above contour tracking framework can work reasonably well with a relatively clear background, but suffers from image clutters. Although techniques such as background subtraction can be used to reduce the clutters in the background, foreground clutters sometimes are not simply avoidable, e.g. textures on clothes during human body tracking. In this paper, we argue that an intermediate-processing step is needed between the above two steps to further reduce the noise caused by image clutters in contour-based tracking.

In this intermediate-processing step, good candidates of the target contours are extracted from the feature map, such as edge maps, by context information. Instead of using feature maps directly, preselected contour candidates from the intermediate step can be used to guide the contour-based tracking as intermediate measurements from images. To summarize, in this intermediate step, we would like to solve the following problem: given a set of predicted contours from either the projections of 3D models or deformed 2D templates, how to extract a set of candidate contours from the feature (e.g. edge) maps using prior knowlege and previous tracking results.

In this paper, we will focus on the robust extraction of lines candidates using context information such as line orientation and the knowledge about kinds of interesting lines, i.e. information contained in the predictions. There are mainly two reasons that the robust extraction of lines is studied here. First, lines are one of the simplest 2D point sets. Moreover, cones and cylinders are widely used in the modeling of human fingers, limbs and torso. Their image projections are line segments. One of the applications of the robust line candidates extraction is 3D tracking of human arm, where the arm is modeled by two cones. Figure 1 shows the modeling of upper arm. Once good candidates of the desired lines can be extracted from the input image, they can be used to confine the search space of the 3D arm movement. This will naturally lead to improvement both in accuracy and processing speed.

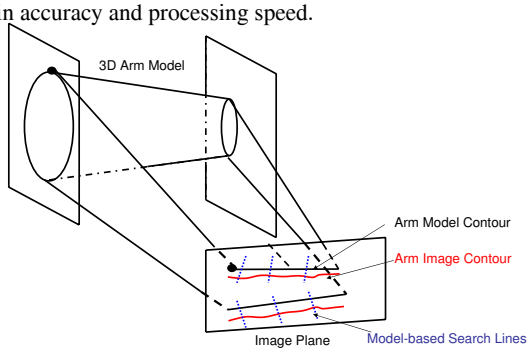

Fig. 1. 3D model-based arm tracking

2. PREPROCESSING OF INPUT FRAMES

In this paper, we assume that the tracking is performed using a static camera with smooth object movement between adjacent image frames. To use this prior knowlege, a preprocessing step is applied to the input frame to locate the region of interest (ROI) and to remove image clutters in the background through ROI extraction and background subtraction.

Assume that the estimation of object contour X in previous frame is correct. Based on the smooth object movement assumption, the object contour in current frame usually is very close to its location in the previous frame. To find the ROI within the current frame, the bounding box of X is firstly located, and then the ROI is obtained

This material is based upon work supported by the National Science Foundation under Grant CISE-RI No. 0403428

by extending length and width of the bounding box by half, while keeping the center unchanged.

Within the ROI, background subtraction is used to extract the foreground object and remove some of the background clutters. Similar to the methods in [2], we used 30 color images as training data to obtain a background image for the fixed camera. The mean $\mu(x, y)$ and standard derivation $\delta(x, y)$ of each HSV color channel are calculated at all the pixel locations. The color distribution of each pixel is modeled as a Gaussian. The intensity change of a pixel at location (x, y) in one of the channel in a frame is calculated using the correpsonding mean and variance by

$$c(x, y) = \frac{|p(x, y) - \mu(x, y)|}{\delta(x, y)} \quad (1)$$

where $p(x, y)$ is the pixel value in the corresponding channel.

Foreground image $f(x, y)$ is then obtained by thresholding the background-subtracted image. The Canny edge detector is then adopted to obtain the foreground object edges. At the same time, the directions of the edge points are also obtained. Figure 2 shows the frame preprocessing results. Some extra lines were added manually inside the arm area to create more clutters.

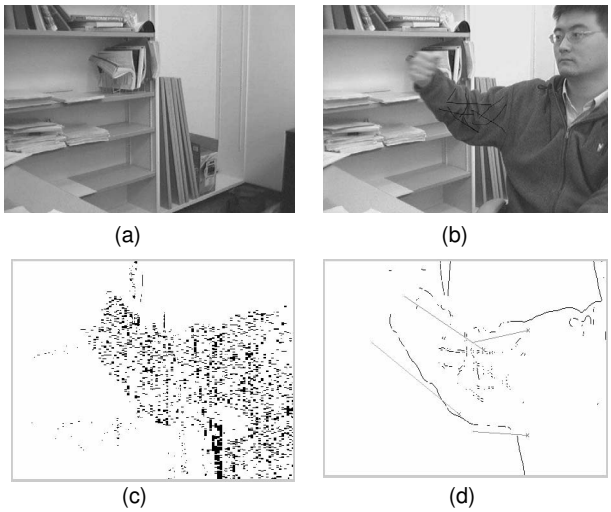

Fig. 2. Image preprocessing results. (a): the background image; (b): an input frame; (c): foreground object within ROI after background subtraction; (d): Canny edge detection results. The dotted straight lines are previous estimated contour lines

3. LINE EXTRACTION

3.1. Line Detected using Hough Transform with Edge Orientation Constraint

Hough transform(HT) is one of most successful methods used for line finding [3],[4]. HT usually uses the normal representation of a line $\rho = x \cos\theta + y \sin\theta$ to map an edge point to a sinusoid curve in the quantized parameter space. The bins along the sinusoid are increased. When the accumulation process is completed, the parameter space is searched for peaks. These peaks indicate the presence of straight line segments in the image, with the peak position as estimates of line parameters.

In the standard Hough transform, the whole range of $[0, 180]$ is taken as possible values for the line slope angle θ. In our method, we take into account of the edge direction information obtained during the edge detection step. Given an edge point $e(x, y)$ with direction ϕ_e, during the Hough transform, it will only contribute to part of the sinusoidal curve for $\theta \in [\theta_a, \theta_b]$, where $\theta_a = max(\phi_e - \eta, 0)$, $\theta_b = min(\phi_e + \eta, 180)$. η controls the range of the angle and was prechosen. Line segments candidates can then be obtained using peak detection. By using the HT with edge orientation constraint, false edge lines that don't have enough points with consistent edge orientations will be removed. Hence the computational resources will be saved, which is extremely important in real time application. Figure 3 shows line detection results using the standard HT and the HT with edge orientation constraint. The thresholds of peak detection for both methods are αV_{max}, where $\alpha = 0.2$ and V_{max} is the maximum value of edge point accumulator. Here we use $\rho = 1$ pixel and $\theta = 1$ degree as the units of the quantities in the transform domain. Using edge point orientation constraint, a much smaller number of candidate lines will be generated.

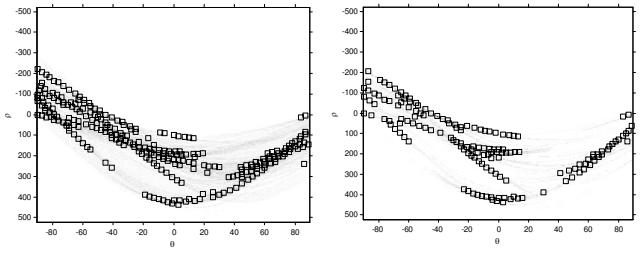

Fig. 3. Line detection using Hough transform. (a): standard Hough Transform; (b): Hough Transform with edge orientation constraints

3.2. Line Segments Reduction Using Contour Lines from Previous Frame

Most of the detected line segments using constrained HT correspond to true edges. By using the movement smoothness assumption, line segments that are far away from contour lines extracted from the previous frame will be removed. Let $l_i(\rho, \theta)$ be the i^{th} line segment detected using the constrained HT. Let $X_j(\rho, \theta) \quad j = 1, 2, \cdots J$ be the set of contour lines from the previous frame. The minimum distance $D(\rho, \theta)$ from $l_i(\rho, \theta)$ to the previous contour line set is computed as follows,

$$D(\rho, \theta) = \min_{j \in 1, \cdots J} ||l_i(\rho, \theta) - X_j(\rho, \theta)|| \quad (2)$$

If $D(\rho, \theta)$ is smaller that a prechosen threshold, $\lambda_{\rho,\theta}$, this line segment $l_i(\rho, \theta)$ will be reserved. This step will again reduce a large number of redundant lines. For example, Fig 4 (b) shows the removed line segments in the HT domain (data points represented by 'o') through this line segment reduction step.

3.3. Line Segments Clustering Using the *k*-means algorithm

Because arm contours are not normally straight lines, small pieces of line segments need to be clustered to get those main lines of the contour. The *k*-means algorithm is implemented [5] to cluster the remaining line segments in the HT parameter space. In our experiments, the number of clusters K was set to $20 - 25$. Because previous estimated values usually give useful information, part of the cluster centers are initialized by taking the parameter values of the contour lines in the previous frame. Each cluster center is calculated as the mean of its (ρ, θ) vectors, weighted by the number of votes. The Euclidean geometric distance $(\rho^2 + \theta^2)$ is used in the *k*-means.

To improve the clustering results, the characteristics of within-cluster and between-cluster are considered. The within-class and

between-class scatter S_w and S_b are computed from the k-means results. The mean of entire data is μ_t. Each cluster has mean μ_j and covariance c_j. $S_w = \sum_j c_j$, $S_b = \sum_j (\mu_j - \mu_t)(\mu_j - \mu_t)^T$. A sequence of output clusterings (Here we choose 10)is produced and the best one with maximum value of $\frac{det(S_b)}{det(S_w)}$ is chosen. After clustering, the main straight lines are generated, as seen in Figure 4. The representation of cluster is the data in one cluster with distance closest to the cluster centroid.

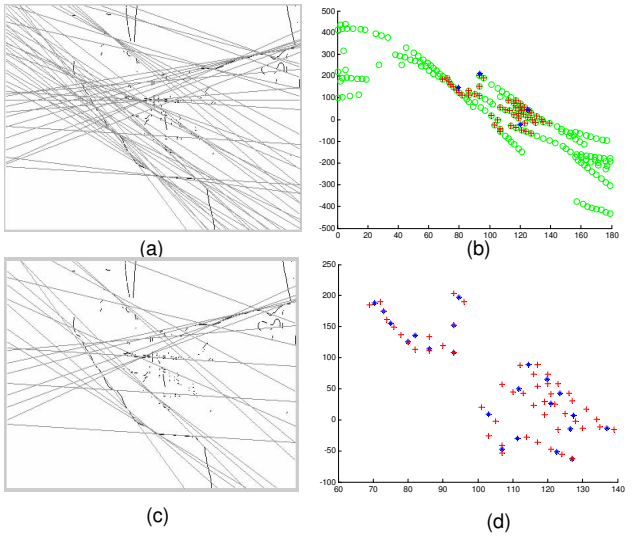

Fig. 4. Line segments clustering results. (a) and (b): the detected lines before clustering in the image and Hough transform domain, respectively. In (b), $\circ$ means discarded lines through the line segments reduction step using previous contour lines, $*$ means previous contour line HT parameters values and $+$ are those remaining line segments; (c) and (d): lines clusters in the image and Hough transform domain, respectively. In (d), $+$ are the input data points to the k-means algorithm and, $*$ are the cluster centers

3.4. Cluster Center Reduction Through Edge Point Assignment

After clustering, edge point are assigned to the cluster centers. For an edge point p at location (x, y) and a cluster center $l_k (k = 1, 2, \cdots, K)$, $d(p, l_k)$, the distance from the point location to this line, and $\theta(p, l_k)$, and absolute direction difference between this point's edge direction and line direction are computed. Then an assignment function $f(p, k)$ is computed as follows:

$$f(p,k) = \min_{k=1,2,\cdots,K} (\hat{d}(p,l_k)^2 + \hat{\theta}(p,l_k)^2) \quad (3)$$

where $\hat{d}(p, l_k) = \frac{d(p,l_k)}{D}$ and $\hat{\theta}(p, l_k) = \frac{\theta(p,l_k)}{180}$. D is pixel length of the image diagonal. D and 180 are used to normalize location and angle distances from the edge point to the line cluster center.

If $f(p, k)$ is less than threshold ε_1 (e.g. 0.1), edge pixdl p will be assigned to one of the line cluster centers that has the minimum value. After all the edge points have been assigned, the number of edge points that each line cluster centers possesses will be counted and normalized by the total number of edge pointed successfully assigned. This normalized edge point counter will be then used as the weights of the line cluster centers. Those line cluster centers with weights less than the threshold ε_2 (e.g. 0.01) will be discarded.

After this step, the remaining line cluster centers will be considered as good line segment candidates. The edge points will reassigned to these candidates again according to equation (3), to reallocate those edge points that were assigned to the line cluster centers discarded during the previous step.

4. CONTOUR LINE REFINEMENT

In this section, we discuss the approach we used to match the line candidates to the contour lines extracted from the previous frame.

4.1. Matching Line Candidates to Contour Lines from the Previous Frame

The Chamfer distance function was firstly proposed by Barrow et al. [6] and improved versions have been used for object recognition and contour alignment. In our approach, Chamfer distance is used to compute the distance between the candidate lines and previous estimated contour lines. The cost function in Chamfer distance is the average value of the squared distances between each point in one line and its closest point in another line. The A and B directed chamfer distance $C(A, B)$ is defined as

$$C(A,B) = \frac{1}{N_a} \sum_{a \in A} min_{b \in B} ||a - b|| \quad (4)$$

where N_a is the point number in A, a and b are points from A and B, $||a-b||$ denotes the Euclidean distance between two points locations a and b. In our framework, A is a set of points assigned the candidate line in the edge map and B is a predicted line, which needs to be matched.

Given a contour line B, the fitting of a candidate line A to B is defined as following:

$$D(A,B) < \lambda_d \quad (5)$$

$$M(A,B) = \alpha C(A,B) + \beta L(A) \quad (6)$$

where $D(A, B)$ is the angle difference of the A and B, $C(A, B)$ is the Chamfer distance between A and B, $L(A)$ is the pixel number of A, α and β are coefficients. In our experiments, $\lambda = 20$, $\alpha = 0.8$ and $\beta = -0.2$. The matching of A and B is that given B, a set of As need to be found such that equation (5) is satisfied and the values of $M(A, B)$ are small. These As will be ranked based on the corresponding values $M(A, B)$. Some of the line candidates can be removed by only keeping top M candidates for each contour line B.

4.2. Edge Point Assignment

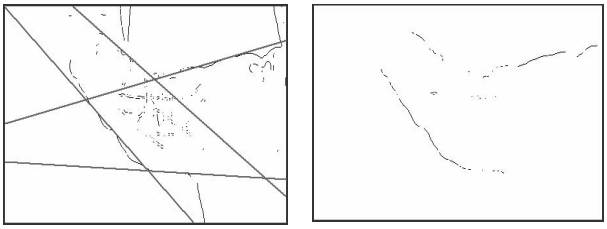

Fig. 5. Contour line matching results. (a): Four extracted lines and (b): edge points assigned to these four lines

After the line candidates have been matched to the contour lines from the previous frame, the edge points will be assigned to the closest one among them. Because the extracted lines are very close to the actual contours, the edge points will be assigned to one of lines with minimum location distance $\hat{d}(p, l)$, if it is less than a threshold ϵ_a. If the corresponding angle difference $\hat{\theta}(p, l)$ is less than a

threshold ϵ_b, this edge point will be assigned this line. Otherwise this edge point will be discarded without being assigned to any of the extracted lines. In our experiments, the values of ϵ_a and ϵ_b are chosen as 0.03 and 0.167, respectively. Fig. 5 shows four extracted lines (a) and edge points assigned to these lines (b).

5. EXPERIMENTAL RESULTS

One example has been presented in the previous sections and another example is given in Figure 6. Some artificial lines were added manually around arm edges and within the arm to create more cluttered environment. So there are more edges in Figure 6 (b). Figures 6 (g),(h) and (i) are the top three matches for each contour line, which satisfy equation (5) and ranked with ascending values of $M(A, B)$ computed using equation (6). The line subtraction from equation (2) will reduce the interference, as seen in (c) and (d). From the results, our approach can find the contour line from the line segment effectively and robustly.

6. CONCLUSION

In this paper, we proposed a robust contour line extraction method using context. As one application of human body tracking, our method is used to extract the arm contour lines from video sequence in a 3D arm tracking system. In order to speed the computation and remove edge noises, the edge orientation is used in Hough transform. Previous estimated object contours are used to extract the desired line candidates. The experiment results show that the robustness and efficacy of our approach. [1]

7. REFERENCES

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

[2] S. Park and J. K. Aggarwal, "Segmentation and Tracking of Interacting Human Body Parts under Occlusion and Shadowing," In *Proceeding MOTION '02*, pages 105-113, 2002

[3] V.F.Leavers, "SURVEY: Which Hough Transform," CVGIP, Image Understanding Vol58, No.2, pages 250-264, 1993

[4] H. KAviSinen, P.Hirvonen, L.Xut and E.Oja, "Probabilistic and non- probabilistic Hough transforms: overview and comparisons," In *Image and Vision Computing* Vol.13 No. 4 May 1995

[5] J.C. Dunn, "A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well- Separated Clusters," J. Cybernetics, vol. 3, 1973, pp. 32-57. Reprinted in Fuzzy Models for Pattern Recognition, J.C. Bezdek and S.K. Pal, eds., IEEE Press, 1992.

[6] H.Barrow, J.Tenenbaum, R.Bolles, and H. Wolf."Parametric correspondence and chamfer matching: Two new techniques for image matching," In *IJCAI*, pages 659-663, 1977

[7] M. Isard and A. Blake. "CONDENSATION C Conditional Density Propagation for Visual Tracking," *International Journal of Computer Vision*, 1998.

[8] C. Sminchisescu and B.Triggs.,"Covariance Scaled Sampling for Monocular 3D Body Tracking," *CVPR*, Vol.1, pp.447-454, 2001

[1] Any opinions, findings and conclusions or recomendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation (NSF).

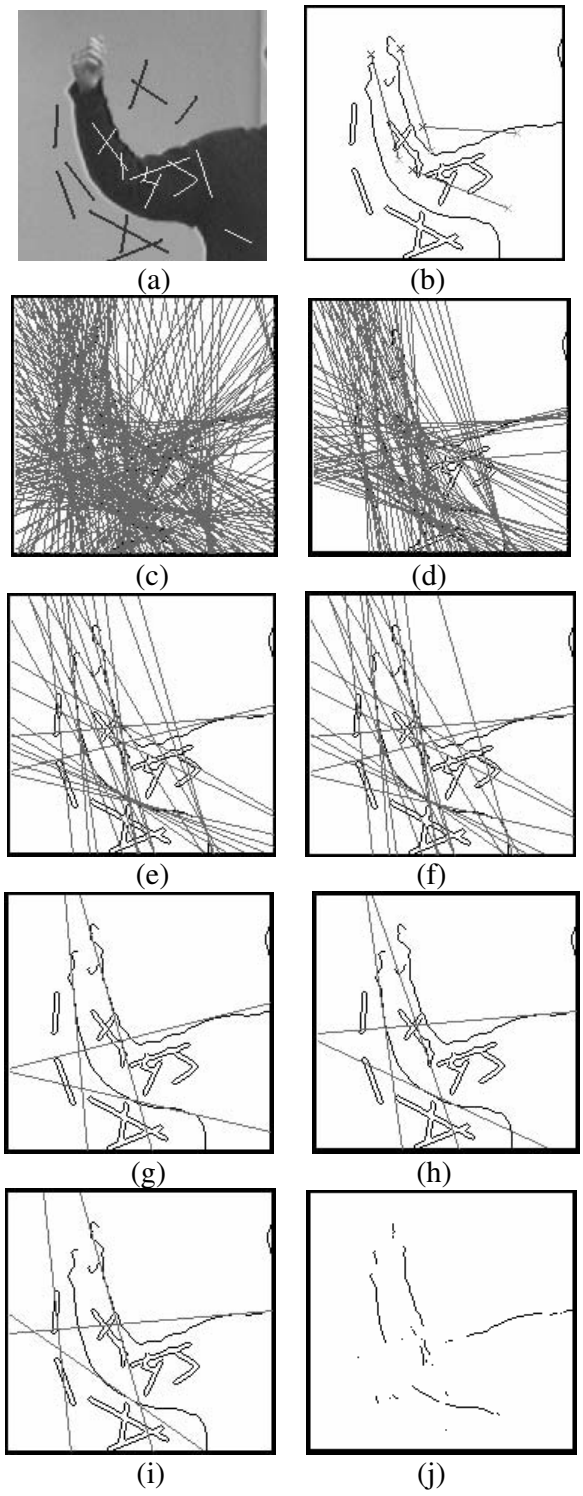

Fig. 6. Example of Contour Line Detection. (a): ROI; (b): detected edges;(c): Detected lines using Hough transform with edge orientation constraints;(d): Line segment reduction using contour previous from the previous frame ; (e): *k*-means clustering; (f): Good line candidates as refined cluster centers via edge point assignment; (g), (h) and (i): Top three (the first, second and third) best matched lines using Chamfer distances, respectively; (j): Edge points assigned to the best matched line candidates