Classifier-based Contour Tracking for Rigid and Deformable Objects

Ali Shahrokni François Fleuret
Pascal Fua*
Computer Vision Laboratory
Ecole Polytechnique Fédérale de Lausanne (EPFL)
CH-1015 Lausanne, Switzerland
{ali.shahrokni,francois.fleuret,pascal.fua}@epfl.ch

Abstract

This paper proposes a machine learning approach to the problem of model-based contour tracking for rigid or deformable objects. The motion of the target is calculated by tracking its contours in a video sequence. We develop a probabilistic representation of contours that allows robust contour tracking in presence of texture and clutter.

We use boosting to train a predictor of the conditional probability of texture transition, given the pixel intensities. The most likely connected contours are obtained by maximizing the posterior probability of object model parameters. The probabilistic formulation allows spatial connectivity of the contours to be formulated in a natural manner. For deformable objects, we use a Hidden Markov Model to calculate the joint law of the conditional probabilities of contour points while for rigid objects geometric properties of the model are used in a framework of random sample consensus algorithm to find the optimal model pose.

We demonstrate that the proposed method is fast and robust for tracking deformable and rigid objects. We also compare our algorithm to several other contour tracking methods.

1 Introduction

In this paper we investigate the use of machine learning techniques to detect boundaries between potentially textured regions. We demonstrate the effectiveness of this approach in the context of tracking of rigid or deformable objects using images that are problematic for other techniques. More specifically, we show how a small set of weak learners can be used to estimate the conditional probability of a texture discontinuity, given its neighborhood. The weighted sum of weak learners can be seen as an approximation of a log-likelihood. This representation of texture discontinuities makes it possible to enforce spatial connectivity of contours quantitatively. Spatial constraints together with

*This work was supported in part by the Swiss National Science Foundation.

trained weak learner responses provide a natural way to detect contours of object which is not attainable through conventional methods such as gradient-based techniques. These characteristics make the resulting contour detection algorithm coherent and simple while remaining general and robust.

In the remainder of this paper, we first discuss related work. We then describe the classification method for contour extraction. Finally Section 4 we go through spatial connectivity constraints used to aggregates the classifier information and report our experiments on tracking and comparison of the performance with some other tracking methods.

2 Related work

The contours for common tracking applications are extracted using different methods. Methods based on local gradient information [2] or edge distance transform [11] are appealing due to their simplicity and speed, but their application is restricted to cases where the contrast is sufficient. Furthermore, they tend to fail in the presence of highly textured objects and clutter, which produce too many irrelevant edges. This is due to failure to exploit the intensity values and structures available in the image.

Texture segmentation techniques such as methods based on graph cuts [1] require computing Markov Random Field (MRF) models which tend to be computationally intensive and therefore not suitable for tracking. Recently Rother et. al. [9] have introduced a fast interactive technique based on graph cuts which yields impressive segmentation results thanks to improvements in the optimization stage and local post-processing. In contrast, rather than making a general hypothesis model for inside and outside texture, in our proposed method we exploit geometric and continuity constraints in conjunction with fast classifier-based estimation of local boundary positions. Our claim is that local boundaries can be picked with more precision with less computation on concentrating the learning on details which is more favorable for tracking proposes.

Machine learning methods and Hidden Markov Models, on the other hand, have been used extensively for object detection [12] and recognition. However, these methods have not yet been applied to low level generic classification problems, such as finding texture discontinuities (cuts) in a narrow image band. The main motivation behind the present work is that such a tool can prove to be useful in a wide variety of applications that require reliable boundary detection.

3 Contour point classification

In this section we show how trained classifiers are used to provide a likelihood measure for the points in the vicinity of a given contour point. The input to each classifier is a narrow image band as described below. The output of each classifier is a binary decision which is '1' if it determines that there is a texture cut in the middle of the band and '0' if otherwise. Furthermore, each classifier is associated with a weight which is related to the total weighted error on the training set. This is explained in detail in section 3.3.

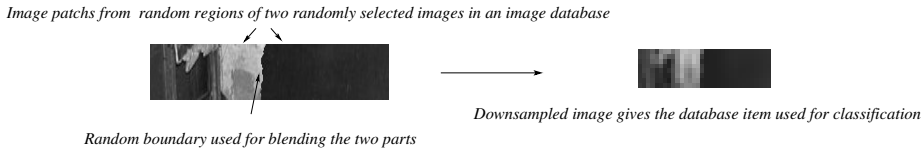


Figure 1: The database samples with a texture transition in the middle are made using blending of random image regions and down sampling.

(a) positive sample (b) negative sample (c) feature parameters

Figure 2: Examples of (a) positive, and (b) negative training samples used to classify texture cut in the middle of the test image. The database items are 32×8 pixels long. (c) features parameters defined on the images.

3.1 Database

The set of training examples consists of images of size 32×8 pixels. The positive examples are composed of two randomly selected patches of size 128×32 from random images collected from the web. These patches are concatenated to each other to form an image of size 256×32 with a texture transition in the middle. To produce more realistic texture transitions, the concatenation is done by stochastic blending of the connecting ends of the two patches as illustrated in Fig. 1. Finally the results are downsampled to give 32×8 images with a smooth texture transition in the middle. The negative samples contain only one downsampled randomly selected image region. Examples of both positive and negative samples are shown in Fig. 2.

3.2 Classifiers

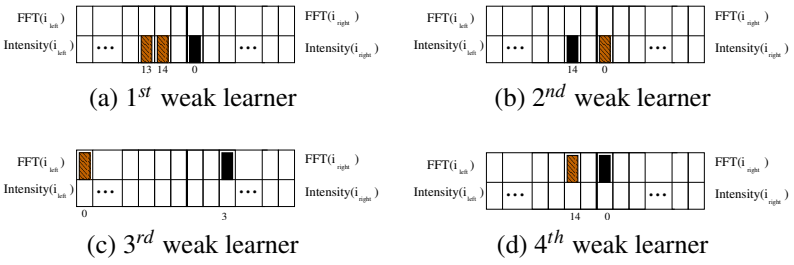
The features that we use for classification of pattern changes are the mean of the different bands on both sides of the database samples in frequency or intensity domains as shown in Fig. 2-c. Therefore, a feature $f(s)$ corresponding to image pixel sequence s can be defined by 4 parameters:

$$f(s) = f_{m,g,n,h}(s) = 1/(g+1) \sum_{k=m}^{g} I_l(k) - 1/(h+1) \sum_{k=n}^{h} I_r(k) \qquad (1)$$

where I_l and I_r are the intensity values or the magnitude of FFT coefficients of the left and right pixel sequences of the image respectively.

3.3 Training classifiers

We associate a feature to each classifier. The training is done using Adaboost [5] that learns by selecting the relevant features with the most discriminating information. We use the algorithm described in [12] for the implementation of the Adaboost and at each stage



(a) 1^{st} weak learner (b) 2^{nd} weak learner

(c) 3^{rd} weak learner (d) 4^{th} weak learner

Figure 3: The first four trained classifiers

r the trained classifier h_r is given a threshold T_r, a parity P_r and a weight w_r. The output label of such classifier is then given by

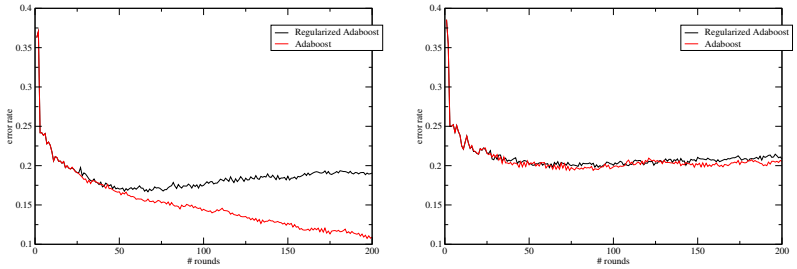
$$h_r(s) = \begin{cases} 1 & \texttt{if } P_r f(s) > P_r T_r \\ 0 & \texttt{otherwise} \end{cases} . \tag{2}$$

The first four trained classifier features are shown in Fig. 3. The bars show the indices of pixel on left and right side of a sample image and whether features are in intensity or FFT domain. The hatched bars indicate the parity of the classifier (i.e. P_r is -1 in Eq. 2 if the hatched bars are on the left side). We see that the first two classifiers compare intensity values of the last pixels on the left side with the first pixel on the right side with different parities. That basically means that the best way to determine whether there is texture cut in the middle of an image is simply by comparing pixels around the potential cut position. We conjecture that the reason why there is one pixel space between the indices compared on both sides in the case of the first two classifiers is that in most edges the two textures tend to blend into each other. The next two classifiers are in FFT domain and compare different frequency bands of the two sides of the image.

A boosted set of classifier with R weak learners then gives a score $x_i = \sum_{r=1}^{R} w_r \times h_r(i)$ to input sequence s. Once the classifiers have been trained and boosted using enough training samples, we can check the performance on the training and test databases for different numbers of weak learners. Fig. 4 shows the error rate of classifications on training and test sets versus the number of weak learners. The graphs shown correspond to trained classifiers using Adaboost and regularized Adaboost [8] techniques.

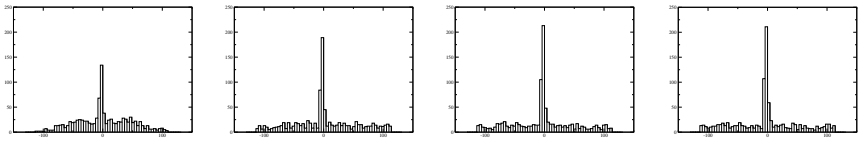
We see that Adaboost error rate decreases exponentially on the training set while it remains constant on the test dataset after about 50 weak learners due to over-fitting. Regularized Adaboost is a way to reduce the effect of outliers. Fig. 4 shows that although the over-fitting is reduced in the case of the training set, no significant improvement is observed on the test set in the case of regularized Adaboost. Therefore, we use non-regularized version of Adaboost algorithm to train of the weak classifiers.

To study the performance of the classifier in detecting the texture boundary in an image the following simulation was conducted. Fig. 5 shows the distribution of error in pixels of the detected texture transition position from the real texture boundary in 1000 randomly generated images using different numbers of weak learners. The images are 256 pixels long and are made by blending two random textures in the middle, therefore we can assume a texture change in the middle of them. The sliding window is 32 pixels long which is the same size as the 4000 used as training examples. The detected boundary



(a) training set (b) test set

Figure 4: Classification error rate vs. number of weak learners trained using 2000 positive and 2000 negative samples. Adaboost (thin curve) and regularized boosting (thick curve) are used for the training of weak learners. (a) is the error rate on the original training set and (b) is the error rate on a test set of 1000 positive and 1000 negative samples.



1 weak learners 10 weak learners 50 weak learners 100 weak learners

Figure 5: Histogram of texture transition detection error in pixels on 1000 test images using different numbers of trained weak learners. The error decreases with higher number of weak learners.

corresponds to the position which maximizes the weighted sum of classifiers response. We can see in Fig. 5 that as the number of weak learners increases, the peak around zero error gets more prominent and the distribution elsewhere becomes flat. Moreover, it can be noted that a small number of weak learners is enough for reliable detection.

3.4 Model of the conditional probability

For each scanline and each location in that scanline, we obtain a response equal to the weighted sum of weak learners as defined in Section 3.3. We propose to combine those responses into a probabilistic model by first converting them into conditional probabilities as follows.

At a given location, we denote by Y a random variable standing for the presence of a cut, by S the pixel intensities in the considered neighborhood on the search direction and by x the weighted sum of weak learners at that location. The posterior probability of having a texture transition at that location is thus given by:

$$P(Y=1\,|\,S=s) = P(Y=1|X=x) = \frac{1}{1+\frac{P(X=x|Y=0)}{P(X=x|Y=1)}} \quad . \tag{3}$$

Under the assumption that $X|Y=0$ and $X|Y=1$ both follow normal laws of expectation μ_0 and μ_1 and of same variance σ, we obtain

$$P(Y=1\,|\,S=s) \quad = \quad \frac{1}{1+\exp\{-\alpha\,(x-\beta)\}} \tag{4}$$

with $\alpha=(\mu_1-\mu_0)/\sigma^2$ and $\beta=(\mu_1+\mu_0)/2$, and μ_0, μ_1 and σ estimated with a trivial likelihood maximization.

Finally, the weighted sum of weak learners X can be seen as an approximation of a log-likelihood $\log\frac{P(Y=1|X)}{P(Y=0|X)}$, similarly to the combination of weak learners in a naive Bayesian predictor [7]. Thus, the parameters α and β stand for a correcting factor for the dependency between weak learners and the prior log-ratio respectively.

In the next section we show how the object model can be used, in conjunction with the classifier responses around the object outlines, to find the most probable model silhouette for tracking applications.

4 Experimental results

In this section we present our experimental results on tracking deformable and rigid objects. Current implementation of our system can reach a speed of 6 to 10 frames per second on a 2.8 GHz Pentium 4. Its speed mainly depends on the number of samples on the model, length of the pixel sequences on each sample point, and the number of weak learners used in the conditional probability classifier. According to our experiments a low number of classifiers such 4 to 10 depending on the complexity of the sequence is enough for reliable contour tracking.

At each frame during tracking we have an initial guess for the object silhouette. The conditional probability model is applied to search lines parallel to the normal directions at each sample point. Each search line gives independently the probability of texture change across it. These probabilities can be aggregated to yield a probability distribution for the most probable contour, i.e. the object boundary.

Based on whether the tracked object is rigid or deformable, we use two methods to treat independent probabilities given by the conditional probability classifier on each search line in order to obtain the object contour in the current frame. These methods are discussed below.

4.1 3–D tracking of rigid objects

For a rigid object robust stochastic search methods such as RANSAC [3] can be used to determine the optimal pose which maximizes the posterior boundary probability. We use a system similar to Drummond and Cipolla's [2] in which the search starts from the estimated projection of a 3-D object model and performs a line search in the direction perpendicular to the projected edges. Our proposed algorithm gives the conditional probability of texture change along these lines. At each iteration of our RANSAC-style algorithm, for each visible model edge two scanlines are selected randomly. Two points are then drawn randomly on those scanlines according to the conditional probability given by the classifier. The straight line defined by those two points is used as the observation for the current model edge for which the RANSAC support is equal to the sum of the conditional probabilities of the pixels on it. After enough iterations we pick up the line with largest

support for each model edge. Pose parameters are then taken to be those that minimize the distance of the model's projection to the selected lines. Fig. 7 shows several frames of tracking results in presence of clutter on a sequence of 600 frames with large and fast camera motions.

For the purpose of evaluation, we have tried the same system using our method and three other edge-based trackers, namely, gradient-based tracker [2], Fisher discriminant classifier [4], and a fast texture boundary detector [10]. The results obtained on a short sequence are shown in Fig. 8 and are discussed below.

Gradient-based tracker: The corresponding implementation of the gradient-based tracker works well and is very fast when the target object stands out clearly against the background. However it tends to fail for textured object whose boundaries are hard to detect unambiguously. An example of such failure is shown in the second row of Fig. 8.

Fisher's discriminant function: As an example of linear discriminant we take Fisher's discriminant [4]. The goal is to find a position that maximizes the between class variance while minimizing the within class variance of intensity values at the same time on either side of the potential texture boundary. Fisher's discriminant analysis has been used successfully in applications such as lip tracking [6] in a framework similar to that described in this paper. As can be expected from its definition, Fisher's criterion for edge-based tracking works well only when the assumption of two classes with normal distribution is valid around the object boundary. The tracking results are shown in the third row of Fig. 8, where some of the detected edges are drifted from the real ones.

Texture boundary detection: Finally, we compare our method with a fast texture boundary detector [10], which assumes uniform prior for all texture distributions and updates the estimated texture distribution as new pixel on scanlines are encountered. This method is fast and adapted for real-time tracking. However, correct estimation of texture distribution relies on a relatively long scanline in order for the transition matrices to converge. The 4^{th} row in Fig. 8 shows tracking results using this method. It can be noticed that edges lacking texture around them cause the tracking to break gradually.

Replacing the boundary detector part of our system with our proposed method based on the classifier and RANSAC not only reduces the length of the search line required to almost half (60 pixels) but also yields good tracking results on the same sequence as shown in the first row of Fig. 8.

4.2 Tracking deformable object

When the model has a high number of degrees of freedom, stochastic search for the optimum model parameters is not trivial. In this case we present a Hidden Markov Model to relate the responses of the classifier along different lines orthogonal to the candidate edge. This is the case when the model does not consist of straight or geometrically well defined edges.

We define a HMM with, as observable state, a set of pixel sequences $\mathbb{S} = \{S_i \,|\, i = 1,...,N\}$ where each sequence S_i has M pixels as illustrated in Fig. 6. The HMM is characterized by the visible state S_i which is the sequence of pixel intensities on line i and the hidden state Z_i which is the location of the real edge along the line i. The likelihood distribution in each of the states, $P(S_t \,|\, Z_t)$, is given by individual sequence conditional probabilities of section 3 and the dependency between successive hidden states, $P(Z_{t+1} \,|\, Z_t)$, is modeled by a Gaussian kernel which ensures connectivity and smoothness of the bound-

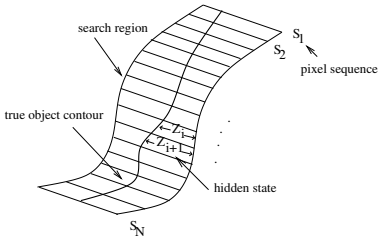


Figure 6: Definition of the HMM on the conditional probability classifier responses over a search image

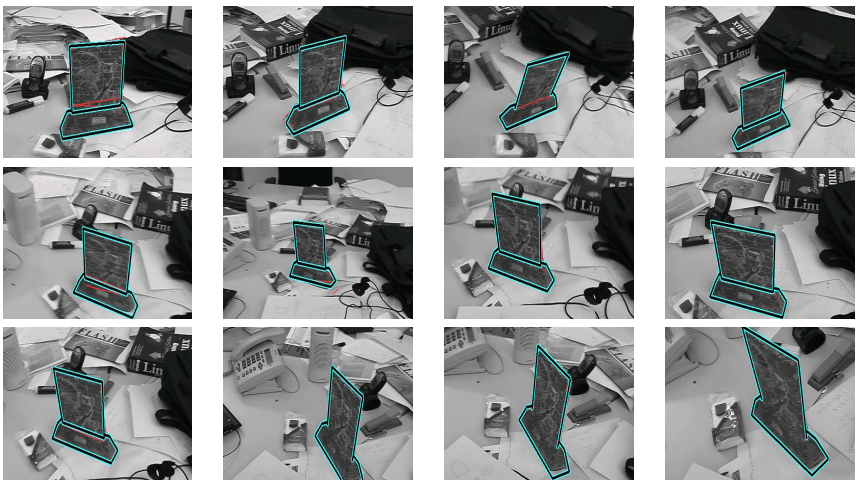


Figure 7: Tracking textured object against cluttered background.

ary. Finally, the initial state distribution $P(Z_0)$ is assumed to be uniform. We wish to maximize the probability of the hidden states for a given set of sequences, $\mathbb{S}$:

$$\arg \max_{z_1,...,z_N} P(Z_1 = z_1, ..., Z_N = z_N \,|\, S_1, ..., S_N) \tag{5}$$

Solving Eq. 5 yields the state sequence which corresponds to the most likely contour in the search line and it can be efficiently done by dynamic programming.

We have applied our algorithm to tracking of the upper body motion as an example of deformable body motion. The user draws an initial path by clicking on some points around the body outline. These points define an spline curve which is used as the starting guess for the body outline. The outline is then obtained using the conditional probability classifier in conjunction with HMM and it serves as the initial guess for the successive frames. Several frames of the tracking results are shown in Fig. 9.

5 Conclusion

We have presented a classification-based method to detect boundaries between potentially textured regions and applied it to tracking of rigid or deformable objects. The classifica-

Figure 8: Comparison between different contour tracking algorithms. First row: Tracking results using our classifier based method and RANSAC. Second row: tracking results using an edge-based tracker. Third row: Tracking results using Fisher discriminant function. Fourth row: tracking results using texture boundary detection

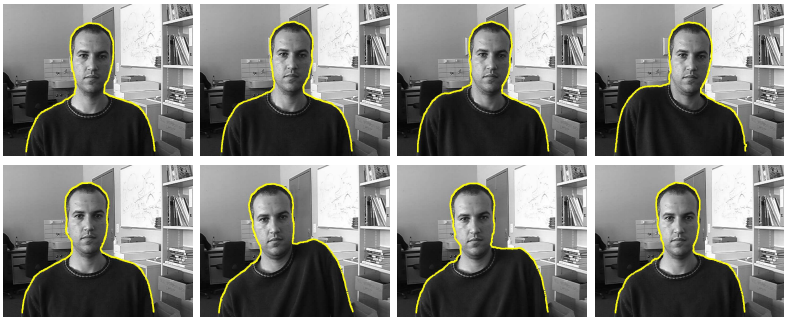


Figure 9: Tracking deforming body outlines.

tion is handled by a small set of trained weak learners which estimate the conditional probability of texture cut, given the neighborhood.

The tracked object model provides constraints which can be combined with the likelihood distribution given in the vicinity of each contour point, thus allowing the extraction of the optimum contour for tracking. These constraints are enforced by either of the following fashions.

HMM. A Hidden Markov Model is defined to calculate the joint law of local conditional probabilities along the contour which is particularly useful in case of tracking free meshed objects.

Stochastic pose estimation in case of rigid objects. RANSAC method is used to find a pose that maximizes the sum of probabilities of contour points given by the classifier.

Training classifiers to distinguish texture cut is a novel approach to contour tracking and is validated through our reported experimental results on tracking deformable and rigid objects. Moreover, a simple comparison shows that it outperforms three other edge-based trackers, namely, gradient-based tracker, Fisher discriminant classifier, and a fast texture boundary detector.

Finally, it is possible to concentrate the training of the weak learners on specific subjects, such as faces, hands or objects. This will be the focus of our future work in that area. Another issue for future work is to study the performance of other features such as cooccurrence matrices as texture transition classifiers.

References

[1] A. Blake, C. Rother, M. Brown, P. Perez, and P. Torr. Interactive image segmentation using an adaptive gmmrf model. In *European Conference on Computer Vision*, volume 1, pages 428–441, 2004.

[2] T. Drummond and R. Cipolla. Real-time visual tracking of complex structures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(7):932–946, july 2002.

[3] M.A Fischler and R.C Bolles. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Communications ACM*, 24(6):381–395, 1981.

[4] R.A. Fisher. The statistical utilization of multiple measurements. *Annals of Eugenics*, 8:376–386, 1938.

[5] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *European Conference on Computational Learning Theory*, pages 23–37, 1995.

[6] Robert Kaucic and Andrew Blake. Accurate, real-time, unadorned lip tracking. In *ICCV*, pages 370–375, 1998.

[7] P. Langley, W. Iba, and K. Thompson. An analysis of bayesian classifiers. In *Proceedings of AAAI-92*, pages 223–228, 1992.

[8] G. Rätsch, T. Onoda, and K. R. Müller. Regularizing adaboost. In *Neural Information Processing Systems*, pages 564–570. MIT Press, 1998.

[9] C. Rother, V. Kolmogorov, and A. Blake. "GrabCut": Interactive Foreground Extraction using Iterated Graph Cuts. *ACM Transactions on Graphics*, 23(3):309–314, 2004.

[10] A. Shahrokni, T. Drummond, and P. Fua. Texture Boundary Detection for Real-Time Tracking. In *European Conference on Computer Vision*, pages Vol II: 566–577, Prague, Czech Republic, May 2004.

[11] A. Thayananthan, P.H.S. Torr, and R. Cipolla. Likelihood models for template matching. In *British Machine Vision Conference*, pages 949–958, 2004.

[12] P. Viola and M. Jones. Rapid Object Detection using a Boosted Cascade of Simple Features. In *Conference on Computer Vision and Pattern Recognition*, pages 511–518, 2001.