Classification-based Probabilistic Modeling of Texture Transition for Fast Line Search Tracking and Delineation

Ali Shahrokni, Tom Drummond, *Member, IEEE,* François Fleuret *Member, IEEE,* and Pascal Fua, *Senior Member, IEEE*

Abstract—We introduce a classification-based approach to finding occluding texture boundaries. The classifier is composed of a set of weak learners which operate on image intensity discriminative features which are defined on small patches and fast to compute. A database which is designed to simulate digitized occluding contours of textured objects in natural images is used to train the weak learners. The trained classifier score is then used to obtain a probabilistic model for the presence of texture transitions which can readily be used for line search texture boundary detection in the direction normal to an initial boundary estimate.
This method is fast and therefore suitable for real-time and interactive applications. It works as a robust estimator which requires a ribbon like search region and can handle complex texture structures without requiring a large number of observations. We demonstrate results both in the context of interactive 2-D delineation and fast 3-D tracking and compare its performance with other existing methods for line search boundary detection.

Index Terms—Computer Vision, Texture, Tracking, Classification, Segmentation.

1 Introduction

Edge-based methods have proved very effective for real-time 3–D model-driven pose estimation [1], [2] and interactive 2–D delineation. Unfortunately, they often fail in the presence of highly textured objects and clutter, which produce too many irrelevant edges. In such situations, it would be advantageous to detect texture boundaries instead. However, because texture segmentation techniques require computing statistics over image patches, they tend to be computationally intensive and have therefore not been felt to be suitable for such purposes.

To dispel this notion, we propose a classification-based approach to finding texture boundaries by looking in the direction normal to an outline that is a very rough approximation of the actual contour. This yields a *line search boundary detection* algorithm that is both extremely fast and robust. This is in contrast with state-of-the-art texture segmentation techniques such as [3], [4], which yield impressive segmentation results but require processing a large portion of the foreground and background regions to model the textures using Gaussian mixture models and locate the boundaries.

More specifically, we show how a classifier composed of a small set of weak learners can be used to estimate the conditional probability of a texture discontinuity, given its neighbourhood. The weighted sum of weak learners' responses can be interpreted as an approximation of the log-likelihood of occluding texture boundaries in natural images. We construct a database which is designed to simulate the occluding textures as captured by digital video cameras and use it to train the classifier. The response of the weighted sum of the weak learners then models the posterior probability of an occluding texture boundary. This method yields a robust estimator of the boundary based on training data which requires a ribbon like search region and can handle complex texture structures without requiring a large number of observations.

The trained weak learner responses combined with spatial constraints provide a natural way to detect object

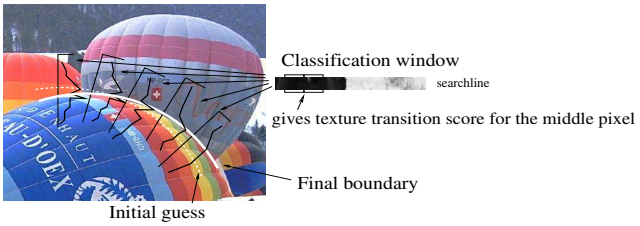


Fig. 1. Contour extraction by classification of texture transition given an initial curve (white dotted curve). Classification score is computed for pixels on searchlines (black lines) by sliding a small classification window along them.

- *A. Shahrokni is with the Department of Engineering Science, University of Oxford, UK.*
  E-mail: ali@robots.ox.ac.uk
- *F. Fleuret is with IDIAP Research Institute, Switzerland.*
- *T. Drummond is with the Engineering Department, University of Cambridge, UK.*
- *P. Fua is with the Computer Science Department, Ecole Polytechnique Fédérale de Lausanne, Switzerland.*

Manuscript received April 19, 2005; revised January 11, 2007.

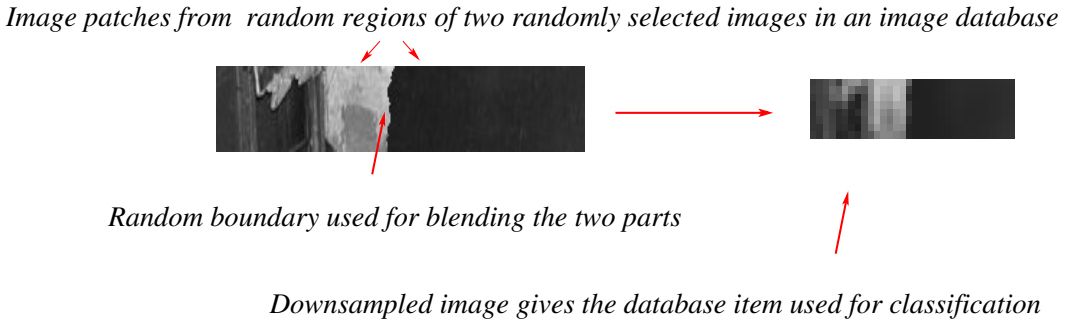


Fig. 2. The database samples with a texture transition in the middle are made by blending random image regions and down sampling.

contours which is not attainable through conventional methods. The resulting contour detection algorithm is versatile, robust, and fast.

We demonstrate this by integrating into real-time 3D tracking and interactive 2D delineation algorithms. We also compare our method with other techniques that are applicable to real-time line search boundary detection. These methods represent different categories of boundary or texture models used in line search algorithms. The first one is the conventional gradient-based line search boundary detection [2]. The second one is based on a parametric Gaussian texture model that uses Fisher's discriminant metric to locate texture boundaries and the final method uses a non-parametric texture model based on a uniform prior for texture process on both sides of the boundary to maximize the texture transition probability in a Bayesian sense [5].

The remainder of this paper is organized as follows. An overview of the related existing classification-based methods for object boundary detection is given in Section 2. We introduce our approach to texture boundary detection in Section 3 and compare its respective merits against several existing line search methods in Section 4. In Section 5, we show how we can exploit the probabilistic model of the classification score to detect object boundaries and delineate and track complex objects using geometric constraints.

2 Related Work

Texture delineation and segmentation techniques work based on estimation of a texture model on both sides of the boundary of an object. In case of real time line search applications, the challenge is to infer an accurate model for the foreground and background textures using small number of observations. Therefore methods based on mixture models or Markov Random Fields (MRF) which seek to find a global solution to the MRF energy [3], [4] are not applicable in the context of line search. Other existing texture segmentation methods based on graph energy minimization over entire image [6], [7] require heavy computation for optimization and therefore are not suitable for fast tracking.

Here we are interested in learning low level generic discriminant characteristics of textures which can be used in finding texture discontinuities (cuts) in a narrow image band. Such a tool can prove useful in a wide variety of applications that require reliable and fast boundary detection. Efficient algorithms have been proposed to maximize a classification score over image features for recognition and object localization. [8] uses this scheme to localize visual words to determine the object's bounding box. Martin *et al.* [9] use features such as brightness, color, and texture measures to estimate the posterior probability of a boundary passing through the center point of an image patch in form of a disc. They use a large database of manually segmented natural images as the training set to model the probability of a pixel being on- or off-boundary conditioned on some set of local image features. These image features consist of brightness, color, and texture features. Their method involves obtaining histograms of intensity, chrominance or banks of texture filter responses on two sides of oriented discs and then calculating the χ^2 distance between the histograms on both sides. The above cues are then combined into a single function that gives the posterior probability of a boundary at each pixel and orientation.

Dollár *et al.* [10] use a large number of generic features calculated over image patches at multiple scales and orientations to describe edges in natural images. The features are then combined with a classifier trained using the probabilistic boosting tree algorithm to detect boundary points. Their method is similar to our approach to edge localization. The basic idea of classification for line-search boundary estimation was first introduced in our earlier work [11] and is developed and further analyzed in this paper. The fundamental difference that characterizes and distinguishes our work from the above edge-classification techniques is the emphasis on line search real-time tracking applications. Therefore, here we are greatly constrained by the choice of discriminative features which can help edge classification with minimal amount of observations and operations as explained in the next section.

3 Contour Point Classification

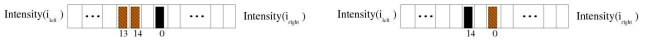
We locally model the texture around an object boundary by two distinct texture processes on each side. We also assume that these processes work along the perpendicular direction to the local boundary orientation. This is a valid local approximation if an estimation of the boundary direction is available. This approach is further justified by the existence of the aperture problem that is associated with the line search methods [2]. Hence, the texture orientation can be locally approximated to be isotropic using the available estimate of the object boundary direction (pose in the previous frame for example). We therefore define a searchribbon as a set of searchlines perpendicular to the estimated boundary direction. The searchribbons can be used in a supervised training scheme by considering a training dataset containing small patches of blended textures. If the training set contains enough examples to accurately model texture transitions we can construct a predictor for object boundary tracking. As illustrated in Fig. 1, such a predictor works on long searchribbons by sliding a small classification window and calculating the score of how well the image area under the classification window corresponds to a patch with a texture transition in the middle.

Here, the input to the classifier is a narrow image band, as shown in the classification window of Fig. 1, and the output is a continuous response over the searchribbon which is larger on the texture transition boundary. In the remainder of this section we show in detail how the classification score can be used to provide an estimate of the posterior probability of texture transition, given the points in its vicinity. We can then combine the probability distributions obtained on several searchribbons to reliably extract object contours.

3.1 Database

The database examples must be designed to simulate occluding contours in natural images. One choice would be to use hand segmented databases of object boundaries. However this might be limiting in the characteristics of the textures which appear in those databases. Instead, we try to simulate the video acquisition of occluding textures by blending two patches through a random path as illustrated in Fig. 2. Furthermore scale invariance can easily be implemented in this scheme by incorporating a pyramidal structure in database reconstruction.

Our 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 arbitrary 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. Finally the results are downsampled to 32×8 images with a smooth texture transition in the middle. The negative samples contain only one downsampled randomly selected image patch.



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

Fig. 3. The first two trained weak learners. The bars show the indices of pixel on left and right side of a sample image. The hatched bars indicate the parity of the classifier.

3.2 Classifier

We use AdaBoost [12] to construct our classifier by linearly combining simple weak learners. Boosting allows to combine weak learners from heterogeneous sets of functionals. Three types of features are used as weak learners in our experiments. Namely, we can compare mean intensity or frequency coefficients at different size-varying regions or frequency bands on two sides of an image. Another type of feature is comparison of cooccurrence frequencies of pixel intensities on the two sides. The intensity features prove to be both efficient and discriminative. We therefore limit our discussion to those features and refer the reader to [13] for further details on other feature types.

Mean Intensity Energy

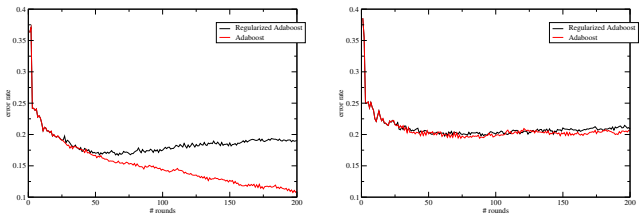
For each searchribbon, there are four parameters to be determined for each intensity feature weak learner that we denote by f_r. These parameters are the positions and length of bands of pixels on the left and right side of the classification window. These bands are used to compare means of intensity on both sides. The total number of intensity features is $[a \times (a+1)/2]^2$ where a is half of the length of the classification window. For efficiency we use a scheme similar to integral images [14] to represent the searchribbon pixel information. This intermediate representation allows computation of a weak learner using only four references to the pre-calculated integral image. Moreover, we pre-compile the parameters of the trained weak learners such as length and threshold to further reduce memory access.

3.3 Training

We combine weak learners using the Adaboost algorithm. At each iteration r the parameters of the weak learner h_r, associated to the feature f_r, are optimized on the database examples. These parameters are a threshold T_r, a parity $P_r = \pm 1$, which is used to account for change in the direction of the inequality, and a weight w_r. The output of such classifier is then given by

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

The first two trained weak learners are shown in Fig. 3. These classifiers compare intensity values of the last pixels on the left side with the first pixel on the right side with different parities. This result is intuitive and it implies that the best way to determine a texture



(a) training set (b) test set

Fig. 4. Classification error rate vs. number of weak learners chosen using 2000 positive and 2000 negative samples. Adaboost (thin curve) and regularized boosting (thick curve) are used for choosing the 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.

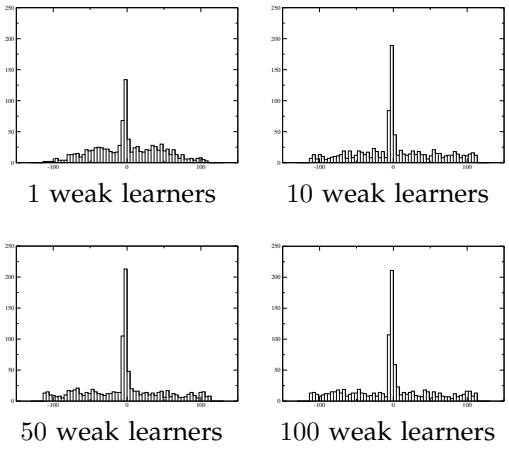


Fig. 5. Histograms of texture transition detection error in pixels on 1000 test images using different numbers of weak learners. The error decreases with higher number of weak learners.

boundary is by comparing pixels around the potential cut position. The one-pixel gap between the compared pixels on both sides can be explained the presence of edge blur and blending of the textures in the training set. The blending also introduces asymmetry in the database which is in turn reflected in the weak learners as can be seen in the difference between the indices used in the first two weak learners in Fig. 3.

A boosted classifier with R weak learners then gives a score $x_i = \sum_{r=1}^{R} w_r \times h_r(i)$ to input sample s. 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 [15] techniques.

We see that Adaboost error rate decreases on the training set as the number of weak learners increases while it converges to a constant value on the test dataset after about 50 weak learners. Fig. 4 shows that although the over-fitting is reduced with the regularized Adaboost on the training set, no significant improvement is observed on the test set. Therefore, we use the non-regularized version for training. The amount of over-fitting can be reduced by using more training samples. For tracking and contour delineation applications however we found that the training database containing 4000 examples produced desirable results.

Fig. 5 shows the distribution of error (in pixels) for the detected texture transition position using different numbers of weak learners. The histograms are obtained using only intensity features on a database composed of 1000 randomly generated images using real textures. Error histograms for other features types have been reported in [13] for further comparison. The images are 256 pixels long and are made by blending two random textures in the middle. The sliding window is 32 pixels long which is the same size as the 4000 used as training examples. The detected boundary corresponds to the position which maximizes the weighted sum of weak learners' responses. The asymmetry in the weak learners and the succession of different parities causes the histograms in Fig. 5 to be slightly skewed. This asymmetry can be reduced by symmetrizing the database by including the mirror image of all database exemplars. We further note that a small number of weak learners (< 10) is enough to obtain above 70 per cent of the detected point within 5 pixels of the true boundary position.

3.4 Model of the Conditional Probability

For each location on the searchribbon, we obtain a score which is equal to the weighted sum of weak learners as defined in Section 3.3. The sum of weak learners' responses yields an approximation of the conditional probability of texture transition at every location on a searchline. At a given location, we denote by Y a random variable standing for the presence of a cut, by S the pixel intensities on the search line 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:

$$\begin{aligned} P(Y=1\,|\,S=s) &= P(Y=1|X=x) \\ &= \frac{1}{1+\frac{P(X=x\,|\,Y=0)}{P(X=x\,|\,Y=1)}}\,. \end{aligned} \tag{2}$$

Under the assumption of normal distribution for the score in the positive and negative datasets with means μ_1 and μ_2 and a unique standard deviation σ, which was verified to be a good approximation on our datasets, we obtain

$$P(Y=1\,|\,S=s) = \frac{1}{1+\exp\{-\alpha\,(x-\beta)\}} \tag{3}$$

where $\alpha = (\mu_1 - \mu_0)/\sigma^2$ and $\beta = (\mu_1 + \mu_0)/2$. μ_0, μ_1 and σ are estimated by likelihood maximization on the training set. Eq. 3 represents the conditional probability of texture transition as a sigmoid function. The weighted sum of weak learners X can be seen as an approximation

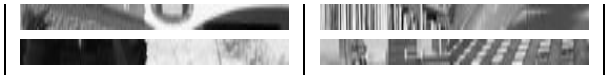


Fig. 6. Magnified examples of the searchribbons of 256×8 pixels generated using patches of natural images. The search ribbons are used as test bed for analysis of different methods.

of a log-likelihood $\log \frac{P(Y=1|X)}{P(Y=0|X)}$, which is similar to the combination of weak learners in a naive Bayesian predictor [16]. Thus, the parameters α and β stand for a correcting factor for the dependency between weak learners and the prior log-ratio respectively.

4 Evaluation and Comparison

In this section we provide analysis of the classifier introduced in Section 3. We use a 32-pixel long sliding window over a searchribbon of 3 pixels wide to obtain a classification score with pre-compiled 30 trained weak learners and integral images. The detected boundary position corresponds to the position which maximizes the weighted sum of weak learners' responses to the subimage under the sliding window or equivalently the probability of boundary given by Eq. 3. We also compare the results with three other techniques associated with the line search approach. The first method is gradient-based line search boundary detection [2] which looks for the first significant change in the intensity gradient along the searchline. The second method models the texture on each side of the boundary by a Gaussian distribution and uses Fisher's discriminant metric [17] to maximize the ratio of intraclass to interclass covariances. The third method is the Bayesian boundary estimation [5], which uses a uniform prior and a non-parametric texture process model on both sides of the boundary to maximize the probability of texture transition in a Bayesian sense using a small amount of observations.

We analyze the distribution of error for each method on a test set which is built by combining random pieces of real images. This database consist of 1000 randomly generated images similar to previous example. Some examples of the images in the test sets are shown in Fig. 6. Note that other texture cuts can appear in the stripes but we are interested in detecting occluding contour type texture transitions which occur at the center of the test images. We use similar searchribbons for all methods with the length of 81 pixels and the width of 5 pixels centered in the middle of each stripe. There is a margin of 16 pixels on each side of the searchribbons to help the convergence of the model-based methods such as Fisher's model and the transition matrix in the Bayesian approach. Therefore the absolute error varies between 0 to 25.

Fig. 7 shows the accumulative histogram of error in pixels from the true texture boundary in the test set. This histogram indicates that for a given error value the classification method always has a higher rate of boundaries founds within that error range.

TABLE 1
Mean computational time for boundary detection on 1000 searchribbons using different methods.

Method	Classification	Transition Matrix	Fisher	Gradient
Time (ms)	0.16	0.2	0.1	0.05

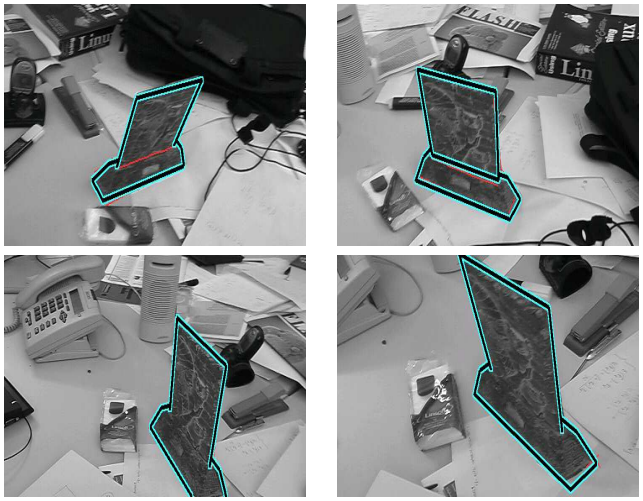


Fig. 8. Tracking textured object against cluttered background.

The mean computational time in milliseconds for all presented methods is shown in table 4. The time is calculated for 1000 images with the searchribbon parameters described above. The classifier uses 30 weak learners and integral images.

5 Results

In this section we discuss the applications of the classification-based boundary estimation to tracking and interactive texture delineation. We consider tracking of deformable and rigid objects.

We assume to have, at each frame during tracking, an initial estimate for the object silhouette. The trained classifier's conditional probability model is applied at each model sample point in a directional search normal to the model edge at that location. Each searchline independently gives the probability of texture change across it. These probabilities are then aggregated using geometric constraints (smoothness or availability of a 3–D model) as explained in [13] to yield a posterior probability distribution for the object contour.

The speed-optimized and precompiled implementation of the classifier-based system can reach a speed up to 150 fps for independent searchribbon tracking on a 2.8 GHz Pentium 4 with 10 weak learners with searchribbons of 81 pixels long and 3 pixels wide. The non-optimized version however runs at 8 fps with the same parameters. Therefore, the use of integral array and precompilation of the weak learners into the tracking

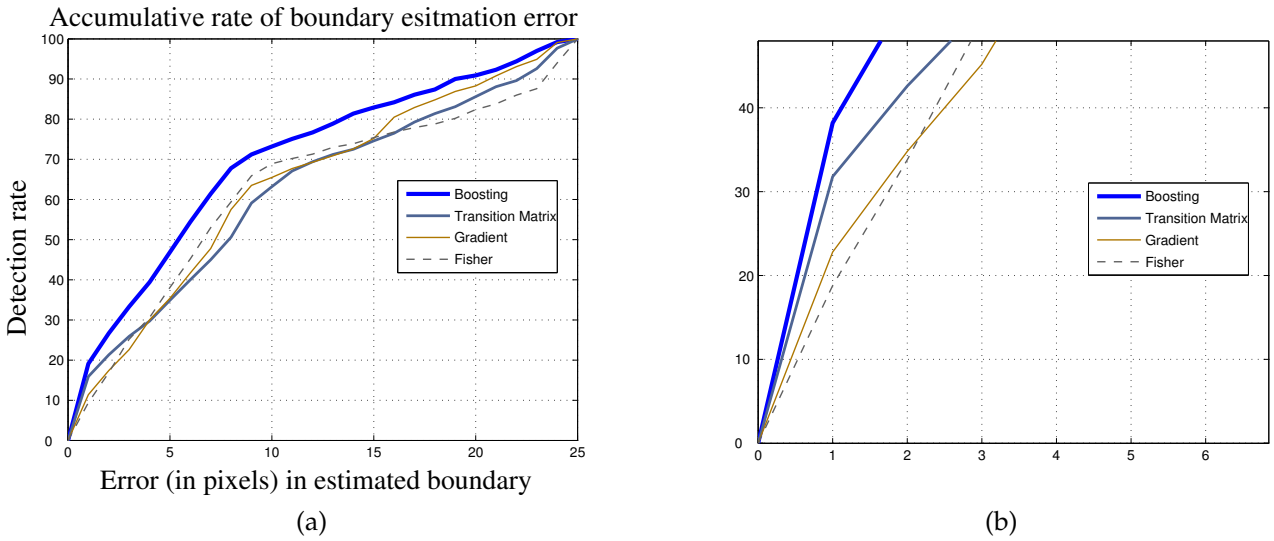


Fig. 7. a) Accumulative histogram of error in pixel for different methods. b) Magnified view of (a) in the region corresponding to small errors. The classification-based method has a higher percentage of detected points within any given error bound from the true boundary position.

system increases the performance by a factor of 20. The processing time of the optimized algorithm varies linearly with the number of samples on the model, and the length of the pixel sequences on each sample point. The number of weak learners used in the conditional probability classifier has negligible effect on the speed because each weak learner involves only 8 memory accesses to the integral array corresponding to the searchribbon and one logical comparison and three additions. However, according to our experiments, depending on the complexity of the sequence, a low number of classifiers (e.g. 4 to 10) is enough for reliable contour tracking.

Fig. 8 shows several frames of the tracking results in presence of clutter on a sequence of 600 frames with large and fast camera motions. We use a modified version of the RANSAC algorithm to fit the model to the probability distribution as explained in detail in [13].

Smoothness constraints can also be readily enforced on the boundary. We use a Hidden Markov Model as explained in [13] for that purpose. This is demonstrated by tracking the upper body deformation. The user defines an initial path by clicking on some points around the body outline. These points are used to fit a spline curve which serves as the starting guess. The outline is then obtained by minimizing the HMM energy over the probabilities given by the classifier. Several frames of the tracking results are shown in Fig. 9. These silhouettes can be applied to a 3–D implicit surface model [18] to track 3–D deformation of the mesh as shown in the last row of Fig. 9.

Tracking results using all the discussed line search methods combined with RANSAC robust model fitting on a short sequence are shown in Fig. 10. The gradient and Fisher's metric-based models fail on object's texture as shown in the first and second row of Fig. 10 respectively. Fisher's criterion works well only when the assumption of two classes with normal distribution is valid around the object boundary. The third row in Fig. 10 shows the tracking results using the Bayesian non-parametric boundary detection method. Correct estimation of texture distribution relies on a relatively long searchlines in order for the transition matrices to converge reliably.

Finally, the classification technique of texture boundary detection described in this paper reduces the length of the search line required to almost half (60 pixels) and yields good tracking results as shown in the last row of Fig. 10.

The classifier with a small set of weak learners can be used for interactive texture segmentation. Examples shown in Fig. 11 use 4 weak learners to calculate the conditional texture boundary probability. The user provided initial curves are shown by the thin lines and the calculated outline are marked by the thick curves, further details on interactive texture segmentation can be found in [13].

6 Conclusion

We have proposed an innovative machine learning technique to generate a probabilistic texture transition model given a searchribbon and a trained classifier. This approach draws together the speed of gradient-based and the robustness of texture segmentation algorithms for the purpose of fast and robust object contour extraction. The classifier is composed of a small set weak learners. The total classifier score is the weighted sum of weak learner responses where training and weights are obtained using the Adaboost algorithm. The posterior probability of texture boundary given the classification score is then

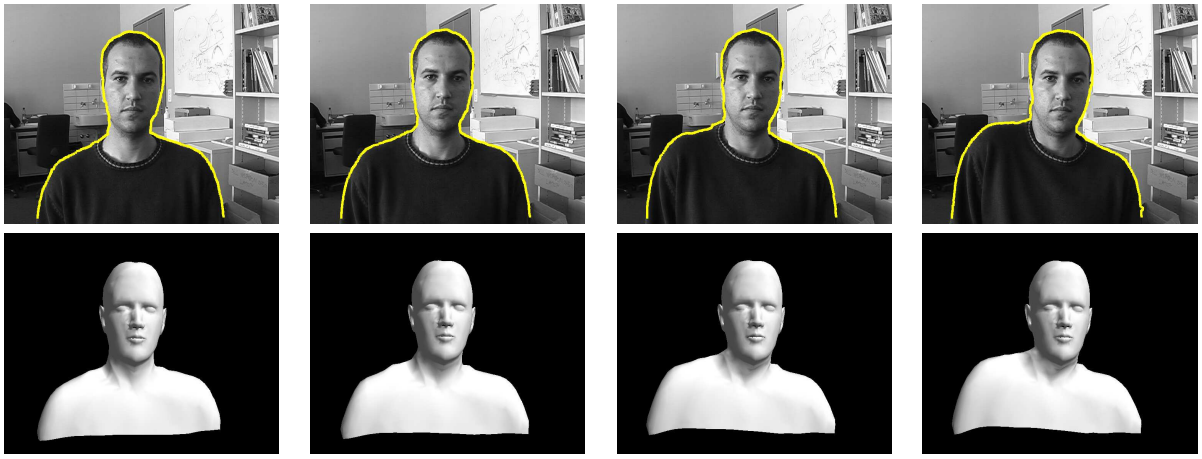


Fig. 9. Tracking of deforming body outlines. The last row shows the 3–D Tracking of deforming body outlines based on the detected outlines.

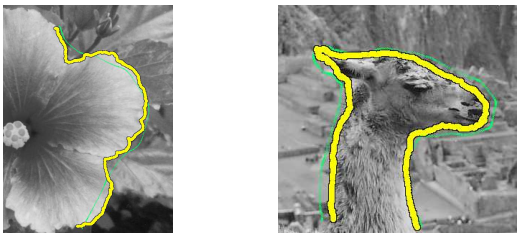


Fig. 11. Interactive segmentation using trained classifiers.

formulated using the Bayes rule with Gaussian priors for the classification score.

The weak learners compare as feature the mean of intensity values on both sides of the classification window. We have used other features such as Fourier coefficients and cooccurrence matrix features as weak learners in this context [13], but experiments show that the intensity features are optimal both in terms of performance efficiency and speed.

Different methods for texture boundary detection have been analyzed through various qualitative and quantitative assessments. The results show that the speed-optimized classification method outperforms Bayesian, gradient- and Fisher metric-based methods in terms of error rate on a dataset composed on random patches from natural images. We have also demonstrated the effectiveness of our method for tracking and interactive texture segmentation in particular in combination with constraints such as rigid model or smoothness of object's outlines as described in [13]. These constraints are enforced by using a RANSAC-based method to find a pose that maximizes the probability of contour points or in terms of an HMM energy minimization which regularizes the probability distribution over the search region to obtain continuous outlines.

The weak learners are designed to detect texture transitions along the window direction. Therefore, this scheme requires that the searchribbons run perpendicular to the estimated boundary directions so that the classification window passes through the actual texture boundary. This is conventional in line search algorithms [2] and has reliable convergence properties as long as there exists an estimate of the boundary orientation. Therefore there is no need for application of multi-directional filters and effectively the texture can be locally modeled as an isotropic pattern. Finally it should be noted that invariance to texture scale can be achieved by introducing a pyramidal texture sampling scheme in the database construction stage. A multiscale training database has successfully been used in tracking a known target in [13]. Further systematic analysis of texture resolution and the effects of the classification window size in the context of tracking might lead to an automatic algorithm for adjusting the classification window size and can be considered as an extension to this work.

REFERENCES

[1] C. Harris, *Tracking with Rigid Objects*, MIT Press, 1992.

[2] T. Drummond and R. Cipolla, "Real-time visual tracking of complex structures," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 7, pp. 932–946, july 2002.

[3] 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*, 2004, vol. 1, pp. 428–441.

[4] C. Rother, V. Kolmogorov, and A. Blake, ""GrabCut": Interactive Foreground Extraction using Iterated Graph Cuts," *ACM Transactions on Graphics*, vol. 23, no. 3, pp. 309–314, 2004.

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

[6] Y. Boykov and M. Jolly, "Interactive graph cuts for optimal boundary and region segmentation of objects in n-d images," in *International Conference on Computer Vision*, Vancouver, Canada, July 2001, pp. Vol I:105–112.

[7] M. Rivera and J. C. Gee, "Two–level mrf models for image restoration and segmentation," in *British Machine Vision Conference*, Kingston University, England, 2004, pp. 809–818.

Fig. 10. Comparison between different contour tracking algorithms. First row: Tracking results using an edge-based tracker. Second row: Tracking results using Fisher discriminant function. Third row: Tracking results using Bayesian texture boundary detection and fourth row: Tracking results using the classification-based method.

[8] C. H. Lampert, M. B. Blaschko, and T. Hofmann, "Beyond Sliding Windows: Object Localization by Efficient Subwindow Search," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2008, to appear.

[9] D. Martin, C. Fowlkes, and J Malik, "Learning to detect natural image boundaries using local brightn ess, color and texture cues," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 5, 2004.

[10] P. Dollár, Z. Tu, and S. Belongie, "Supervised learning of edges and object boundaries," in *CVPR*, June 2006.

[11] A. Shahrokni, F. Fleuret, and P. Fua, "Classifier-based Contour Tracking for Rigid and Deformable Objects," in *British Machine Vision Conference*, Oxford, UK, 2005.

[12] Y. Freund and R.E. Schapire, "Experiments with a New Boosting Algorithm," in *International Conference on Machine Learning*. 1996, pp. 148–156, Morgan Kaufmann.

[13] A. Shahrokni, *Probabilistic modeling of texture transition for fast tracking and delineation*, Ph.D. thesis, EPFL, 1015 Lausanne, 2005, Thesis, no 3377.

[14] P. Simard, L. Bottou, P. Haffner, and Y. LeCun, "A fast convolution algorithm for neural networks and signal processing," in *Advances in Neural Information Processing Systems 11*. 1999, MIT Press.

[15] G. Rätsch, T. Onoda, and K. R. Müller, "Regularizing adaboost," in *Neural Information Processing Systems*. 1998, pp. 564–570, MIT Press.

[16] P. Langley, W. Iba, and K. Thompson, "An analysis of bayesian classifiers," in *Proceedings of AAAI-92*, 1992, pp. 223–228.

[17] Robert Kaucic and Andrew Blake, "Accurate, real-time, unadorned lip tracking," in *International Conference on Computer Vision*, 1998, pp. 370–375.

[18] S. Ilić and P. Fua, "Implicit Meshes for Surface Reconstruction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 28 , no. 2, 2006.