Tree-based Classifiers for Bilayer Video Segmentation

Pei Yin[1] Antonio Criminisi[2] John Winn[2] Irfan Essa[1]
[1]School of Interactive Computing, Georgia Institute of Technology, Atlanta, GA 30332, USA
[2]Microsoft Research Ltd., Cambridge, CB3 0FB, United Kingdom

Abstract

This paper presents an algorithm for the automatic segmentation of monocular videos into foreground and background layers. Correct segmentations are produced even in the presence of large background motion with nearly stationary foreground. There are three key contributions. The first is the introduction of a novel motion representation, "motons", inspired by research in object recognition. Second, we propose learning the segmentation likelihood from the spatial context of motion. The learning is efficiently performed by Random Forests. The third contribution is a general taxonomy of tree-based classifiers, which facilitates theoretical and experimental comparisons of several known classification algorithms, as well as spawning new ones.

Diverse visual cues such as motion, motion context, colour, contrast and spatial priors are fused together by means of a Conditional Random Field (CRF) model. Segmentation is then achieved by binary min-cut. Our algorithm requires no initialization. Experiments on many video-chat type sequences demonstrate the effectiveness of our algorithm in a variety of scenes. The segmentation results are comparable to those obtained by stereo systems.

1. Introduction and related work

This paper addresses the problem of extracting a foreground layer from *monocular* video. Applications for the proposed technology include background substitution, compression, adaptive bit-rate video transmission and tracking. These applications require: (i) robust segmentation to survive from strong distracting events such as people moving in the background, camera shake or illumination change; (ii) efficient separation to attain live streaming speed. This paper focuses on the common scenario of video chat.

Recent research in this area has produced compelling, real-time algorithms, either based on stereo [8] or motion [5]. Other algorithms require initialization in the form of a "clean" image of the background [20].

Stereo-based segmentation [8] seems to achieve the most robust results. In fact, background objects are correctly separated from foreground, independently from their motion/stasis characteristics. This paper aims at achieving a similar behaviour *monocularly* (*cf.* fig. 1).

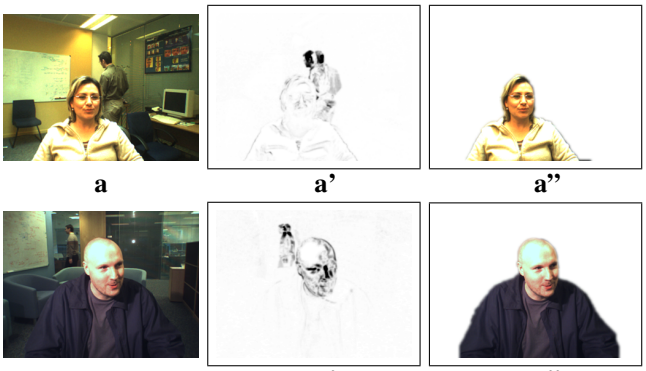

Figure 1. **Achieving robust layer extraction monocularly. (a,b)** Original frames from the "IU" and "JM" sequences from [8], respectively. **(a',b')** Temporal derivatives (dark indicates large values). The foreground person is nearly stationary while the background one is moving. In this case, background subtraction techniques or conventional motion-based algorithms would tend to classify the background person erroneously as foreground. Furthermore, inaccurate classification may be produced in the textureless and motionless areas of the foreground. **(a",b")** Segmentation obtained by the proposed algorithm. Correct foreground/background separation has been achieved (the extracted foreground is shown in original colours).

The static background assumption of some monocular systems [20] makes segmentation prone to camera shake (*e.g.* for a webcam mounted on a laptop screen), changes in illumination and large objects moving in the background. The algorithm in [5] avoids the need for a clean background image. However, the segmentation still suffers in the presence of large background motion. Also, initialization is sometimes necessary in the form of global colour models.

The work in [24] has started an important line of research in using geometric models (*e.g.* planar motion) for the segmentation of optical flow fields. However, in video-chat data the foreground motion cannot be described well by such rigid models. Furthermore, we wish to avoid the complexities associated with optical flow computation.

The algorithm proposed in this paper exploits motion *and* its spatial context as a powerful cue for layer separation. The correct level of geometric rigidity is automatically learnt from training data. Our algorithm benefits from novel

a **b** **c**

Figure 2. **Ground-truth layer labelling in video frames**. **(a)** A frame from a monocular training sequence. In this video both the closer and the farther persons are moving. **(b)** *Depth*-based layer labelling (white for foreground, black for background and gray for uncertain), as used in [8]. Here only the closest person is labelled as foreground. **(c)** *Motion*-based layer labelling, as used in [5]. Both moving objects are labelled as foreground. In this paper we *only* use depth-based labelling. This encourages our monocular system to learn to "imitate" stereo.

motion features, called *motons*. Motons (related to *textons*) were inspired by recent research in motion modeling [5], and object and material recognition [13, 15, 19, 22, 25]. Motons are then combined with *shape-filters* [19] to model long-range spatial correlations (shape). These new features prove useful at capturing visual context and filling-in missing, textureless or motionless regions.

Fused motion-shape cues are discriminately selected by supervised learning. Key to our technique is a classifier trained on *depth-defined* layer labels like those used in stereo [8], as opposed to motion-defined layer labels like in [5] (compare fig. 2b and fig. 2c). Thus, the classifier is forced to combine other available cues accordingly to induce depth in the absence of stereo, while maintaining generalization.

A general taxonomy of classifiers is described which enables us to interpret common algorithms such as AdaBoost, Decision Trees, Random Forests and Cascaded Boosting as variants of a single tree-based classifier. In turn, this allows us to compare fairly the different algorithms and select the most efficient or accurate for the application at hand.

Pixel-wise segmentation is finally achieved by fusing motion-shape, colour and contrast with local smoothness prior in a Conditional Random Field model [10, 11]. The final binary segmentation is achieved through min-cut [3].

The result is a segmentation algorithm which is efficient, robust to distracting events and requires no initialization.

2. Motons and shape filters

This section describes the motion-shape features used in our segmentation algorithm. We build upon the motion-vs-stasis likelihood model of [5], and combine it with concepts borrowed from recent literature in object recognition [19]. This leads to a powerful set of features which simultaneously capture motion and its long-range spatial context.

Notation. Given an input sequence of images, a frame is represented as an array $\mathbf{z} = (z_1, z_2, \cdots, z_n, \cdots, z_N)$ of pixels in YUV color space, indexed by the pixel position n. A frame at time t is denoted $\mathbf{z^t}$. Temporal derivatives

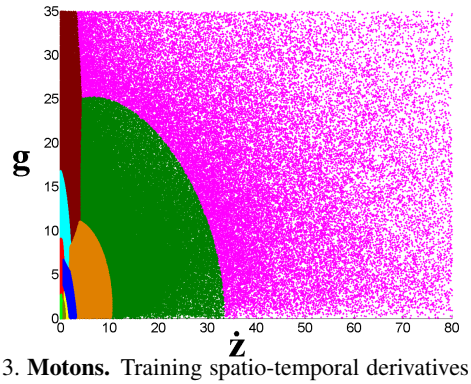

Figure 3. **Motons.** Training spatio-temporal derivatives clustered into 10 clusters (motons). Different colours for different clusters.

are denoted $\dot{\mathbf{z}} = (\dot{z_1}, \dot{z_2}, \cdots, \dot{z_n}, \cdots, \dot{z_N})$ and at each time t, are computed as $\dot{z_n^t} = \mid G(z_n^t) - G(z_n^{t-1}) \mid$ with $G(\cdot)$ a Gaussian kernel at the scale of σ_t pixels. Spatial gradients $\mathbf{g} = (g_1, g_2, \cdots, g_n, \cdots, g_N)$ where $g_n = \mid \nabla z_n \mid$, are computed by convolving the images with DoG kernels of width σ_s. Here we use $\sigma_s = \sigma_t = 0.8$, approximating a Nyquist sampling filter. Spatio-temporal derivatives are computed on the Y channel only. Motion observables are $\mathbf{O_m} = (\mathbf{g}, \dot{\mathbf{z}})$. The segmentation task is to infer the binary label $x_n \in \{Fg, Bg\}$ from observed data.

Motons. Here we follow a procedure similar to that for constructing textons [13]. First, motion features $\mathbf{O_m}$ are computed for all training pixels. Those 2D vectors are then clustered into M clusters via Expectation Maximization. The M resulting cluster centres are called *motons*. An example with $M = 10$ motons is shown in fig. 3. This operation may be interpreted as building a vocabulary of motion-based visual words. Our visual words capture information about motion and "edgeness" of image pixels, rather than their texture content as in textons.

Clustering (i) enables efficient indexing of the joint $(g, \dot{z})$ space while maintaining useful correlation between g and $\dot{z}$, and (ii) reduces sensitivity to noise. A dictionary size of just 10 motons has proven sufficient. Also, our moton representation is shown to yield less segmentation errors than using $\mathbf{O_m}$ directly.

In [5], it is observed that strong edges with low temporal derivatives usually correspond to background regions, while strong edges with high temporal derivatives are likely to be foreground. Textureless regions tend to have their log likelihood ratio close to zero due to uncertainty. Those stasis/motion discrimination properties are retained by our quantized representation; which is not yet sufficient for separating moving background from moving foreground.

Given a dictionary of motons, each pixel in a new image can be assigned to its closest moton by Maximum Likelihood (ML). Therefore, each pixel can now be replaced by an index into our small visual dictionary [25]. An example of the resulting moton map is shown colour coded in fig. 4b.

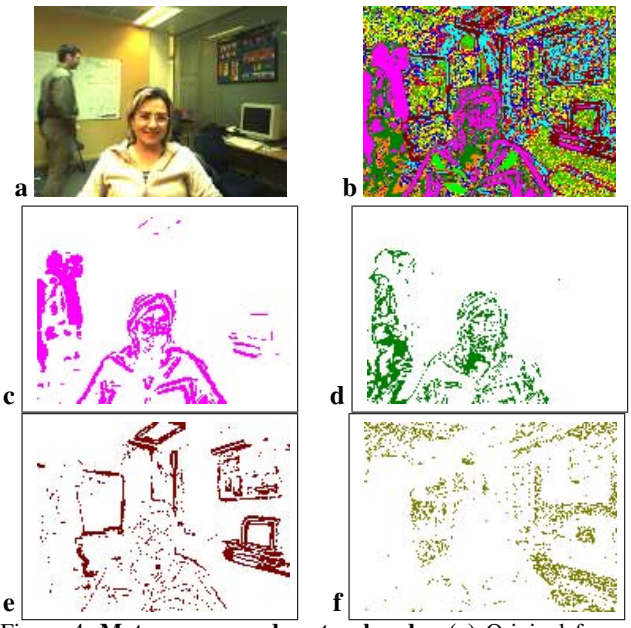

Figure 4. **Moton maps and moton bands. (a)** Original frame from the "IU" sequence. **(b)** Corresponding moton map with $M = 10$ motons. Same colour corresponds to same moton. **(c)** A moton band showing all the pixels associated to a "moving-edge" moton. **(d)** Pixels associated to a moving, "weak-texture" moton. **(e)** Pixels associated to a "stationary-edge" moton. **(f)** Pixels associated to a stationary, "weak-texture" moton.

Then, a moton map can be decomposed in its M component bands, namely "moton bands". Thus we have M moton bands $I^k, k = 1, \ldots, M$ for each video frame $\mathbf{z}$. Each I^k is a binary image, with $I^k(n)$ indicating whether the n^{th} pixel has been assigned the k^{th} moton or not. Fig. 4c-f shows some example moton bands.

Shape filters. In video-chat type sequences the foreground object (usually a person) moves non-rigidly and yet in a structured fashion. This section shows how to capture the spatial context of motion adaptively.

Similar to [19] a shape filter is defined as a moton-rectangle pair (k, r), with k indexing in the dictionary of motons and r indexing a rectangular mask whose four corners are chosen within a detection window (bounding box) about the size of the video frame (fig. 5). A whole set of d shape filters $\mathcal{S} = \{(k_i, r_i)\}, i = 1, \ldots, d$ is then defined by randomly selecting moton/rectangle pairs (see details later). For each pixel position n, the associated feature ψ_n is computed as follows. Given the moton k we center the detection window at n and count the number of pixels in I^k which fall in the offset rectangle mask r. This count is denoted $v_n(k, r)$ (see fig. 5). The feature value $\psi_n(i, j)$ is simply obtained by subtracting moton counts collected for the two shape filters (k_i, r_i) and (k_j, r_j), *i.e.* $\psi_n(i, j) = v_n(k_i, r_i) - v_n(k_j, r_j)$. The i and j indices are selected randomly ($\leq d$). The feature ψ_n can be computed efficiently by integral image processing [23] for every moton band I^k. Our shape filters may be interpreted as a generalization of the features used in [23].

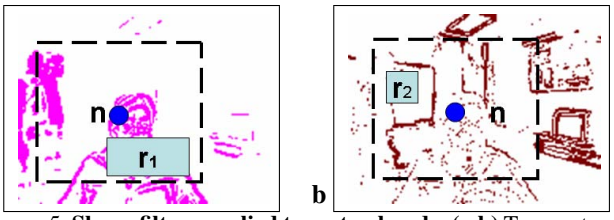

Figure 5. **Shape filters applied to moton bands. (a,b)** Two moton bands with rectangular masks r_1 and r_2 (centred in the same pixel, n) superimposed. Given n and the shape filter (k, r), $v_n(k, r)$ counts the number of pixels associated to k within r; see text.

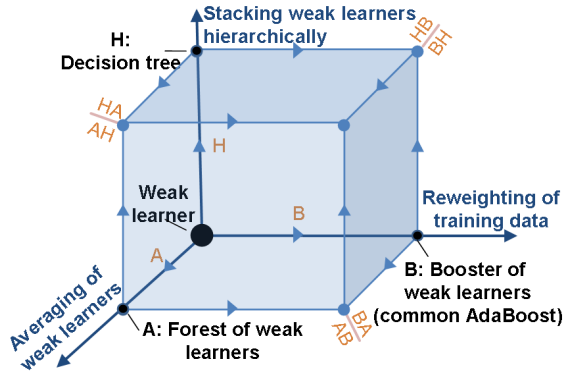

Figure 6. **The tree cube taxonomy of classifiers** captures many classification algorithms in a single structure. See text for details.

Next, our features are discriminatively selected and combined by a classifier for the estimation of our Fg/Bg unary potentials (see Section 5). The following section presents a taxonomy of tree-based classifiers and shows how common classifiers may be interpreted as instances of the same general algorithm. Such taxonomy then helps us to select the classifier that performs best in our application.

3. The Tree Cube taxonomy

Common classification algorithms such as Decision Trees [16], Boosting [7] and Random Forests [1, 4] share the fact that they build "strong" classifiers from a combination of "weak" classifiers (learners), often just decision stumps. The main difference between these algorithms is the way the weak learners are combined. This section presents a useful framework for constructing strong classifiers by combining weak classifiers in different ways.

The three most common ways of combining weak classifiers are: i) hierarchically (`H`), ii) by averaging (`A`) or iii) via boosting (`B`). In Fig. 6 the origin represents the weak learner (*e.g.* a decision stump), and the axes `H`, `A`, `B` represent those three basic combining "moves".

- The `H` move hierarchically combines weak classifiers into decision trees. During training a new weak classifier is iteratively created and attached to a leaf node where needed based on information gain. It can be

Path	Classification algorithm
A	voting by committee [2]
B	booster of stumps
H	decision tree
HA	forest of trees (decision forest)
HB	booster of trees
AH	tree of forests (of stumps)
AB	booster of forests (of stumps)
BA	committee of boosters
BH	tree of boosters
HAB	booster of forests of trees
HBA	committee of boosters of trees
BAH	tree of committee of boosters (of stumps)
BHA	committee of trees of boosters
ABH	tree of boosters of forests (of stumps)
AHB	booster of trees of forests (of stumps)

Table 1. **Tree Cube classifiers.** Fifteen of the many possible classification algorithms encoded in the taxonomy of fig. 6.

shown that the H move reduces classification bias [2].

- The B move, instead, linearly combines weak classifiers. After the insertion of each weak classifier, the training data is reweighted/resampled [16]. Classification of one instance involves evaluating *all* the tests in the strong classifier. The B move includes AdaBoost and Gentle Boost. Boosting reduces the empirical error bound by perturbing the training data [7].
- The A move creates strong classifiers by averaging the results of many weak classifiers. Note that the weak classifiers added by the A move face the same problem, while those sequentially added by the H and B moves face different problems/distribution. Thus, the main computational advantage is that each weak classifier can be trained independently from each other and in parallel. The A move gives rise to Random Forests when the weak classifier is a random tree. The averaging move reduces classification variance [2].

Paths along the edges of the cube in fig. 6 correspond to different combinations of weak classifiers and thus different strong classifiers. Restricting each of the three basic moves to be used only once produces three order-1 algorithms (excluding the base learner itself), six order-2 and six order-3 algorithms, as listed in table 1. Many known algorithms are conveniently mapped into paths through the tree-cube. For example: Boosting (B), Decision Trees (H), Booster of Trees (HB) and Random Forests (HA). Also, note that the widely used Attention Cascade [23] can be interpreted as a one-sided tree of boosters. The tree-cube taxonomy also enables us to explore new algorithms (*e.g.*HAB) and compare them to other algorithms of the *same* order (*e.g.*BHA).

Next, we explore which classifier performs best for the segmentation of video-chat sequences. Following the tree-cube strategy we compare two popular second-order models: Random Forests of trees (RF) and Booster of Trees (BT). As a sanity check we also compute the results of conventional booster of stumps (GB).

- Initialize weights of N training points $w_n = 1/N$, $n = 1, 2, \ldots, N$ and initialize strong classifier $F(n) = 0$.
- Repeat for $\ell = 1, 2, ...L$
  1. fit the regression function $h_\ell(.)$ by minimizing $\sum_{n=1}^{N} w_n (h_\ell(n) - y_n)^2$, with $y_n \in \{-1, +1\}$ the ground-truth label of pixel n.
  2. update strong classifier $F(n) \longleftarrow F(n) + h_\ell(n)$
  3. update training weights $w_n \longleftarrow w_n * e^{y_n h_\ell(n)}$, and re-normalize $\sum_{n=1}^{N} w_n = 1$
- Strong classifier is $F(n) = \sum_{\ell=1}^{L} h_\ell(n)$

Table 2. **Training Gentle Boost**.

4. Random Forests vs Booster of Trees

The base weak classifier used in this paper is the widely used decision stump. A decision stump applied to the n^{th} pixel takes the form $h(n) = a\delta(\psi_n(i,j) > \theta) + b$, where $\delta(\cdot)$ is a 0-1 indicator function, $\psi_n(i,j)$ is the shape filter response for the i^{th} and j^{th} shape filters (as described in section 2). Positive values of the real valued $h(n)$ output indicate that pixel n belongs to foreground and vice-versa. Now we look at different ways of combining stumps into strong classifiers. We begin with the H move.

Decision tree. When training a tree, at each iteration a new decision stump is fitted by finding the θ, a and b values which yield either the least square error [19] or the maximum entropy gain, as described later. During testing, the output $F(n)$ of a tree classifier is the output of the leaf node. Next we analyze the details of the B combination move.

Gentle Boost. Out of the many versions of boosting, here, we focus on the Gentle Boost algorithm [7] because of its robustness properties [14, 21]. For the n^{th} pixel, a strong classifier $F(n)$ is constructed as a linear combination of stumps $F(n) = \sum_{\ell=1}^{L} h_\ell(n)$. For completeness the full algorithm is summarized in table 2. Here Gentle Boost is applied both to stumps (B in fig. 6) and decision trees (HB in fig. 6). We abbreviate the first algorithm as GB and the second as BT. We also combine the stumps into Random Forests (the HA path in fig. 6)

Random Forests. A forest is made of many trees, and its output $F(n)$ is the average of the output of all trees (the A move). A Random Forests (denoted RF) is an ensemble of decision trees trained with *random features*. In this case, each tree is trained by adding new stumps in the leaf nodes where maximum information gain can be achieved. However, unlike boosting, the training data is not reweighted for different trees. RF has been applied to recognition problem in vision, *e.g.* OCR [1] and keypoint recognition [12].

Randomization. GB, BT and RF are trained effectively by optimizing each stump only on a few (1000 in our implementation) randomly selected shape filter features. This re-

duces the statistical dependence between weak learners [1], and it provides increased efficiency without significantly affecting the accuracy [19, 6].

In all three algorithms the classification confidence is computed by softmax transformation [7, 19] $\widetilde{P}(x_n = Fg|\mathbf{O_m}) = \frac{\exp(F(n))}{1+\exp(F(n))}$. Next we describe how those motion-shape based classifiers are combined with colour, contrast and spatial smoothness to obtain binary segmentation.

5. Layer segmentation

Segmentation is cast as an energy minimization problem where the energy to be minimized is similar to the one in [8], with the only difference that the unary potential of stereo match U^M is replaced by our motion-shape unary

$$U^{MS}(\mathbf{O_m}, x; \Theta) = \sum_{n=1}^{N} log(\widetilde{P}(x_n|\mathbf{O_m})). \quad (1)$$

The CRF energy is as follows:

$$E(\mathbf{O_m}, z, x; \Theta) = \quad (2)$$
$$\gamma_{MS} U^{MS}(\mathbf{O_m}, x; \Theta) + \gamma_C U^C(z, x; \Theta) + V(z, x; \Theta),$$

Similar to [8], U^C is the colour potential (a combination of global and pixel-wise contributions) and V is the widely-used contrast-sensitive spatial smoothness term. Model parameters are incorporated in Θ and relative weights γ_{MS} and γ_C are optimized discriminatively from training data. The final segmentation is inferred by binary min-cut. No complex temporal model [5] is used here. Finally, background edge abating [20] could also be exploited here if a pixel-wise background model were learned on the fly.

6. Experimental Results

Our new motion-shape likelihood, eq.(1) is validated in section 6.1; while the segmentation accuracy achieved by the complete CRF model, eq.(2) is assessed in section 6.2.

We have collected a database of 28 monocular sequences [1] which we have then pixel-wise labelled every fifth or tenth frame into foreground, background and uncertain (in the difficult, mixed-pixel regions), according to their distance from the camera (fig. 2b). In our experiments, 46 labelled frames from 7 clips are chosen randomly for training and 2 clips for validation. All the 426 labelled frames of the remaining 19 clips are used for testing.

6.1. Comparing unary classifiers

GB and BT were trained by minimizing the empirical loss as required by boosting, while RF were trained by maximizing the information gain as required by C4.5 [16]. All three algorithms share the same set of motons. Their testing errors are then measured and compared with one another.

[1] http://research.microsoft.com/vision/cambridge/i2i/DSWeb.htm
For the six sequences that are captured in stereo setting, only the left-camera videos are used here, and *only* for testing.

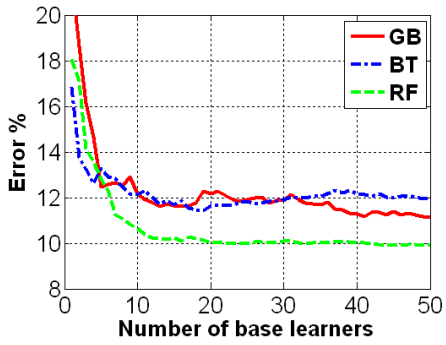

Figure 7. **Comparing accuracy of classifiers.** Testing unary accuracy with respect to the complexity of the ensembles in one trial. Five trials have been run and RF has consistently outperformed both the GB and BT algorithms.

Stereo	Monocular			
Stereo [8]	Motion [5]	Learned motion-shape		
		RF	GB	BT
5.55%	23.66%	**9.93%**	11.76%	11.42%

Table 3. **Comparison with state of the art**. Comparison between the proposed classifiers and existing stereo and monocular unaries.

The ensemble size for GB, BT and RF are set to 195 stumps, 19 trees and 47 trees respectively to maximize the accuracy on the validation set. The trees in BT and RF have 50 nodes, which is optimal for BT on validation set.

Next, we evaluate the unary classification accuracy with the 426 testing frames. Pixel classification into foreground and background is carried out by thresholding at $F(n) = 0$[2]. The error rate is averaged over $426 * 160 * 120 = 8.18 * 10^6$ pixels, and the processing rate (frame per second) is measured with our non-optimized Matlab code.

Accuracy of unary potentials. Fig. 7 compares the classification accuracy of GB, BT and RF when varying the number of base learners. Assuming balanced trees, evaluating one binary decision tree with 50 nodes roughly equals evaluating $\log_2 50 \approx 6$ stumps (depends on the balance of the tree). Thus we have scaled down the curve of GB along the x-axis by a factor 6, so that the expected number of stumps evaluated are the same for GB, BT and RF. Random Forests consistently yield the lowest testing errors.

From the GB curve in fig. 7 we can also see that there are not many pixels which can be correctly classified with a few stumps, therefore, we wouldn't expect a cascade to give a significant speedup in our application.

Table 3 compares the accuracy of our motion-shape potentials U^{MS} with the stereo potentials of [8][3] and the mo-

[2] The final segmentation results are significantly improved by integrating color, contrast, and spatial priors using the CRF model, shown next.

[3] Here, the accuracy of the stereo likelihood has been improved with respect to [8] by setting the log likelihood ratio to zero in low texture areas (uncertain for stereo). The pixel-wise stereo error rate would increase to 17.51% without such postprocessing. This further illustrates the importance of shape information in our bilayer segmentation application.

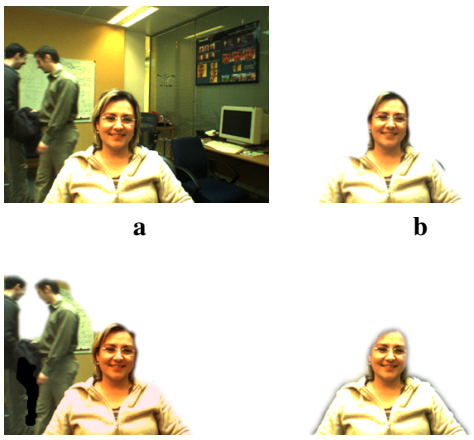

Figure 8. **Segmentation results on the "IU" test sequence**. **(a)** Original, **(b)** stereo-based segmentation from [8], **(c)** monocular segmentation from [5], where background motion pops into foreground, **(d)** monocular segmentation from the proposed algorithm.

tion potentials of the monocular system in [5]. Our motion-shape unary potentials lead to accuracy comparable to those of stereo and superior to the motion-based ones.

Accuracy vs efficiency. Table 4 compares the three classifiers both in terms of accuracy and speed. The first three columns report the lowest achieved classification error and the corresponding frame rate for each of the three algorithms at their "optimal" parameter setting according to validation. Having confirmed that RF produces the lowest errors we then evaluate the speed of RF when it is forced to produce the same error level as GB and BT (4^{th} column). In the last two columns, the ensemble of RF is forced to run at the same speed as GT or BT, and observe its classification error. In all cases RF outperforms the other two classifiers.

6.2. Assessing segmentation accuracy

This section analyzes segmentation results obtained by the full CRF model with U^{MS} estimated by RF.

Qualitative evaluation. Figure 8 compares the segmentation of the "IU" test sequence obtained by our algorithm with those in [5, 8]. In this sequence, two people walk behind the foreground person. Varying scene illumination constitutes a further source of difficulty. The motion based method in [5] classifies the background people as foreground (as it is designed to do). The proposed algorithm, instead, produces a segmentation similar to that of the stereo system in [8], where background motion is effectively ignored. Fig. 1, 9, 10, 11 provide more segmentation results.

Quantitative evaluation. Fig. 12 shows segmentation errors with respect to ground truth for four of our test sequences. The median error is around or below 1%. Median errors for 10 test sequences are also reported in table 5.

Automatic initialization. Fig. 13 illustrates how the system initializes itself. At the beginning the subject is stationary and the segmentation inaccurate. However, a small motion of the head is sufficient to achieve the correct segmentation (Fig. 13d). This "burn-in" effect may also be observed for a different test sequence in the error plot in fig. 12d.

Figure 10. **More segmentation results**. **(a, b)** Original frames from test sequence "56", where the picture on the TV set changes. **(a', b')** Corresponding segmentations. **(c-d')** More segmentation results on test sequence "43" and "50".

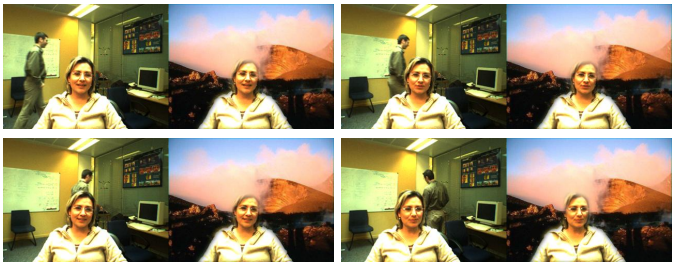

Figure 11. **Background substitution on the "IU" test sequence from [8]**. Original and corresponding segmented frames with background substitution. People moving behind the foreground person are correctly classified as background.

Test Sequence	41	43	50	51	54
Seg. Err. (%)	0.80	**0.02**	1.31	1.06	0.33
Test Sequence	56	58	60	IU [8]	JM [8]
Seg. Err. (%)	0.93	0.79	**6.33**	2.56	0.27

Table 5. **Segmentation errors for ten of the test sequences.**

The plot in fig 12a demonstrates how our algorithm can recover automatically from possible mistakes. In fact, in frames 60-85 of the "JM" sequence the subject leans very close to the camera and so the image looks very different from the training frames, and segmentation errors occur. The segmentation recovers promptly following this error.

Inaccurate segmentations. Fig. 14a',b' show examples of inaccurate segmentation. Under harsh lighting conditions unary potentials may not be very strong and thus the Ising smoothness term may force the segmentation to "cut through" shoulders and hair regions. Similar effects may be noticed in stationary frames. Noise in temporal derivatives also affects the results. This situation can be detected by monitoring the magnitude of motion, and enabling a temporal model such as that in [5] may help reduce the problem.

Algorithm	GB (best)	BT (best)	RF (best)	RF (same err as GB and BT)	RF (same speed as GB)	RF (same speed as BT)
Classif. error (%)	11.76	11.42	**9.93**	11.23	**9.93**	**10.11**
Speed (fps)	1.2	2.9	1.2	**7.7**	1.2	2.9

Table 4. **Comparing testing accuracy and efficiency** for GB, BT and RF in one of five testing trials. See text.

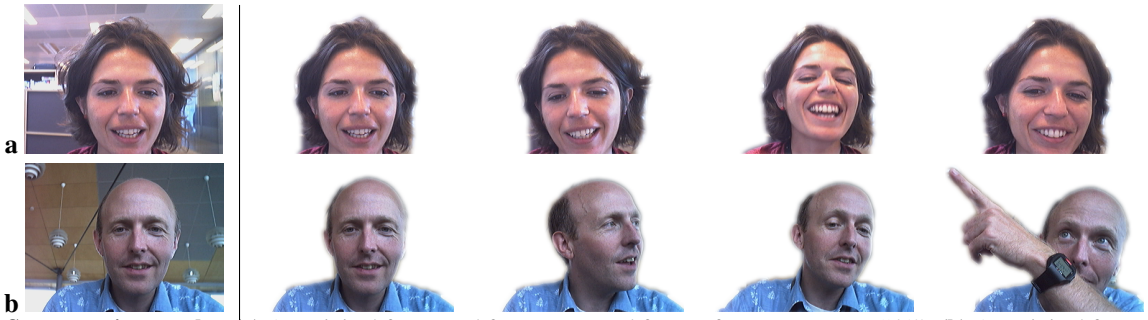

Figure 9. **Segmentation results**. **(a)** An original frame and four segmented frames for test sequence "41". **(b)** An original frame and four segmented frames for test sequence "54". Border matting [17] could be applied here to improve the hair regions. This paper is concerned with binary segmentation only.

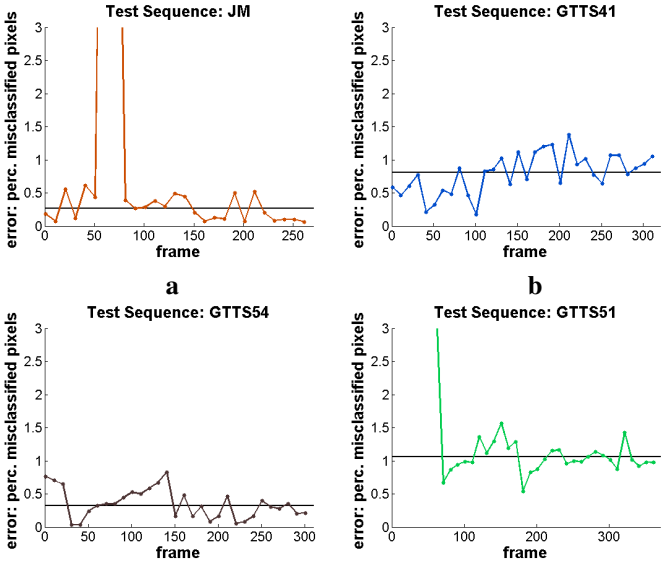

Figure 12. **Accuracy of segmentation**. **(a)** Percentage of misclassified pixels on the "JM" test sequence from [8]. Note how the system promptly recovers from possible mistakes. The median error (horizontal line) is well below 0.5%. **(b,c,d)** Percentage of misclassified pixels for the test sequences "41", "54", "51", respectively. **(d)** After an initial "burn-in" period the segmentation converges to a good accuracy level (around or below 1% misclassified pixels).

Figure 13. **Self-initialization**. **(a)** original frame from test sequences "58". Harsh lighting conditions make the segmentation problem challenging. **(b-d)** Segmentation for different frames in chronological order. After an initial "burn-in" period the algorithm converges to the correct Fg/Bg separation. A "clean" background image or other types of initialization were not necessary.

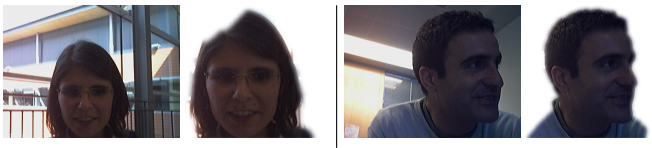

Figure 14. **Some inaccurate segmentation**. **(a)** A frame from test sequence "41". **(b)** A frame from test sequence "60". **(a',b')** corresponding segmentations. Bad scene illumination produces inaccurate segmentation in the hair region of (a') and in the shoulder region of (b'). See also the errors in table 5.

6.3. Discussion on bias and variance

For a classification algorithm, bias describes its modelling power while variance describes its stability [9]. Note that bias and variance are different than the mean error and the variance of error. The bias and variance decomposition of the RF, GB and BT algorithms (in table 6) helps us to understand their behaviour, and the nature of our task. This decomposition is computed from five trials on the testing set, and the results discussed below.

(1) BT and RF yield lower bias than GB.
The linear combination property of the `B` and `A` moves requires that the decision boundary of the classification task is additive in terms of the decision boundary of the weak learners *i.e.* the capacity of the base learning algorithm matches the complexity of the problem [7]. By moving along the `H` direction, bigger trees are constructed which are capable of modeling higher order interaction between variables. *e.g.* a decision stump only contains one feature, while the typical decision path of a 50-node binary tree

Method	Bias	Variance
RF (%)	10.20	0.39
GB (%)	10.31	1.48
BT (%)	9.95	2.09

Table 6. Bias/variance analysis. Bias and variance of RF, GB and BT

equals a conjunction term of 5-6 features. Therefore, (1) indicates that the segmentation task is complex, and its decision boundary is better approximated by deeper trees.

(2) BT has lower bias than RF.
This result confirms that boosting achieves additional reduction in bias by aggressively perturbing the training process to focus on the difficult samples [4]. However, this sometimes result in overfitting.

(3) RF has the lowest variance.
Boosting persistently increases the minimum margin (of a few incorrectly classified samples) at the potential cost of decreasing the average margin (over all training data) [26]. Therefore boosting does not generalize well in the presence of *label noise*[4]. In contrast, RF is quite robust to such kind of noise. In fact, the effect of a few mistakenly labelled samples is restricted to particular leaf nodes locally, without sacrificing the accuracy of other nodes or trees. Therefore, it is not surprising to see that overfitting makes the error of `B` higher than `A`. Similar phenomena have also been reported in [4].

7. Conclusion and Future Work

This paper has presented an algorithm for the efficient segmentation of foreground and background in monocular video sequences. Our algorithm is capable of inferring bilayer segmentation monocularly even in the presence of distracting background motion and without the need for manual initialization.

We have: (i) introduced novel visual features which capture motion and motion context efficiently; (ii) provided a general understanding of tree-based classifiers, which in turn (iii) has helped us select an efficient and accurate classifier in the form of Random Forests.

Experiments and the related analysis on existing test data and our own database confirm accurate and robust segmentation. Similar to stereo-based system, our approach manages to separate foreground and background even when distracting background motion occurs.

Next, we would like to apply our classifier taxonomy to other domains and applications to assess the merits of different types of classification algorithms in various situations. Further comparisons between tree-based classifiers and Kernel Machines [18] (*e.g.* SVM) are also necessary.

[4] unavoidable in segmentation problems

References

[1] Y. Amit and D. Geman. Shape quantization and recognition with randomized trees. *Neural Comput.*, 9(7):1545–1588, 1997.

[2] C. Bishop. *Neural Netowrks for Pattern Recognition*. Oxford University Press, 1995.

[3] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate engergy minimization via graph cuts. *IEEE PAMI*, pages 1222–1239, 2001.

[4] L. Breiman. Random forests. *UC Berkeley TR567*, 1999.

[5] A. Criminisi, G. Cross, A. Blake, and V. Kolmogorov. Bilayer segmentation of live video. In *IEEE CVPR*, pages 53–60, 2006.

[6] T. Deselaers, A. Criminisi, J. Winn, and A. Agarwal. Incorporating on-demand stereo for real time recognition. In *CVPR*, 2007.

[7] J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: a statistical view of boosting. *Annals of statistics*, 38:337–374, 2000.

[8] V. Kolmogorov, A. Criminisi, A. Blake, G. Cross, and C. Rother. Bilayer segmentation of binocular stereo video. In *IEEE CVPR*, pages 407–414, 2005.

[9] E. Kong and T. Dietterich. Error-correcting output coding corrects bias and variance. In *Proc. of ICML*, pages 313–321, 1995.

[10] S. Kumar and M. Hebert. Discriminative random fields: A discriminative framework for contextual interaction in classification. In *Proc. of IEEE ICCV*, pages 1150–1157, 2003.

[11] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilitstic models for segmentaing and labeling sequence data. In *18th Int. Conf. on Machine Learning*, pages 282–289, 2001.

[12] V. Lepetit, P. Lagger, and P. Fua. Randomized trees for real-time keypoint recognition. In *Conference on Computer Vision and Pattern Recognition, San Diego, CA*, June 2005.

[13] T. Leung and J. Malik. Representing and recognizing the visual appearance of materials using three-dimensional textons. *IJCV*, 43(1):29–44, 2001.

[14] R. Lienhart, A. Kuranov, and V. Pisarevsky. Empirical analysis of detection cascades of boosted classifiers for rapid object detection. In *DAGM*, pages 297–304, 2003.

[15] F. Perronnin, G. Dance, C. Csurka, and M. Bressan. Adapted vocabularies for generic visual categorization. In *IEEE ECCV*, 2006.

[16] J. Quinlan. Bagging, Boosting, and C4.5. In *Proc.of National Conf. on Artificial Intelligence*, pages 725–730. AAAI Press, 1996.

[17] C. Rother, V. Kolmogorov, and A. Blake. "grabcut": interactive foreground extraction using iterated graph cuts. *ACM Trans. Graph.*, 23(3):309–314, 2004.

[18] B. Scholkopf. Statistical learning and kernel methods. *MSR-TR 2000-23*, 2000.

[19] J. Shotton, J. Winn, C. Rother, and A. Criminisi. Textonboost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In *ECCV*, 2006.

[20] J. Sun, W. Zhang, X. Tang, and H. Shum. Background cut. In *Proc. of ECCV*, pages 628–641, 2006.

[21] A. Torralba, K. Murphy, and W. Freeman. Sharing features: efficient boosting procedures for multiclass object detection. In *CVPR04*, pages 762–769, 2004.

[22] M. Varma and A. Zisserman. A statistical approach to texture classification from single images. *IJCV*, 62(1-2):61–81, 2005.

[23] P. Viola and M. Jones. Robust real-time object detection. *IJCV*, 57(2):137–154, 2004.

[24] J. Wang and E. Adelson. Representing moving images with layers. *IEEE Trans. Image Process*, 3(5):625–638, 1994.

[25] J. Winn, A. Criminisi, and T. Minka. Object categorization by learned universal visual dictionary. In *Proc. of ICCV*, pages 1800–1807, 2005.

[26] P. Yin, I. Essa, and J. M. Rehg. Segmental boosting algorithm for time-seris feature selection. *Tech. Report GIT-GVU-06-24*.