Towards Interactive Generation of "Ground-truth" in Background Subtraction from Partially Labeled Examples

Etienne Grossmann, Amit Kale and Christopher Jaynes
UK Center for Visualization and Virtual Environments
Lexington KY 40507
{etienne,amit,jaynes}@cs.uky.edu

Abstract

Ground truth segmentation of foreground and background is important for performance evaluation of existing techniques and can guide principled development of video analysis algorithms. Unfortunately, generating ground truth data is a cumbersome and incurs a high cost in human labor. In this paper, we propose an interactive method to produce foreground/background segmentation of video sequences captured by a stationary camera, that requires comparatively little human labor, while still producing high quality results. Given a sequence, the user indicates, with a few clicks in a GUI, a few rectangular regions that contain only foreground or background pixels. Adaboost then builds a classifier that combines the output of a set of weak classifiers. The resulting classifier is run on the remainder of the sequence. Based on the results and the accuracy requirements, the user can then select more example regions for training. This cycle of hand-labeling, training and automatic classification steps leads to a high-quality segmentation with little effort. Our experiments show promising results, raise new issues and provide some insight on possible improvements.

1 Introduction

Ground truth is very important in computer vision because of the central role it can play in the empirical analysis and development of algorithms. Labeled image databases, for example, are used to train image classifiers and detectors [20] or face recognition methods [13]. Ground truth also provides "perfect" data to test existing methods that were built on the premise that this data is available. For example, having exact silhouettes of walkers helps in the evaluation of gait recognition algorithms [15]. Perhaps the most common usage of ground truth is in quantitative performance evaluation, where it sets the gold standard, whether for ego-motion estimation [1],three-dimensional reconstruction [19], background subtraction [6, 16], or others tasks.

Unfortunately ground truth is usually hard to obtain. Labeling image databases or videos requires a person to view and annotate hundreds or thousands of images [7]. Generating the ground truth for background subtraction requires that the user hand-segments each image, a cumbersome task which may require two to thirty minutes per image [6].

We propose an off-line method to generate near-perfect foreground segmentation of video sequences requiring significantly less manpower than extensive hand-labeling. The approach utilizes the power of supervised learning [18] and off-the-shelf weak classifiers to lift the burden of labeling from the user. Given a sequence, the user indicates, with a few clicks in a GUI, a few rectangular regions that contain only foreground or background pixels. Adaboost then builds a classifier using the output from a set of weak classifiers applied to this data. The resulting classifier is run on the remainder of the sequence. Based on the results and the accuracy requirements of the ground truth data, the user can then select more example regions for training.

This cycle of *hand-labeling, learning and automatic classification rounds* will eventually lead to a satisfactory level of precision, as long as the automatic classifier is capable of achieving an arbitrary precision on the training data. Since we use Adaboost [18] to train the classifier, this condition is equivalent to Adaboost's "weak learner" condition.

In our case, the weak classifiers are ordinary image and video filters, as shown in Figure 1 (b-d) as well as post-processing operators applied to the boosted classifier. Our method requires a single parameter to be set viz. the desired classification error of Adaboost on the training dataset or the total number of training steps. In short, we use supervised learning and image processing tools to learn the distinction between foreground and background.

Several attempts have been made towards achieving semi-automatic generation of ground truth for background subtraction. Black et al. [3] used consistency in shape and appearance between successive frames is used to build approximate ground truth. Liu and Sarkar [12] used a hidden Markov model coupled with a eigen stance model to gener-

ate silhouettes for evaluation of gait recognition algorithms. It should be noted however that the quality of foregrounds obtained using such techniques which rely on such "tracking" between keyframes depends on how well the assumptions of spatio-temporal continuity hold and may lead to increasing (and unknown) errors in the result. Furthermore refinement of the computed foregrounds using these methods is not straightforward since it requires the user to manually label the foreground for frames where tracking fails. In contrast, we make minimal assumptions about the dynamic scene to obtain high quality foreground segmentation. Furthermore iterative refinement of the quality of segmentation is relatively straightforward to achieve. By utilizing an iterative boosting method, applied to each frame, the approach is free from tracking failures and robustness issues.

Recently there has been an upsurge in the use of supervised learning methods in computer vision. The main reason is that supervised learning methods e.g. Adaboost, decision trees, and neural networks combine simple decision rules (weak classifiers, stumps, neurons) to obtain classifiers that outperform ad-hoc methods [22]. Moreover the latter require more field-specific knowledge from the designer. Examples of recent uses of supervised learning approaches in computer vision include novel view generation [9, 8], face or pedestrian detection [22, 23].

The reader should note that, while these approaches solve existing computer vision problems, our method serves to alleviate the human effort required for accurate foreground segmentation. In this respect our work is related to that of Agarwala et al. [2] where human interaction is combined with energy minimization based curve tracking to produce rotoscope sequences, saving a huge amount of labor with respect to prior methods. Our contribution, then, will be apparent in the ease of high quality "ground truth" production for domains that typically require significant effort and will, hopefully, lead to more complete and commonplace use of ground-truth both in the development and analysis of vision algorithms. Finally, our work gives some insight on the background segmentation problem, as the composition of the boosted classifiers suggests an answer to the question: "what are the useful features in background subtraction in a particular sequence?"

2 Methodology

Having presented the principle of labeling, learning and validation rounds, we show how Adaboost is used to learn classifiers.

2.1 Supervised learning with Adaboost

We use Adaboost [10, 18] for many reasons, one of them being that its theoretical properties have been studied extensively [18, 11] and it has been observed to generalize well. Moreover, the algorithm itself requires a single parameter to be set, the number of training rounds T.

The training process in Adaboost results in a classifier

$$H(X) = \text{Sign}\left(\sum_{t=1}^{T} \alpha_t h_t(X)\right) \in \{-1, 1\}, \qquad (1)$$

where $X \in \mathcal{X}$ is the observed data used in classification, $h_t : \mathcal{X} \longrightarrow [-1, 1]$ is a "weak classifier" belonging to a class of functions $\mathcal{H}$ and $\alpha_t \in \mathbb{R}$ is a weighing coefficient. G returns $+1$ to indicate foreground and -1 to indicate background.

Adaboost requires that the weak classifiers perform "better than guessing." Mathematically, this means that there exists a positive ϵ (which does not need to be known), such that, given a sample of data $(X_1, y_1), ..., (X_N, y_N)$, where $y_n \in \{-1, 1\}$ represents the class (background and foreground, in our case) of input X_n, and given positive weights $D(1), \ldots, D(N)$ that sum to one, there exists a classifier $h \in \mathcal{H}$ such that its error

$$\sum_{n=1}^{N} D(n) [\![h(X_n) y_n < 0]\!]$$

is less than $1/2 - \epsilon$, where $[\![h(X_n) y_n < 0]\!]$ is 1 (resp. 0) if $h(X_n)y_n$ is negative or not, i.e if h wrongly (resp. correctly) predicts the class of X_n. If this assumption holds, then the classifier (1) built by Adaboost will have an error on the training data that decreases exponentially with T.

The input X may e.g. be the RGB values and location of a pixel, and may also include information on its spatial and temporal neighborhood. At most, X could include, beyond the pixel location (x, y) and RGB value, the whole image and perhaps the whole sequence too. What the set $\mathcal{X}$ is exactly is not important here because Adaboost only "sees" the values $h(X)$, for $h \in \mathcal{H}$, and not X itself.

The weak classifiers h_t and weights α_t in Equation (1) are determined by the Adaboost rule at the t^{th} training step. The training data consists of examples $(X_1, y_1), ..., (X_N, y_N)$. At each training step, Adaboost chooses the classifiers $h_t \in \mathcal{H}$ and weights $\alpha_t \in \mathbb{R}$ that minimize a criterion related to the error. In the present work, we use the criterion used in [18, Sec. 4].

2.2 Image filters as weak classifiers

We now detail how image filters, i.e. image processing operations, can be used as weak classifiers suitable for Adaboost. Assume we have image filters $f_1, \ldots, f_M$ (listed below) that produce, for a given pixel location and value $X \in \mathcal{X}$, a value $f_m(X) \in \mathbb{R}$[1]. Then, for every $m \in$

[1] or $\{1, \ldots, 255\}$.

$\{1, \ldots, M\}$ and every threshold $\tau \in \mathbb{R}$, we define the weak classifier

$$h^{m,\tau}(X) = \text{Sigmoid}\left(f_m(X) - \tau\right). \quad (2)$$

The set of weak classifiers is: $\mathcal{H} = \{h^{m,\tau} \mid 1 \leq m \leq M, \tau \in \mathbb{R}\}$.

Optimal threshold determination: it is of practical importance to note that the optimal threshold τ that minimizes the classification error can be determined with complexity proportional to the number of examples - surprisingly, we did not find prior reference to this fact, so we explain here how this is done. This results from the image filters taking only a finite number of values, e.g. $0, 1, \ldots, 255$, so that only thresholds $\tau_1 = -0.5, \tau_2 = 0.5, \ldots, \tau_{257} = 255.5$ need to be considered. By noting that the sum

$$S_i = \sum_{n \text{ s.t. } f(X_n)=i} D_t(n) y_n$$

is the difference between the cost of threshold τ_i and that of threshold τ_{i+1}, one easily shows that the best threshold τ_i is that corresponding to the minimum of $\sum_{j<i} S_i$. The absolute error is then $\sum_{n \text{ s.t. } y=-1} D_t(n) + \sum_{j<i} S_i$. We may thus determine in linear time the optimal threshold for both the filters $f(X_n)$ and $-f(X_n)$. This is obviously an improvement over the naive procedure, which has complexity $O(N \log N)$ at each boosting step.

More generally, when the filter may take R different values, the optimal threshold can always be found in time $O(NR)$ after performing a preprocessing step of complexity $O(RNlogN)$ in which the values the filter takes on the training examples are sorted.

Having presented some general aspects of the boosting procedure, we now specify the filters that are used in our experiments. The output of these filters, on image 120 of the MIT sequence used in Section 3, is shown in Figure 1.

2.2.1 Spatial Correlation filter

Classification with these filters is based on spatial correlation of different sized neighborhoods of the input images with a mean background-only image obtained from the beginning of the sequence. The correlation for each pixel in the output image is computed as:

$$f_m^{\text{Corr}}(X) = \text{Corr}\left(B(N_m, X), T(N_m, X)\right), \quad (3)$$

where $B(N_m, X)$ (resp. $T(N_m, X)$) is a neighborhood of width N_m around X in the input (resp. mean background) image. The correlation will be low in foreground regions and high in background regions. We used nine neighborhood sizes $N_m \in \{3, 5, 7, 9, 11, 15, 21, 27, 33\}$, leading to varying levels of smoothing in the outputs. Figure 1, (b) shows the filter values with $N_m = 3$.

2.2.2 Spatio-temporal Filters

In these filters, image pixels are classified based on spatio-temporal consistency in successive images. The output image is generated as follows: the current frame t is spatially smoothed by a binomial filter of width σ, resulting in values $\hat{X}_t^\sigma$. These values are time-smoothed using a first order AR filter:$\hat{X}_t^{\sigma,\lambda} = \lambda \hat{X}_t^\sigma + (1-\lambda)\hat{X}_{t-1}^\sigma$. Finally, the filter is

$$f_m^{ST}(X_t) = |\hat{X}_t^{\sigma,\lambda} - \hat{X}_{t-1}^{\sigma,\lambda}| \quad (4)$$

In Section 3, we use 12 ("hall-monitor" sequence) or 48 ("MIT" sequence) classifiers, corresponding respectively to $\sigma \in \{2, 16\}$, $\lambda \in \{1/2, 16/17\}$ and to $\sigma \in \{0, 2, 4, 16\}$, $\lambda \in \{0, 1/2, 4/5, 16/17\}$; each of these filters were applied in three color spaces, RGB, HS and V. Figure 1, (c) shows the output of these filters, for the V component, with $\sigma = 2$ and $\lambda = 1/2$.

2.2.3 Pixel-wise probabilistic filters

Classification with these filters is based on the probability of the current RGB (or HSV or Laplacian of Gaussian(LOG)) value X at a given pixel to belong to the background, assuming a kernel probability model:

$$f_m^{\text{Kernel}}(X) = \sum_{i=1}^{P} \Pi_{j=1}^{d} \frac{1}{\sqrt{2\pi}\sigma_{m,j}} e^{-(X_j - Z_{i,j})^2 / 2\sigma_{m,j}^2}, \quad (5)$$

where $Z_1, ..., Z_P$ are $d = 3$-dimensional vectors of RGB (or HSV or LOG) background values observed in the first P frames of the sequence, which are supposed to be background-only. The parameter $\sigma_{m,j}$ for each pixel is allowed to take 10 different values around the value suggested in in Elgammal et al [5]. We thus have $10 \times 3 = 30$ different pixel-wise probabilistic filters at our disposition. Figure 1, (d) shows a possible output of the filter (negated), computed on the RGB representation the image.

2.2.4 Morphological operators

Morphological operators are applied at each training step $1 < t \leq T$ to the current of the unthresholded classifier,

$$H_t(X_n) = \sum_{s=1}^{t-1} \alpha_s h_s(X_n), \;\; n \in \{1, \ldots, N\}.$$

We use grey-level operators of erosion, dilation, opening and closing, with radii of 1, 2, 3 and 4 pixels, which result in grey-level images. There are thus 16 morphological operators at our disposition.

Figure 1: **From left to right:** Image 120 of MIT Indoor sequence; Output of a correlation filter (Sec. 2.2.1); of a spatio-temporal filter (Sec. 2.2.2)(negated to be more visible); of a probabilistic filter (Sec. 2.2.3)(negated to be more visible); region segmentation (Sec. 2.2.5).

2.2.5 Region-voting

The region-voting classifier is based on a segmentation of the original image and on the classification at the current training step. For any pixel X belonging to a region consisting of R pixels $\{X_{r_1}, X_{r_2}, \ldots, X_{r_R}\}$, the region-voting classifier is defined by:

$$f_m^{Reg}(X) = \frac{1}{R} \sum_{i=1}^{R} H_t(X_{r_i}) \tag{6}$$

A different region-voting weak classifier family is defined for each possible region segmentation. In the present work, we use the segmentation method of Comaniciu and Meer [4] with three different values of the color histogram distance threshold, resulting in three families of weak classifiers, corresponding to different levels of granularity of segmentation. Note that only the regions matter in this filter, not the color given to each region by the segmentation method. Figure 1, (e) shows the most detailed region segmentation.

Note that using morphological operators and region-voting changes theoretically the boosting framework, since the set of weak classifiers varies during boosting. Our study tends to show that this change has no adverse consequence in theory and that it may be useful in practice.

Altogether, in the following section, we use (9+48+28+16+3=) 104 filters on the MIT sequence, and (9+12+28+16+3=) 68 in the "hall-monitor" sequence. Even if, as explained above, each one yields a family of weak classifiers indexed by the threshold τ in Equation (2), this is a relatively small number of weak classifiers.

3 Experiments

Having detailed our methodology, we show how it performs in practice and analyze the relative importance of each type of weak classifier in the classifier built by Adaboost. We also compare the results of the presented method with those obtained when one uses high quality hand-segmented images for training.

To avoid confusion, we use the term "learning round" to indicate the rounds of the labeling-training-validation cycle, while "training steps" denotes the successive steps of Adaboost.

Hall-monitor We first use the well-known "hall-monitor" sequence, which consists in 300 352×288 images. For validation, we will use a hand-made segmentation[2] (Figure 2, top).

For training, in the first round, we used the rectangular regions in images 121, 126, 131 and 136 shown in Figure 3, top. The lighter (and greener) boxes are regions of foreground, while darkened (reddened) regions are background. Adaboost reduced the training error to zero in five steps, as shown in Fig. 3, left, circle-curve, which plots the training error vs. the number of training steps. The validation error remained approximately constant, with a slight increase, after the fifth Adaboost step (Fig. 3, left, full curve). This indicates that Adaboost overfits slightly the data. After examining the output of the classifier on the training images (this output is not shown), we added some boxes, resulting in the training set shown in Figure 3, bottom.

This data was used in the second round of training. This time, 17 rounds were needed to zero the training error. The same slight overfitting trend as in the first round was observed. Surprisingly, the validation error increased with respect to the first round (Fig. 3, left, the dashed curve, is above the full curve), despite the fact that the output appears visually improved (Fig. 2, third row vs. second). This is better understood by looking at the false positive and false negative rates: in the first round, these were 0.88% (FP) and 3.51% (FN), while in the second round, they were 1.15% (FP) and 2.36% (FN). Indeed, in the second round of labeling, we focused on reducing the false negatives, by adding labeled boxes mainly in foreground regions. While this goal was achieve, the false positive rate increased slightly. Since the data is overwhelmingly background, this results in a global increase of the misclassification rate.

[2]Prof. Erdem [6] provided hand-labeled ground-truth of the left sides of images 32-240 and the authors hand-labeled the right sides of images 70, 100, 130, 160, 190 and 220

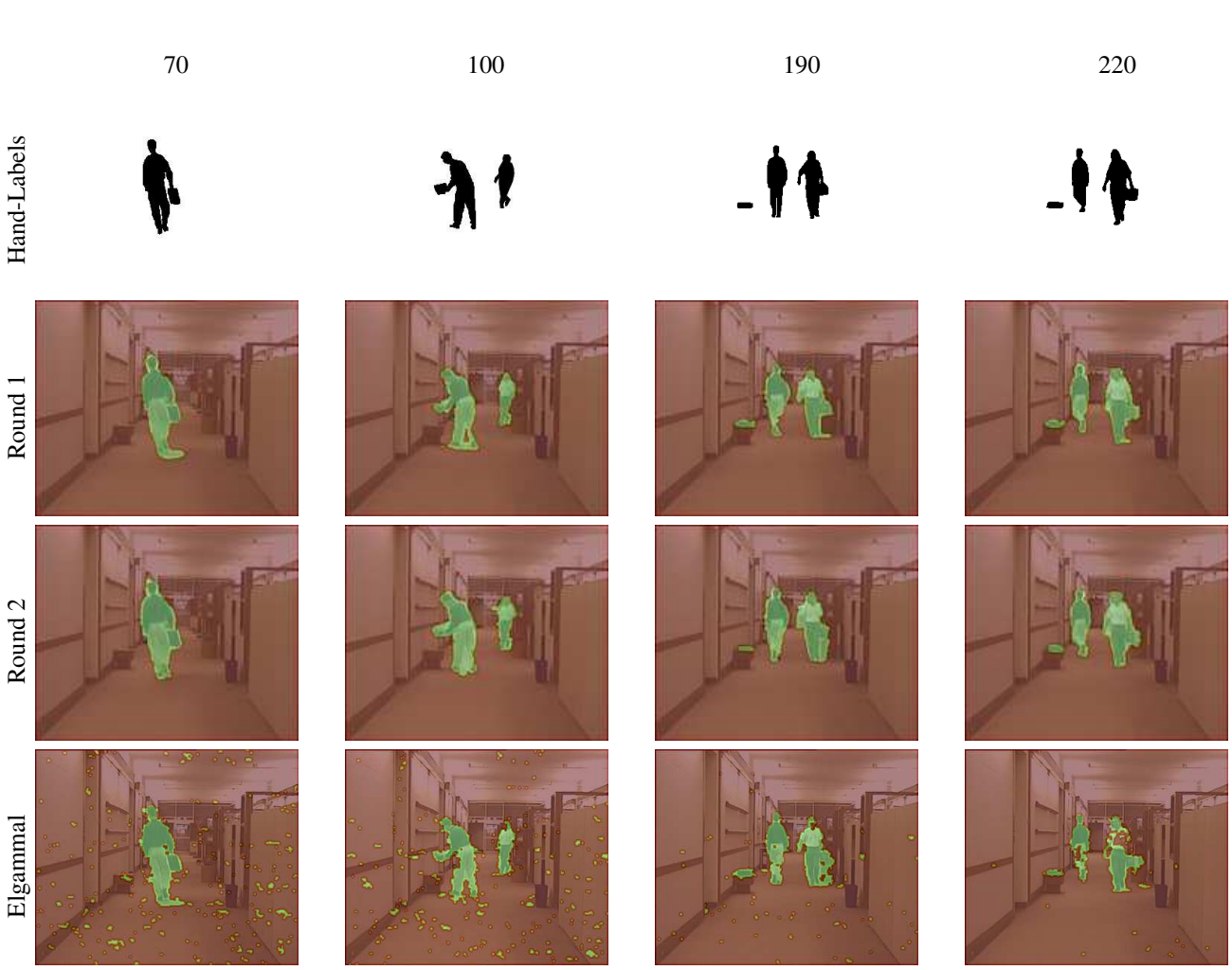

Figure 2: **Row 1:** Hand-labeled ground truth for images 70, 100, 190 and 220 of the "Hall-monitor" sequence. **Row 2:** Output of the boosted classifier trained with the first set of boxes, superposed on the hall-monitor images. **Row 2:** Output of the boosted classifier trained with the second set of boxes. **Row 4:** Output of the method of Elgammal et al [5].

These results were compared -this is not a fair comparison- with the unsupervised Kernel-based approach of Elgammal et al. [5] (shown in row 4). The false positive rate using their method was found to be 0.37% (compare with 0.88%, 1.15% for rounds one and two) and , the false negative rate was 15.7% (compare with 3.51% 2.36% for rounds one and two) while the overall error was 1.76%. (we reach 1.01 and 1.23% on the first and second rounds). Although the difference in error is not huge, we achieve a much lower false negative rate and our false positives form a visually acceptable "fat border" around the true foreground region.

A fairer comparison is with our previous work in which carefully hand-labeled images images, like those in Figure 2, top, are used for training. In this case, using images 70, 100, 130, 160 and 190 for training and image 220 for validation, and performing 20 boosting steps, we obtained a training error of 0.51%, a validation error of 0.96%, a false positive rate of 0.89% and a false negative rate of 2.8%. Again, while these error rates appear good, the output is in fact visually less appealing than that obtained when training with boxes and shown in Fig. 2.

Beyond the classification error, it also worth studying the relative importance of the various weak classifier: in the second round, the final classifier consisted in the sum of 10 image-processing filters (6 probabilistic, 3 spatio-temporal filters, and one correlation filter), 6 morphological operators, 2 region-voting filters and 1 correlation filter. The contributions of these classes of filters in Eq. (1) were respectively 36%, 56% and 7.5%. Figure 3, left, light dashed curve is the validation error of a classifier trained on the second round data, but without the morphological operators. While this suggests that morphological operators are detrimental in terms of validation error, it should be noted

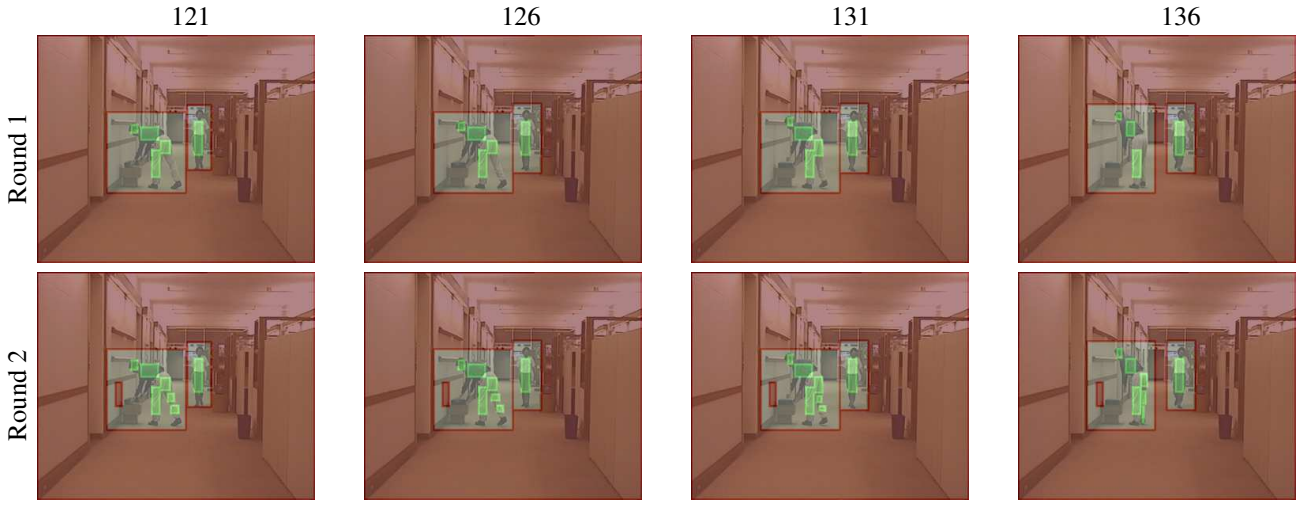

Figure 3: Training data for the "Hall-monitor" sequence. Foreground regions are indicated as lightened green boxes and background is indicated by the darkened red interior or exterior of boxes. with superposed boxes labeling foreground and background regions. **top:** data used in the first training round. **bottom:** data used in the second round. Fig. 2 shows the output of the boosted classifiers on another set of images.

that this operator improves the visual aspect of the segmentation, by cleaning off artifacts produced by the other weak classifiers. A more detailed analysis is provided in the next dataset.

These experiments provide some insight on how to further develop interactive methods for offline background subtraction. These ideas are discussed in Section 4.

MIT indoor data This image sequence was taken at the MIT AI Laboratory and provided by Joshua Migdal [14], together with some hand-labeled frames. It consists of a person entering the field of view of the camera on the right, walking to the left out of the camera's view. The top row of Figure 5 shows the hand-labeled segmentation of frames 115, 125, 130 and 135, that we use for validation.

Five to 12 other images were used in training. At each round, some new boxes were labeled and new images added to the training set. Fig. 5, rows 2-4 show the output of the classifier on the validation images at the first, second and fourth learning rounds.

At the second learning round, we studied the importance of each type of weak classifier. This was done by comparing the performance obtained when weak classifiers were limited to (1) Image filters only (those of Secs. 2.2.1, 2.2.2 and 2.2.3); (2) Image filters and morphological operators; (3) Image filters and region voting; (4) Image filters, morphological operators and region voting. For each combination of weak classifier families, eight Adaboost classifiers were trained, one for each subset of eight training images chosen in $\{113, 116, 120, 123, 128, 133, 136, 138, 139\}$. The average performance of these eight classifiers is recorded and plotted in Fig. 3, right. These curves show clearly that morphological operators are detrimental in terms of validation error, whether region voting is used or not. In this dataset, contrarily to the "hall-monitor" dataset, the global error was decreased from the first to the second learning round. Without the morphological operators, the validation error, false positive and false negative rates are, after the first learning round, 1.00%, 0.81% and 4.03%. After the second round, these figures are: 0.96%, 0.86% and 2.50%. After the fourth round, these figures are: 1.29%, 1.47% and 1.27%. These last figures indicate a tendency to overfit when more complete training data is used, even when morphological operators are not use. This effect is comparable to that obtained when training an image segmenter for the "hall-monitor" sequence, using hand-labeled data for training.

4 Conclusions

In this paper we presented an iterative approach based on supervised learning to generate high quality foreground segmentation for video sequences which involves the user marking a few rectangular regions, as opposed to performing laborious hand-segmentation. In addition, by training multiple classifiers on slightly different datasets, as we did with the MIT sequence, it is possible to get a confidence measure for the method.

Our objective is, in the future, to improve the quality. Achieving this objective would bring the following benefits:

- Quantitative evaluation of unsupervised background

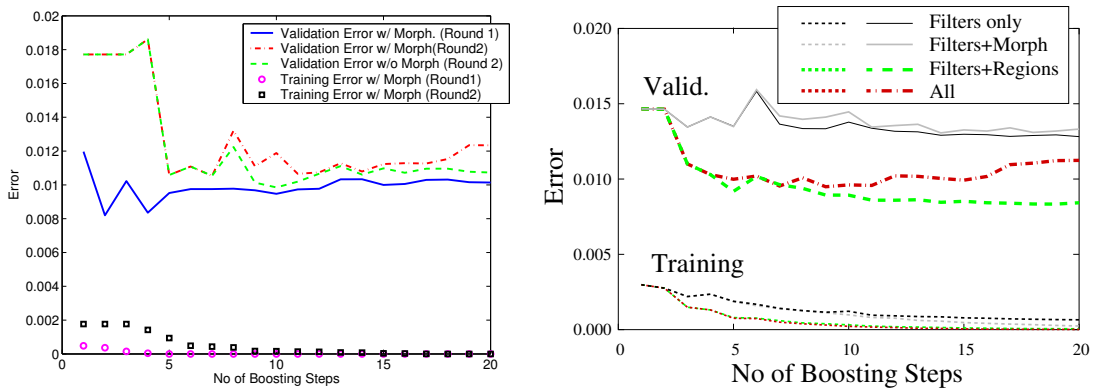

Figure 4: **Left:** Validation and training error as a function of number of boosting steps, on the "Hall-Monitor" sequence. From bottom to top: training error in first learning round, in second learning round, validation error in first learning round, in the second learning round (without morphological classifiers) and in the second learning round with all classifiers. See the text for a complete explanation. **Right:** Average validation and training error obtained with the training data of the second learning round the MIT sequence. Each curve corresponds to a combination of weak classifiers.

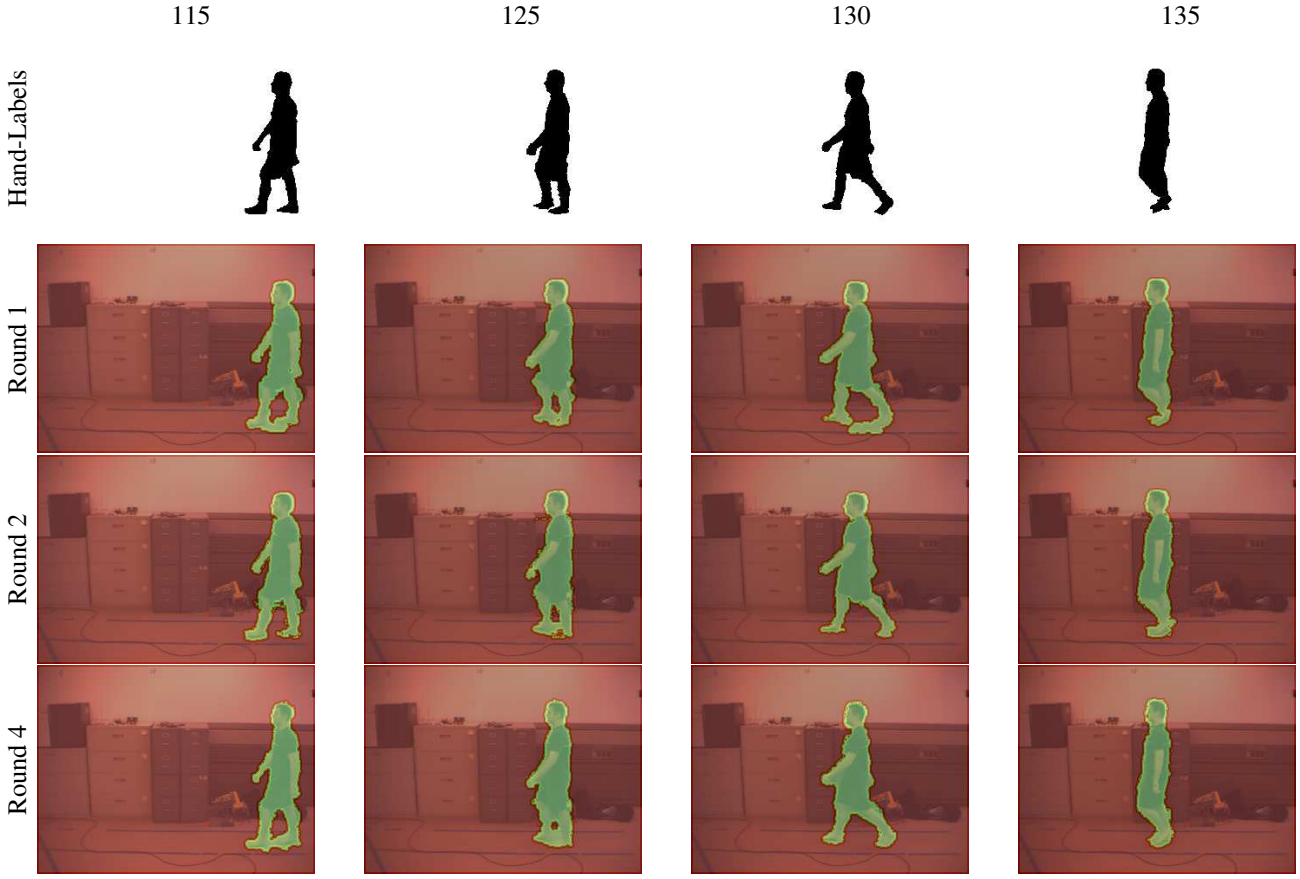

Figure 5: **Row 1:** Hand-labeled ground-truth for the MIT Indoor sequence, images 115, 125, 130 and 135. **Rows 2-4:** Output of boosted classifier after 1,2 and 4 rounds of training.

subtraction methods, as done in [6], would be possible with very little extra labor.

- The availability of large amounts of segmentation data enables new approaches to background subtraction, for

example approaches based on supervised learning.

The results obtained thus far when training from boxes are quantitatively comparable to those obtained when training from hand-segmented images. Of course, the results are better than that obtained with unsupervised learning. We have seen that early in the interactive learning process, results are very appealing visually, owing to the low false negative rates and to the absence of patches of false positives. We observed that false positives form a "fat border" around the true positives.

However, attempting to reduce this border of false positives does not give good results with our current approach. Adding labeled boxes to the training set quickly yields results comparable to that obtained when training from hand-segmented images, i.e. the limit of infinitely many labeled boxes.

We also showed experimentally that all image filters and post-processing operators are not equally useful. This suggests using different -more powerful- image operators, e.g. coarse "tracking" filters similar to those in Black et al [3].

We also met with the discrepancy between classification error and visual appeal of a segmentation. This suggests to use measures of quality closer to human perception [21], not only for performance evaluation, but also during training. This last point would require changes to the boosting framework, beyond that of using post-processors as weak classifiers. In future work, we also consider to used semi-supervised learning methods to automatically identify unlabeled regions that would be particularly informative for the learner.

In summary, we have already obtained relatively promising results and the results of our experiments suggest many directions for improvements.

References

[1] H. Adams, S. Singh, and D. Strelow. An empirical comparison of methods for image-base motion estimation. In *IROS*, Lausanne, Switzerland, 2002.

[2] A. Agarwala, A. Hertzmann, D. H. Salesin, and S. M. Seitz. Keyframe-based tracking for rotoscoping and animation. *ACM Trans. Graph.*, 23(3):584–591, 2004.

[3] P. Black, T. Ellis, and P. Rosin. A novel method for video tracking performance evaluation. *IEEE Int. Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance (VS-PETS)*, pages 125–132, 2003.

[4] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Trans. Pattern Analysis Machine Intell.*, 24(5):603–619, 2002.

[5] A. Elgammal, R. Duraiswami, D. Harwood, and L. S. Davis. Background and foreground modeling using nonparametric kernel density estimation for visual surveillance. *proc. of the IEEE*, 90(7):1151–1163, 2002.

[6] C. E. Erdem, A. M. Tekalp, and B. Sankur. Metrics for performance evaluation of video object segmentation and tracking without ground-truth. In *IEEE Intl. Conf. on Image Processing (ICIP)*, 2001.

[7] R. Fisher, J. Santos-Victor, and J. Crowley. CAVIAR: Context aware vision using image-based active recognition. http://homepages.inf.ed.ac.uk/rbf/CAVIAR/, 2003. EC IST project IST 2001 37540.

[8] A. Fitzgibbon, Y. Wexler, and A. Zisserman. Image-based rendering using image-based priors. *Proceedings of the International Conference on Computer Vision*, October 2003.

[9] W. T. Freeman and E. C. Pasztor. Learning low-level vision. In *International Conference on Computer Vision*, volume 2, pages 1182–1189, 1999.

[10] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

[11] J. Friedman, T. Hastie, and R.J. Tibshirani. Additive logistic regression: a statistical view of boosting. *Annals of Statistics*, 2:337–374, 2000.

[12] Z. Liu and S. Sarkar. Effect of silhouette quality on hard problems in gait recognition. *IEEE Trans. on SMC*, 35(2):170–178, April 2005.

[13] A.M. Martinez and R. Benavente. The AR face database. 24, Computer Vision Center of the Universitat Autònoma de Barcelona, 1998.

[14] J. Migdal and W.E.L Grimson. Background subtraction using markov thresholds. *Proceedings of IEEE Workshop on Motion and Video Computing*, January 2005.

[15] P. J. Phillips, S. Sarkar, I. Robledo, P. Grother, and K. W. Bowyer. The gait identification challenge problem: Data sets and baseline algorithm. *Proc of the International Conference on Pattern Recognition*, 2002.

[16] P.L. Rosin and E. Ioannidis. Evaluation of global image thresholding for change detection. *Pattern Recognition Letters*, 24(14):2345–2356, 2003.

[17] C. Rudin, R. Schapire, and I. Daubechie. Analysis of boosting algorithms using the smooth margin function: A study of three algorithms. *submitted somewhere (As of Oct. 1 2004)*, 2004.

[18] R. E. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37(3):297–336, 1999.

[19] D. Scharstein, R. Szeliski, and R. Zabih. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms, 2001.

[20] A. Torralba, K. P. Murphy, and W. T. Freeman. The MIT-CSAIL database of objects and scenes. web.mit.edu/torralba/www/database.html.

[21] Ranjith Unnikrishnan, Caroline Pantofaru, and Martial Hebert. A measure for objective evaluation of image segmentation algorithms. In *Proceedings of the 2005 IEEE Conference on Computer Vision and Pattern Recognition (CVPR '05), Workshop on Empirical Evaluation Methods in Computer Vision*, June 2005.

[22] P. Viola and M. Jones. Robust real-time object detection. In *Proc. ICCV workshop on statistical and computational theories of vision*, 2001.

[23] P. A. Viola, M. J. Jones, and D. Snow. Detecting pedestrians using patterns of motion and appearance. In *ICCV*, pages 734–741, 2003.