# A Framework for Feature Selection for Background Subtraction

Toufiq Parag
*Dept. of Computer Science*
*Rutgers University*
*Piscataway, NJ 08854*
*tparag@cs.rutgers.edu*

Ahmed Elgammal [1]
*Dept. of Computer Science*
*Rutgers university*
*Piscataway, NJ 08854*
*elgammal@cs.rutgers.edu*

Anurag Mittal [2]
*Real-time Vision & Modeling Dept.*
*Siemens Corporate Research*
*Princeton, NJ 08540*
*anurag_mittal@yahoo.com*

## Abstract

*Background subtraction is a widely used paradigm to detect moving objects in video taken from a static camera and is used for various important applications such as video surveillance, human motion analysis, etc. Various statistical approaches have been proposed for modeling a given scene background. However, there is no theoretical framework for choosing which features to use to model different regions of the scene background. In this paper we introduce a novel framework for feature selection for background modeling and subtraction. A boosting algorithm, namely RealBoost, is used to choose the best combination of features at each pixel. Given the probability estimates from a pool of features calculated by Kernel Density Estimate (KDE) over a certain time period, the algorithm selects the most useful ones to discriminate foreground objects from the scene background. The results show that the proposed framework successfully selects appropriate features for different parts of the image.*

## 1 Introduction

Background modeling and subtraction forms a core componentL in many vision systems. The main idea behind such a module is to automatically generate and maintain a representation of the background that is then used to classify any new observation as background or foreground. The information provided by such a module can then be utilized for performing high-level object analysis tasks such as object detection, tracking, classification and event analysis.

Various methods of increasing complexity have been considered in the literature. In [17], a single Gaussian was considered to model the statistical distribution of the background at each pixel. Friedman et. al.[3] use a mixture of three Normal distributions to model the visual properties in traffic surveillance applications. Three hypothesis are considered - road, shadow and vehicles. The EM algorithm is used which gives very good model-fitting but is computationally expensive. Grimson et. al. [6] extend this idea by using multiple Gaussians to model the scene and develop a fast approximate method for updating the parameters of the model incrementally. Such an approach is capable of dealing with multiple hypothesis for the background and can be useful in scenes such as waving trees, beaches, escalators, rain or snow. The mixture-of-Gaussians method is quite popular and was to be the basis for a large number of related techniques [8, 7]. In[4], a statistical characterization of the error associated with this algorithm is studied. When the density function is more complex and cannot be modeled parametrically, a non-parametric approach able to handle arbitrary densities is more suitable. Such an approach was used in [1] where the use of Gaussian kernels for modeling the density at a particular pixel was proposed.

Another class of background modeling methods try to model the short-term dynamical characteristics of the input signal. Several authors [9, 10] have used a Kalman-filter based approach for modeling the dynamics of the state at a particular pixel. A simpler version of the Kalman filter called *Weiner filter* was considered in [16] that operates directly on the data. Such modeling may further be performed in an appropriately selected subspace [18, 13, 12].

Besides modeling the statistical distribution of the data, another factor in background modeling is the choice of the transformation that is applied to the original data in order to obtain the features that are used. Several features have been considered including raw color, normalized color, spatial gradients, texture and optical flow. The basic idea behind feature selection is that one wants to be invariant to certain types of changes in the visual space while maintaining a good detection for the foreground objects. For instance, in outdoor scenes, the classification must be invariant to a change of illumination that might occur due to the sun, clouds or light from a nearby light source. Similarly, in dynamic scenes such as ocean waves or waving trees, in-

[1]Also affiliated with the Dept. of Computer Science and Automatic Control, University of Alexandria, Egypt.

[2]Currently with Dept. of Computer Science and Engg., IIT Madras, Chennai, India.

variance to such periodic motion is critical. Each feature has its strength and weakness and is particularly applicable for handling a certain type of variation. For instance, normalized color, spatial gradients or texture features may be considered for obtaining invariance to illumination[5, 8], while optical flow might be useful in handling dynamic scenes [11, 14]. On the other hand, such features may not be very suitable in other regions. For instance, spatial gradients or texture is not very suitable for a region that has low spatial gradients since such feature will be unable to detect any object that has a low gradient itself. Similarly, optical flow cannot be computed accurately in regions that have low texture and is thus not very useful in such regions.

In most background subtraction algorithms, the features are chosen arbitrarily and the same feature are used globally over the whole scene. No framework is known to the authors for selecting suitable features for different parts of the scene. The *primary contribution* of this paper is a generic formulation that is able to automatically select the features that obtain the best invariance to the background changes while maintaining a high detection rate for the foreground objects. We propose to address the problem as a classification problem where we classify the foreground objects from background pixels. An ensemble learning method, namely boosting classifier, seems appropriate in this scenario. Boosting algorithms usually generates a weighted linear combination of some weak classifiers that perform only a little better than random guess. We learn weak classifiers from the feature values at a pixel and combine the ones performing better than the others to produce a strong classifier. Thus we are effectively selecting different features at each pixel to distinguish foreground objects from the background. Once selected, we expect this combination of features to perform successfully afterwards, unless there are major changes in the scene.

The organization of the paper is as the following. Section 2 gives a general overview of the framework. Sections 3 and 4 describes the general Realboost algorithm and how the it was used in background subtraction, respectively. Section 5 explains the experimental setup and the results. Finally, we conclude by a brief discussion over the findings of this study and on directions for future work in section 6.

## 2 Feature Selection Framework

In background subtraction, we generally have a sequence of $\mathcal{T}$ training images. Typically a statistical model is learned from feature values for each pixel over $1 \leq \tau \leq \mathcal{T}$ images of a sequence which, is later used to identify the objects that do not belong to the scene. For the rest of the paper, we will model every pixel individually and carry out the same operation on each of them. Given $\mathcal{T}$ observed values $x^1, x^2, \cdots x^{\mathcal{T}}$ of certain features of a particular pixel, we can build a statistical model for the probability $\rho(x) = p(x|x^1, x^2, \cdots x^{\mathcal{T}})$ of any new observation value $x$ at that pixel given its history. We can use single Gaussian, mixture of Gaussians [6], or Kernel Density estimation (KDE) [1, 11] to generate this model.

**Which Feature to Use:** Let us suppose we have $M$ different types of features and $\mathbf{x}_j$, $1 \leq j \leq M$ is the $j$-th feature value (e.g. R,G,B, intensity, spatial gradients, temporal gradients, optical flow, etc.) for a certain pixel of an image. As discussed in the introduction, pixels of different part of the image can exhibit different characteristics. This observation leads us to claim that different features rather than a single one may produce better accuracy in modeling the background distribution at each pixel. Our objective is to find out the more useful features from a pool of $M$ features, use weights to quantify their importance and use their *weighted* combination to differentiate foreground objects from the background. The selected subset of features can be different from one pixel to another based on the characteristics of each pixel in the background (tree waving, water, sky, static road, etc.). This process of feature selection is supposed to be done only once at system initialization (e.g. while installing a new camera at a new scene). Once this process is done, a feature map is generated for the scene background depicting which features are best for which pixel in the image. This feature map can be used afterward to model the background. This process does not need to be performed again unless there is some significant structural change in the scene.

**Model Parameter Selection:** The choice of the parameters of the estimation process also affects the background model by a huge amount. For example, we know that KDE is vulnerable to the choice of the bandwidth parameter $\sigma$. It can produce totally erroneous output for a wrong choice of $\sigma$. Consider the situation when the kernel bandwidth does not correspond to the variance of the feature values we are trying to model. For pixels in an outdoor scene, the variance will be substantially different from one another. To argue in favor of this statement, let us draw histograms of pixels of an outdoor image shown in Figure 1. The histograms for

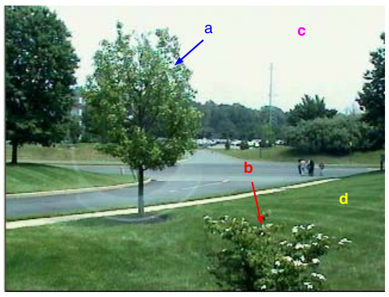

**Figure 1.** An image from an outdoor scene

only green color values for pixels at a, b, c, d indicated in the input image (Figure 1) over 200 image frames are shown in Figure 2. As we can see, pixels in tree leaves (a, b) have higher variances than that of pixels in more static regions like sky and grass (c, d). Therefore, kernel estimation with same bandwidth will fail to estimate the density of many pixels.

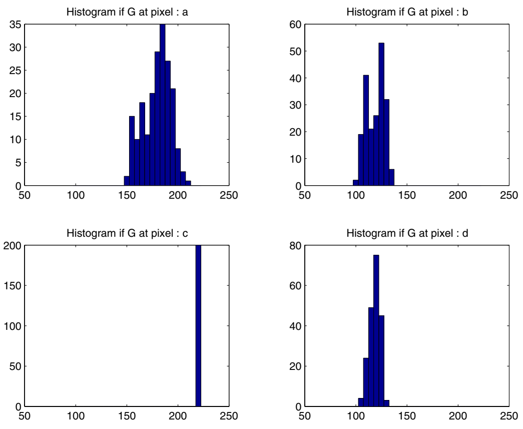

**Figure 2.** Histograms of G at different pixel position

**Boosted Background Classifier:** The framework introduced in this paper provide a solution for these problems by choosing best features and parameters at each pixel out of a pool of feature/bandwidth combinations. We want to generate a classifier for pixel $\mathbf{x}$ of the following form

$$F(\mathbf{x}) = \sum_{t=1}^{T} w_t f_{j_t}(\mathbf{x}) \tag{1}$$

such that we will classify a certain pixel to be a background pixel if $F(\mathbf{x}) > 0$ and a foreground object if $F(\mathbf{x}) \leq 0$. In equation 1, $f_{j_t}(\mathbf{x})$ is some function of $\mathbf{x}_{j_t}$,, where $j_t$ is the index of $t$-th feature chosen and $w_t$'s are the values to measure the importance of $f_{j_t}(\mathbf{x})$. From what we have already discussed, some function of the probability estimates of $\mathbf{x}_j$ may be a choice for $f_{j_t}(\mathbf{x})$ in equation 1. The summation in equation 1 is over $T$ of these functions where $T$ is a small number, representing the top features selected to model the background at each individual pixel. We can generate these estimates for all $M$ feature types and for a range of parameters of the estimation process. Therefore, we need an algorithm to select optimal feature and parameter from this pool of features and parameters for background subtraction. The algorithm should also calculate the weights for it's choice of features simultaneously.

The boosting algorithm, RealBoost [15] seems ideal for this scenario. Unlike Adaboost (which combines weak hypotheses having outputs in {-1, +1}), RealBoost algorithm computes real-valued weak classifiers given real numbered feature values, and generates a linear combination of these weak classifiers that minimizes the training error. We can expect that, using the density estimates, the Realboost algorithm will be able to select the features most appropriate for any specific pixel.

## 3 RealBoost

Let $S = \{(\mathbf{x}^1, y^1), \cdots, (\mathbf{x}^N, y^N)\}$ be a sequence of training examples where each instance $\mathbf{x}^i, i = 1, 2, \cdots, N$ belongs to an instance space $\mathcal{X}$ and each label $y^i$ belongs to a label set $\mathcal{Y}$. We will consider the case of binary classification where $\mathcal{Y} = \{-1, +1\}$. Suppose we have M different types of features for each $\mathbf{x} \in \mathcal{X}$. Denote $\mathbf{x}_j \in \Re$ as the $j$-th feature value of $\mathbf{x}$ and $\rho_j(\mathbf{x}) = g(\mathbf{x}_j)$ where $g : \Re \to [0, 1]$ is some function applied to the feature values. The aim of boosting algorithm is to find a linear combination of some functions $h_j(\mathbf{x})$ of $\rho_j(\mathbf{x})$ of the form

$$H(\mathbf{x}) = \sum_{t=1}^{T} \alpha_t h_{j_t}(\mathbf{x}) \; , \quad \text{where} \;\; h_j(\mathbf{x}) = f(\rho_j(\mathbf{x})) \tag{2}$$

where $T$ is a small number of selected functions. According to this equation, an instance $\mathbf{x}$ can be classified as an example of class $y^i = +1$ if $H(\mathbf{x}^i) > 0$ and as an example of $y^i = -1$ otherwise [2, 15]. The $h_j(\mathbf{x}) : [0, 1] \to \{-1, 1\}$ are called the weak hypothesis [1]. At each round of boosting [2, 15], we choose a particular $h_{j_t}(\mathbf{x})$ from all $h_j(\mathbf{x})$ , $j = 1, 2, \cdots M$; i.e. we add one more term to the summation. Each weak hypothesis is given a weight $\alpha_t$ based on its performance in classifying the training examples. These weights quantify how much confident we are on the corresponding hypotheses in predicting the unknown test cases.

In [15], the weak hypotheses were modified to produce real-valued output, i.e. $h(\mathbf{x}) : \Re \to \Re$ for suitably chosen $h(\mathbf{x})$ on $\mathbf{x}$ so that it absorbs the corresponding confidence weights $\alpha_t$'s into themselves. We will adopt this strategy so that the weak hypotheses in equation 3 can be defined as $h_j(\mathbf{x}) : [0, 1] \to \Re$. This can be accomplished by designing a weak learner algorithm such that the $sign(h_j(\mathbf{x}))$ denotes the predicted label for $\mathbf{x}$ and $|h_j(\mathbf{x})|$ denotes the 'confidence' of the prediction. Then the form of the strong classifier becomes

$$H(\mathbf{x}) = sign\Big(\sum_{t=1}^{T} h_{j_t}(\mathbf{x})\Big). \tag{3}$$

Let $\Psi_j$ be the set values obtained by applying the $\rho_j(\cdot)$ function at the training instances, i.e. $\Psi_j = \{\rho_j(\mathbf{x}^1), \; \rho_j(\mathbf{x}^2), \cdots \rho_j(\mathbf{x}^N)\}$, where $j = 1, 2, \cdots, M$. One approach to compute $h(\mathbf{x})$ is to partition the set $\Psi^j$ into disjoint subsets $\Psi_j^1, \; \Psi_j^2, \; \cdots \; \Psi_j^B$ and assign the same $h_j^*$ to all $h_j(\mathbf{x}^*)$ such that $\rho_j(\mathbf{x}^*) \in \Psi_j^b$, where $1 \leq b \leq B$. The idea is to partition $\mathcal{X}$ into smaller sets containing $\mathbf{x}$'s that are similar in some sense according to the value of the

[1]Notice that, $h_j$ is actually a function of $\rho_j(\mathbf{x})$. We use the notation $h_j(\mathbf{x})$ instead of $h(\rho_j(\mathbf{x}))$ for simplicity.

function $\rho_j(\cdot)$ and then use a single representative value of $h_j^*$ for all instances in the same partition. Let us assume that we know how to partition $\Psi^j$ into $B$ subsets and define the term

$$W_{t,c}^{j,b} = \sum_{i:\rho_j(\mathbf{x}^i)\in\Psi_j^b \,\wedge\, y^i=c} D_t(i)$$

where $c \in \mathcal{Y}$ and $D_t(i)$ is the normalized weight associated with instance $\mathbf{x}^i$ at iteration $t$ of the boosting [15]. It has been shown in [15] that the cost function (i.e. the error term) is minimized for a particular feature $j$ at iteration $t$ of boosting if $h_j(\mathbf{x}) = \frac{1}{2}\ln[(W_{t,+}^{j,b})/(W_{t,-}^{j,b})]$ for all $\mathbf{x}$ such that $\rho_j(\mathbf{x}) \in \Psi_j^b$. The minimized value of the cost function becomes $Z_t^j = 2\sum_b \sqrt{W_{t,+}^{j,b} W_{t,-}^{j,b}}$; $j = 1, 2, \ldots, J$ at iteration $t$. The algorithm determines $j_t$ as follows

$$j_t = \arg\min_{j\in J} Z_t^j.$$

Analyzing the formulae above, we can interpret the quantity $W_{t,+}^{j,b}/W_{t,-}^{j,b}$ as a measure of discriminative power of the weak hypothesis for $\mathbf{x}$ such that $\rho_j(\mathbf{x}) \in \Psi_j^b$.

All that we are left with is to find out a way to partition the instance set $\Psi_j$ into $\Psi_j^1, \Psi_j^2, \ldots, \Psi_j^B$. The function (or estimate) $\rho_j(\mathbf{x})$ of feature values lies within the range $[0, 1]$. We can always build a histogram within the range $[0, 1]$ with $B$ bins: $bin_b = [\frac{(b-1)}{B}, \frac{b}{B})$ for $b = 1, 2, \ldots, B$. Then we define the subset $\Psi_j^b = \{\rho_j(\mathbf{x}) : \rho_j(\mathbf{x}) \in bin_b\}$. *Notice that this is a histogram of the values of the $\rho_j$ functions and not a histogram of the feature values themselves.* Essentially we are creating partitions $X_1, X_2, \ldots X_B$ of the instance space $\mathcal{X}$ comprising instance $\mathbf{x}$ with close $\rho_j(\mathbf{x})$ values; i.e. with similar properties that are being quantified by $\rho_j$ [2]. Given these settings, we can describe the RealBoost algorithm as:

```
• Initialize the weights $D_1(i) = \frac{1}{N}$
• For $t = 1, 2, \ldots, T$
    1. For $j = 1, 2, \ldots, M$, Decompose $\Psi^j$ into $\Psi_1^j, \Psi_2^j, \ldots \Psi_B^j$
    2. Calculate
       $$W_{t,c}^{j,b} = \sum_{i:\rho(\mathbf{x}^i)\in\Psi_j^b \wedge y^i=c} D_t(i),$$
       for $j = 1, 2, \ldots, M$; and $b = 1, 2, \ldots, B$; where $c \in \{-1, +1\}$.
    3. Set the output of $h$ for all $\mathbf{x}$ such that $\rho_j(\mathbf{x}) \in \Psi_j^b$ as
       $h_j(x) = \frac{1}{2}\ln\left[(W_{t,+}^{j,b} + \epsilon)/(W_{t,-}^{j,b} + \epsilon)\right]$ where $\epsilon$ is a
       small positive constant for numerical stability.
    4. Select $h_{j_t}$ minimizing the normalization factor $Z_t^j =
       2\sum_{b=1}^{B}\sqrt{W_{t,+}^{j,b} W_{t,-}^{j,b}}$ ; $j_t = \arg\min_j Z_t^j$
    5. Update $D_{t+1}(i) = D_t(i)e^{-\left(y^i h_{j_t}(\mathbf{x}^i)\right)}$
• The strong classifier is: $H(\mathbf{x}) = sign\left[\sum_{t=1}^{T} h_{j_t}(\mathbf{x})\right]$
```

[2] A complete theoretical step by step construction and analysis of the boosting algorithm can be found in [15].

## 4 Feature Selection in Background Subtraction

### 4.1 Statistical Background model

The density functions of the feature values constitutes the statistical model for the background. Using the notation of section 2, the probability density function (pdf) for $j$-th feature for any pixel is estimated by Kernel Density Estimation (KDE) [1, 11] as the following

$$p(\mathbf{x}_j) = \frac{1}{\mathcal{T}}\sum_{\tau=1}^{\mathcal{T}} K_\sigma(\mathbf{x}_j - \mathbf{x}_j^\tau), \tag{4}$$

where $K_\sigma$ is a kernel function. We use such probability estimate as the function $\rho_j(\cdot)$, i.e. $\rho_j(\mathbf{x}) = p(\mathbf{x}_j)$, which (as defined in section 3) maps from the real-valued feature to the $[0, 1]$ range. Choosing $K$ to be a Gaussian kernel $N(0, \sigma)$, the pdf can be written as

$$\rho_j(\mathbf{x}) = \frac{1}{N\cdot C\cdot\sigma}\sum_{\tau=1}^{\mathcal{T}} exp(-\frac{1}{2\sigma^2}(\mathbf{x}_j - \mathbf{x}_j^\tau)^2). \tag{5}$$

Here $\sigma$ is the bandwidth parameter of the kernel function. We will use these estimates to learn the weak classifiers.

### 4.2 RealBoost in Feature Selection

We divided the training set into two parts- first $\mathcal{T}$ images are used to generate the statistical model for the background and then next $N$ images are used for boosting. All these $\mathcal{T} + N$ images are assumed to be free of foreground objects. In this study, we consider background pixels as the positive examples. We have $M$ one dimensional features for each pixel to choose from (e.g. R,G,B, intensity, spatial gradients, temporal gradients, optical flow, etc.). For each of the features, we also need to select the parameters of KDE that produces the best weak classifier. Therefore, we calculate the KDE estimates of the feature values using different parameters and let RealBoost chose the best one.

Suppose the set of bandwidths used is $\Sigma = \{\sigma_1, \sigma_2 \ldots \sigma_Q\}$. So, for every pixel, we have $M \times Q$ feature/parameter combinations; i.e. we have $Q$ functions $\rho_{jq}$ for feature $j$ with bandwidth $\sigma_q \in \Sigma$. Consequently, we also have $M \times Q$ number of $\Psi_j(q)$ sets. Each of these sets will then be partitioned into $B$ bins to calculate the corresponding weak classifiers $h_j(\mathbf{x})$ (for each bin).

Let the $j$-th feature values for any pixel in $N$ consecutive images be $\mathbf{x}_j^1, \mathbf{x}_j^2, \ldots, \mathbf{x}_j^N$. Given these values, the KDE estimates with parameter $\sigma_q$ are $\Psi_j(q) = \{\rho_{jq}(\mathbf{x}^1), \rho_{jq}(\mathbf{x}^2), \cdots \rho_{jq}(\mathbf{x}^N)\}$. These values will be used to generate the weak classifiers in the boosting procedure (see section 3). The negative examples are generated randomly from a uniform distribution. The KDE estimates of these random negative examples are also provided to the boosting algorithm along with the $\rho_{jq}(\mathbf{x})$s. Finally, we will have a strong classifier $H$ for each pixel as the output of the RealBoost algorithm by combining $T$ out of $M \times Q$ weak classifiers. Any new value $\mathbf{x}^{new}$ of $\mathbf{x}$ will be classified as a background pixel (positive example) if $H(\mathbf{x}^{new}) > 0$ and as a foreground pixel otherwise.

As discussed in section 3, in each bin, the weak classifier quantifies the discriminative power of the corresponding feature (along with the associated parameter for KDE). We may also view the approach as a histogram approach for estimating the distribution of the $\rho_{jq}(\mathbf{x})$ values for some feature $j$ and bandwidth $\sigma_q$. Let us call the KDE estimates of positive examples as $\rho(\mathbf{x}_+)$ and that of negative examples as $\rho(\mathbf{x}_-)$ for any feature values computed with some bandwidth. For a feature to be useful in background subtraction, the probability estimates of feature values of positive examples should be substantially different than that of negative examples; i.e. the distributions of the $\rho(\mathbf{x}_+)$ need to be fairly separated from that of $\rho(\mathbf{x}_-)$. In other words, the KDE with a specific bandwith is supposed to keep the amount of 'mixing 'of $\rho(\mathbf{x}_+)$ and $\rho(\mathbf{x}_-)$ as low as possible. For example, in Figure 4, the red and blue bars corresponds to the histogram bins of $\rho(\mathbf{x}_-)$ and $\rho(\mathbf{x}_+)$ respectively. The labels of the sub-images state the feature-bandwidth combination used for respective histogram. In this figure, the feature R with bandwidth 200 will be the most useful feature-parameter combination. In terms of probability, this is the same as saying a good feature-parameter combination should keep the estimated probability (approximated by the histogram) of $\rho(\mathbf{x}_+)$ to lie in some range of values as different as possible than that of $\rho(\mathbf{x}_-)$ to lie in the same range. This is precisely what is being done when we are minimizing the cost function $Z_1^j = 2\sum_{b=1}^{B} \sqrt{W_{1,+}^{j,b} W_{1,-}^{j,b}}$. Recall that the first iteration of boosting all $D_1(i)$'s are equal. We are choosing the feature and bandwidth value that decreases the probability that $\rho(\mathbf{x}_+)$ and $\rho(\mathbf{x}_-)$ assume values in the same range. Therefore, the weak hypothesis calculates the difference between the log-probabilities of $\rho(\mathbf{x}_+)$ and $\rho(\mathbf{x}_-)$. In the subsequent iterations, the $D_t(i)$ is changed based on the performance of the of $h_{j_{t-1}}$. Then, the examples that were missed in the last iteration are emphasized to minimize the probability that both $\rho(\mathbf{x}_+)$ and $\rho(\mathbf{x}_-)$ are present in bin $b = 1, 2, \ldots B$. Thus, the strong classifier simply gives us the differences of log-likelihoods for $\rho(\mathbf{x}_+)$ and $\rho(\mathbf{x}_-)$.

Since we are using random examples, the estimates for negative examples will be distributed all over the range $[0, 1]$. This is apparent from the histograms of estimates in Figure 4. The presence of estimates for negative examples in the intervals where the estimates of positive examples are concentrated, will pull down the confidence level of the corresponding weak hypothesis. Consider the scenario where, pixel $\mathbf{x}$ is a background pixel (and hence a positive example) but there is at least one weak classifier $h_r(\mathbf{x})$ in $H(\mathbf{x})$ that does not classify $\mathbf{x}$ correctly. If the confidence value of $h_r(\mathbf{x})$ is high and the rest of the weak classifiers had their confidence values reduced, the value of $H(\mathbf{x})$ may become negative. Therefore, we cannot rely on the theoretical bias of 0 anymore. We use a bias $\delta < 0$ and modify the strong classifier in equation 3 as follows:

$$H(\mathbf{x}) = sign\Big(\sum_{t=1}^{T} h_{j_t}(x) - \delta\Big). \tag{6}$$

This bias $\delta$ is the only control parameter in our algorithm. One method to determine the optimum $\delta$ is to change it's value iteratively until a certain rate of false positives is attained [1].

## 5 Experiments & Results

We implemented our algorithm on several indoor and outdoor (both color and gray level)image sequences. All the images used in the experiments had dimensions $320 \times 240$ pixels. In total 9 types of features, namely three color values R,G,B and spatial derivatives for each of these color channels in both $x$ and $y$ directions for each pixel of a color image, were used in this study. Some experiments only used a subset of these features. For intensity images, we used 3 features- the pixel intensity value and its spatial gradients in horizontal and vertical directions.

The first image sequence was a video taken from outside an office. The images in this video contains trees with moving leaves; cars and pedestrians on the road etc. We used $\mathcal{T} = 100$ kernel points for KDE. The first experiment was carried out with only color values R,G,B as features and with kernel bandwidths of $\Sigma_R \in \{40, 160, 200\}$, $\Sigma_G \in \{60, 160, 240\}$ and $\Sigma_B = \{60, 100, 240\}$. We generated one dimensional KDE estimates for all these color features with each of the bandwidths in the corresponding $\Sigma$ set. Therefore, if we wish to produce a strong classifier with $T$ weak classifiers, there can be $\binom{9}{T}$ possible combinations to select from. We used 200 frames to learn the RealBoost classifier, i.e. to learn which features are performing better than the others. In this experiment the negative examples are 400 samples from a uniform distribution over [0, 255] for color values.

We anticipate the boosting process to choose features with larger bandwidth estimates in the regions surrounding pixels a and b in Figure 1, and to select smaller bandwidth estimates in regions surrounding c. Figure 3, shows some sample classifier maps. Classifier maps are binary images $\mathcal{I}_C$ to describe the feature and parameter selection . For any feature/parameter combination, we set a pixel in $\mathcal{I}_C$ to 1, if the feature and bandwidth were chosen by RealBoost for that pixel. We are using $T = 3$ for the boosting process, i.e. each pixel can be chosen at most three times in all the classifier maps. Each classifier map is labeled by the feature name and kernel bandwidth at the bottom in Figure 3. As we can see (refer to Figure 1), in the low variance regions (e.g. the sky), boosting method selected weak classifier calculated using smaller bandwidth KDE estimates. Similarly, in regions with high variance in the feature values, weak hypotheses calculated with large bandwidth estimates were selected. Also notice that, the hypothesis using the green value as feature were selected more in number within the leaves of the tree and in the herb. The selection

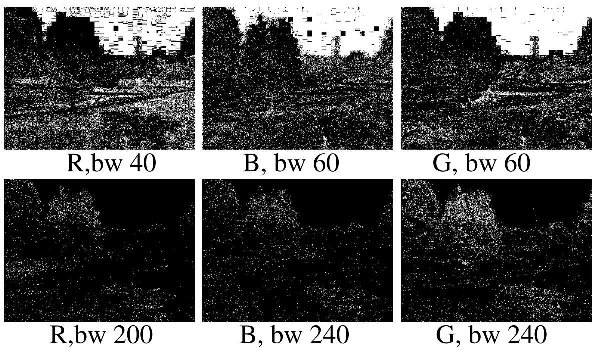

**Figure 3.** Classifier maps for the hypotheses chosen

of the hypotheses can be explained better by the plots of the histograms of densities of the $\rho(\mathbf{x})$ at pixel $(101, 31)$ given in Figure 4. The black and white bars represent the frequencies of $\rho_j(\mathbf{x}_+)$ and $\rho_j(\mathbf{x}_-)$ respectively in the corresponding bins. The number of bins used to build the histogram of $\rho(\mathbf{x})$ is $B = 10$ for all the experiments performed in this study. We observe that the lower bandwidth estimates

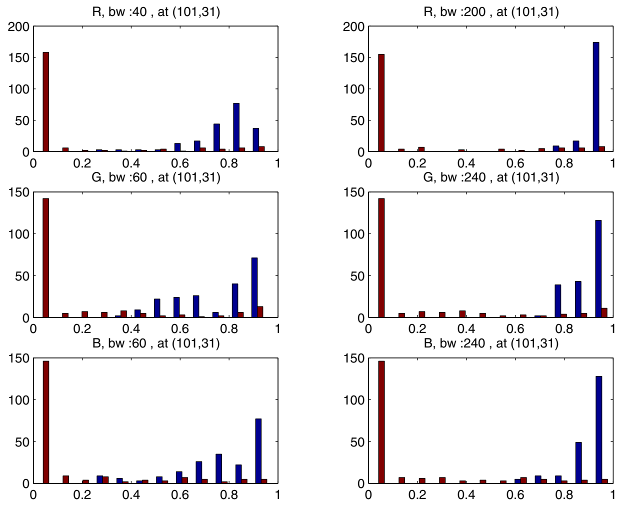

**Figure 4.** Distribution of KDE estimates of features with different bandwidths

are spread out in a wider range, $[0.4,\ 0.92]$ approximately, whereas the estimates of higher bandwidths are concentrated in $[0.8\ 1)$. This will make the value of the cost function $Z_j^t$ (defined in section 3) of the weak hypothesis constructed from higher bandwidth estimates lower than that of the others. Consequently, the weak hypothesis constructed from higher bandwidth estimates will be selected. Furthermore, among the features estimated with higher bandwidth, R has the least spread in its distribution of probability estimates. Therefore, feature R with kernel bandwidth 240 will be the feature to choose for pixel $(101, 31)$ provided that all weights of the training samples are equal.

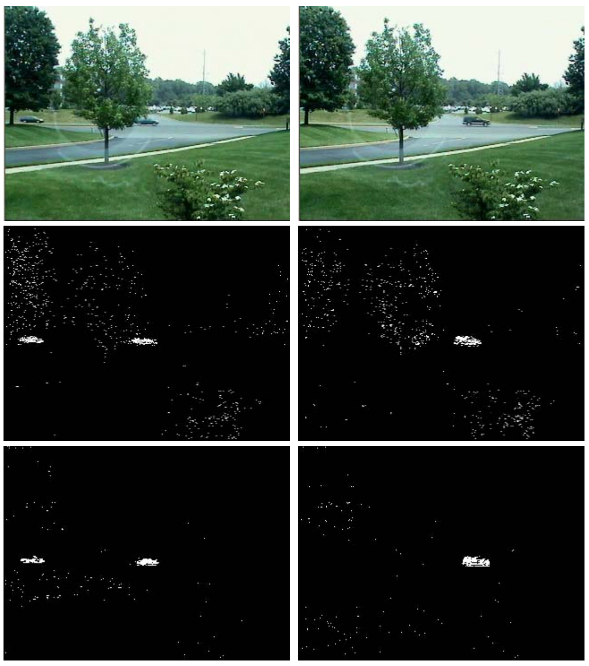

**Figure 5.** Detection results using R,G,B only. The second row uses the method of [11] and the bottom row shows the result of our approach.

We used the threshold value of $-13$ for the first sequence (at a false positive rate $5 \times 10^{-3}$ pixels/frame), which gives us the output shown in Figure 5, bottom row. It should be made clear that no post-processing operation (e.g. removal of connected components) was applied to any of the outputs of the experiments we performed. Inclusion of spatial gradients produced similar output for these images.

Figure 5 also compares the output using our approach with that of [11]. In [11], normalized color values along with the optical flow vector were used as features to generate the background model using Adaptive Kernel Density Estimation (AKDE). The approach of [11] was shown to be superior to the original KDE approach in [1]. For a fair comparison, the outputs of [11] were also displayed without any post-processing. Clearly, our method produces significantly fewer number of false detections than that of [11].

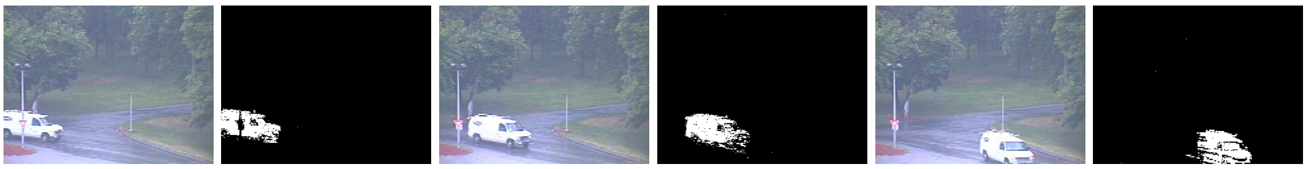

**Figure 6.** Outputs from a rainy outdoor sequence

To make a quantitative comparison, we used an image sequence without any foreground objects and imposed a synthetic object to pass over the regions where it is most likely to have foreground objects. We used a circle of radius 10 pixels as the synthetic image generated by the method described in [1]. Figure 7 depicts the ROC curves for both the algorithms. The proposed method supersedes the AKDE method both in terms of false positives and detection rate when used without the post-processing steps. The strategy of adaptive selection of elementary features for each pixel is shown to significantly outperform the method of background subtraction using adaptive statistical models with additional complex features.

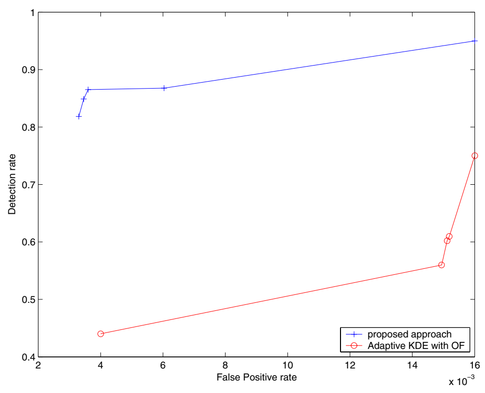

**Figure 7.** Receiver-Operator Characteristic (ROC) curves for results using proposed method and using the method in [11] (without post-processing)

We also tested our method with the same fetures, on an image sequence of a rainy day. For color channels R,G,B, three bandwidth triplets $\{140, 100, 100\}$, $\{40, 30, 30\}$ and $\{80, 80, 80\}$ were chosen to estimate their density. The first 50 images were used to learn the density and the next 170 images were used to learn the RealBoost classifier. A total of 400 negative examples were sampled from $[0, 255]$. Figures 6 and 8 shows some of the sample outputs and classifier maps respectively. We observe that the learning algorithm was able to segment the image into different regions that are most suitable for a certain feature. In this image sequence, we see that the hypothesis that uses B value as the feature and estimates the density with a kernel bandwidth 20 is most appropriate for the road and the hypothesis using G with the same bandwidth is appropriate for the trees and grasses. We keep changing the bias until a specific false positive rate is achieved. For this sequence, we had a false positive rate of almost zero for a bias of $-8$. The algorithm was next tested on an indoor sequence. The

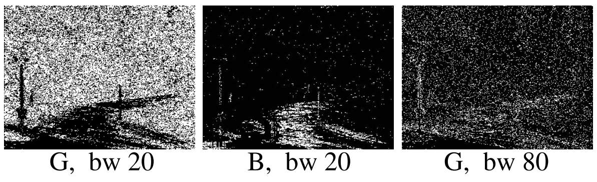

**Figure 8.** Classifier maps for the rainy sequence.

use of only R,G,B as features in indoor images gives rise to lots of false positive due to shadows. So, we tested our algorithm adding six spatial gradients of these three features. The result, shown in Figure 9, implies that use of spatial gradients suppresses the shadows totally. It also produces many false negatives within the body of foreground object. Since usually there is not much variation inside the region of a foreground object, the feature values of the pixels inside the forwground object region produces high estimates given the background model. The reason behind this is that, in this case, the background model for spatial gradients is centered around zero. The two sets of kernel bandwidth used were $\{50, 160\}$ and $\{40, 10\}$ respectively for color values and spatial gradients. The bias value used for the strong classifier is $-12$. The images used to learn the model and to choose the features were 80 and 120, respectively.

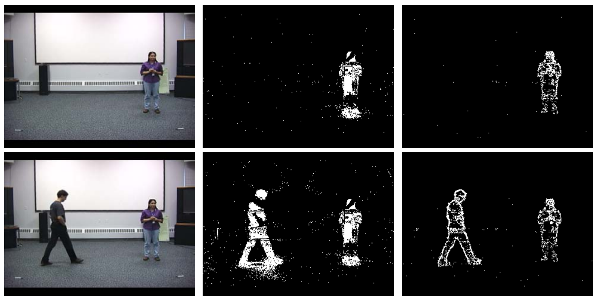

**Figure 9.** Results for the indoor sequence. Left- Input, middle- Output with R,G,B, right-Output with R,G,B + Spatial gradients.

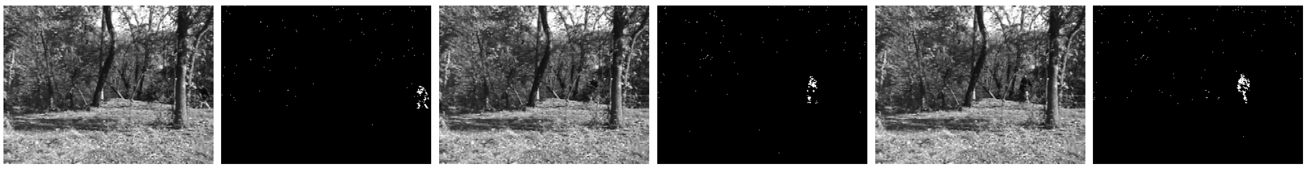

**Figure 10.** Intensity image output

Our final experiment was carried out on a sequence of gray level images taken in the woods. This sequence is also considered comparatively difficult since the foreground object often becomes occluded by tree branches and the background also has moving parts like tree leaves. Two sets of kernel bandwidths $\{15, 50, 150, 200\}$ and $\{60, 160, 200\}$ were used for the two types of features, namely intensity and spatial gradients. A bias of $-15$ for the final strong classifier gives the output as in Figure 10. Again, the boosting algorithm was able to pick a higher bandwidth for the upper regions with moving tree leaves and a lower bandwidth for the static portion of the image (Figure 11).

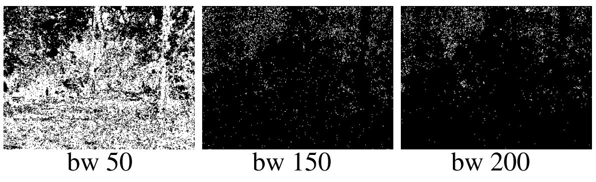

**Figure 11.** Classifier maps for the intensity image.

## 6 Conclusion

A new framework for background subtraction has been proposed which addresses the problem as a classification problem. The classification algorithm will be generalized in the sense that after it has been learned for the first time, it will be able to satisfactorily subtract background from the image unless there are some structural changes in the scene. The choice of the classifier facilitates combining different features, which is analytically more justifiable and has been experimentally shown to perform better than previous methods [1, 11].The learning method has been shown to follow a clear pattern to choose features which is interpretable in terms of the input image. However, the use of 1D features like spatial gradients introduces some negative examples which is inevitable in our experimental setup. As a future work, a combination of higher dimensional features with some additional constraints may be tried so that the adverse effect of one feature can be compensated by the contribution of the others.

## References

[1] A. Elgammal, D. Harwood, and L. S. Davis. Non-parametric model for background subtraction. In *6th European Conference on Computer Vision*, June/July 2000. Dublin, Ireland.

[2] 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.

[3] N. Friedman and S. Russell. Image segmentation in video sequences: A probabilistic approach. In *Thirteenth Conference on Uncertainty in Artificial Intelligence(UAI)*, Aug. 1997.

[4] X. Gao, T. Boult, F. Coetzee, and V. Ramesh. Error analysis of background adaption. In *CVPR*, pages I: 503–510, Hilton Head Island, SC, June 2000.

[5] M. Greiffenhagen, V. Ramesh, D. Comaniciu, and H. Niemann. Statistical modeling and performance characterization of a real-time dual camera surveillance system. In *CVPR*, pages II:335–342, Hilton Head, SC, 2000.

[6] W. Grimson and C. Stauffer. Adaptive background mixture models for real-time tracking. In *IEEE International Conference on Computer Vision and Pattern Recognition*, June 1999. Ft. Collins, CO.

[7] M. Harville. A framework for high-level feedback to adaptive, per-pixel, mixture-of-gaussian background models. In *ECCV*, page III: 543 ff., Copenhagen, Denmark, May 2002.

[8] O. Javed, K. Shafique, and M. Shah. A hierarchical approach to robust background subtraction using color and gradient information. In *MVC*, pages 22–27, Florida, Dec. 2002.

[9] K.-P. Karmann and A. von Brandt. *V Cappellini (ed.), Time Varying Image Processing and Moving Object Recognition*, volume 2, chapter Moving Object Recognition Using an Adaptive Background Memory. Elsevier, Amsterdam, The Netherlands, 1990.

[10] D. Koller, J. Weber, and J. Malik. Robust multiple car tracking with occlusion reasoning. In *ECCV*, pages 189–196, Stockholm, Sweden, May 1994.

[11] A. Mittal and N. Paragios. Motion-based background subtraction using adaptive kernel density estimation. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2004.

[12] A. Monnet, A. Mittal, N. Paragios, and V. Ramesh. Background modeling and subtraction of dynamic scenes. In *ICCV*, pages 1305–1312, Nice, France, Oct. 2003.

[13] N. Oliver, B. Rosario, and A. Pentland. A bayesian computer vision system for modeling human interactions. *PAMI*, 22(8):831–843, August 2000.

[14] R. Pless. Spatio-temporal background models for outdoor surveillance. *Journal on Applied Signal Processing*, 2005.

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

[16] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers. Wallflower: Principles and practice of background maintenance. In *ICCV*, pages 255–261, Kerkyra, Greece, Sept. 1999.

[17] C. Wren, A. Azarbayejani, T. Darrell, and A. Pentland. Pfinder: Real-time tracking of the human body. *PAMI*, 19(7):780–785, July 1997.

[18] J. Zhong and S. Sclaroff. Segmenting foreground objects from a dynamic, textured background via a robust kalman filter. In *ICCV*, pages 44–50, Nice, France, Oct. 2003.