Visual Tracking Lab 1

Master IN COMPUTER VISION AND ROBOTICS

Centre Universitaire Condorcet - UB, Le Creusot

18 October 2017

Submitted to: Prof Desire Sidibe

Submission by: Mohit Kumar Ahuja

1. Introduction

The goal of this Visual Tracking module is to learn about, and, more importantly, to learn how to use basic tracking algorithms and evaluate their performances. We will start with a very simple and effective technique called Background Subtraction which can be used to initialize the tracker, i.e. to find the target's position in the first frame of a sequence, or to track the target through the entire sequence.

1.1 **Background Subtraction**

Background subtraction (BS) is widely used in surveillance and security applications, and serves as a first step in detecting objects or people in videos. BS is based on a model of the scene background that is the static part of the scene. Each pixel is analyzed and a deviation from the model is used to classify pixels as being background or foreground.

As an example, you can see the car sequence in file "back_sub_car.m". We want to track the car in this sequence. We first needed to detect the car's position in the first frame of the sequence, or provide that location manually. If we have a model B of the static part of the scene, then moving objects can be detected in an image I, just by taking the difference I - B.

{Note: This report is the explanation of the code submitted for Lab-1 of Visual Tracking}

2. Part 1: Frame Differencing

In this method, the background model at each pixel location is based on the pixel's recent history. The history can be the average or the median of the previous n frames:

$$B_i(x;y) = median\ \{ I_{i-n+1}(x;y); I_{i-n}(x;y); :::; I_{i-2}(x;y); I_{i-1}(x;y) \}$$

So, the same has been implemented in Matlab as:

```
for i = 1:VIDEO_HEIGHT
    for j = 1:VIDEO_WIDTH
        I = median(ImSeq(i,j,:));
        Background(i,j) = I;     % Background Image
    end
end
```

A pixel belongs to the foreground if:

$$| I_i(x;y) - B_i(x;y) | > T$$

So, the same has been implemented in Matlab as:

```
for m = 1:size(ImSeq,3)
    M = ImSeq(:,:,m);                  % Current Frame
    Foreground = M- Background;        % Foreground Image without Threshold
    Foreground_Threshold = Foreground>25;% Foreground With Threshold
end
```

Conclusion: Frame differencing technique is differentiating Background and Foreground in every frame. Background Image is the median of all the pixels in every image as shown in figure1.

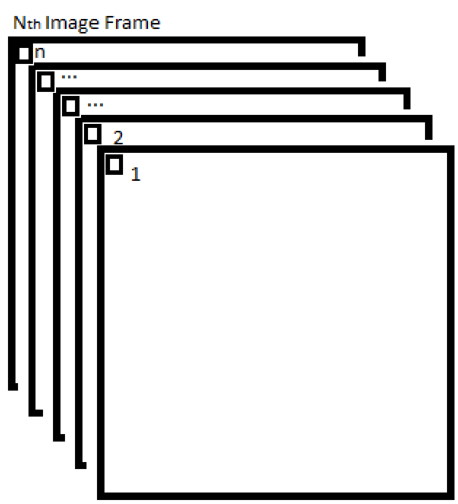

Figure 1: Computing Background Image

Foreground Image is computed using the Background Image and subtracting the current frame from the background which will lead us to only moving part of the image i.e. Foreground.

Results:

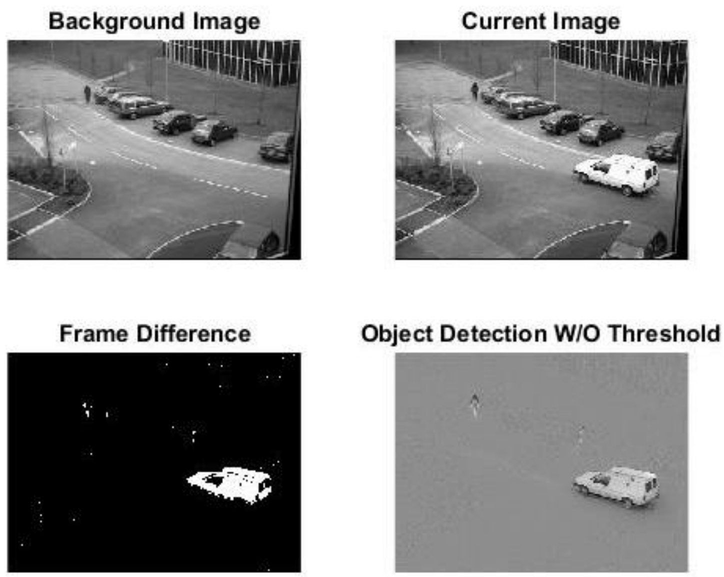

Figure 2: At Starting of the process: (a) Background Image, (b) Current Frame, (c) Object Detection using Frame Difference with threshold, (d) Object Detection without threshold

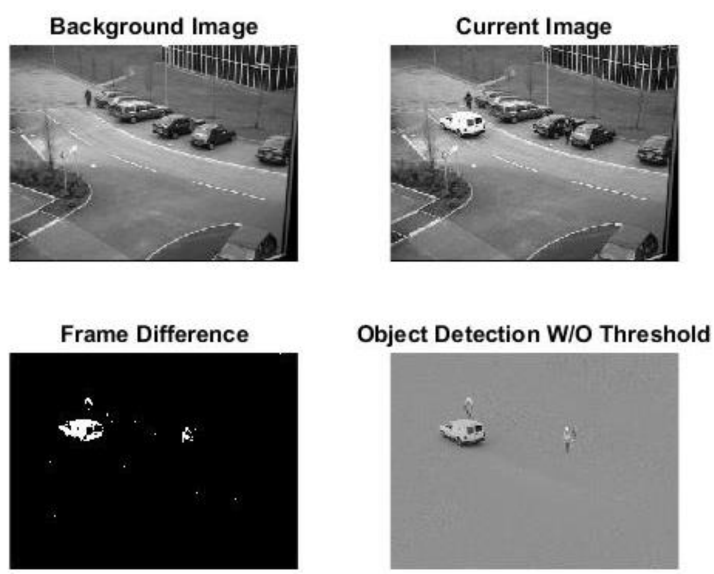

Figure 3: At the Ending of the process: (a) Background Image, (b) Current Frame, (c) Object Detection using Frame Difference with threshold, (d) Object Detection without threshold

3. Running Average Gaussian

In this method, each pixel's recent history is modeled as a Gaussian probability density function (pdf). In order to avoid fitting the pdf from scratch at each new frame time t, a running (or online cumulative) average is computed.

The pdf of every pixel is characterized by a mean and a variance. To accommodate for changes in the background, illumination variations or non-static background objects, at every frame t, each pixel's mean and variance is updated as follows:

$$Mean\,(x, y) = \alpha * I_t(x, y) + (1 - \alpha)\; Mean\,(x, y)$$
$$Variance\,(x; y) = d^2 * \alpha + (1 - \alpha)\; Variance\,(x; y)$$

So, in Matlab we initialized the mean and the variance first:

```
mean = ImSeq(:,:,1);  % Initialising  Mean
sigma = 1000*ones(VIDEO_HEIGHT,VIDEO_WIDTH);
```

Then we computed the mean and the variance for every frame in Matlab as:

```
for m = 1: size(ImSeq,3)

    mean(:,:,m+1) = alpha*ImSeq(:,:,m)+(1-alpha)*mean(:,:,m);
    d = abs(ImSeq(:,:,m)-mean(:,:,m+1));
    sigma(:,:,m+1) = (d.^2)*alpha+(1-alpha)*(sigma(:,:,m));

    final = d>2*sqrt(sigma(:,:,m+1));

end
```

Conclusion: In running average Gaussian method, we differentiate the foreground from background by using the equation:

$$\begin{cases} I(x,y) \text{ is foreground if } \dfrac{|I_t(x,y) - Mean(x,y)|}{Variance} > T \\ I(x,y) \text{ is background if } \dfrac{|I_t(x,y) - Mean(x,y)|}{Variance} < T \end{cases}$$

So, if the value of $\dfrac{|I_t(x,y) - Mean(x,y)|}{Variance}$ is greater than T then it will be considered as and foreground otherwise as background.

Result:

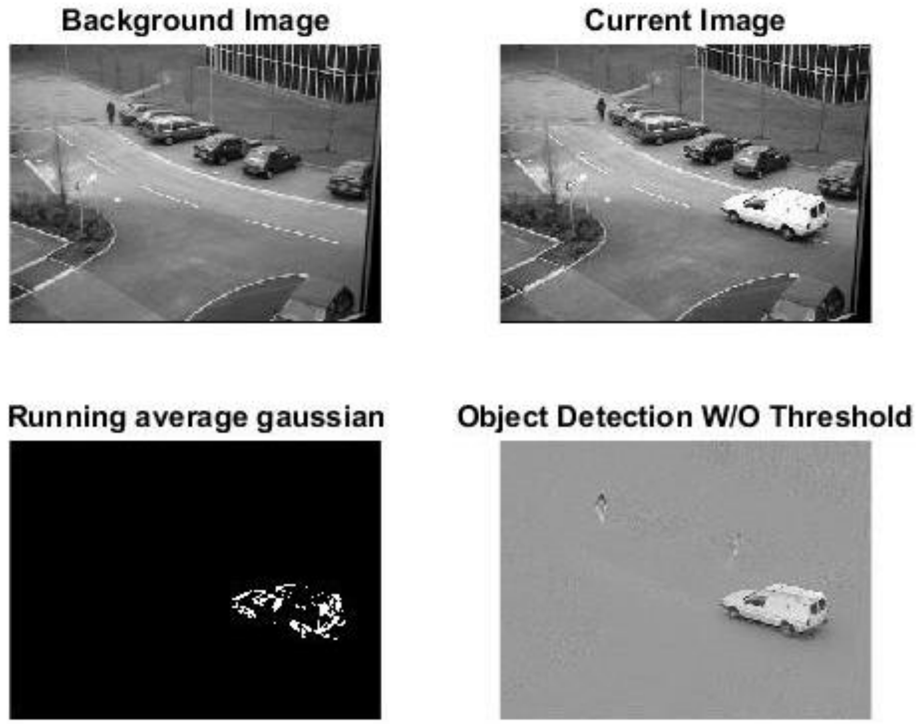

Figure 4: At Starting of the process: (a) Background Image, (b) Current Frame, (c) Object Detection using Running Average Gaussian, (d) Object Detection without threshold

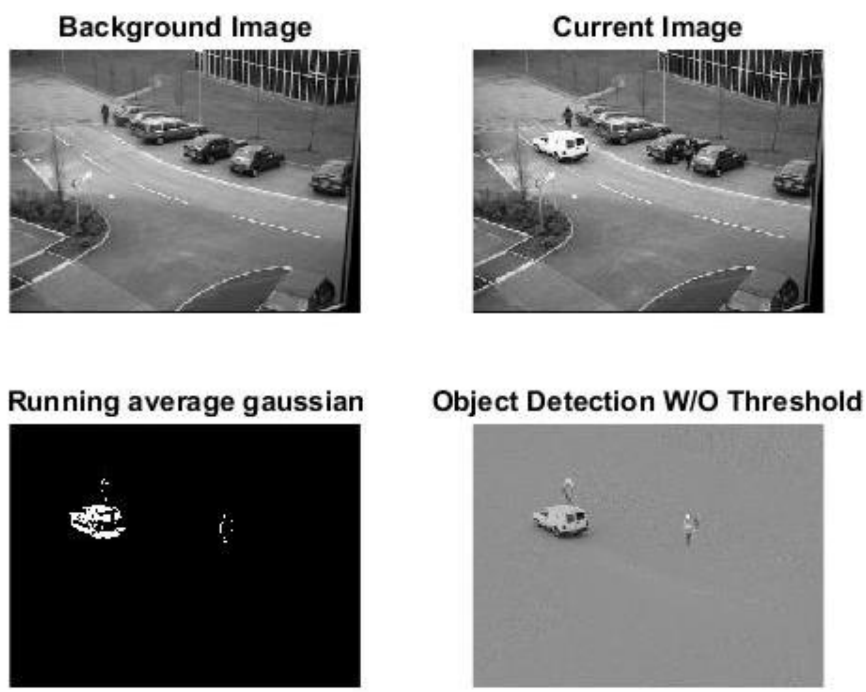

Figure 5: At Starting of the process: (a) Background Image, (b) Current Frame, (c) Object Detection using Running Average Gaussian, (d) Object Detection without threshold

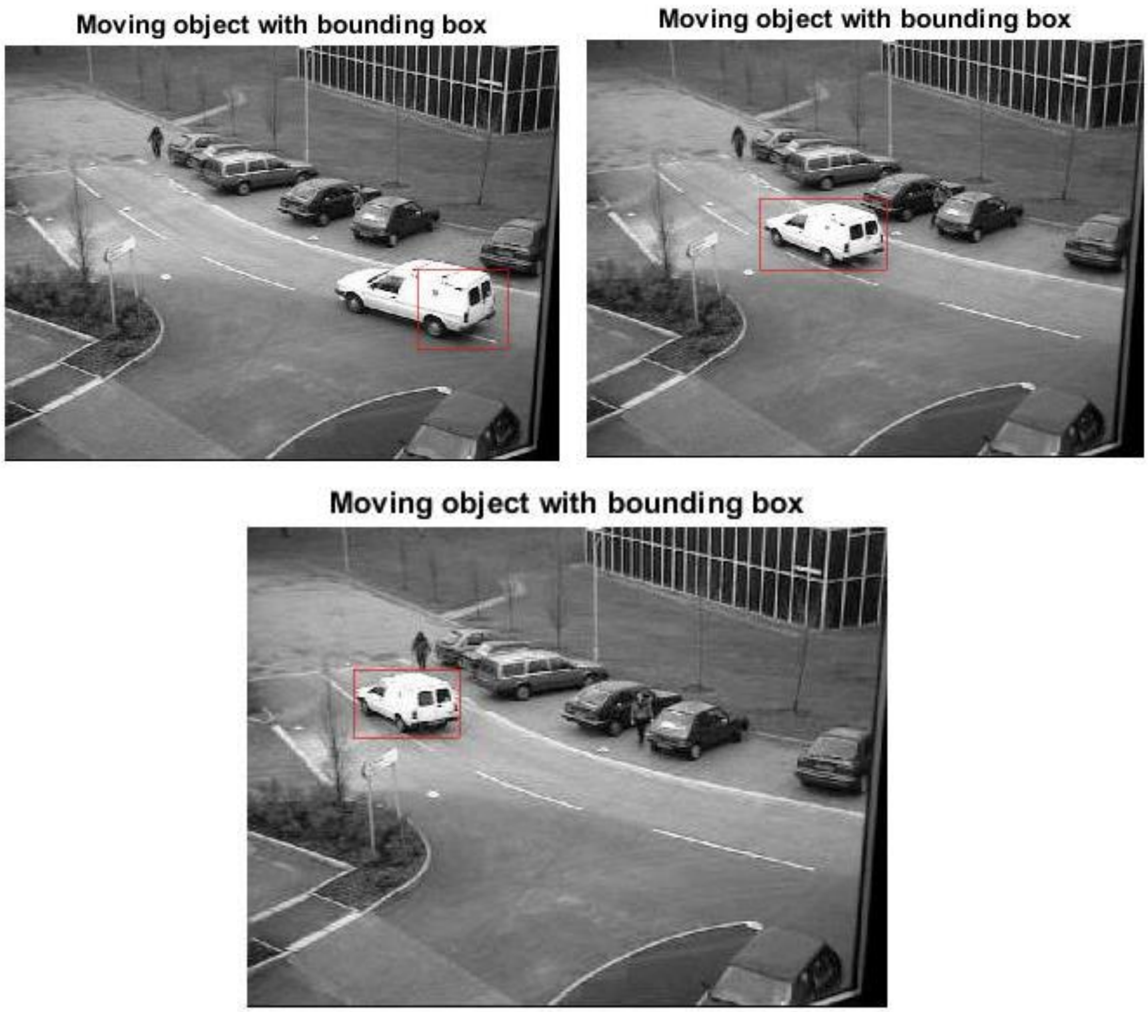

Figure 6: Object Detection using Running Average Gaussian with Bounding Box

4. Eigen Background

The Eigen-background model describes the range of variations in intensity values that have been observed by building an Eigen space that models the background. First, each frame It of the sequence is represented as a column vector Xt of dimension d = w*h, where w and h are the size of the images. The model is formed by taking a sample of N images. The mean image is computed as:

$$m = \frac{1}{N}\sum_{i=1}^{N} x_i$$

The mean-normalized image vectors are then put as column of a matrix X:

$$X = [x_1 - m, x_2 - m, \ldots\ldots, x_N - m]:$$

The columns of X corresponding to frames in the video, all lie in a low-dimensional subspace of Rd. This lower dimension space can be recovered by performing a singular value decomposition (SVD) of the matrix X:

$$X = U \sum V^T$$

So, I implemented this in Matlab as:

```
% The Mean Image
mean = 1/N*sum(x(:,1:N),2);
% Compute mean-normalized image vectors
Normalised_Mean = x - repmat(mean,1,size(x,2));
% SVD of the matrix X
[U, S, V] = svd( Normalised_Mean, 'econ');
```

Where U and V are orthogonal matrices and $\sum$ is diagonal matrix with singular values, in decreasing order, in its diagonal. We can approximates the subspace spanned by the columns of X, quite well, by considering as a basis only the first k columns of U, where k « N. This is also known as principal component analysis (PCA). The matrix U_k obtained by keeping the first k columns of U is called the Eigen-background. A new image y can be projected onto the reduced subspace as:

$$\hat{y} = U_k * p + m$$

Since U_k is orthogonal, p is easily obtained as

$$p = U_K^T (y - m)$$

So, I implemented this in Matlab as:

```
p = EIgen_Background' * (Project - mean);
Sub_Image = EIgen_Background * p + mean;
```

Finally, by computing and thresholding the absolute difference between the input image and the projected image, we can detect moving objects in the scene:

$$|\hat{y} - y\,| > T$$

So, I implemented this in Matlab as:

```
%Detect moving object
Mask = abs(Sub_Image - Project) > T;
```

Conclusion: Eigen Background method focuses on finding the mean and variance of the image and then use the normalized mean for singular value decomposition to obtain the Eigen background and to obtain the foreground, we compare the image with T, if the difference is greater than T then the object detected id from foreground otherwise not. And after all this, I applied some morphological functions to improve the quality of image for better detection like: eroding and dilating. And After that on the final output of the morphological functions, I applied the bounding Box on the image sequence.

Results:

Figure 6: Object Detection using Eigen Background with Bounding Box

5. Working With Highway Sequence

This is a sequence of 1700 color images of a highway, thus showing several moving cars. The sequence comes with ground truth segmentation of the moving objects in each frame. You can use the 470 first frame to initialize your background model. Then detect moving objects from frames 470 to 1700.

So in the code, I used first 470 images for detecting the background and then rest of the images are used for all other options. One of the major issue for doing this was also that we don't have ground truth for first 470 images.

I just added the ground truth path and added the ground truth images and rest of the code is the same.

I am using the same code which was used for "Car" sequence but with computing the precision, recall and Fscore. Nad also the sequence starts from 470 till end (1700).

5.1 Frame Difference

The same code is used for highway sequence except that the background has been detected form first 470 frames and the rest of the frames will be used for computing the F_Score.

```
for i = 1:VIDEO_HEIGHT
    for j = 1:VIDEO_WIDTH
        % Taking median of frame by frame
        I = median(Image_Seq(i,j,1:470));
        Background(i,j) = I;
    end
end
```

And to obtain the F_Score, I initialized all the parameters as zero and also used some morphological functions to improve the quality of the image and to improve the F_Score.

```
True_Positive = 0;True_Negative = 0;False_Positive =
0;False_Negative = 0;
% Moving object in every frame with applying threshold.

for m = 470:size(Image_Seq,3)

    M = Image_Seq(:,:,m);
    Foreground = M- Background;
    Foreground = Foreground>20;   % Thresholding
    imshow(Foreground,[]);
    title('Moving object Detection with Frame Difference');

    Foreground = imfill(Foreground,'holes');
     Foreground = imopen(Foreground, strel('rectangle', [2
     2]));
```

```
    True_Positive = True_Positive + nnz(Foreground &
    Ground_Truth_Seq(:,:,m));
    True_Negative = True_Negative + nnz(~Foreground &
    ~Ground_Truth_Seq(:,:,m));
    False_Positive = False_Positive + nnz(Foreground &
    ~Ground_Truth_Seq(:,:,m));
    False_Negative = False_Negative + nnz(~Foreground &
    Ground_Truth_Seq(:,:,m));
end

disp('%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%');
disp('<------ For Frame Difference ------>');
precision = True_Positive/(True_Positive+False_Positive)
recall = True_Positive/(True_Positive+False_Negative)
F_Score = 2*precision*recall/(precision + recall)
disp('%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%');
```

Conclusion: So, by using the morphological functions, it improved the fscore as well as improved the quality of the image. By doing this the output of the system is :

%%
<------ For Frame Difference ------>
precision = 0.7710
recall = 0.1346
F_Score = 0.2291

%%

5.2 Running Average Gaussian

The same code is used for highway sequence except we start the sequence from image number 470. Also the formulas are the same as used in Frame difference as shown below:

```
True_Positive = 0; True_Negative = 0; False_Positive = 0;
False_Negative = 0;
for m = 470: size(Image_Seq,3)
    %Mean
    mean = alpha*Image_Seq(:,:,m)+(1-alpha)*mean;
    %Variance
    d = abs(Image_Seq(:,:,m)-mean);
    sigma = (d.^2)*alpha+(1-alpha)*(sigma);

    final = d>2*sqrt(sigma);
    imshow(final,[]);
    title('Running average gaussian');

    final = imfill(final,'holes');
    final = imopen(final, strel('rectangle', [2 2]));
    True_Positive = True_Positive + nnz(final &
    Ground_Truth_Seq(:,:,m));
```

```
    True_Negative = True_Negative + nnz(~final &
    ~Ground_Truth_Seq(:,:,m));
    False_Positive = False_Positive + nnz(final &
    ~Ground_Truth_Seq(:,:,m));
    False_Negative = False_Negative + nnz(~final &
    Ground_Truth_Seq(:,:,m));
end

disp('%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%');
disp('<------ For Running Average Gaussian ------>');
precision = True_Positive/(True_Positive+False_Positive)
recall = True_Positive/(True_Positive+False_Negative)
F_Score = 2*precision*recall/(precision + recall)
disp('%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%');
```

Conclusion: So, by using the morphological functions, it improved the fscore as well as improved the quality of the image. By doing this the output is:

%%%
<------ For Frame Difference ------>
precision = 0.8102
recall = 0.2880
F_Score = 0.4250
%%%

5.3 Eigen Background

The same code is used for highway sequence except we start the sequence from image number 470. Also the formulas are the same as used in Frame difference and the output of the system is shown down:

%%%
<------ For Frame Difference ------>
precision = 0.3422
recall = 0.2652
F_Score = 0.2988
%%%

Result:

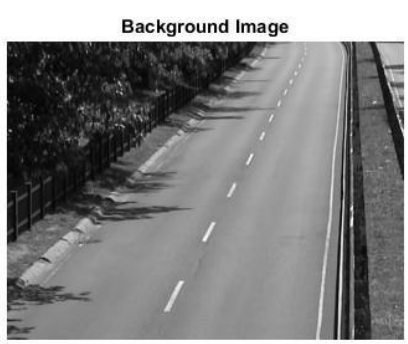

Figure 7: Background Image of Highway sequence

For Frame Difference:

Figure 8: Frame Difference

For Running Gaussian Average:

Figure 9: Running Average Gaussian

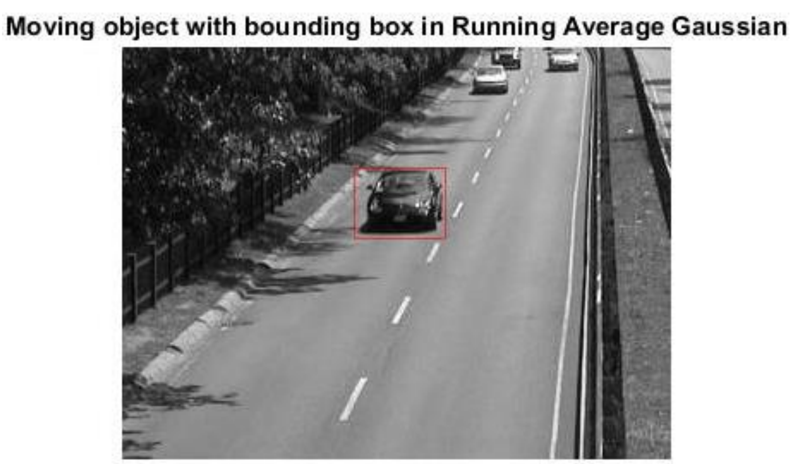

Figure10: Moving object detection using Running Average Gaussian with Bounding Box

Figure11: Moving object detection using Running Average Gaussian with Bounding Box

Eigen Background:

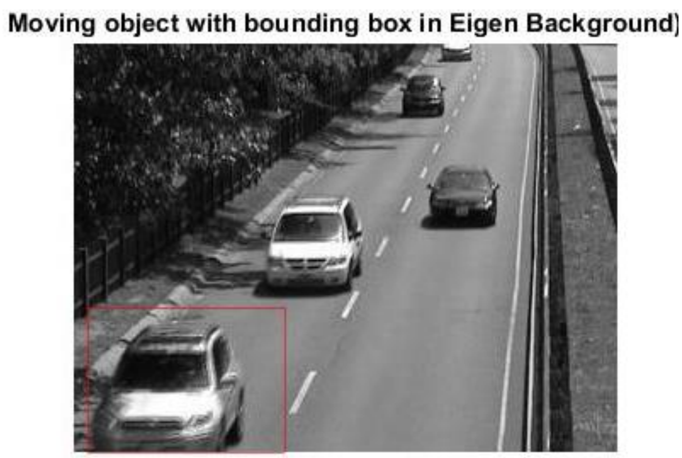

Figure12: Eigen Background

Chart For comparison:

	Frame Difference	**Running Gaussian**	**Eigen Background**
Precision	0.7710	0.8102	0.3422
Recall	0.1346	0.2880	0.2652
F_Score	0.2291	0.4250	0.2988

{Note: Code is fully commented for better understanding of the reader}