# Visual Tracking
# Background Subtraction
# Université de Bourgogne

Sandeep Manandhar

October 7, 2015

## 1 Frame differencing

In this method, we prepare a background model for each pixels from the median of past few frames.
We stack last few frames of the sequence and calculate median of each pixels. As we can notice in fugure 1, the median pixel will probably be the one with high number of membership in the stack. The background, as it is supposed to be in the scene through out the sequence, its pixel indeed will have more membership in the stack. This is how we model a background pixel. After creating the median image, we simply subtract the incoming frame from this image to separate foreground objects. Then we apply various morphological operators like opening to delete small objects, dilate and fill to close holes in objects.

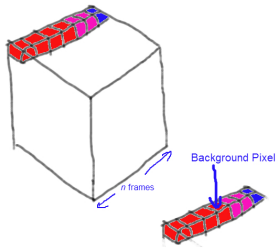


Figure 1: Median for each pixels across frames

We have tested the algorithm by varying the number of frames to create the background median image. The figure 2 shows a test case of taking last 110 frames. We can see the artifacts on background image. The far end of the highway has more cars which seem to move slower. Hence the foreground pixels in that region ought to remain there for longer period increasing their membership in the stack. So we can see the pixels from the car showing up as the background pixels in our model.

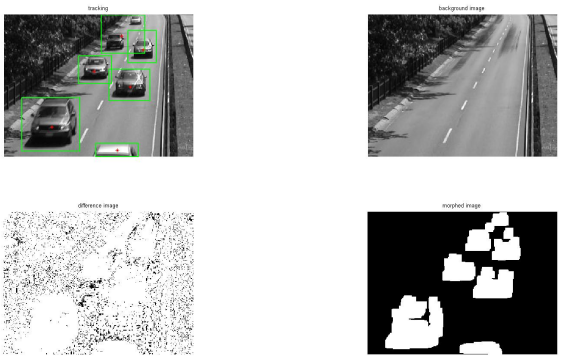


Figure 2: 1)Tracking foreground objects 2) Background model 3) frame difference 4) after morphological operation

(a) 5 previous frames (b) 15 previous frames (c) 55 previous frames (d) 100 previous frames

Figure 3: Background model with different number of frames

Figure 3a is created using just 5 frames. Here, we have more foreground pixels embedded as background pixels. As we increase the number of frames, this misclassification of pixels in background model itself diminishes.
Instead of calculating median of last frames, we also tried updating an initially created background model. This decreases the computation time of the algorithm. Further, we have applied few morphological operators like matlab's bwareaopen, imdilate, imclose, etc. to remove the outliers and create bounding boxes. The best F-score that we received for this method during our trial was 78.

## 2 Running Average Gaussian

In this method, we create a gaussian model for each pixels and test the new pixels against this *pdf* and classify as a background or a foreground pixel. The mean and variance is update at each frame. The results rely on the initialization of mean and variance. For the tests, we have used 0 mean and unit variance. The $\alpha$ parameter determines the learning rate. As we can see, many of the background

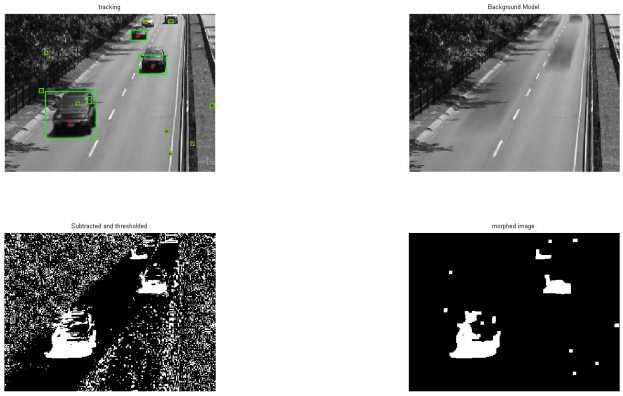


Figure 4: 1)Tracking foreground objects 2) Background model 3) thresholded 4) after morphological operation

pixels have been mis-classified as foreground pixels (in figure 5 (3).The reason could be, we have used a unimodal gaussian. The background has been modeled as the outliers of the foreground pixels. So depending upon a global threshold, we classify the background and foreground pixels. We do not consider separate gaussian models for foreground and background pixels. The Fscore that we

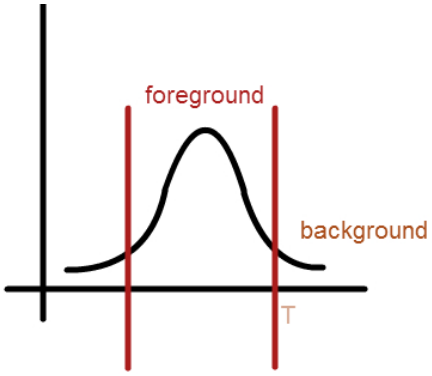


Figure 5: Gaussian modeling of image pixel; T determines the classification

received upon threshold of 1.05 was 47.

# 3 Eigen Background

In this method, we use a number of frames to create eigen space. Using certain number of eigen vectors, we create a background model and test it against incoming frames. As we know that, vectors corresponding to higher eigen values encode subtle changes(low frequencies), we use these vectors to create the background model. In figure 6, we depict the backgrounds created from differ-

(a) 1-2 eigenvectors (b) 1-10 eigenvectors (c) 100-200 eigenvectors (d) 300-400 eigenvectors

Figure 6: Background model with different ranges of eigen vectors

ent ranges of eigenvectors. The first two vectors (6a) model a background with subtle changes. During the sequence, the background in almost static. The first 10 vectors (6b) model backgrounds with shadow-like patches. Even though the model during the sequence is dynamic, these shadow-like patches are smooth. The higher vectors (6c and 6d) model backgrounds with fringe like structures and highly changing during the sequence. With high number of eigen vectors to create background model, foreground objects can be separated in more dynamic scene.
The scene we have used is not much dynamic and we have used just two first eigenvectors to gain Fscore of 80.

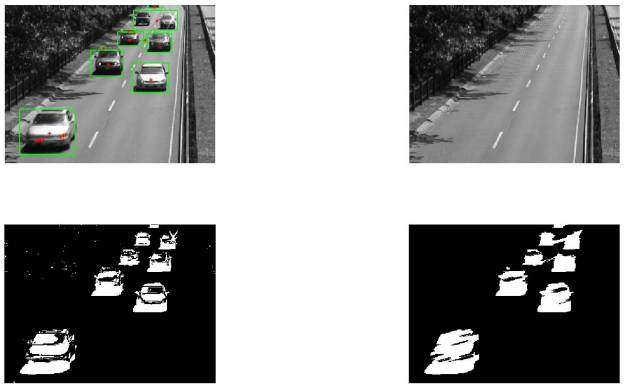

Figure 7: 1)Tracking foreground objects 2) Background model 3) thresholded 4) after morphological operation

### 3.1 PCA space

Since the dataset provided is huge ($240 \times 320 \times 470$ images for background model), the PCA calculation is not straightforward. We have calculated the eigen components of $A'A$ instead of $AA'$. This will create a covariance matrix of $470 \times 470$ instead of $76800 \times 76800$.

$$A'Av = \lambda v \tag{1}$$

Multiplying by $A$ on both sides, we get

$$AA'(Av) = \lambda(Av) \tag{2}$$

As we can see in 1, $v$ is an eigen vector of $A'A$. Then from 2, we can say $Av$ is an eigen vector of $AA'$. The eigen vectors thus obtained are not normalized as they have been stretched by $A$. To normalize then, we divide them by singular values of $A$ i.e. $\lambda^{0.5}$. Hence, we can get back to our original eigen vectors by reduced number of multiplications and additions.

## 4 Morphological operation

Various morphological operations have been used to filter and determine the ROI of objects to be tracked. Depending upon these operations, we can tweak the tracking of foreground objects and improve the Fscore. Following Matlab's functions have been employed for these operations.

1. bwareaopen(Image, n) - This function removes the connected component with less than n pixels. This function has been used to remove small background objects which appear in lump as well as the pixels joining the cars when they are nearby.

2. imfill - This function with 'holes' as parameters, fills the holes in the background image.

3. imerode - This function erodes an image with elementary components eg. disk, square. This function has been used to remove salt noise from the thresholded images.

4. imdilate - This function dilates an image with elementary components eg. disk, square. This functions has been used to increase the presence of foreground pixel which can increase the F-score to certain extent.

5. regionprops - This function finds the centroid and the bounding box of connected components in our morphed images.