COL780
Assignment 1

Avani Jain
2020MT10792

1 Method for Background Subtraction model

At first, I have converted the colored image frames to grayscale as the results observed were better with grayscale, using the standard weighted formula,

$$Grayscale = 0.299R + 0.587G + 0.114B$$

The assignment implements the Gaussian Mixture Model for Background Subtraction with exponentially decaying weights for the past values (Y=0). Each pixel is considered mutually independent of all other pixels, and a sum of gaussians is used to model it.

$$Pr(x) = \sum_{k=1}^{K} w_k \eta(x; \mu_k, \sigma_k)$$

where, $\sum_{k=1}^{K} w_k = 1$.
Intensity distribution of every point at any time t, is modeled as a GMM and online K means approximation is used to update the parameters.

At each pixel, using the previous values, distribution parameters are learned, and they are used to classify the new value, whether it belongs to background or foreground.

$$\{x^1, x^2, \dots x^t\} \implies w_i, \mu_i, \sigma_i \ \forall i = \{1, 2, \dots k\}$$

x^i are pixel values.
Initially, it has been assumed, $w = 1$, $\mu = x$, and a large value of σ, then, iteratively, it is checked if x_t matches any existing gaussian.

$$M_k(x_t) = \begin{cases} 1, & \text{if } \frac{x^t - \mu_k^t}{\sigma_k^t} \leq 2.5 \\ 0, & \text{otherwise} \end{cases}$$

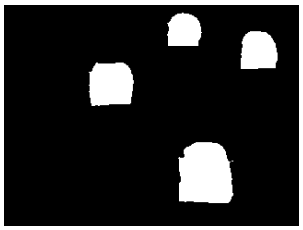
- If there is a match, the parametes are updated,

$$(w_k^{t-1}, \mu_k^{t-1}, \sigma_k^{t-1}) \Rightarrow (w_k^t, \mu_k^t, \sigma_k^t)$$

- If not, and if there already exist K gaussians, the gaussian with lowes weight is removed and replaced with a new gaussian, having same weight, and the new value as a mean, and high variance

For the background subtraction process, the k gaussians are sorted in decreasing order of the ratio of their weight and standard deviation. And first few gaussians are selected such that the sum of their weights is less than the threshold value, T_b. Now, if x^t matches some gaussian that is background, then the pixel is classified as background, or else foreground.

(a) Input



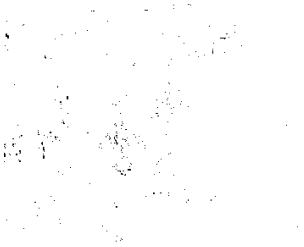
(b) Groundtruth

(c) Result

Figure 1

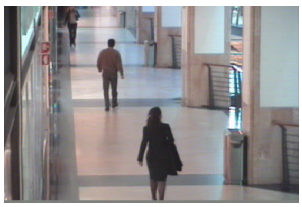
(a) Input

(b) Groundtruth



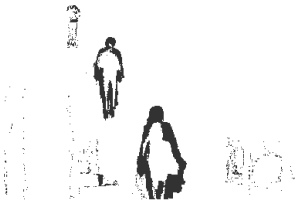
(c) Result

Figure 2



(a) Input

(b) Groundtruth



(c) Result

Figure 3

2 Failure cases

- It fails to remove shadows of the foreground objects. A comparison of groundtruth and the result obtained is as shown in Figure 1 for the input dataset HighwayI. It is because shadows are also moving in the same was as an object, and hence the model is not biased towards it.

- In the absence of any movement of the object in the input, the model learns a lot of noise and classifies it as foreground and forgets the object that was initially classified as foreground. This is expected as a result of the algorithm that subtracts the background by looking for changes in pixel values and as the changes in the pixel values stop, the person is no longer classified as foreground.
  A comparison of groundtruth and the result obtained is as shown in Figure 2 below for the input dataset Candela_m1.10.

- The model fails to show the person as a whole in the dataset CAVIAR1 as show in 3 and just shows the outline of that person as a foreground object. This is because, in the input, these humans are darkened, only a figure of them is observed, and hence the model fails to detect any change in the pixels at the centre of the body because there is no change in pixel intensity there. It is again an expected failure as a result of the algorithm used.

- The model also blackens or treats as foreground, the tiny blobs of pixels that cover the leaves moving in background due to wind, or some noise in video due to light changes as with dataset Candela_m1.10.

3 Analysis of Comparison with other method

The results achieved are compared with the Zuhaib Ul Zamann's results, who had Y = 2. I had Y = 0, i.e, exponentially decaying weights for the past values, and Zuhaib had Y=2, i,e, kernel-density based method with decaying weights

- The results obtained by Zuhaib's method has more noise compared to the method that I worked on, as is apparent from all the below figures, 4, 5, 6, 7. This could be due to some error, or high learning rate, or maybe due to the nature of the algorithm itself.

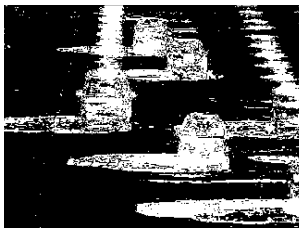
- In some cases the method other than mine takes time to forget old classifications (this can be seen in Figure 4), and in some cases, my method fails to forget quickly. This memory property shouldn't vary much for both the cases though, since both of them use exponentially decaying weights. So, high chances that this variation is just a difference in hyperpaprameter tuning.

- The method used by Zuhaib gives more finished boundaries at some places compared to mine. Please refer to Figure 6 to observe this. This is also explainable by the nature of algorithms used, a kernel density based method is expected to give better results.

In all, both the methods use exponentially decaying weights, just that a kernel-density based method as implemented by Zuhaib, gives better approximations and predictions.

In kernel density based approach, pixel probability density function is given by $\mathrm{E}[Kernel(x_i, \Sigma)]$, where x_i is the pixel intensity of the ith frame, calculated using exponentially decaying weights. Whereas, in the other approach, pixel probability density function is just taken as a normal distribution.
The kernel bandwidth, Σ reflects the local variance in pixel intensity due to the local variation from image blur and hence using this method now, the pixel intensity changes largely reflect the changes in objects.
Hence the comparisons make sense as kernel does smoothens the results for better predictions, or in other words is a better predictor of the population based on a few samples. Hence, statistically, the kernel based approach is expected to give more sharp boundaries.

(a) Y = 0



(b) Y = 2

Figure 4

(a) Y = 0

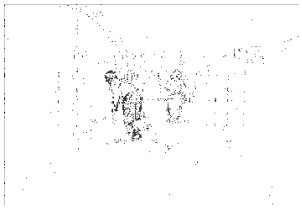
(b) Y = 2

Figure 5

(a) Y = 0

(b) Y = 2

Figure 6



(a) Y = 0

(b) Y = 2

Figure 7

4 Method for Foreground Pixel Aggregation

The model then uses the method of foreground pixel aggregation first, which starts by evaluating integrals of images using the dynamic programming induced formulas as derived in the class. And then using a value of "s", or size such that the whole picture is divided into small sxs patches. After that, a check is done for the value of the pixels in that patch, if the value is above the threshold, then the whole patch is declared as foreground, else background.

The other cleaning method used after this is that of bounding boxes. In which again using the calculated values of integrals of images, non maximal suppression is used to find the bounding rectangles for foreground objects.
Patches of size l x b are taken. Then for every possible rectangle in the image, sum of pixel values is calculated in that rectangle. Then the rectangles are filtered, all those having the score or pixel sum above a certain threshold are kept, and rest discarded. In this way, a list of all possible rectangular boxes is maintained. After that NMS or non maximal supression is used to get the most probable rectangles.
It involves following steps -

- Picking the rectangle having largest score value
- Removing all rectangles which have intersection over union above a threshold with the chosen rectangle
- Repeating this until all rectangles are either chosen or removed

5 Video links of Generated Outputs

Click on this link for the videos

The videos having "part-1" in their name are the results obtained by running just the background subtraction algorithm on the datasets, i.e, after doing the question 1 of the assignment.
The videos having "part-2" in their name are the results obtained after cleaning the previous results using foreground aggregation method, i.e, after doing the question 3 of the assignment.

All the code files as well as the Jupyter notebook, alongwith the requirements.txt file, have been included in the main folder.