Assignment 3: Real Time Tracking

Indian Institute of Technology, Gandhinagar

Patel Vrajesh
20110134
Google Drive : Link

Problem Statement: Implement the real-time object tracking algorithm using background subtraction and a Mixture of Gaussians.

Dataset - (Waving trees+moving car) The video attached in this Link was considered for background modeling. The resolution of the video is (240*426*3) where height=240, width=426, and number of channels=3.
The video is around 5 seconds and the total number of frames captured is 138. The frames per second (fps) captured was around 30. **The reason for choosing this video is to also highlight the limitation of this algorithm (unstable camera) along with the application.**

Methodology

1) The input video was captured using opencv2 to extract frames out of it. A total of 138 frames were captured for a 5 second video. Resolution of each frame is 240*426*3.
2) For each image location, we stored pixel values across all the frames in an array and passed it as input to the Gaussian_Mixture function to represent the pixel values as a mixture of two Gaussians (K=2).
3) One of the Gaussians will represent the background and the other will represent the foreground of a particular pixel across all the frames. In general, the gaussian with the highest weight represents the background and the rest of the gaussians represent the foreground.
4) As the background remains the same for most of the frames, the mean of the gaussian that has more weight and less standard deviation is used for constructing the background.
5) To track the object, we subtract this background from each frame.
6) Now, I have saved the output frames in a separate folder.
7) Using the builtin GMM, MOG, and MOG2 functions, I created another three sets of output images to compare with.

Algorithm (Gaussian Mixture Model)

1) Represent the pixel values across all the frames as a mixture of two Gaussians.
2) Start by initializing the values of mean and variance for both the gaussians.

3) Calculate the likelihood of each observation x_i using the initial values of mean and variance.
4) For both the gaussian clusters, calculate our data's probability density (pdf) using the initial mean and variance values.

$$f(\mathbf{x}|\mu_k, \sigma_k^2) = \frac{1}{\sqrt{2\pi\sigma_k^2}} \exp(-\frac{(\mathbf{x}-\mu_k)^2}{2\sigma_k^2})$$

5) Then, calculate the likelihood of a given example x_i to belong to the kth cluster.

$$\mathbf{b}_k = \frac{f(\mathbf{x}|\mu_k, \sigma_k^2)\phi_k}{\sum_{k=1}^{K} f(\mathbf{x}|\mu_k, \sigma_k^2)\phi_k}$$

6) Using Bayes Theorem, get the posterior probability of the kth Gaussian to explain the data. That is the likelihood that the observation x_i was generated by kth Gaussian. I have initialized the weights to 1 to one of the gaussians.

7) In the next step, re-estimate the learning parameters as follows.

$$\mu_k = \frac{\sum \mathbf{b}_k \mathbf{x}}{\sum \mathbf{b}_k} \qquad \sigma_k^2 = \frac{\sum \mathbf{b}_k (\mathbf{x}-\mu_k)^2}{\sum \mathbf{b}_k} \qquad \phi_k = \frac{1}{N}\sum \mathbf{b}_k$$

8) Repeat the same process until the values of mean, variance and weights of both the Gaussians converge.

Limitations

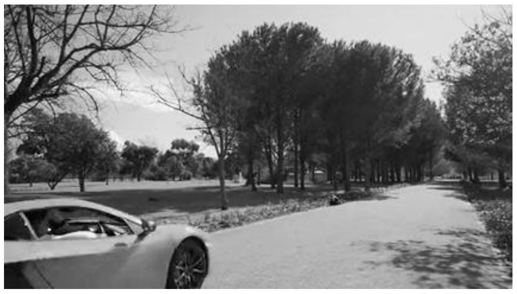
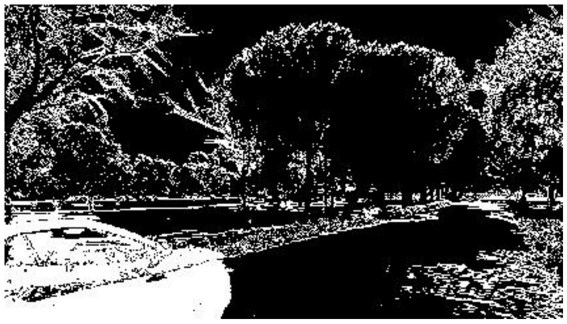
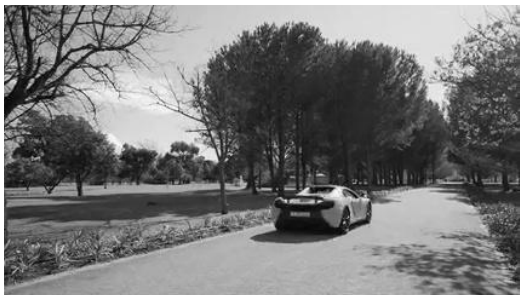
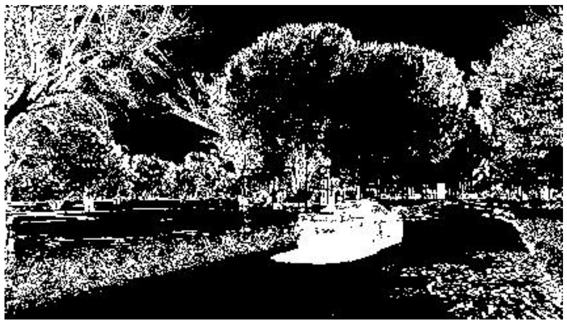
- The implementation is not an optimized one in terms of time-complexity(took me around an hour for a 5 second video) and is only intended to deliver the concept.
- It will not give accurate results when the device capturing video is not stable. As it can be seen here that due to slight instability, it detects the leaves of the trees. This sometimes works in our favor if we have a stable setup, it will capture even the slightest of the movement.
- The higher fps tentatively creates a copy of the same moving object in the video. Reducing the fps helps to reduce this issue. Also the time period of processing will also decrease if fps are reduced.

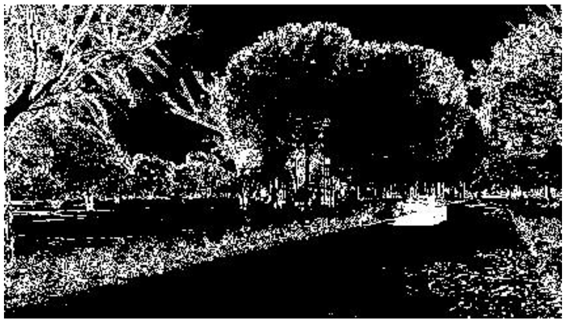
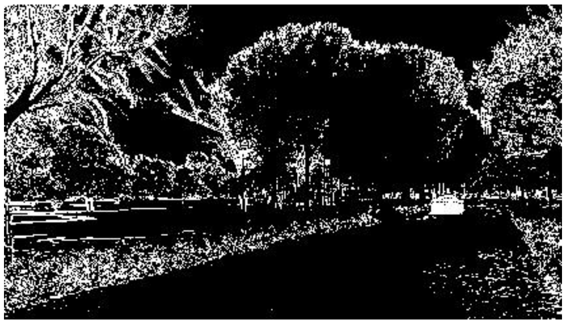
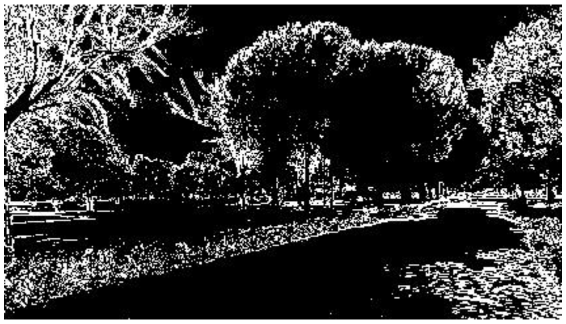
Results

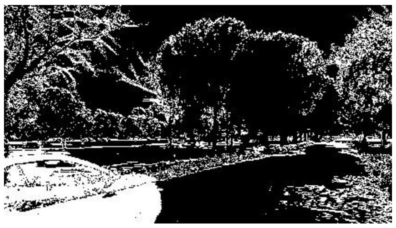
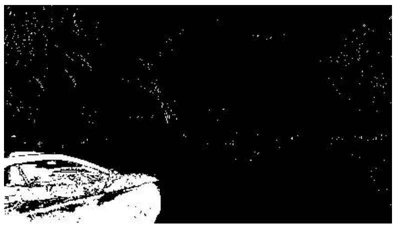
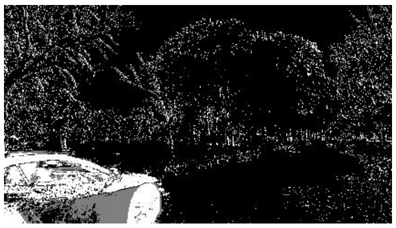
Link of MOG video: Link
Link of MOG2 video: Link
Link of output video: Link

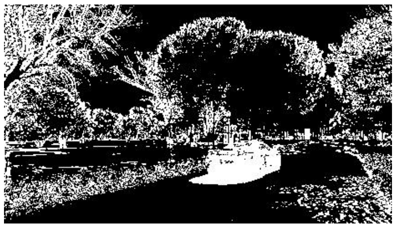
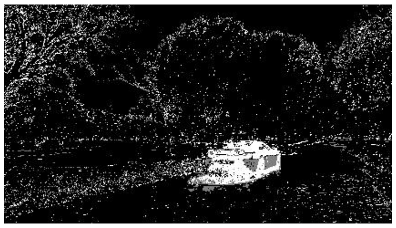
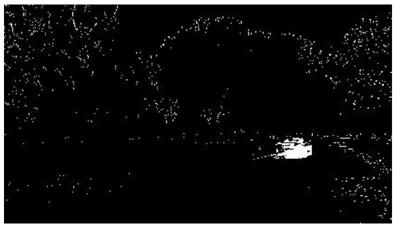
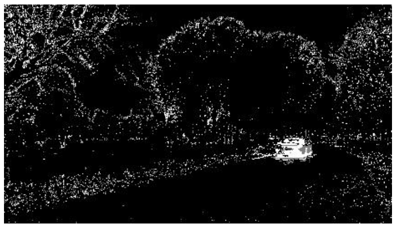
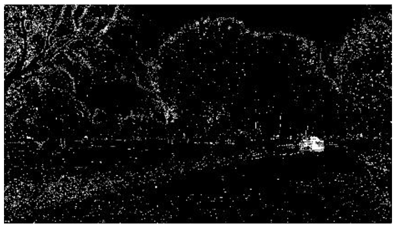
It can be observed that the built-in MOG and MOG2 try to nullify the effect of slight change in environment. However, the algorithm proposed in the paper detects the finest of changes. The proposed algorithm took too much of a time(around 1.5 hours) to give the final output video while the MOG and MOG2 were able to do the background modeling in about one minute. Videos with higher fps might be very time and memory intensive when we use the proposed algorithm.

Some of the video frames and their detected foregrounds are shown below
Kindly zoom for better visibility.

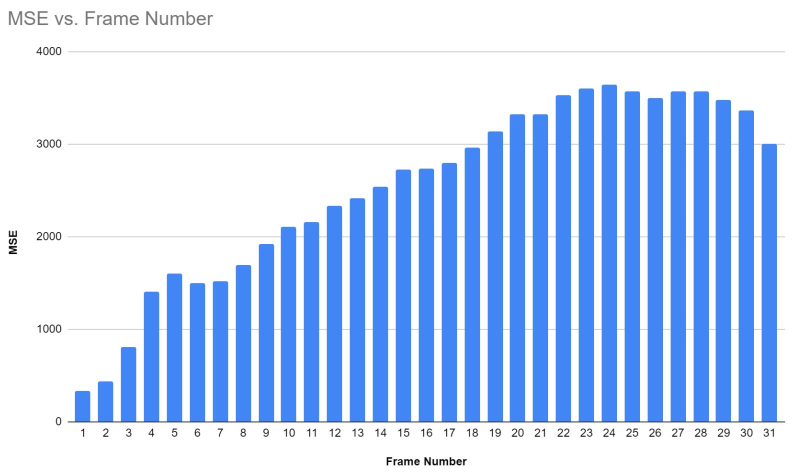
Frame No.	Video Frame(Original)	Modeled Frame
20		
40		

60		
80		
100		

Output frame using my algorithm	**Output frame using in-built MOG**	**Output frame using in-built MOG2**
		

I have considered MSE as a parameter of accuracy.
MSE is calculated for the first 31 frames when compared with MOG2:



References:

- C. Stauffer and W. E. L. Grimson, "Adaptive background mixture models for real-time tracking," Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149), 1999, pp. 246-252 Vol. 2, doi: 10.1109/CVPR.1999.784637.
- Thierry Bouwmans, Fida El Baf, Bertrand Vachon. Background Modeling using Mixture of Gaussians for Foreground Detection - A Survey. *Recent Patents on Computer Science*, Bentham Science Publishers, 2008, 1 (3), pp.219-237. ⟨hal-00338206⟩
- https://www.sciencedirect.com/science/article/abs/pii/S1574013718303101?via%3Dihub
- https://medium.com/@prantiksen4/background-extraction-from-videos-using-gaussian-mixture-models-6e11d743f932
- https://www.geeksforgeeks.org/background-subtraction-opencv/
- https://www.geeksforgeeks.org/python-opencv-background-subtraction/?ref=lbp
- https://www.geeksforgeeks.org/project-idea-motion-detection-using-background-subtraction-techniques/
- https://towardsdatascience.com/how-to-code-gaussian-mixture-models-from-scratch-in-python-9e7975df5252