CS771 Project

Separation of foreground frame from background with high accuracy in real time

Department of Computer Science & Engineering,
IIT Kanpur

Submitted by

Amit Kumar	13094
Gaurav	13274
R Sundararajan	13523

Under the guidance of
Prof. Harish Karnick

Abstract

This project is being submitted as the CS771 course project under guidance of Prof Harish Karnick. The purpose is to classify the frames as background or foreground with close to perfection. In this report, we provide the results of the various algorithms we tried to achieve better accuracy. We first tried standard OpenCV MOG library that resulted in about 92% accuracy after suitably processing the input frames. Then we shifted to Vibe algorithm which lead to rapid increase in accuracy to about 98%. The source code we used for ViBe was available from the authors on request. Then we implemented the CodeBook Algorithm on our own, proposed by Kyungnam Kim and applied certain parameters tweeks to increase accuracy level to 98.37%. Observaing the problems in Coodbook we tried to apply edge detection operator on Coodbook results to further enhance the accuracy. Even though the accuracy level improved but the average frame rate fell dramaticaly that lead us to discard the improvement.

1 Algorithms Tried

1.1 MOG (Mixture Of Gaussians)

MOG characterise each pixel by its intensity in the RGB space.The probability of observing a pixel is modelled using a mixture of Gaussian distributions.For each new incoming pixel, the Mahalanobis distance to the distibution is calculated and depending on the threshold, the incoming pixel is classified as background or foreground.In practice, the unmodified output of MOG gives pretty noisy results whereas after blurring the output frame followed by erosion and dilation to we get better contours.

1.1.1 Drawbacks

- **Foreground Aperture problem :**With backgrounds having fast variations, it becomes difficult to model properly modelled using a few Gaussians.To counter this, if the model is approximated by a large number of Gaussians then slow moving foreground pixels are absorbed into the background resulting in a high false negative rate.
- **Shadow problem :**Shadows due to objects outside the frame lead to mis-classification of frames as foreground. And if we tend to remove the shadow Mog tend to remove the entire object on which the shadow falls leading to some foreground frame being classified as back-ground.

1.2 ViBe

Vibe is a standard background subtraction algorithm that extracts background features and information from moving frames. For each pixel values in the past at its location are stored to maintain adn develop history model.Now when we need to classify a pixel, we create a sphere of radius R centered a the location. A pixel is classified as background if the number of samples enclosed in the sphere corresponding to its model are greater than a threshold value.Finally, when the pixel is found to be part of the background, its value is propagated into the background model of a neighboring pixel.

1.3 Drawbacks

- **Stop targets :**If a moving target stops in the field of view for a long time it sends the the target into background thereby misclassifying the frame.
- **Broken targets :** Splitting the entire target into several pieces because of some parts of target very similar to the background.This leads to quite low bounding box-accuracy

1.4 CodeBook Algorithm

CodeBook algorithm works in two phases, namely learning phase and update phase.Sample background values at each pixel are quantized into codebooks which represent a compressed form of background model much like the bag-of-words representation in text sequences.This helps in capturing background variation due to periodic like motion over a long spam of time with limited memory being used.One of the advantages over other algorithm is that it does not need to be trained on strictly empty frames.Can somewhat handle illumination variations and moving backgrounds but this is better handled using Layered Model Detection.

2 Drawbacks of CodeBook

- CodeBook has quite poor frame rate of 8-9 frames/sec as compared to MOG 20 frames/sec and Vibe 33 frames/sec.
- For objects moving far away from the camera, if the contrast between the object and the road is low, it fades into the background.
- Without layered model detection, the noise rate exceeds the tolerable noise but the static foreground objects are not sent into the background.
- With layered model detection, the noise is drastically reduced whereas the static foreground objects start being sent into the background albeit very slowly.
- The parameters have to be separately tuned for different camera angles for different times of the day and the algorithm has to be trained separately for the above different cases.

3 Improving CodeBook

3.1 Layered Model Detection(Cache based detection)

While testing, a cache model (as mentioned in the paper by Kim et al.) is maintained in addition to the original background model generated during training. This helps backgrounds which were obtained during the detection phase.For this we define an additional cache model and for each incoming pixel at detection time we try to matchin

- Another way to handle the problem of noisy bounding boxes due to movement of shadows.

- Results in movement of small stationary foreground pixels into the background thereby drastically reducing the noise.

3.2 Combining codebook's result with an edge detection operator

The result of the codebook algorithm is combined with an edge detection algorithm (we used Sobel operator). The main idea is to highlight the boundaries of objects in a scene.This usage of an edge detector will verify if foreground pixels detected by the codebook algorithm belong to an object or not.

- We observed an improvement of (5-7 %) over the basic codebook algorithm.
- We obtained an interesting result when working on convex hulls of the contours instead of directly drawing the bounding rectangle of a contour.

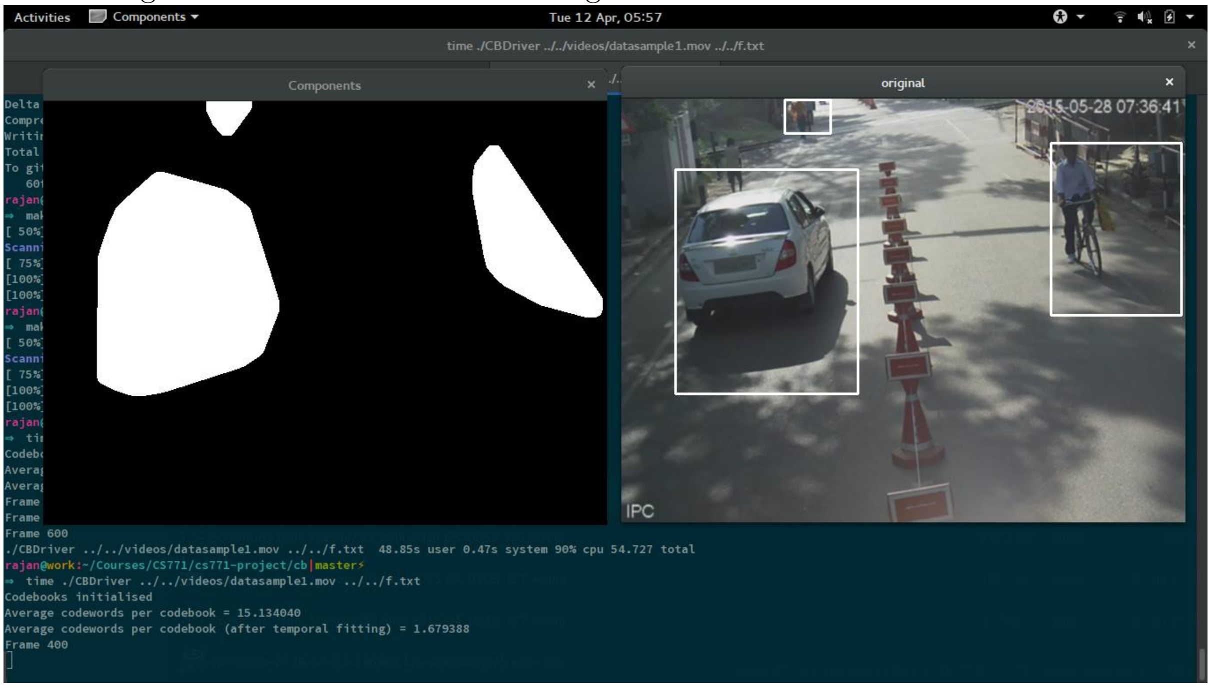
Figure 1: CodeBook with Sobel edge detector used to draw convex hulls

```
Frame 600
./CBDriver ../../videos/datasample1.mov ../../f.txt  48.85s user 0.47s system 90% cpu 54.727 total
rajan@work:~/Courses/CS771/cs771-project/cb|master⚡
⇒  time ./CBDriver ../../videos/datasample1.mov ../../f.txt
Codebooks initialised
Average codewords per codebook = 15.134040
Average codewords per codebook (after temporal fitting) = 1.679388
Frame 400
```

4 Problems faced and their workarounds made

4.1 Implementation level problems

- We coded up the CodeBook algorithm on our own
- **Problem** : We initially coded up the algorithm in Python but it gave an extremely poor frame rate of 30 frames per minute. This is due to the dynamic typing of python and high overhead associated with the operations on lists and dicitionaries.
- **Solution** : We then switched over to C/C++ which significantly improved the frame rate to 8-9 frames per second. Not too heavy on the size of the codebase (close to 500 lines of code)

4.2 Presence of Noisy contours

- **Problem** : Movement of shadows of trees, illumination changes and shaking of leaves started generating contours of their own.
- **Solution** : **Knowing our data set** to be of the following form, we could improvise on the accuracy.
  - All the video is such that no video can have small contours very close to the camera. In fact, these contours are usually due to the movement of the shadows formed by the leaves or due to illumination changes.
  - So we increased the minimum area of contours to be drawn close to the camera.
  - The number of noisy contours detected was drastically reduced.

5 Testing and Results

5.1 Steps taken for testing

- We annotated frames by adding bounding boxes on our own.
- We measured accuracy for the **rectangle (bounding box) overlap** and for overall **foreground/background binary classification**.
- For accuracy using rectangle overlap area, we used a test frame set of randomly chosen 35-40 frames from each video sequence.
- For accuracy of classifying a frame as foreground or background (0-1 accuracy), the test set containing around 700 frames had a 20 % presence of background frames to ensure their realistic representation in validation.

5.2 Results

	MOG	ViBe	Codebook (edge detection)
Frame Rate	20	33	8-9
Accuracy (Background/Foreground)	92.64	98.07	98.37
Accuracy (Framewise Rectangle Overlap)	86.39	81.07	88.13
Accuracy (Overall Rectangle Overlap)	90.30	87.08	90.83

5.2.1 Comments

- The first rectangle overlap accuracy is calculated by measuring the accuracy for each frame and then averaging it out over all frames in the test set.
- The second rectangle overlap accuracy is calculated by summing the intersection areas for all test frames and dividing it by the sum of union areas for all frames in the test set.

6 Conclusions and Future Scope

- We do not have an extremly high accuracy to show for our efforts inspite of trying out quite a lot of variations. The accuracy of our implementation of the CodeBook algorithm is better than those of MOG/ViBe but it is not a significant improvement.
- The rectangle overlap accuracy cannot be improved to reach a very high percentage such as 99 % because even a small error in drawing the bounding box of a large foreground object results in a great decrease in the accuracy.
- But we believe that with better tuning and training, the CodeBook algorithm can achieve greater than 99 % accuracy for the case of background foreground binary classification.

7 References

- **ViBe : A universal background subtraction algorithm for video sequences** (Barnich et al.)
- **Real time foreground background segmentation using codebook model** (Kim et al.)
- **Foreground-Background Segmentation Based on Codebook and Edge Detector** (Mousse et al.)