INDIAN INSTITUTE OF TECHNOLOGY, DELHI

REPORT

Assignment 1: Background Subtraction

ARNAV TULI | ENTRY NO. 2019CS10424

Course - COL780 | Prof. Anurag Mittal

Due February 5, 2023

Contents

Chapter 1

Background Subtraction

1.1 Introduction

My entry number ends with $24 \equiv 0 \pmod 4$. So, I have implemented Gaussian Mixture Model (GMM) for background subtraction with exponentially decaying weights. The algorithm used is same as mentioned in the paper by Stauffer and Grimson (CVPR 1999). The method is *online* and parametric, involving the following parameters:

- K: Number of gaussians in the mixture for every pixel in the image
- α: Learning rate of the GMM
- ω: Initial weight for the new gaussian in the mixture
- σ^2: Initial variance for the new gaussian in the mixture
- T: Background cut-off threshold for cumulative sum of *weights*

In practice, no one set of parameters works for all the scenarios due to volatile parameters like lighting, shadows, nature of foreground objects (static/slow dynamic/fast dynamic) and non-volatile parameters like camera noise and variance. Hence, the parameters must be fine-tuned to the setting under study. Therefore, in this report, I will divide my analysis of the technique into two parts:

1. Analysis of GMM across different scenarios with fixed set of parameters
2. Analysis of GMM with another background subtraction algorithm on different scenarios with fine-tuned parameters

The analysis in (1) will only consider the *raw* output of the GMM without any filter. However, analysis in (2) will, in addition to the *raw* output, also consist of output *filtered* using integral images. Following datasets are used for all of my analysis:

- Candela_m1.10
- CAVIAR1
- HallAndMonitor
- HighwayI
- IBMtest2

The generated videos for both parts of analysis (*Common* (1) and *FineTuned* (2)) can be found in their respective folders at the following link: COL780_A1_Results.

Note: The exact algorithm used for *filtering* the image will be discussed in section 1.3.1.
Note: OpenCV's contour detection method (findContours()) is used draw a bounding box around detected objects. To avoid small internal contours, boxes with area less than 400 were ignored.

1.2 *Common* Analysis

In this section, I fix the parameters of the GMM and apply it in different scenarios. The values of parameters used are: $K = 3$, $\alpha = 0.03$, $\omega = 0.08$, $\sigma^2 = 600$ and $T = 0.7$.

Note: The code was initially tested on the *IBMtest2* dataset. As a result, these parameters are fine-tuned to *IBMtest2*, leading to better results. Nevertheless, as long as these parameters are kept fixed, a fair comparison of the GMM model can be made across different settings. As will be seen below, *IBMtest2* acts as a decent baseline for comparing GMM's results on other datasets.

Overview:

	Datasets				
	IBMtest2	Candela_m1.10	CAVIAR1	HallAndMonitor	HighwayI
Mean mIoU (raw)	0.64	0.22	0.17	0.22	0.36

Table 1.1: Mean mIoU scores for raw video output (no filter) of GMM across different datasets. Scores greater than 0.5 are *good*, greater than 0.3 are *moderate*, and others are *poor*.

1.2.1 IBMtest2

This video involves two people (foreground objects), who are walking in a hallway. The model is able to correctly detect the people walking with a mean mIoU of 0.64, which is good. The figures below show model predictions at different frames: (a) input; (b) groundtruth; (c) GMM output

(a) (b) (c)

Figure 1.1: Frame#31- First person in frame (mIoU = 0.71)

(a) (b) (c)

Figure 1.2: Frame#45- Both people in frame (mIoU = 0.72)

(a) (b) (c)

Figure 1.3: Frame#62- Second person in frame (mIoU = 0.71)

Some reasons why GMM works well in this situation:

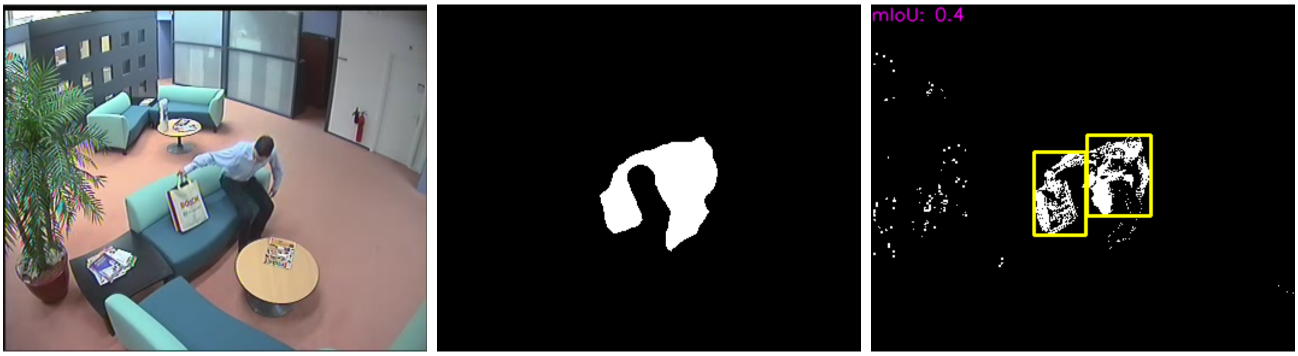
- Static background, and fast dynamic foreground. This makes it easy for the GMM to distinguish foreground from background.
- No prominent shadows or other lighting effects. This leads to less false positives and noise.
- Foreground objects are well separated in both space and time. This makes it easy for the GMM to adapt to background in between the arrival of the foreground objects, thus, reducing false negatives. If the foreground objects are not well separated, GMM can adapt to one object, and mark subsequent objects as background.

1.2.2 Candela_m1.10

This video involves a person and a bag (foreground objects). The person is detected correctly as long as its walking, but the model fails when it gets seated on the sofa and stops moving. The figures below show model predictions at different frames: (a) input; (b) groundtruth; (c) GMM output

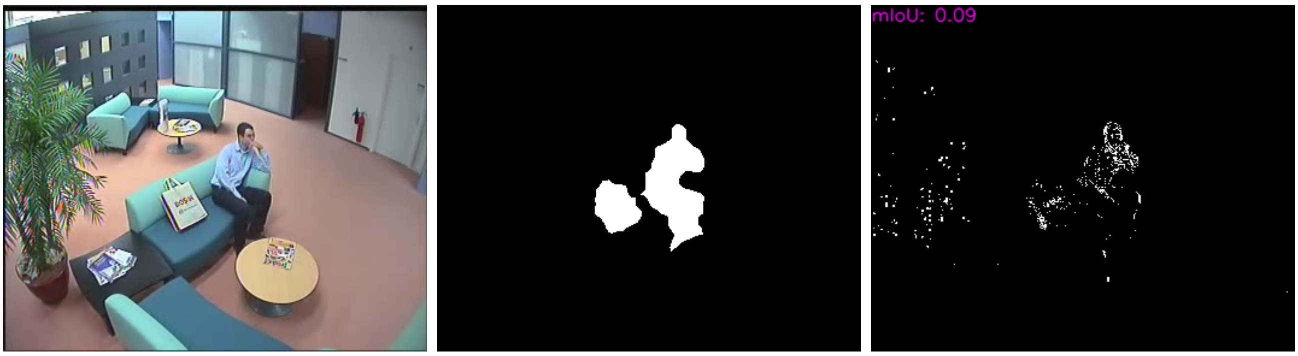
(a) (b) (c)

Figure 1.4: Frame#33- Person and bag in frame (walking) (mIoU = 0.70)



(a) (b) (c)

Figure 1.5: Frame#69- Person and bag in frame (sitting) (mIoU = 0.40)



(a) (b) (c)

Figure 1.6: Frame#115- Person and bag in frame (seated) (mIoU = 0.09)

(a) (b) (c)

Figure 1.7: Frame#319- Person and bag in frame (walking) (mIoU = 0.50)

Some reasons why GMM does not work well in this situation:

- Slow/Static dynamic foreground. This causes blending of foreground into the background due to adaptive nature of GMM, thus, leading to false negatives. (Figure 1.6)
- Dynamic background. Parts of background like the *plant* is continuously moving, which is making it hard for GMM to adapt to it. This is leading to false positives which is evident in the output frames above.

There are no prominent shadows or lighting variation, and the foreground objects are well separated in space. Therefore, in situations where the foreground is fast and dynamic, the GMM does well in detecting it, for example mIoU is 0.7 in Figure 1.4.

1.2.3 CAVIAR1 and HallAndMonitor (H&M)

Both CAVIAR1 and HallAndMonitor involve a hallway in which people are walking. The camera points directly in the direction of the hallway. As a result, people walking in the same direction are not well-separated in pixels over time. This issue is compounded if they are moving slowly as well. This is a prominent reason why GMM adapts to the foreground objects over time leading to false negatives in the output. The figures below illustrate this issue: (a) input; (b) groundtruth; (c) GMM output

(a) (b) (c)

Figure 1.8: CAVIAR1: Frame#47- Person walking (slowly) (mIoU = 0.34)

(a) (b) (c)

Figure 1.9: CAVIAR1: Frame#260- Person standing (mIoU = 0.05)

(a) (b) (c)

Figure 1.10: CAVIAR1: Frame#359- People walking (slowly) (mIoU = 0.09)

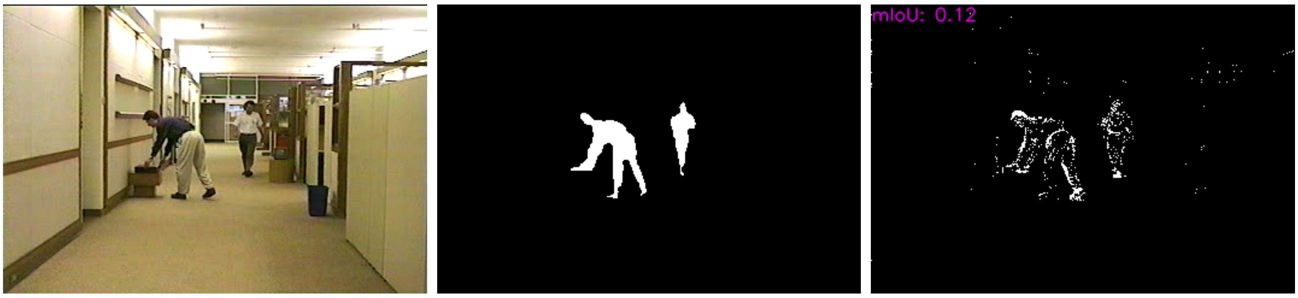
As can be seen from above figures, the foreground objects are not well separated in space because they are moving in the same direction along which the camera is aligned, and are also moving slowly. Due to this, the main body of foreground is blending with the background (because of adaptive nature of GMM), and only their boundary is indicated as the foreground.

(a) (b) (c)

Figure 1.11: H&M: Frame#34- Person coming out of room (mIoU = 0.63)

(a) (b) (c)

Figure 1.12: H&M: Frame#82- One person stopping, other coming into the hallway (mIoU = 0.19)



(a) (b) (c)

Figure 1.13: H&M: Frame#124- One person stopped, other walking in hallway (mIoU = 0.12)

(a) (b) (c)

Figure 1.14: H&M: Frame#291- Person walking out of hallway (mIoU = 0.28)

Figure 1.11 to 1.14 show that both:- direction of movement and speed are factors in determining blending of foreground objects with background. For example, in Figure 1.11, 1.12 and 1.14, objects are either coming into the hallway or going out. Since, the direction is different from the camera angle, the objects become well-separated in space over time, and avoid blending into the background. However, in Figure 1.13, one person is moving, but in the direction of the hallway (less spatial variation). Consequently, object's body blends into the background and only the boundary is visible like in CAVIAR1.

1.2.4 HighwayI

This video involves a front-view of a busy highway. The cars (foreground objects) are moving along the direction of camera, but are well-separated in space over time due to their high speed. Hence, the GMM does not face the same issues as in previous section. However, prominent shadows are present in the scene, which the GMM is unable to adapt to due to fast moving cars (and shadows). The figure below illustrates this issue: (a) input; (b) groundtruth; (c) GMM output

(a) (b) (c)

Figure 1.15: Frame#119- Fast moving cars (mIoU = 0.51)

As can be seen, the shadows towards the right of the frame are static (possibly due to some trees or other obstacle), therefore, GMM is able to adapt to them over time. However, the shadows towards the left, which are associated with the cars are fast-dynamic, and thus, the GMM is unable to adapt to them. This leads to the presence of false positives in the final output.

1.2.5 Limitations of GMM

From the analysis done in this section, I can list down some basic limitations of the GMM:

- **Slow-dynamic/Static foreground objects:** Due to adaptive nature, the model starts treating foreground objects as background over time, leading to **false negatives**. This issue is compounded if the foreground objects are also moving directly away or towards the camera, causing less separation in pixel-space over time. (Candela_m1.10, CAVIAR1, HallAndMonitor) *How to fix?* Can be fixed to some extent by decreasing the learning rate. However, low learning rate has its own limitations:
  - Foreground objects can leave lasting impressions on the image if they are also present in the initial frame. Depending on how many foreground objects are present in the initial frame, this can lead to a disastrous amount of *false positives*!
- **Dynamic background:** It is not rare to have dynamic elements in the background. For example, the *plant* in Candela_m1.10. It is often hard for GMM to adapt to such variations leading to false positives and noise. *How to fix?* Can be fixed to some extent by increasing the initial variance of the model. However, this may effect true positive detection negatively.
- **Prominent shadows and lighting effects:** It is hard for GMM to adapt to shadows if they are prominent and associated with fast moving objects. *How to fix?* Shadows can be filtered away using integral images and thresholding. (see analysis on next page)

1.3 *FineTuned* Analysis

In this section, I fine-tune the parameters of the GMM to fit different datasets better. This involves decreasing the learning rate for Candela_m1.10, CAVIAR1 and HallAndMonitor and increasing variance for Candela_m1.10. Further, to clean up the results, foreground mask will be cleaned using integral images. This will also take care of the shadows in HighwayI dataset. Before, I present my analysis, I will describe my filtering algorithm below:

1.3.1 Filtering algorithm

Filtering foreground mask is a two-step process:

1. Apply integral image filter with rectangle dimension 15×9 (height by width), and threshold probability $\frac{70}{135}$. Zero-out the pixels which are not part of any threshold-clearing rectangle.
2. Apply integral image filter with user-input rectangle dimensions and threshold probability (custom filter). Zero-out pixels which are not part of any threshold-clearing rectangle. For other pixels, either choose to fill them with *white* (255), or let them be. If choosing to fill, then, choose to either reduce blob formation using non-maximum suppression, or let it be.

First filter is fixed, and basically acts like a noise remover. The second filter, on the other hand is custom, and can be used for object detection and shadow removal (HighwayI).
Due to options available in (2), I can get three different filtering techniques: *filter* (no filling), *filter_fill* (filling with no NMS), *filter_fill_nonmax* (filling with NMS). Table 1.2 shows mean mIoU score for each of these technique on the different datasets.

	Datasets				
Mean mIoU	IBMtest2	Candela_m1.10	CAVIAR1	HallAndMonitor	HighwayI
raw	0.64	0.53	0.52	0.45	0.43
filter	0.67	0.57	0.56	0.54	**0.61**
filter_fill	0.57	**0.59**	0.40	0.42	0.38
filter_fill_nonmax	**0.73**	0.55	**0.57**	**0.61**	0.50

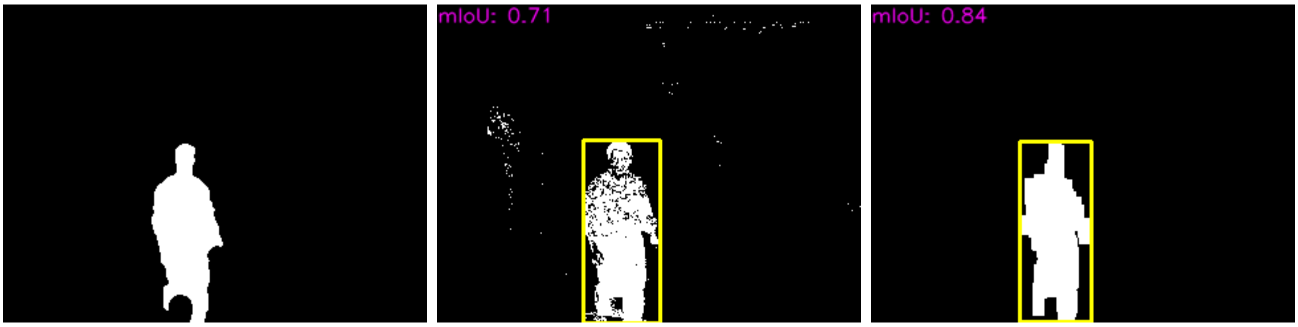
Table 1.2: Mean mIoU scores for video output of GMM model across different datasets (different filtering techniques). Pink background indicates maximum value.

<u>**Note:**</u> The above scores are for fine-tuned parameters. Consequently, the *raw* scores in Table 1.2 are greater than or equal to the raw scores in Table 1.1.

1.3.2 Parameter values and results

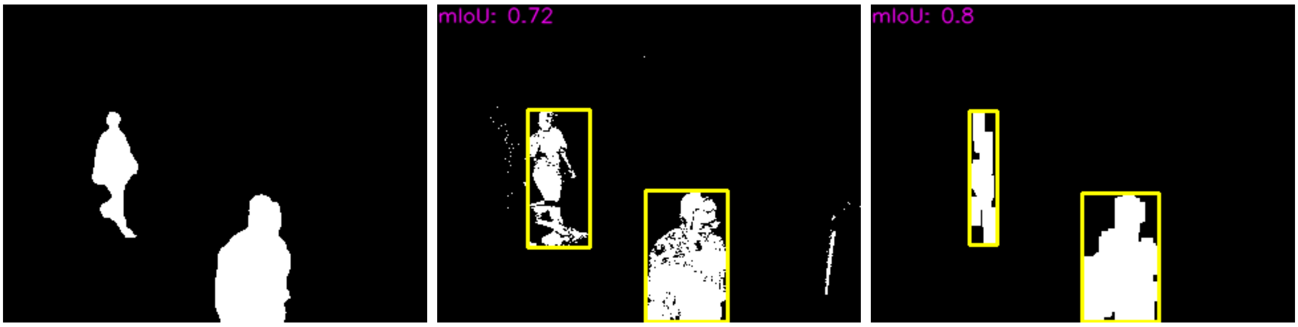
IBMtest2

The GMM parameters were already fine-tuned for IBMtest2 dataset. Filter parameters are: rectangle = 15×9 with threshold = 70 pixels. Now, I will compare the *raw* output with the *best* output (filter_fill_nonmax) next: (a) groundtruth; (b) raw; (c) best



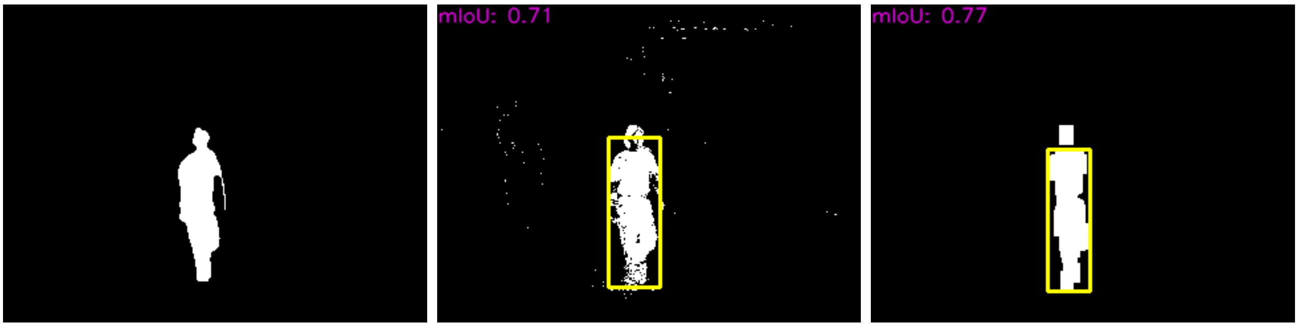
(a) (b) (c)

Figure 1.16: Frame#31- First person in frame (mIoU = 0.84)



(a) (b) (c)

Figure 1.17: Frame#45- Both people in frame (mIoU = 0.80)



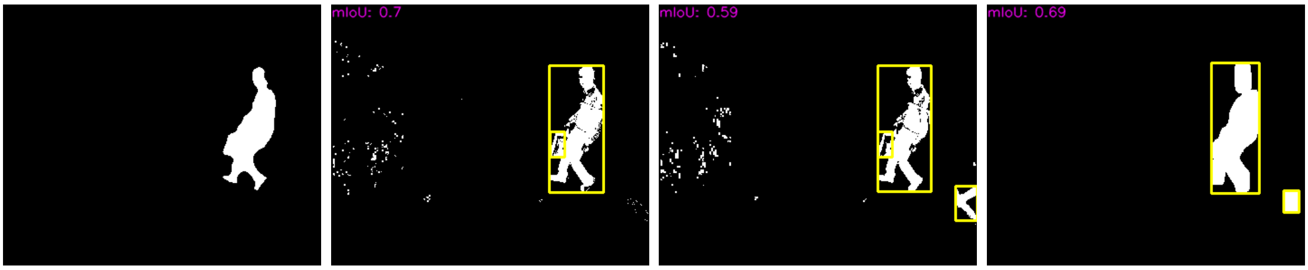
(a) (b) (c)

Figure 1.18: Frame#62- Second person in frame (mIoU = 0.77)

Candela_m1.10

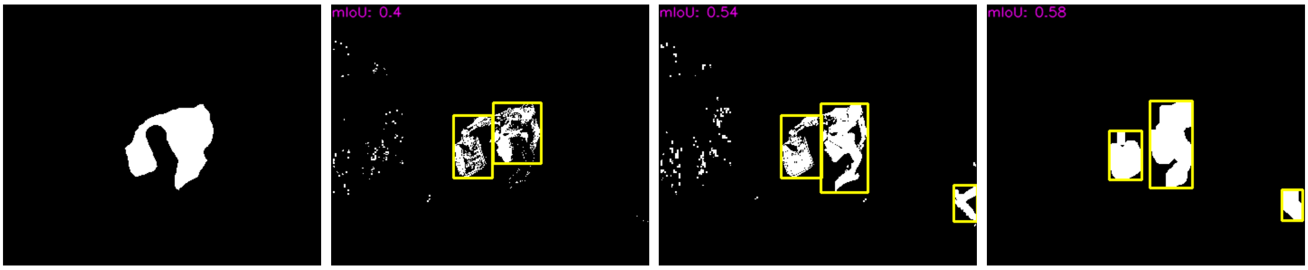
The fine-tuned parameters for this dataset are: $K = 3$, $\alpha = 0.0003$, $\omega = 0.06, \sigma^2 = 800, T = 0.7$, and filter rectangle $= 15 \times 9$ with threshold $= 100$ pixels. The figures next show the old raw and new raw GMM output alongside the best filtered output (filter_fill) for reference: (a) groundtruth; (b) old raw; (c) new raw; (d) best

Note: Due to low learning rate, there is a lasting impression on the foreground mask, which can be observed below (bottom right location in the mask).



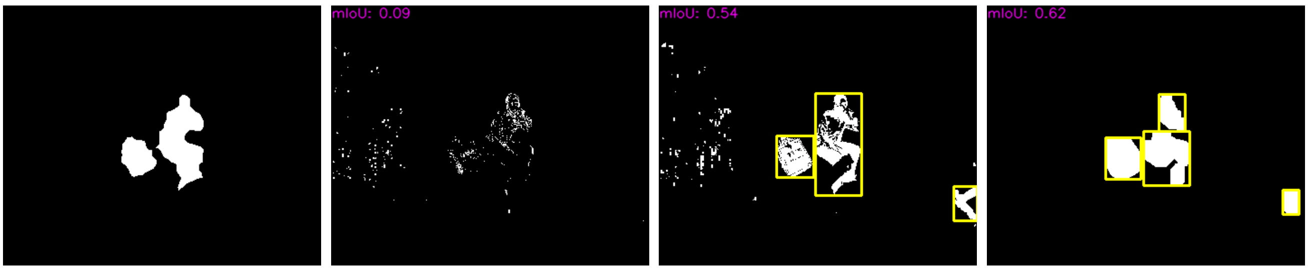
(a) (b) (c) (d)

Figure 1.19: Frame#33- Person and bag in frame (walking) (mIoU = 0.69)



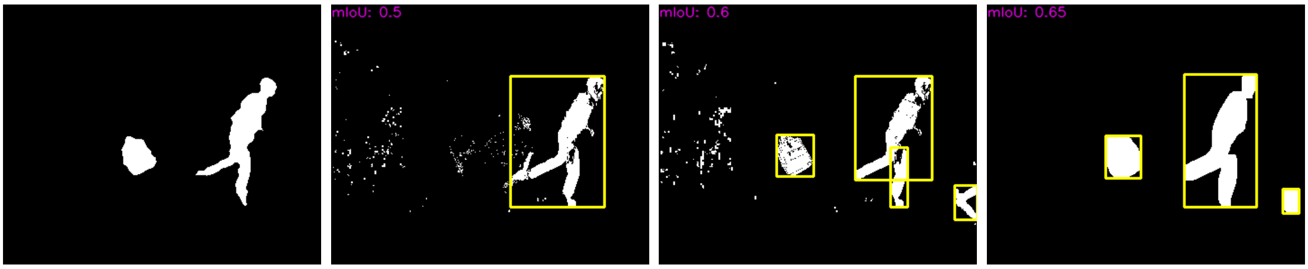
(a) (b) (c) (d)

Figure 1.20: Frame#69- Person and bag in frame (sitting) (mIoU = 0.58)



(a) (b) (c) (d)

Figure 1.21: Frame#115- Person and bag in frame (seated) (mIoU = 0.62)



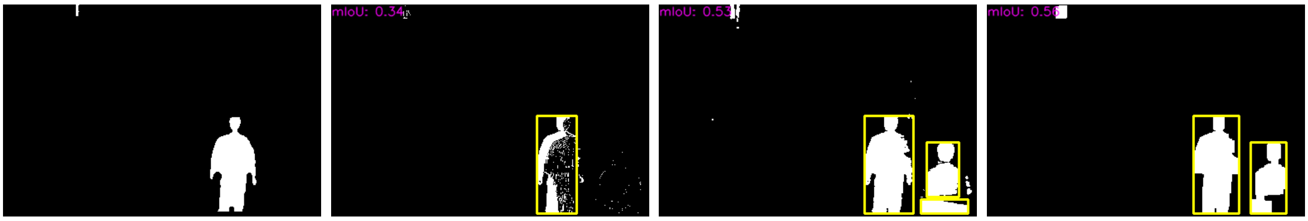
(a) (b) (c) (d)

Figure 1.22: Frame#319- Person and bag in frame (walking) (mIoU = 0.65)

CAVIAR1

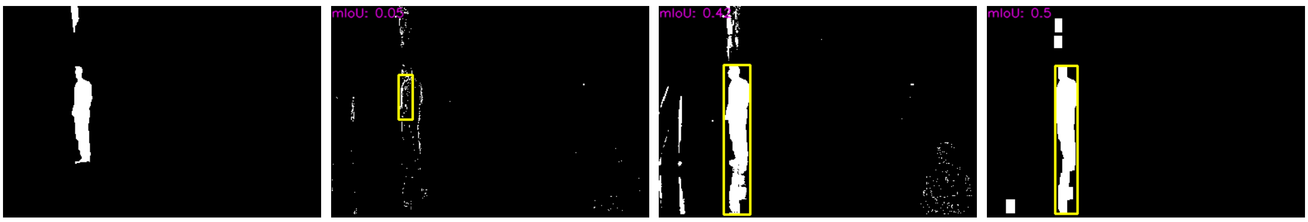
The fine-tuned parameters for this dataset are: $K = 3$, $\alpha = 0.002$, $\omega = 0.08, \sigma^2 = 300, T = 0.7$, and filter rectangle = 15×9 with threshold = 70 pixels. The figures next show the old raw and new raw GMM output alongside the best filtered output (filter_fill_nonmax) for reference: (a) groundtruth; (b) old raw; (c) new raw; (d) best

Note: Due to low learning rate, there is a lasting impression on the foreground mask, which can be observed below (bottom right location in the mask).



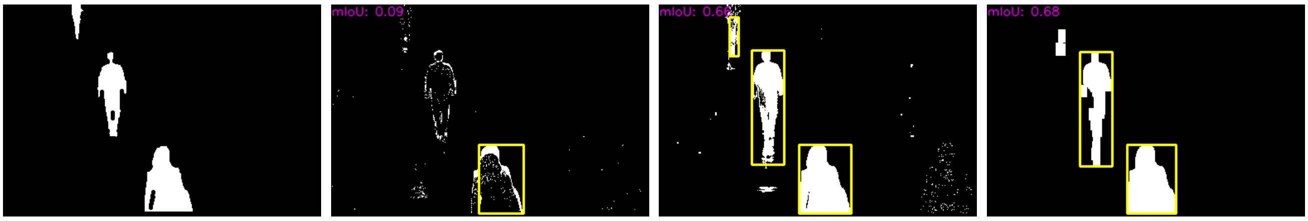
(a) (b) (c) (d)

Figure 1.23: Frame#47- Person walking (slowly) (mIoU = 0.56)



(a) (b) (c) (d)

Figure 1.24: Frame#260- Person standing (mIoU = 0.50)

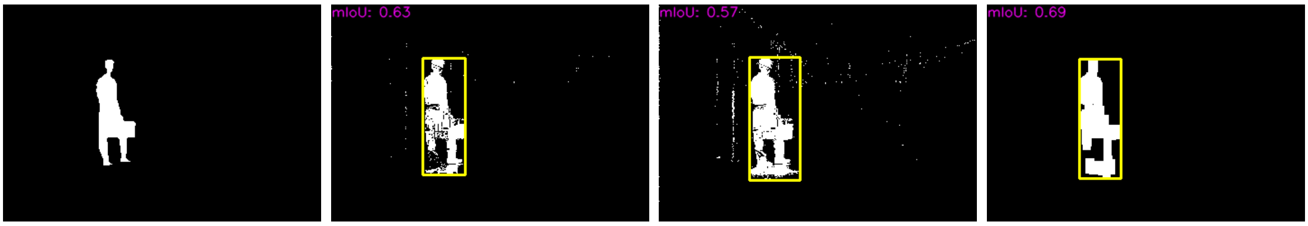


(a) (b) (c) (d)

Figure 1.25: Frame#359- People walking (slowly) (mIoU = 0.68)

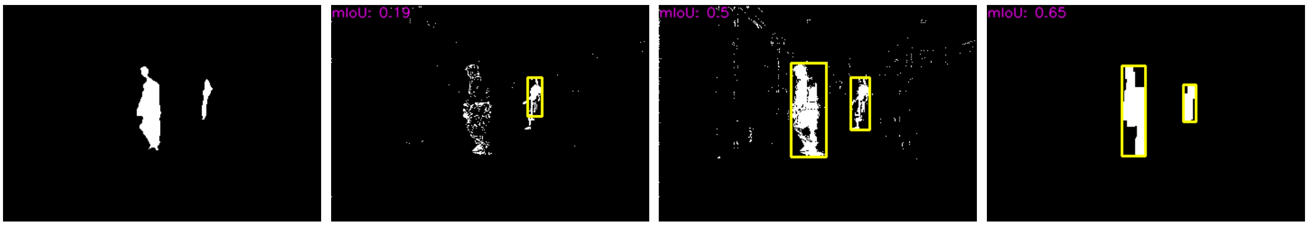
HallAndMonitor

The fine-tuned parameters for this dataset are: $K = 3$, $\alpha = 0.0003$, $\omega = 0.08, \sigma^2 = 300, T = 0.7$, and filter rectangle = 15×9 with threshold = 70 pixels. The figures next show the old raw and new raw GMM output alongside the best filtered output (filter_fill_nonmax) for reference: (a) groundtruth; (b) old raw; (c) new raw; (d) best



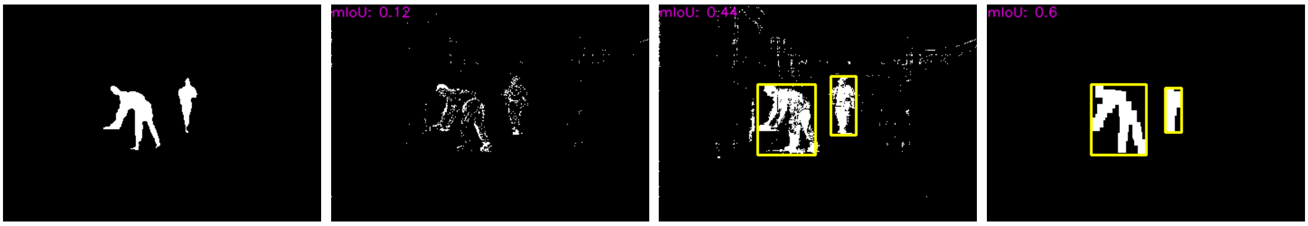
(a) (b) (c) (d)

Figure 1.26: Frame#34- Person coming out of room (mIoU = 0.69)



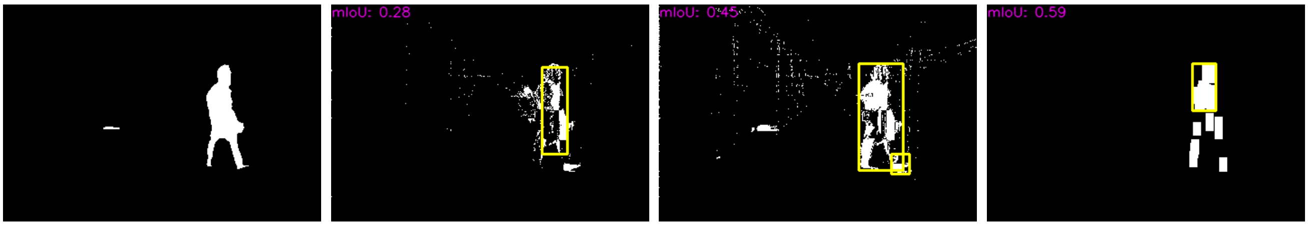
(a) (b) (c) (d)

Figure 1.27: Frame#82- One person stopping, other coming into the hallway (mIoU = 0.65)



(a) (b) (c) (d)

Figure 1.28: Frame#124- One person stopped, other walking in hallway (mIoU = 0.60)

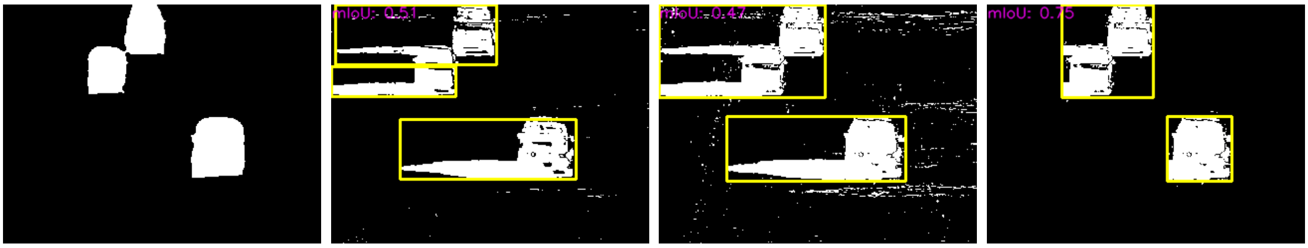


(a) (b) (c) (d)

Figure 1.29: Frame#291- Person walking out of hallway (mIoU = 0.59)

HighwayI

The fine-tuned parameters for this dataset are: $K = 3$, $\alpha = 0.03$, $\omega = 0.05, \sigma^2 = 100, T = 0.7$, and filter rectangle $= 60 \times 10$ with threshold $= 300$ pixels. The figures next show the old raw and new raw GMM output alongside the best filtered output (filter) for reference: (a) groundtruth; (b) old raw; (c) new raw; (d) best



(a) (b) (c) (d)

Figure 1.30: Frame#119- Fast moving cars (mIoU = 0.75)

1.3.3 Conclusion

Most of the limitations mentioned in section 1.2.5 have been addressed by fine-tuning parameters and filtering (like prominent shadows and foreground blending with background). The best mean mIoU scores are all above 0.5, indicating good performance of the GMM on the given datasets.

1.4 Comparison with other technique

In this section, I compared my results with Akshat (2019CS50418), who implemented kernel density based method with decaying weights. Table 1.3 shows the resulting mean mIoU scores of the two methods (without filtering)

	Datasets				
Mean mIoU	IBMtest2	Candela_m1.10	CAVIAR1	HallAndMonitor	HighwayI
GMM	**0.64**	**0.53**	**0.52**	0.45	0.43
KDE	0.57	0.44	0.38	**0.47**	**0.47**

Table 1.3: Mean mIoU scores for raw video output of GMM model and KDE model across different datasets. Pink background indicates maximum value.

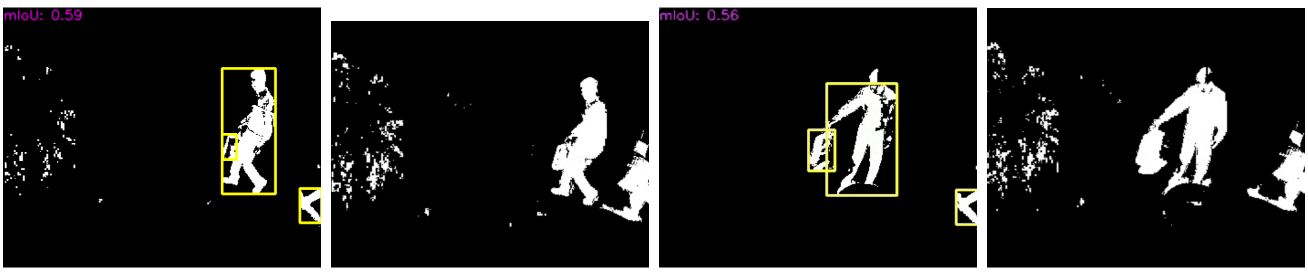
1.4.1 Quantitative assessment

Directly from the mIoU scores, the following things can be concluded:

- GMM performs better than KDE on the following datasets: IBMtest2, Candela_m1.10 and CAVIAR1, whereas, KDE performs better than GMM on the following datasets: HallAndMonitor and HighwayI.
- Both GMM and KDE perform well on IBMtest2 dataset, with scores above 0.5. On the other hand, both GMM and KDE perform moderately on HallAndMonitor and HighwayI datasets, thus, indicating that its relatively harder to do background subtraction correctly in these settings without any filters.
- The difference in performance exists on the following two datasets: Candela_m1.10 and CAVIAR1. GMM performs well, but KDE performs moderately. As discussed later, this quantitative margin can be attributed to false detection of certain qualitative features in KDE, which is not there in GMM.

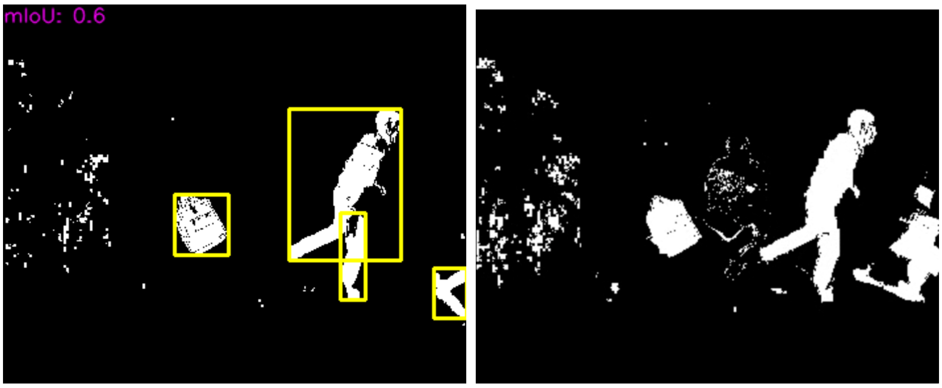
1.4.2 Qualitative assessment

Candela_m1.10



(a) (b) (c) (d)

Figure 1.31: (a) GMM vs (b) KDE; (c) GMM vs (d) KDE



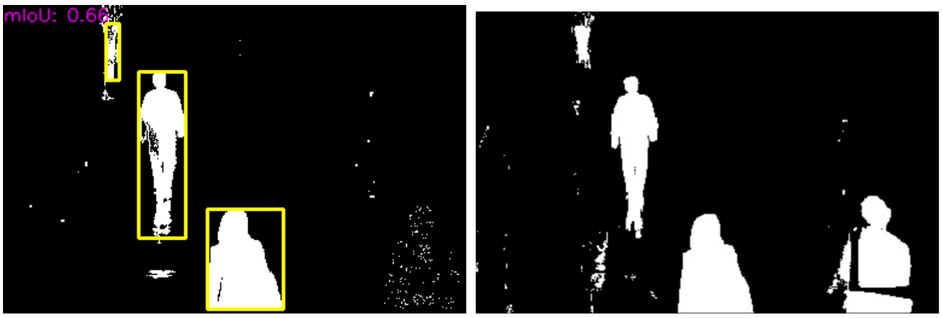
(a) (b)

Figure 1.32: (a) GMM vs (b) KDE

Observations:

- The person (foreground) has a larger lasting impression in case of KDE as compared to GMM. This can be seen as a bigger white patch in bottom right region in Figure 1.31 (b) and (d), and Figure 1.32 (b). This shows that GMM is quicker to adapt to image setting than KDE. In other words, GMM is learning at a faster rate.
- GMM model is able to adapt to the moving *plant* (background) to a greater extent (than KDE) and completely hid it in Figure 1.31 (c). On the other hand, this plant noise is consistently visible in case of KDE output.
- Similar to first point, the person walking away leaves its impression in case of KDE but not in case of GMM (Figure 1.32), reinforcing the fact that GMM is learning to adapt faster.

CAVIAR1

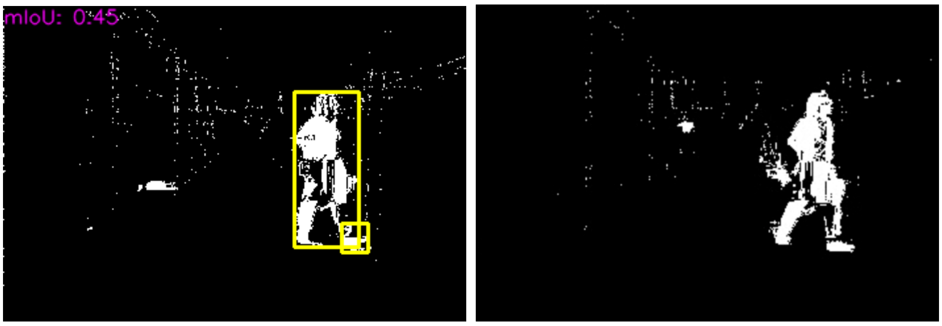


(a) (b)

Figure 1.33: (a) GMM vs (b) KDE

Like in case of Candela_m1.10, the lasting impression of foreground object is more pronounced in case of KDE as compared to GMM. This is a primary reason for under-performance of KDE in these two settings.

HallAndMonitor



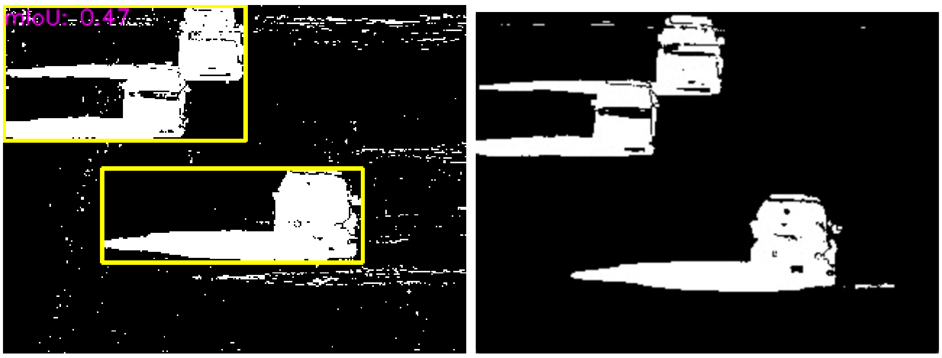
(a) (b)

Figure 1.34: (a) GMM vs (b) KDE

The behaviour of GMM and KDE in this setting is almost identical. There is no issue of lasting impressions in this case in either of the two algorithms. There are, however, two minor differences:

- Noise (false positives) is slightly lesser in KDE than in GMM. This might be a reason why mIoU of KDE is slightly higher than GMM (0.47 > 0.45).
- The bag that the person was holding is being correctly detected in GMM (middle-left region), but not in KDE. Since, the bag has small contribution to the mIoU, the overall effect (with noise) is coming out to be negative.

HighwayI

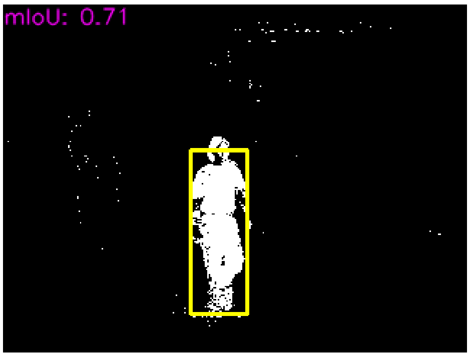


(a) (b)

Figure 1.35: (a) GMM vs (b) KDE

As can be seen from Figure 1.35, KDE output has lesser noise than GMM. This is the trend that is seen for all frames, leading to a higher mIoU (0.47 > 0.43). Other than this, the performance of GMM and KDE was identical in this setting.

IBMtest2



(a)

(b)

Figure 1.36: (a) GMM vs (b) KDE

The performance of both GMM and KDE in this setting was good. The only minor difference was shadows. GMM was able to correctly model the shadows by incorporating it into the variance parameter of the gaussian, which the KDE was unable to do. Prominent shadows and lighting variation are still a weakness of the GMM, but light shadows and lighting effects can be modelled using the variance parameter, and that gives it an edge over KDE, at least in this setting.

1.4.3 Conclusion

There is no clear winner between these two. In some settings, GMM may perform better than KDE, and in some, KDE may perform better than GMM. Hence, the choice of algorithm will depend on the setting under study. However, some key takeaways from the above discussion are mentioned below:

- **Slow-dynamic/Static foreground objects:** GMM is able to handle such settings better. KDE can detect the foreground objects correctly in these settings, but it also leads to formations of lasting impressions (false positives), which degrade the quality of detection. (like Candela_m1.10 and CAVIAR1)
- **Light shadows and lighting effects:** GMM is able to better model such variations in lighting effects using the variance parameter of the gaussians, which KDE is unable to do. (like IBMtest2)
- **Noise cancellation:** In general cases (fast-dynamic foreground), KDE is able to better cancel out noise than GMM. (like HallAndMonitor and HighwayI)

Chapter 2

Appendix: Code Overview

There are two code files `main.py` and `backgroundSubtractor.py`. The code assumes the following directory structure:

```
directory
├─ Dataset/
│  ├─ Candela_m1.10/
│  ├─ CAVIAR1/
│  ├─ HallAndMonitor/
│  ├─ HighwayI/
│  └─ IBMtest2/
├─ main.py
└─ backgroundSubtractor.py
```

`main.py` has support for the following command line arguments:

- `--dir`: Dataset directory (IBMtest2, HighwayI etc.)
- `--K`: Number of gaussians in mixture
- `--A`: Learning rate
- `--wt`: Initial weight
- `--var`: Initial variance
- `--height`: Height of rectangle (integral image)
- `--width`: Width of rectangle (integral image)
- `--thresh`: Rectangle pixel threshold (integral image)
- `--filter`: Filter flag (use `--no-filter` to stop filtering)
- `--fill`: Fill flag (use `--no-fill` to stop filling (if filtering))
- `--nonmax`: NMS flag (use `--no-nonmax` to stop NMS (if filling))

Run command: `python main.py` ⟨ `args` ⟩
Output: A video is saved in the directory specified with `--dir`, and mean mIoU is printed.