CMPEN 454 – Project 3

Yassen Saad – Rodrigo Olaya – Neel Sakhrani

# **Summary**

The project aimed to implement and evaluate four motion detection algorithms: Simple Background Subtraction, Simple Frame Differencing, Adaptive Background Subtraction, and Persistent Frame Differencing. By studying lecture 18 and using the pseudocodes given, we applied the mentioned algorithms to video sequences to detect motion and generate output frames that make up a video. The main objectives were to gain experience in efficient MATLAB programming, understand the principles of motion detection algorithms, and assess their performance in various scenarios.

# **Outline**

From experience with project 2, we knew the implementation had to follow a structured approach to make it easier on our future selves to debug. This is why we split the code into two sub-scripts. We start by defining the input parameters in the script proj3\_caller.m. We then read the frames provided and convert them to grayscale to simplify processing in the proj3\_main.m script. The motion detection algorithms were then sequentially applied to the frames, with separate background models maintained for each algorithm. Output frames from each algorithm were added in sequence into a quad frame for visualization. The implementation focused on modularity and efficiency, with clear separation of algorithmic logic and input/output operations.

## Flowchart:

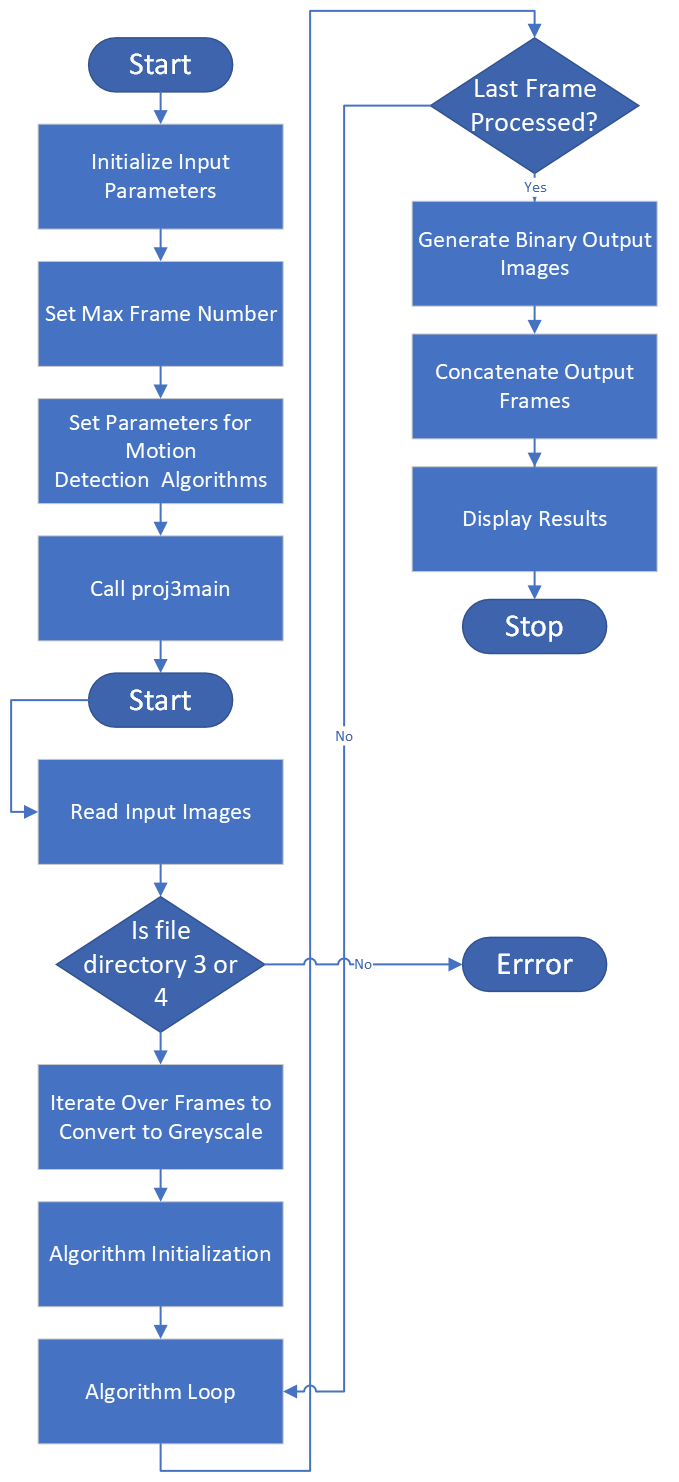


Figure 1: Flowchart to depict general structure of Matlab script.

# **Observations**

During experimentation, several key observations were made regarding the behavior of the implemented algorithms:

* Parameter Sensitivity: Adjusting parameters such as the absolute difference threshold and alpha blending parameter had a significant impact on algorithm performance. We played around with the numbers until we got the best possible output. Adaptability to Scene Changes: Adaptive Background Subtraction demonstrated robustness in handling changes in the background scene, thanks to its dynamic adaptation mechanism. This algorithm proved effective in scenarios with varying lighting conditions and background movement.
* Challenges with Persistent Frame Differencing: Persistent Frame Differencing posed challenges in achieving consistent intensity decay. Despite attempts to adjust the gamma parameter, the algorithm exhibited weird behavior in some cases. We solved this by realizing that the issue was not in the algorithm code itself but actually in the display code. This was a very frustrating realization as we had spent hours trying to debug assuming there was an issue with the greyscale.

## Parameter specific change:

1. Whenever we set the absolute difference threshold to 0, or to +40 the results were terrible, we found that somewhere in the range of 30-40 was the best output. Higher than 40 and it starts getting darker, and less than 30 and you can see very odd spotting.

A black and white image of a person's face

Description automatically generated

Figure 2: Effects on output when we set absolute difference threshold to 0.

A black background with a white object in the middle

Description automatically generated

Figure 3: Effects on output when we set absolute difference threshold to 70.

A white object in the sky

Description automatically generated with medium confidence

Figure 4: Effects on output when we set absolute difference threshold to 35.

1. For the alpha parameter, as we approach and at 0 it is very similar to the simple background subtraction algorithm, and then as we approach and at 1 it is similar to the simple frame differencing.

A white and black image of a person in a suit

Description automatically generated with medium confidence

Figure 5: Effects on output when we set alpha to 0.



Figure 6: Effects on output when we set alpha to 0.5.

A white object in the sky

Description automatically generated

Figure 7: Effects on output when we set alpha to 1.

1. In regard to linear decay, it is only taken into account for the persistent frame differencing, and through trial and error we deduced that at 0 the trace is almost all white, and as the linear decay increases the fading is clearer. By around 100 we noticed it was very similar to the simple frame differencing.

A black background with white text

Description automatically generated

Figure 8: Effects on output when we set linear decay to 0



Figure 9: Effects on output when we set linear decay to 35.

A black background with white lines

Description automatically generated with medium confidence

Figure 10: Effects on output when we set linear decay to 100.

**Results**

We can see that in the case where the lighting conditions are not good, such as in Figure 11, the best algorithm is simple frame differencing. In the case of figure 11, we also see that simple background subtraction is very bad quality. For other purposes, such as tracking the direction of motion and trace the previous position of object, persistent frame differencing is best. We can see in figure 12 that the tracing helps us see where the ants have previously been. An important observation is that Simple Frame Differencing and Adaptive background Subtraction are very accurate on the ship deck. They can detect the motion of the sea and the boats, but they do not detect the movement of the boat. This is the objective of these algorithms because we don’t want the boat to be detected as moving, since it is not moving with respect to the camera.

A collage of images of a planet

Description automatically generated

Figure 11: Possum.

A group of white objects on a black background

Description automatically generated

Figure 12: Ants.

A black and white image of a dog

Description automatically generated

Figure 13: Deer.

A screenshot of a video game

Description automatically generated

Figure 14: Ship Deck.

Overall, we would choose Adaptive Background Subtraction. As noted previously, it can detect motion while ignoring the objects that are not moving with respect to the camera. It is also very accurate when working with low quality images. It is also very polyvalent, since by changing the alpha parameter, we can reproduce Simple Background Subtraction or Simple Frame Differencing.

# **Contributions**

Overall, it was very hard to come up with division because all the aspects of the project were related to each other. This meant that we had to take a step-by step approach even though we knew we had to modularize the code for each algorithm. As a group, we would meet up and work on the project by brainstorming ideas and tackling each segment together.