**Name: Dhamini Vootkuri**

**UID: U01111434**

[**vootkuri.3@wright.edu**](mailto:vootkuri.3@wright.edu)

**CEG 7560 Visualization for image Processing for Cyber Security**

**Report: Video Processing and Object Tracking**

**1. Introduction**

In this assignment, I worked with two CCTV videos from the VAST Challenge dataset.  
My goal was to detect moving objects, remove noise, track each object and show all results in one combined display.  
I also generated a final output video and saved a few screenshots automatically.

**2. Preprocessing**

I loaded both videos using OpenCV.  
Since both videos had slightly different brightness and content, I resized the second video to match the first.  
I used MOG2 background subtraction to separate moving objects from the background.

To clean the masks, I used:

* Gaussian blur
* Thresholding
* Morphological open + close
* Median blur

These steps helped me to reduce shadows and noise.

A collage of cars driving on a street

AI-generated content may be incorrect.

*Figure 1: Binary mask generated after background subtraction and cleanup.*

**3. Object Detection**

After getting the binary mask, I used contour detection to find moving objects.  
Very small contours were ignored to avoid noise.  
For every valid contour, I created a bounding box and calculated the centroid.

A screenshot of a video camera

AI-generated content may be incorrect.

*Figure 2: Detected moving objects with bounding boxes and centroids.*

**4. Tracking**

I used a simple centroid-based tracking method.  
Each object got an ID, and the ID stayed the same until the object disappeared.  
I also drew a movement trail using previous centroids.  
This helped me see the direction and path of each moving object.

A screenshot of a video camera

AI-generated content may be incorrect.

*Figure 3: Object detection with ID labels and tracking trails*

Even though this is a simple tracker, it still worked well for the video.  
Crowded scenes naturally created many IDs, which is expected.

**5. Combined Display**

I created a 2x2 layout:

* Top-left: Video 1
* Top-right: Video 2
* Bottom-left: Mask 1
* Bottom-right: Mask 2

I also added the timestamp on all four panels.

A screenshot of a video

AI-generated content may be incorrect.A collage of images of cars and buildings

AI-generated content may be incorrect.

*Figure 4. Full 2x2 layout showing both camera views (top) and their corresponding foreground masks (bottom).*

This layout made it easy for me to compare original videos with the masks and tracking results.

**6. Output Generation**

I saved:

* The processed combined video (CombineVideosOutput.avi)
* Screenshots every 150 frames

VTK was used to display the frames while the program ran.  
Execution was stable on my machine. I tested multiple segments and saved screenshots and a final video.

Following are some saved screenshots:

A collage of a street with cars and a black screen

AI-generated content may be incorrect.A two images of a street

AI-generated content may be incorrect.A comparison of a street and a street

AI-generated content may be incorrect.A screenshot of a video

AI-generated content may be incorrect.A collage of cars driving on a street

AI-generated content may be incorrect.A collage of a street corner

AI-generated content may be incorrect.

**7. Results and Observations**

* The system tracked vehicles and some people correctly.
* When the street became crowded, the mask became noisy. This is normal for MOG2.
* The timestamps were accurate and visible in all panels.
* The bounding boxes and trails helped visualize movement patterns clearly.

Overall, my program met the assignment requirements.

Limitations: simple tracker fails in heavy occlusion and mask noise increases when the scene is crowded

**8. Conclusion**

I successfully processed both videos and built a full visualization system using OpenCV and VTK.  
The code detects motion, removes noise, tracks objects, draws boxes and shows everything in one combined view.  
I also saved output videos and screenshots. The results look correct and the system works as expected.