Image Processing

Applied Programming with Dr. Lusk

ENGR 10573 SEC 081

By: Ebby Randall and Drew Steigleder

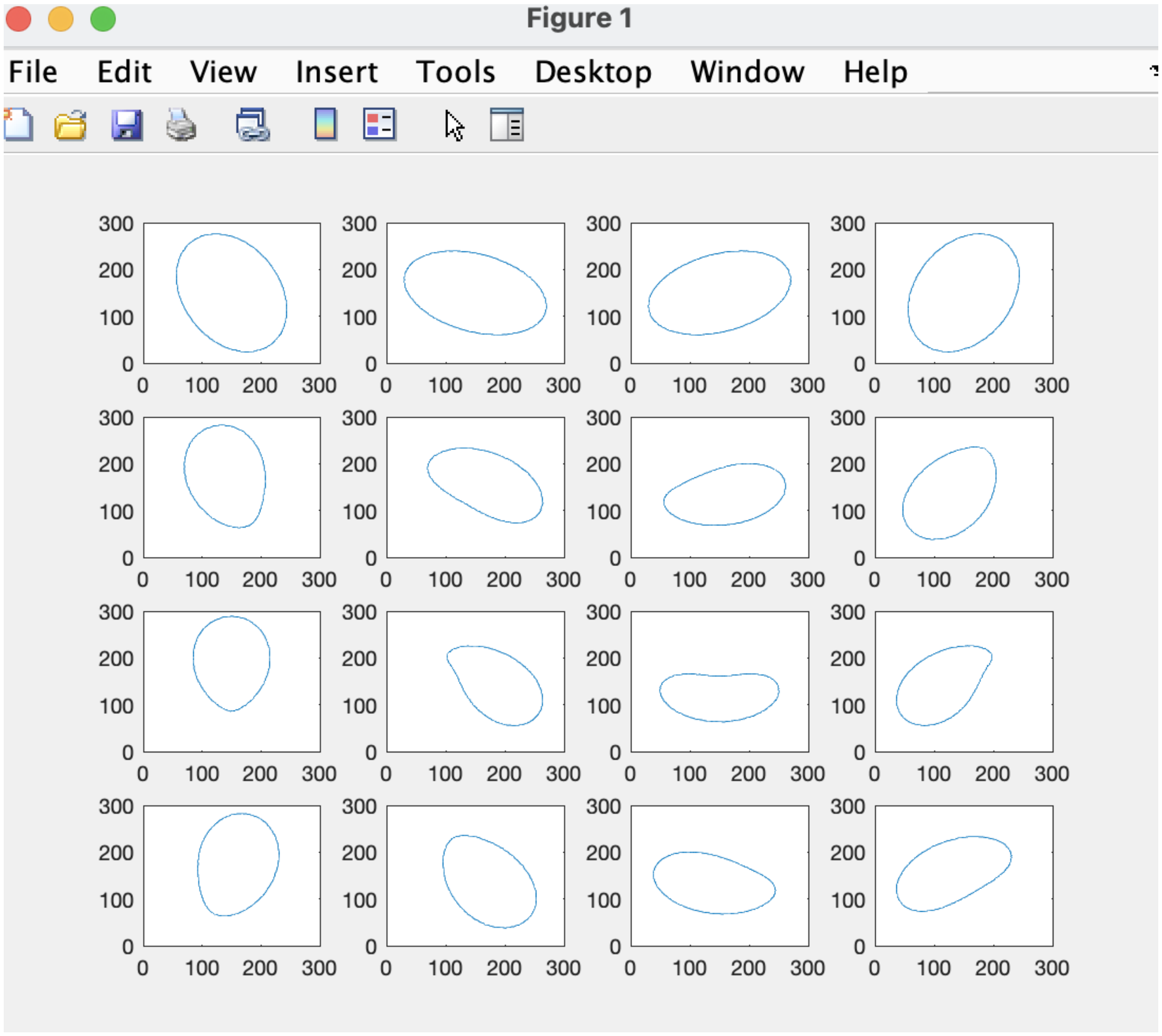
Goal:

The goal of this project was to write program scripts with supporting functions, completed in the past homework assignments, and some given functions to create our own image processing algorithm. We were given an input video of multiple moving black dots on a background made up of black and white pixels. Afterwards, we were given a list of tasks and functions to apply to the video with code in order to be able to fully track the movement of the dots. Our algorithm will take this video, reduce the noise around our desired points, and place a tracking cross on each of the moving points, relative to the dot size. This project examines how computer "vision" is created, and foreshadows other technological advancements, such as: ring doorbell tracking, surveillance cameras, and even facial identification scanners. It is surprising how this type of code implementation can be seen in so many different aspects of the world today.

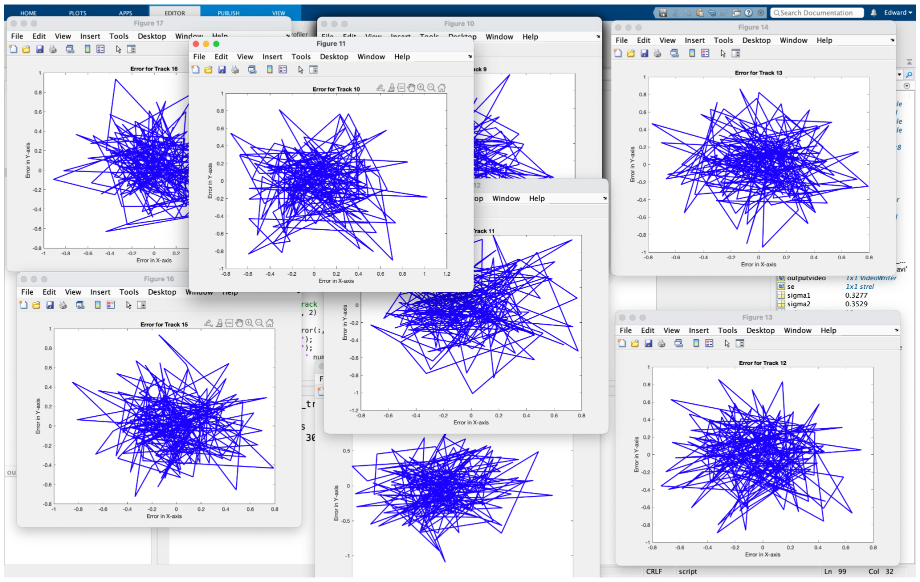
Methods:

Many coding methods learned in this unit and past were called to action, these methods include preallocation spacing, for and while loops, the ability to implement functions and call other files of code, and the ability to conform and change various aspects of an image/video file by different standards. To start, we first began to create a duplicate storage file of the original video and created it to be blank so that the for-loop could just add frames as it progressed through the original video, to the new one. However, in order to do this we had to find the amount of frames that the original video consisted of. After that, in our for-loop we were able to change each frame individually, removing noise, changing the color, and completing the rest of the tasks given to us, using the built in MATLab functions and provided .m files on the project report. Finally, we were able to compare our tracking to the .m file tracking and estimate the error our code had in the tracking of the dots. This error could come from various little details such as size of the “x” like trackers on the dots, how fast they moved, and their orientation on the center of the dot.

Results



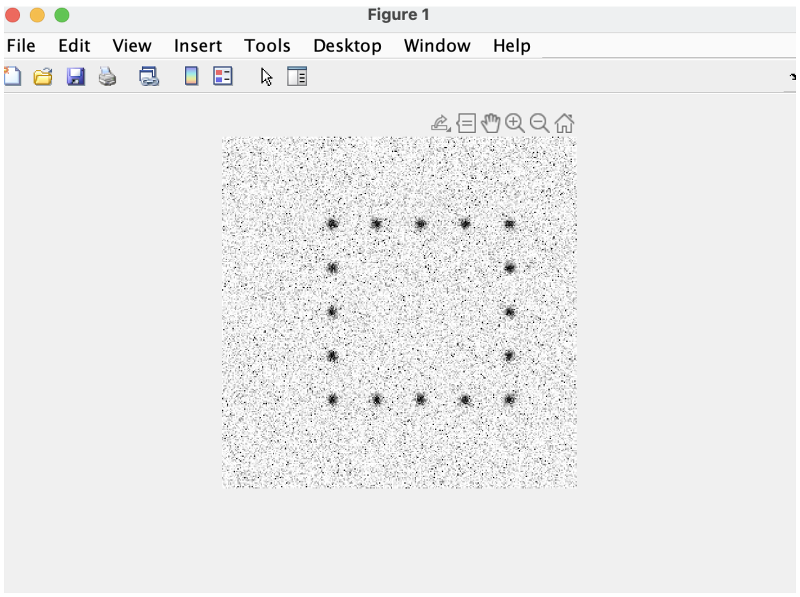
This image above represents the tracked paths of the dots we are trying to build the image tracking algorithm for. Each graph represents one point and its path across the frames of the video.

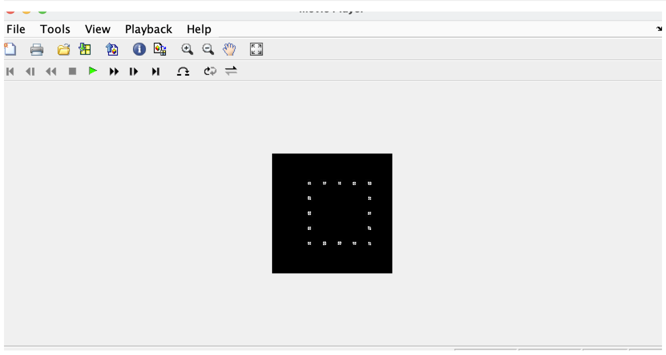


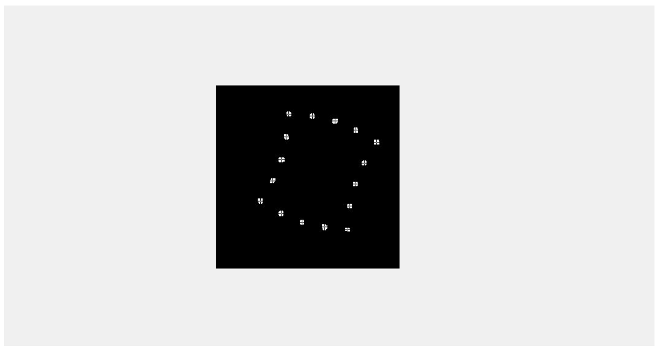
I did a poor job graphically representing the error for each point we tracked. I used a for loop to graph the error of each iteration and got back graphs looking like this.

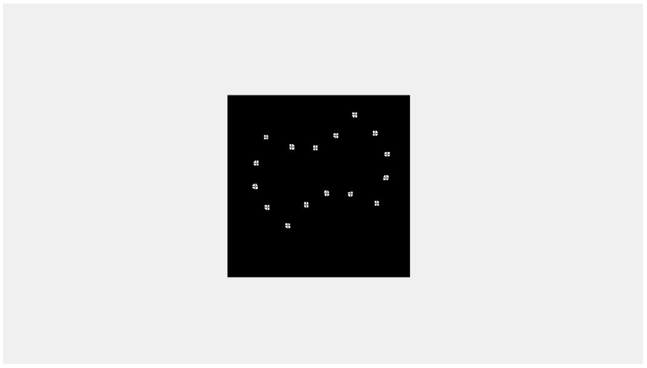
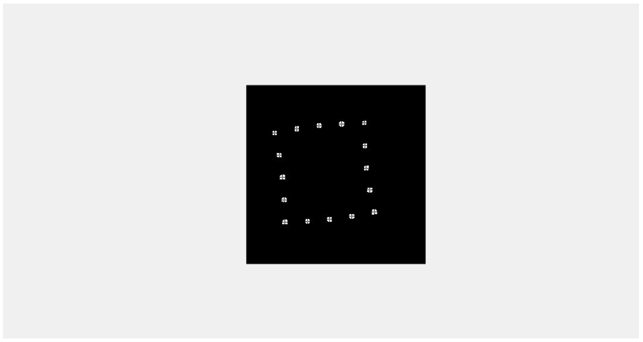
The image tracking is a video but I will take a few screenshots to represent the tracking. The first image shown in the beginning state of the frames we are processing. Our goal is to put a white

“X” on each black dot which will then trace across each frame.







Summary Statement

This project involves processing and analyzing imagery data using MATLAB's Image Processing Toolbox and associated toolboxes. The implementation entails performing image filtering, thresholding, morphological operations, and target localization to isolate and track objects in the imagery. The ultimate goal is to obtain the centroid of each target and analyze the performance of the developed algorithm by comparing it to a given truth file. A thoughtful report detailing the findings and outcomes is expected.

Discussion

The image processing pipeline for this project starts with filtering to remove shot and Gaussian noise, followed by thresholding to produce a binary image. Morphological functions like imclose and imerode are employed to solidify dots and remove spurious pixels. The centroids of the targets are then calculated using regionprops, and tracks are drawn on the image frame to visualize the results. To assess the algorithm's performance, the calculated tracks are compared to a given truth file, and error statistics are generated. A comprehensive report should discuss the various MATLAB functions and toolboxes utilized, the implementation process, the results obtained, and potential improvements that could be made to the algorithm for better performance.