

George Fang, Lily Davoren

Matt Zucker

ENGR027

16 February 2024

Project 1 Writeup

In this project, we tracked the movement of specified objects in a video using the computer vision techniques we learned in class, specifically thresholding to distinguish objects of interest from the background, applying morphological operators to fix noise and imperfections, and blob tracking. Our overall approach involved defining several constants and parameters, color selection, a video processing loop, contour detection, object tracking, and displaying the results.

The constants defined at the beginning of the script are used as object tracking criteria. In regard to our approach to color selection, our program prompts users to click on the brightest part of the object(s) they want to track in the input video. This functionality allows the user to select multiple colors for tracking so that objects of different colors can be tracked simultaneously. The selected colors are then stored in a list of target colors for tracking. To process the video, the script reads the input video specified by the user and processes each frame in the loop. Color thresholding is then applied to the video to detect objects of the selected color(s) in the current frame. Morphological operations, specifically opening, are applied to eliminate noise and refine the binary mask. Contours are then detected in the thresholded image, and for each contour, its area and mean location are calculated using moments. Regarding our approach to object tracking, objects are tracked based on the color thresholding results and using the contours generated by OpenCV. Objects are then created or updated based on their proximity to existing objects, and their locations are stored. Objects are considered “gone” and no longer trackable if they are missing for a specified number of frames. The program ultimately displays the original video with contours drawn around the tracked objects with object IDs overlaid.

The use of color thresholding to detect objects based on colors selected by the user is effective for tracking specific objects. In addition, object tracking is implemented with a consideration for object permanence, which helps to handle temporary obstructions. However, the color-based tracking can be sensitive to variations in lighting conditions and shadows. The object tracking approach is not robust with all input videos, especially when objects are close to each other or undergo rapid changes (see `HIGHWAY_TRAFFIC.mp4` and `LIGHT_SABER.mp4`). To improve upon the issue of fluctuating illumination in future iterations of this script, we could implement adaptive thresholding techniques, which will more effectively handle changes in lighting conditions and shadows. Overall, the effectiveness of the scripts is heavily dependent on the characteristics of the input video. One of the most important characteristics is the color contrast of the object being tracked with the rest of the colors in the video. If the contrast is low, then it will be difficult to detect objects using color thresholding. We attempted to use temporal averaging for background subtraction, but it was incredibly difficult to handle regions of the video where the performer moved their body outside of their averaged location.

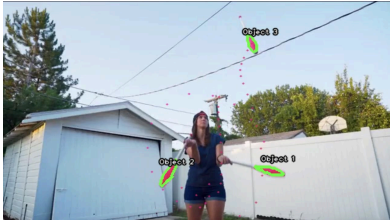


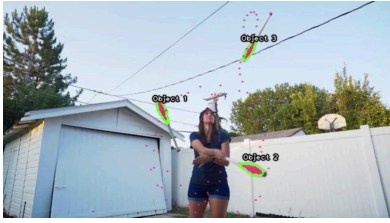
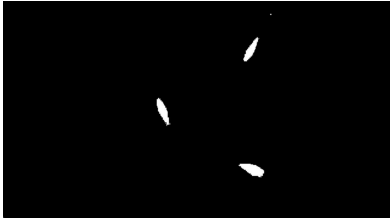

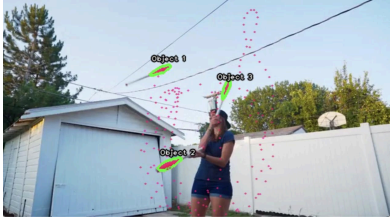
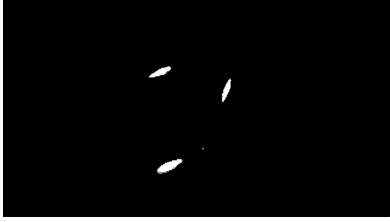

The tracking only remains robust if the selected colors in the input video remain distinguishable under varying brightnesses. In addition, the success of the tracking depends on the user’s ability to accurately choose the brightest part of the objects they want to track. Because a minimum pixel area has been defined, and objects are identified based on this minimum pixel area, objects smaller than this minimum pixel area may not be detected accurately or at all. A maximum object jump distance has also

been defined, which means that the system assumes that objects do not move beyond a certain distance between consecutive frames. If objects move beyond this threshold, they might not be tracked properly or at all. The object permanence threshold is defined as the number of frames an object can be missing before it is considered gone, which assumes object continuity between frames.

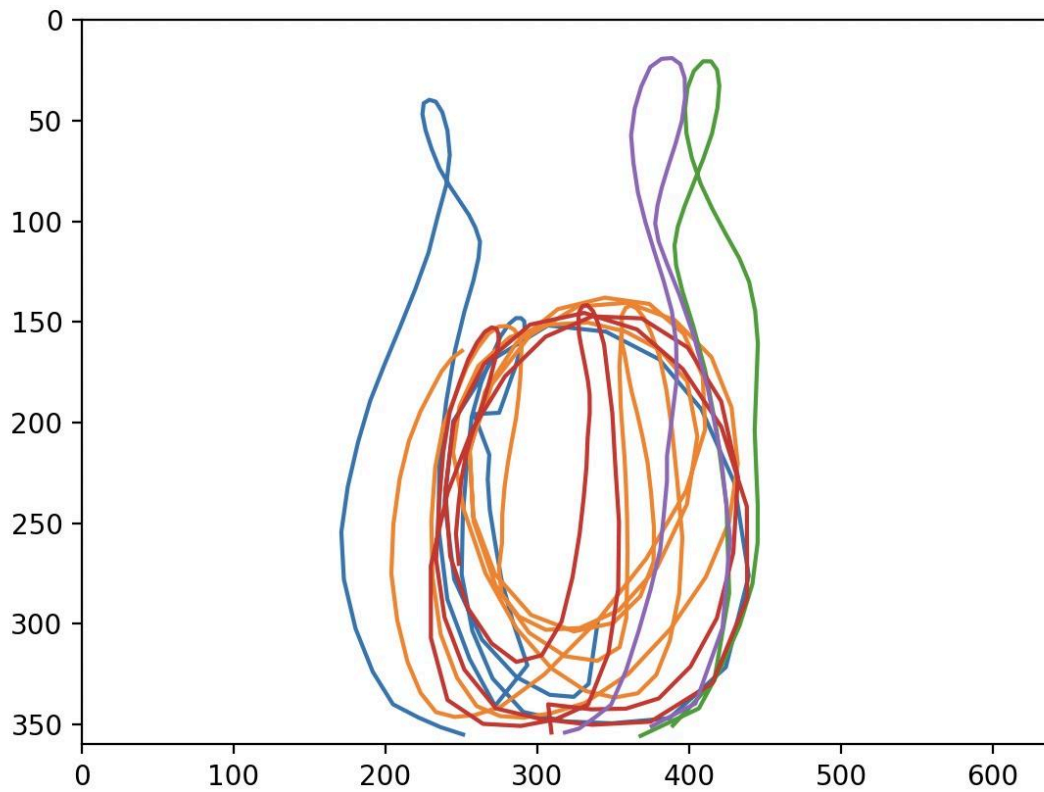
In the juggling pins video, when one of the pins exits the frame, it is identified and tracked as a new object, which is an issue we attempted to remedy, but was beyond the scope of our abilities. Instead, we abstracted away the complexity by declaring the objects as “gone.” In addition, at times the pins overlap each other and the object identification and tracking switches the object IDs and mistakes one for the other. We attempted to estimate the object’s intended next position by using its last change in position, but it was too difficult to get working.

Additional cool factors we implemented into this system beyond the minimum project requirements are the functionality that tracks objects based on colors selected by the user in an intuitive UI and the user can slow down and speed up the video in real time.

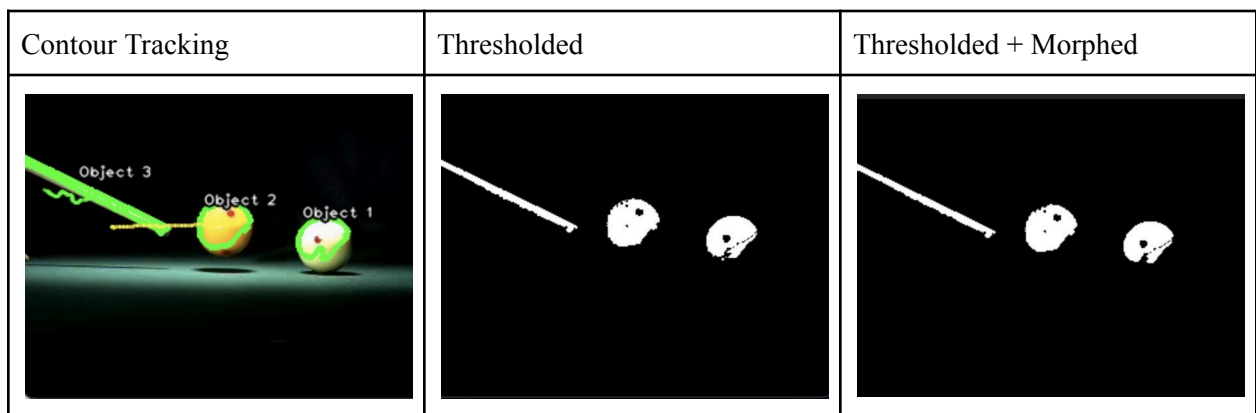
JUGGLING_PINS.mp4

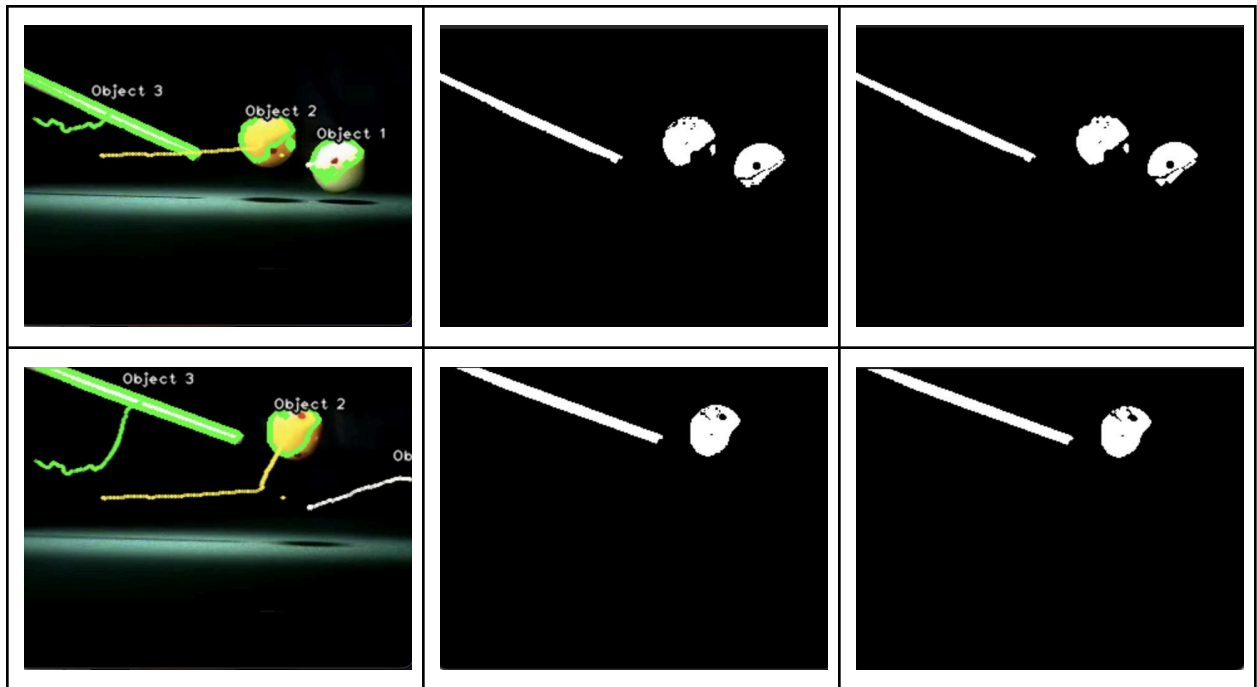
Contour Tracking	Thresholded	Thresholded + Morphed
		
		
		

Final Pixel Locations of Tracked Pins

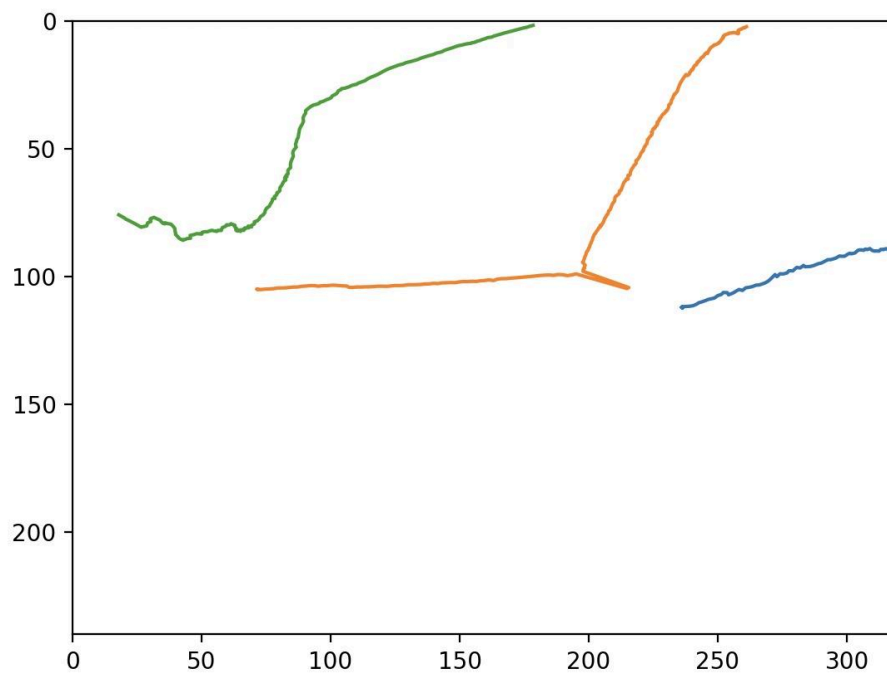


POOL_HIT_CUT.mp4





Final Pixel Locations of Tracked Pins



Video Links:

<https://www.youtube.com/watch?v=GVh5fu8m-vk>

<https://www.youtube.com/watch?v=avFjRgzTL-w>