CS6640 A1

Jake Bergquist

9/5/18

1) General Architecture for developing an image processing system for analyzing video sequences for moving vehicles.

In order to analyze a video to identify moving vehicles within the video an architecture would generally have to do the following: Take in the series of still frames that make up the video, distinguish what in the video constitutes a moving vehicles, and return an output that describes the vehicles that were identified.

The only file that should be needed would be the video as an input. The video consist of a series of still images taken at a regular interval.

These frames should be loaded into the function in an easy to access way. For matlab, given that these are rgb frames, a cell array with each cell containing one frame should do well. These frames should be treated as read only in order to preserve the video, and any operations to be done on these frames should be done on dedicated copies within the algorithm.

Once the frames have been imported and stored a next logical step would be to isolate a background from the video. All moving elements in the video can be isolated by comparing the frames to the background. The moving elements will be seen as different from the background. With the background eliminated each of the frames can be compared to it to determine which ones have movement. From here thresholding, image correlation, edge detection, and other such processing techniques can be used to detect what is moving within the frame, and compare it across the other frames. With these metrics the algorithm can build up a consensus of what is moving, if fits the criteria to be a car, as well as where to goes and how many other cars move in the image. The algorithm should return an analysis that includes details about how many cars were detected, what they looked like, and where they went.

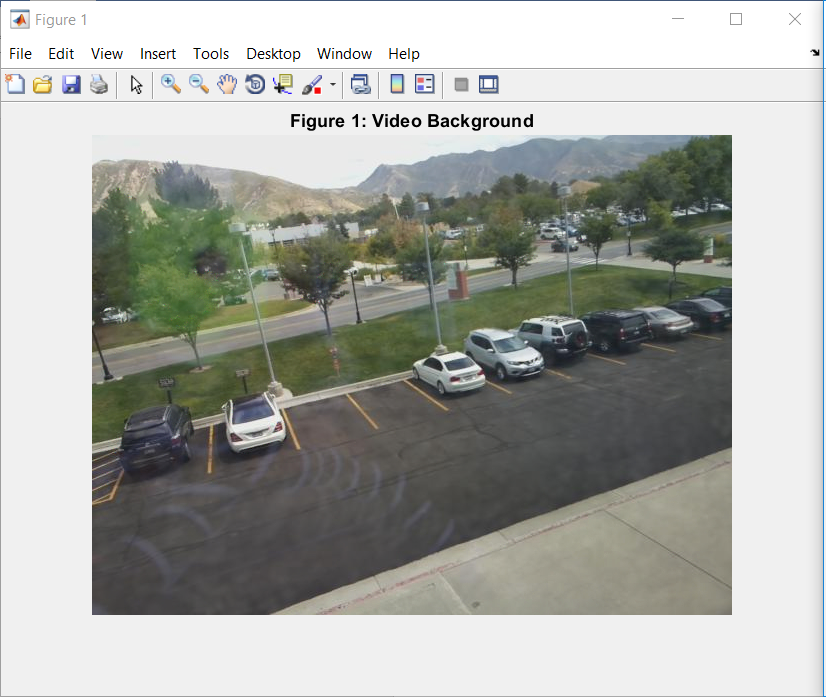
2) Acquiring a 5 second video from the ip camera and turning in the files for the frames and the video.

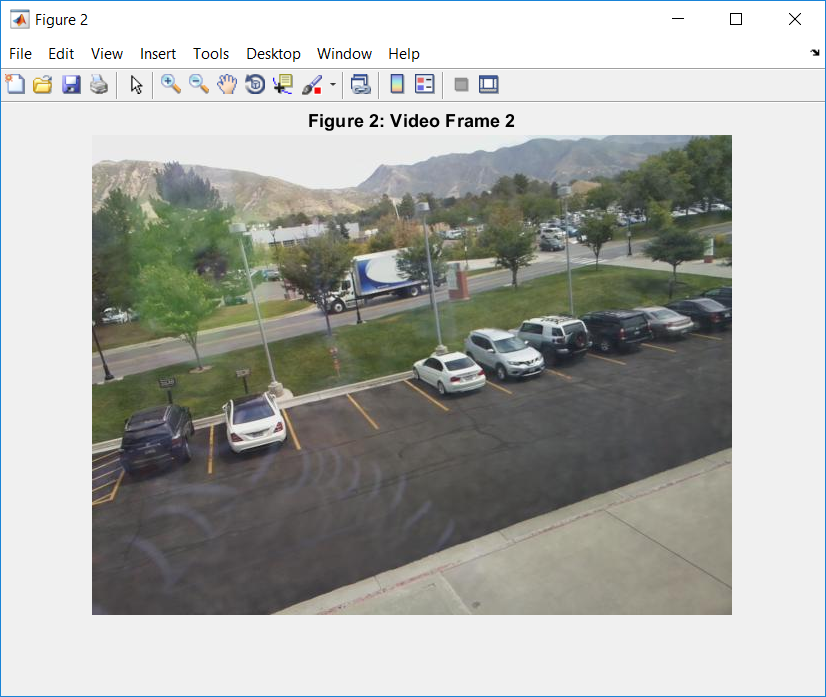
Files turned in with report.

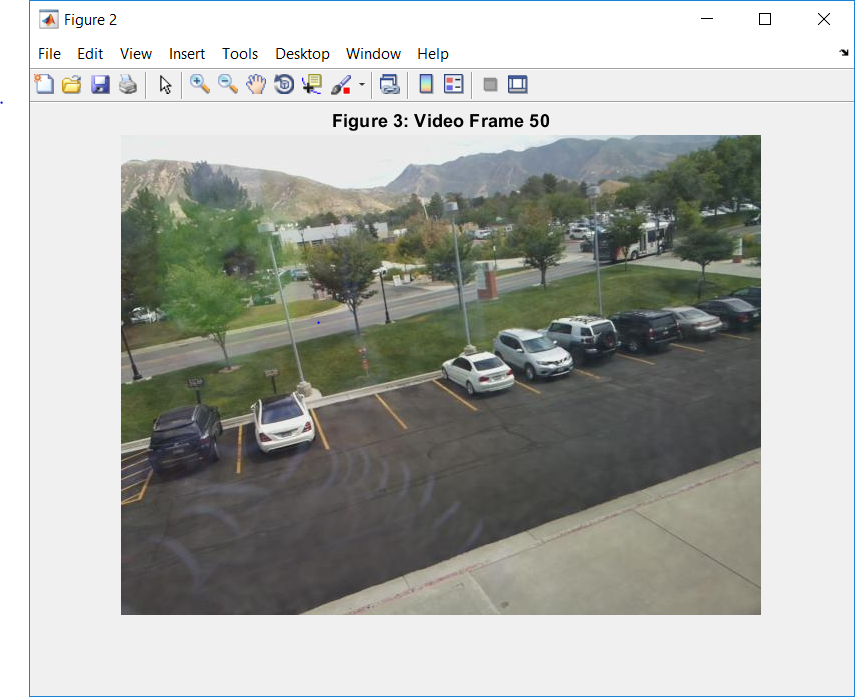
3) Developing code to attain the background of the video from the IP cam.

In order to isolate the background of the image the first step is to load in the frames from the video object. By setting the video time to 0 we ensure that we are starting at the beginning of the video. We can then calculate the number of frames by multiplying the frames per second by the duration of the video. This results in a total number of 75 frames for my specific video (5 second duration with 15 fps). Next we iteratively use the readFrame function to read each of these frames and save them out into a cell array, where each cell contains one frame. Now that we have the frames extracted we can work on attaining the background image for all the frames. Given that this is a video where most of the things are not moving and the things that do move, a few cars, go across the video rather quickly, we can say that most of the pixels in the video spend most of their time as background. As such if we take the mode of all of the RGB values for each pixel that will give us a good background image. To do so we first have to extract all the red, green, and blue values for each frame and group them. We can then use the mode function to find the mode for each pixel. We then use this mode to reconstruct an image which is our background for the video. As a last step we make sure to convert the double output of the mode function back to an int8 so that it can be displayed properly as an image.

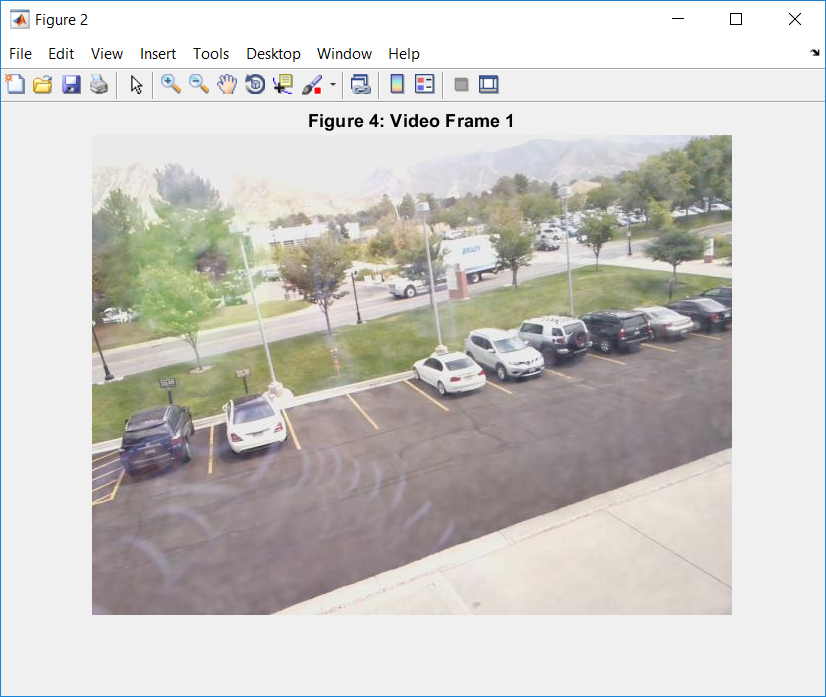
The resulting background image (**Figure 1**) can be compared to the second frame (**Figure 2**) and the fiftieth frame (**Figure 3**)



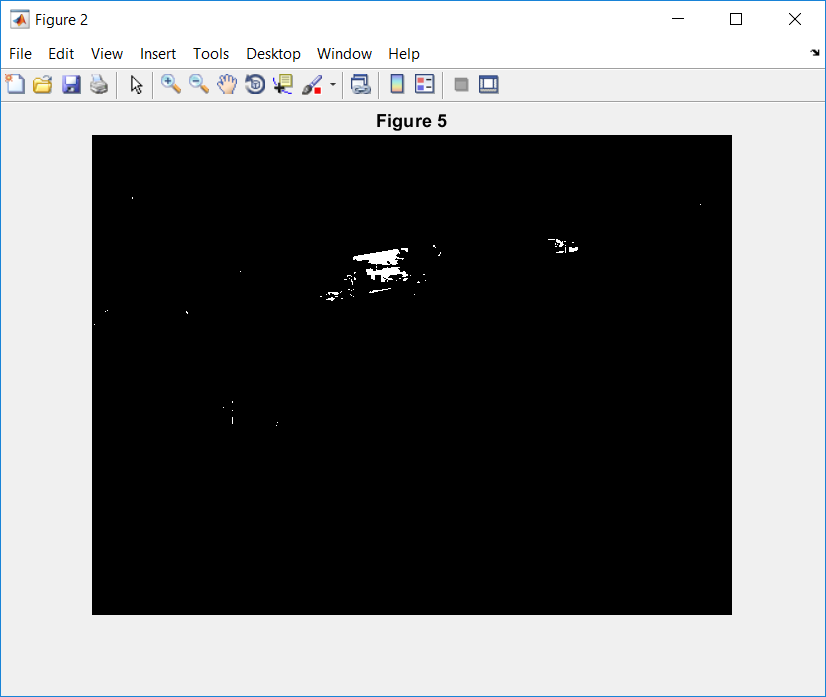




In frames 1 and 50 we see cars (a truck in frame 2, and a bus in frame 50) that are driving across the road. Ideally we would like to remove these to form our background which is composed of all of the stationary things from the video. We see that this is accomplished in our background frame. Of note, the first frame is much brighter than the others (**Figure 4**) but this does not show up on the background frame. This should be considered for future analysis of the video. It could have been caused by an artifact of the IP camera first transmitting, or perhaps there was an excessive glare during that frame. This is not seen in the subsequent frames.



As a sanity check we can subtract the background image from frame 2 and run a threshold. This should show us anything from that frame that was moving relative to the background. This can be seen as the car from frame 2 is highlighted in **Figure 5**.



We can see the moving car from frame 2 is mostly captured, except for the trees blocking its path. Additionally it would seem that a few other elements were identified such as the slightly moving trees. Overall this demonstrates that our background image processing function is successful.

4) Investigating the use of correlation of a small section of the image with the rest to segment sections.

I started by using a sub section of the image containing a bumper of one of the parked cars to see if I could find other car bumpers in the image. Using the section in **Figure 6** taken from frame 2 (**Figure 2**) I ran a correlation across the entirety of frame 2 using xcorr2(image, filter). I then scaled the values of the correlation from 0 to 255, and performed a threshold at corrOutput > 185. All pixels that passed this threshold were colored red (by setting their R value to 255 and their G B values to 0). As we can see the white car bumper correlated most strongly with itself and the very white sky (**Figure 7**). To elucidate this I wrote a quick function to outline the areas that pass the threshold with a red box the size of the input filter for the correlation (**Figure 8**); As we can see in **Figure 8** the car that we used as our input image, as well as the sky and a bit of the sidewalk had a high enough correlation to pass our threshold.