ELEN90095 — AI For Robotics — Sim-2-Real Part A

Tianchen Lou – 1078368 – tianchenl@student.unimelb.edu.au
Arya Araban – 1439683 – aaraban@student.unimelb.edu.au

A.1:

Below are some examples of images collected during stationary operation of the car.

A.2:

Parameters:

The most important parameters for the detection of lines in the images are the hue, lighting, and saturation values. These values help highlight the distinctive green line by either detecting the presence of green colours, or the brightness of the line against its surroundings. Parameters for the quantity and angle of the detection arc is also a key parameter to detect curvature in the line as it progresses.

Limitations:

Some of the most difficult images for line detection are ones under the table, where the overall lighting is darker. This reduces contrast for hue and saturation as there is less reflectivity from the tape, and makes lighting as a parameter very difficult to operate. Additionally, images with parallel or branching lines may also be challenging, as the algorithm is only capable of detecting one path.

Challenges:

Challenges faced during this experiment mainly revolved around tuning parameters for line detection to work in all situations. Using further parameters as well as morphological operations such as erosion and dilation helped alleviate difficulties in line detection. Otherwise there were occasional connection issues and bugs with the benchmarking software, but those were able to be troubleshooted as well (and we appreciate how much time was spent developing and testing these!!).

A.4:

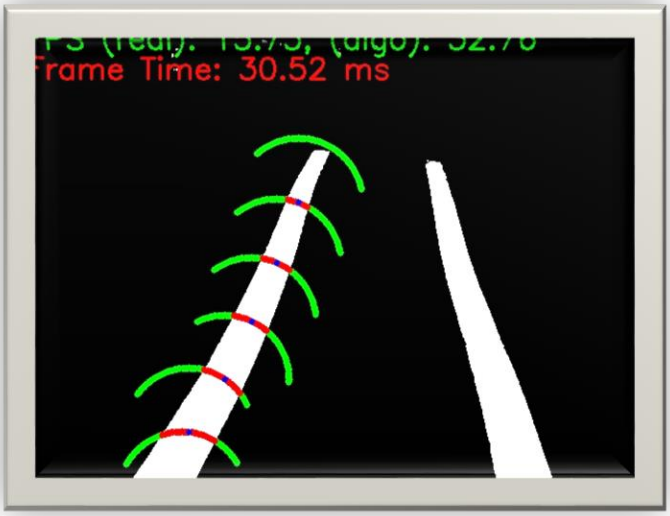
Below is an example of a blurry image collected while the car was in motion. It is blurrier than the stationary images.

A.5:

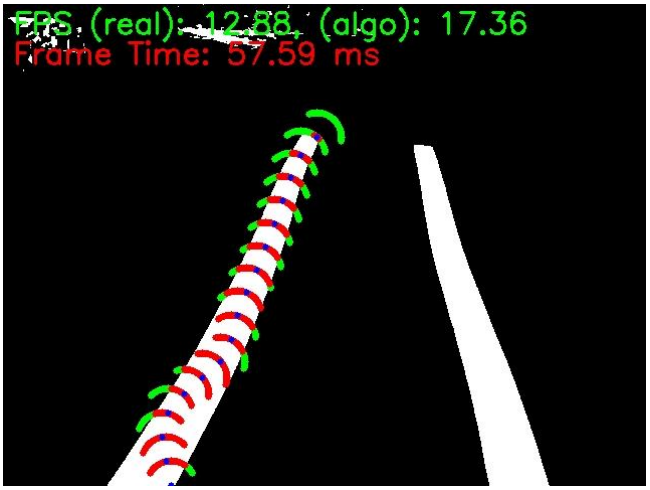
Parameter sets	HSL filtering	Morphological operations	Detection arc	Computation time
Set 1	Saturation: 100 - 255	None	Radius: 65 Arc: -60 - 60	29.4ms
Set 2	Light: 147 – 255 Hue: 34 – 111	None	Radius: 25 Arc: -60 - 60	62.2ms
Set 3	Saturation: 55 – 246 Light: 62 – 244	Erosion: kernel 3 Dilation: kernel 3	Radius: 10 Arc: -60 – 60	128.1ms

Parameter values were chosen based on overall performance across all possible line conditions, including noisy and dark lines. The values of the 3 final parameter sets are detailed above, with variations between their selected HSL values and other factors. The performance of all 3 configurations are displayed below

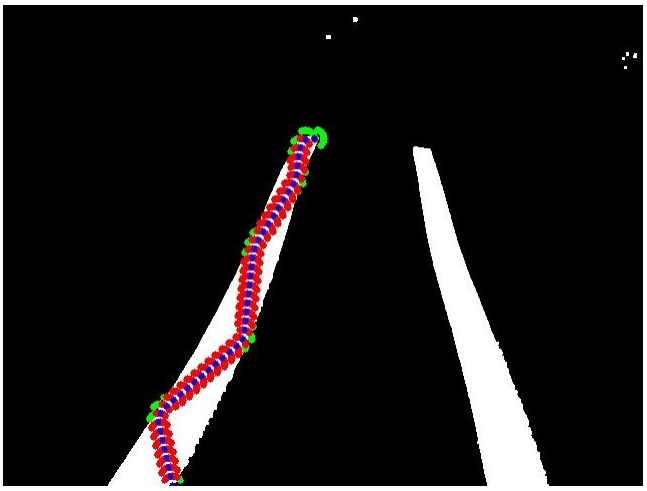
Config 1:



Config 2:



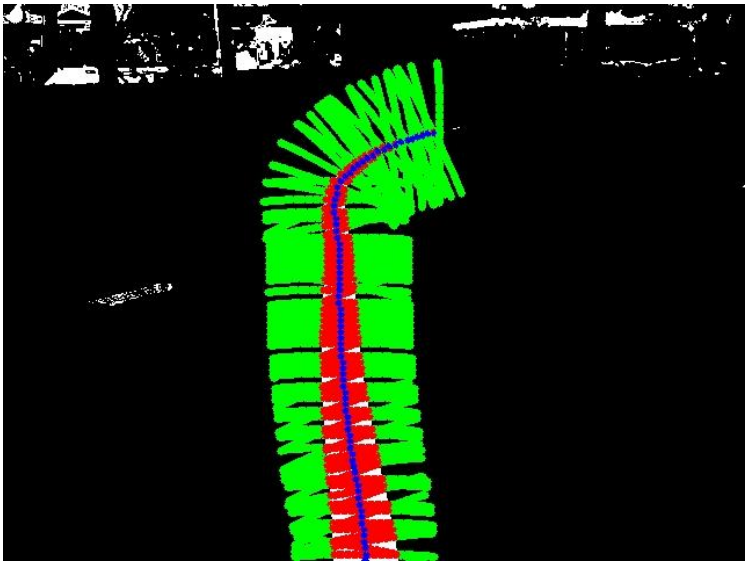
Config 3:



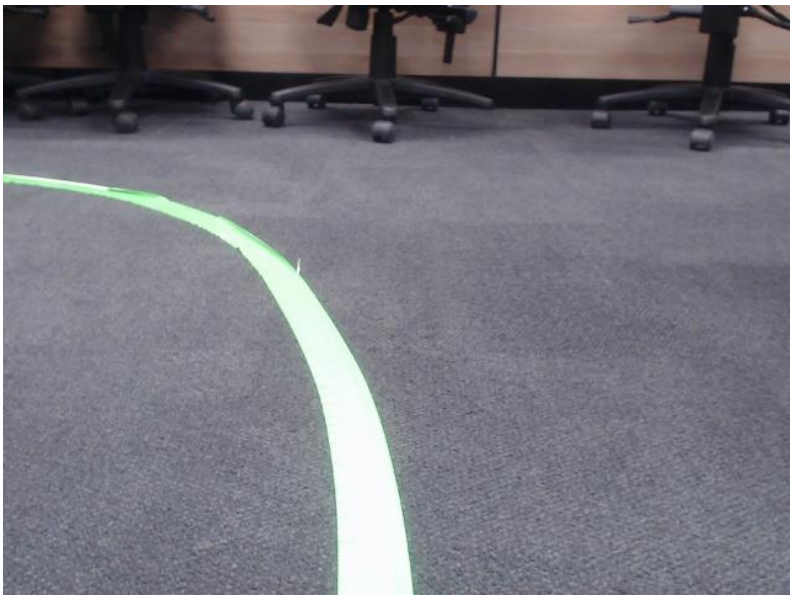
A.6:

Benchmark parameter	HSL filtering	Morphological operations	Detection arc
Benchmark set 1	Saturation: 100 – 255	None	Ellipse A: 70 Ellipse B: 5 Arc: -60 – 60

The benchmark parameter set was the most accurate model of path prediction used testing. The configuration for it is displayed above. Below is an example of a line detection process from the benchmark parameters.



Given the following base picture:



Performance of the sets against the benchmark are displayed below for the base image.

Parameter set	Performance against benchmark MSE
Set 1	45.9
Set 2	55.7
Set 3	160.9

A.7:

The live line detection results are displayed in the image below, using the configuration set 1. From the table in A.5, it can be seen that the computation time is overall similar relative to the stationary detection time.

When increasing the camera frame rate for faster control loop response, a computation buffer is required between the frame interval and line detection processing time to prevent lag. At 5 FPS, the 200ms interval (calculated as 1/5 x 1000) is sufficient for the 39.40ms algorithm. Although as the frame rate increases, the interval between frames decreases. For example, at 30 FPS the interval is 33ms, which is less than the 39.40ms processing time. In this case, a buffer of 6ms would be required, calculated as the difference between the processing time and frame interval.

A.8:

We will use configuration set 1 for the line detection parameters in the control loop due to its balance of accuracy and speed. This configuration achieved high benchmark accuracy with a low mean squared error of 45.9. More importantly, it had the fastest computation time at 29.4ms per frame compared to the other configurations. The fast computation enables using a higher camera frame rate in the control loop, allowing faster reactions to changes during autonomous control. The higher sensor data throughput improves control loop responsiveness.

A.9:

To verify the line detection algorithm for a particular application, additional testing should be conducted using diverse images captured under the expected real-world operating conditions. This will validate performance under different lighting, backgrounds, speeds, etc. Furthermore, the algorithm's accuracy can be quantified by comparison to manually labeled ground truth images showing the true line location. Testing should also involve profiling the processing time on the target hardware to ensure it meets real-time timing requirements at the desired frame rate. Automated regression testing may also be utilized on collected image datasets to catch errors during development.