AINT308 - OpenCV Assignment 2 2022

Student No. 10618407
School of Engineering,
Computing and Mathematics
University of Plymouth
Plymouth, Devon

***Abstract*—Machine vision is a mature technology that is becoming more prevalent within modern engineering practises. It is being utilised more in the rapidly evolving fields of autonomy and automation. This report outlines some of the functionalities of a popular C/C++ based computer visions library *OpenCV*. The Assignment has been split into three tasks; Task 1, Task 2, and Task 3. The first task, Task 1, is to evaluate the colors of pixels in a picture to determine the colour of a given object in the frame (car). The second task, Task 2, was to track on object across multiple frames of a video to track its motion (swinging pendulum). The final task, Task 3, was to identify and cross correlate components on a circuit to check for any missing components.**

Keywords:

Computer Vision, OpenCV, Object Detection, C++, Object Tracking

I. Task 4: Disparity Mapping

A. Introduction

B. Solution

C. Further Improvements

D. Conclusion

II. Task 5: Self-driving Car Lane Detection

A. Introduction

B. Solution

1) Use a method to detect the lane markings

The first part of lane detection was to convert the video frames to grey scale, this was done to reduce the informational load to compute. Although Canny Edge detection works less effectively using a grey scale image [1], the lines for the edges of the road were still found and detected as edges. The frames are converted to grey scale using OpenCV's `cvtColor` function - insert REF – insert figure of the code.

After the frames are converted to grey scale, the frame is then blurred using OpenCV's `blur` function – insert REF – insert fig of the code. This was done to reduce the noise in the image, this makes the edge detection perform better [2].

Once the image was blurred, a rectangular mask was placed over the top half of the image using the `rectangle` function - insert REF – insert fig of the code. This was used to eliminate the top half of the image which would not be useful for finding the road markings.

Canny Edge detection was then conducted on the masked frame using OpenCV's `Canny` method - insert OPENCV REF here – insert fig of the code. Canny Edge was chosen because it is deemed to be one of the best edge detection algorithms [3]. Canny Edge detection uses non-maximum suppression [4] and Hysteresis Processes [5]. Non-maximum Suppression and Hysteresis Process reduce the number of false edges and thus create a better starting point for further processing such as Hough Transforms.

Use Hough lines to create the lines -

2) Display a video with lanes detected

C. Further Improvements

D. Conclusion

References

[1] S. Malik and T. Kumar, "Comparative analysis of edge detection between gray scale and color image," *Communications on Applied Electronics*, vol. 5, pp. 38–43, 05 2016.
[2] K. Aishwarya, A. S., and V. Singh, "A comparative study of edge detection in noisy images using bm3d filter," *International Journal of Engineering Research and*, vol. V5, 09 2016.
[3] J. F. Canny, "Canny edge detection," 2009.
[4] J. Hosang, R. Benenson, and B. Schiele, "Learning non-maximum suppression," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 6469–6477.
[5] M. Sornam, M. S. Kavitha, and M. Nivetha, "Hysteresis thresholding based edge detectors for inscriptional image enhancement," in *2016 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)*, 2016, pp. 1–4.

The code can be found on GitHub here!

APPENDIX