Deliverables Summary

Michael McDonnell

October 25, 2019

Introduction

The thesis focus for the year was attempting to develop a system to allow a self driving vehicle to identify road features to set the conditions for navigation through features such as intersections.

The development attempted to follow an AGILE approach with iterative increases in functionality. A simulation system was developed to provide camera feed inputs to a Python process. The core deliverables discussed were:

- **Journal or Conference paper.**
- **Simulation with Inter Process Communication.**
- **Navigation localisation system.**

During the course of development familiarity was gained in a range of additional areas. Edge detection was investigated as an initial road detection attempt and the use of convolutional neural networks was investigated as an alternate option. As a result of this work outside the core system, the following additional deliverables have been developed:

- **MATLAB assignment detail for Computational Problem Solving.**
- **Sprint review example.**
- **???Machine learning familiarisation activities**

1 Journal or Conference Paper

It was identified that *Robotics and Autonomous Systems* is a suitable journal to submit a paper to. This journal deals with the core subject material, has an impact factor of 2.928 and has a clearly detailed submission process. If the paper is not accepted into this journal alternative options will be considered including conferences including optimistically, the 17th International Conference on Ubiquitous Robots (UR 2020) - Kyoto with a submission date of early 2020. A more flexible option is one of the various International Conferences on Control, Automation, Robotics and Vision Engineering throughout 2020.

The initial draft is aligned to the Robotics and Autonomous systems general format however content, style and length can be massaged based on submission target.

2 Simulation with Inter Process Communication

TODO: Intro, tidying up and commenting, documentation, extensions

3 Navigation localisation system

The provided system includes test data for a proof of concept. In addition to the provided files, **cv2** and **numpy** are both required. Provided files are as follows:

- *nav_localisation.py* - This is the core file which is run. The default method here calls a single *main* method and passes test data (see below). The main work in this script is from the *process_frame* method which maintains a simple state using boolean flags to determine if detection or tracking code is run. This script also contains visualisation code as well as video saving function.
- *road_surface_detection.py* - Class to use as a road detector, implements Histogram backprojection road surface detection. *get_road_surface_from_new_frame* is the Class method called to detect the road surface in a new frame.
- *feature_tracker.py* - Obtain initial feature mask using *get_feature_masks* and check for feature detection using *check_feature*. This includes argument for optical flow which is estimated using *GetUpdatedRoadFeatureLocationFarneback*.
- *bezier.py* - Helper script to develop bezier curve mask using *get_curve_mask*. Generic in that *draw_bezier* can be called directly to draw bezier curve to any image.
- *test_data.py* - Implements the data interface required (see following subsection) and contains test methods to confirm data.

3.1 Data interface

The system expects a data interface with the following methods:

- *get_next_feature()* - Returns next feature or None if no more available.
- *get_next_frame()* - Returns next image frame (either from video stream or saved images).
- *get_inverse_perspective_matrix()* - Returns the inverse perspective matrix developed for this dataset.
- *get_ipm_mask()* - Returns a mask which defines the area of information for the inverse perspective matrix.

The provided code demonstrates the concept of the system and is suitable for extension, generalisation and improvement. Any script that implements the data interface can be passed to the main method in *nav_localisation.py* for processing which allows easy implementation of multiple different input source files. The data interface provided uses images and an inverse perspective transformation matrix included in the *TestData* subfolder.

4 MATLAB assignment detail for Computational Problem Solving

It was identified that filter convolution is a (relatively) simple algorithm to implement and may be a suitable teaching case for a practical implementation of loops.

A candidate assignment task was developed with an example solution included. This has the dual effect of providing a practical example and passing on some conceptual understanding of how image filtering works. **TODO: Polish example solution, Create assignment task**

5 Sprint review example.

The feature development of the system and simulation was managed using an AGILE approach. This involved incremental feature delivery with a fortnightly review. Fortnightly reviews included triaging of future work but also the provision of an update to interested stakeholders. The review includes the 'velocity' over the previous fortnight which is a quantified measure of work based off the estimated 'story points' per task that were completed in the sprint. Following this a presentation of the current capability (generally focussing on impactful demonstrations of new features developed) occurs and discussion with stakeholders. After the sprint review, tasks are selected for work over the following fortnight based off the existing velocity.

This assists with planning time allocations (using existing knowledge to estimate small task difficulty) while retaining flexibility to rapidly pivot in focus.

The sprint reviews throughout the year morphed into an informal brief review due to the difficulty in locking in standing appointments however the theory was maintained. An example of a formal sprint review is included as an example to individuals in future that may be interested in adopting this approach and are looking for a starting point.

6 ???Machine learning familiarisation activities

The use of Convolutional Neural Networks (CNN) for road surface detection was seriously considered. As part of analysing the potential effectiveness of this solution an effort was made to develop a robust understanding of neural network (and more generally, machine learning algorithms) implementation from first principles as well as small 'toy' projects to gain a working familiarity with implementation details. A focus on interpretable AI and a desire to have a more complete solution (with driving lines and intersection tracking) meant that a CNN approach was not pursued however a good understanding of the application was developed.

The Stanford University online Machine Learning course was undertaken which provided a mathematical understanding of multiple machine learning algorithms. The certificate of completion of this is included. On completion of this course familiarisation projects were developed using Python and Keras which is a Tensorflow wrapper.

TODO: Projects undertaken TODO: How they work

TODO: Wording of this deliverable title

Final words

The project this year was ambitious, challenging and allowed development of conceptual understanding over a range of topics. In addition to the core academic deliverable requirements it is hoped that some of the additional delivered elements may assist others in delving deeper into some of these topics.