Autonomous Driving Robot

CS39440 Major Project Report

Author: Josh Wells (jow102@aber.ac.uk)

Supervisor: Patricia Shaw (phs@aber.ac.uk)

24th April 2022

Version 1.0 (Draft)

This report is submitted as partial fulfilment of a BSc degree in
Artificial Intelligence & Robotics (GH76)

Department of Computer Science
Aberystwyth University
Aberystwyth
Ceredigion
SY23 3DB
Wales, UK

Declaration of originality

Consent to share this work

Acknowledgements

I am grateful to…

- Ben Weatherley (bew46@aber.ac.uk)
- Patricia Shaw (phs@aber.ac.uk)

Abstract

Increasingly cars are becoming more autonomous, giving warnings for speed limit signs and controlling the speed of the vehicle. Meanwhile, fully autonomous cars are being developed and tested on the roads in a variety of situations. There are many problems surrounding autonomous cars, and this project will start to explore some of those complexities.

The goal of this project was developing a Robot Operating System (ROS) based program to work with Turtlebot3, a relatively simple robot, inside a simulated environment (Gazebo) but with the potential for it to be transferred to a physical robot. The project aimed to tackle the issues surrounding lane detection, and the adjustment of the robot's driving behavior to accurately follow the detected lane.

The main body of this project involved finding the best computer vision techniques, using OpenCV, to achieve an accurate lane detection algorithm that is also fast enough to be ran on a live video feed with minimal delay. The project explored multiple different approaches to lane detection, investigating each techniques advantages and disadvantages.

The end-goal of the project was the development of a robot that can autonomously navigate a simulated road environment while obeying UK traffic laws.

Contents

1. Background, Analysis & Process

1.1. Background

I did not have any previous practical experience with implementing computer vision techniques, but I have an interest in computer vision and its applications. I very much wanted to learn how to successfully implement the techniques I had studied into a functioning program. The coupling of computer vision and robotics is what motivated me to select this autonomous driving robot as my project.

In preparation for this project, I taught myself methods for implementing OpenCV to work alongside ROS on a live, simulated robot inside a simulated environment mainly using a YouTube tutorial about the basics ***(Murtaza's Workshop – Robotics and AI, 2020)***. I have plenty of practical experience programming in C++, so decided to continue the project using C++ over Python. I began by researching implementation techniques for OpenCV in C++, and created a few simple practice programs to help me get used to the OpenCV library.

In order to get OpenCV to work alongside ROS I had to make use of the cv_bridge ***(ROS.org, 2010)*** library available for C++. This required some research and a small bit of practice to get the live raw image from the robot into a format which I could then use for image processing with OpenCV.

I assessed a number of different existing systems, including a paper entitled 'Automatic Driving on Ill-Defined Roads: An Adaptive, Shape-constrained, Colour-based Method' ***(Marek Ososinski and Frédéric Labrosse, 2013)***. This paper investigated the creation of a system that can identify the size and shape of the road ahead of it, before navigating along the road. The method discussed in this paper was effective, but complicated. Designed for use on ill-defined roads (e.g. country roads or lanes), this method adds multiple layers of complexity that my project did not require. As my project would take place inside a simulated, well-marked road environment there would be no need for the extra steps taken by this method to identify the drivable area in front of the robot.

Another method which I investigated made use of a series of filters, that are available within the OpenCV library, that would be able to detect the lines on either side of the road ***(Pysource, 2018)***. This method works well when there are clear markings on the road, once the edges of the lane/road are detected I would then be able to successfully calculate the robots movement path along the center of the lane.

1.2. Analysis

The problem involves being able to successfully detect the edges of a lane/road, from the background work I can conclude that there are many different ways that one can achieve this goal. The approach taken by Marek Ososinski and Frédéric Labrosse ***(2013)*** provides a robust solution to detecting the width of the road ahead of the robot. By making use of a shape-constrained, colour-based method they were able to determine the exact edges of the road and draw a trapezoidal shape based on the shape of the detected road edges. This shape would then be able to determine the drivable area in front of the robot and use this information to steer, keeping the robot in the center of the road at all times. While this approach is extremely robust and would provide an accurate lane following behavior for this project, it adds a number of layers of complexity that I felt would not be achievable in the time-span of this project.

In light of this, I felt a better method for this project would be to attempt to identify the marked lines on either side of the road before then calculating the centre of the lane for the robot to drive along. This approach, while not as robust as the previously mentioned one, seemed far more achievable in the time-span of this project.

The main tasks of the project were as follows:

- Create a simulated road environment for the robot to drive in.
- Develop an algorithm that can successfully identify the marked lines on the road (left and right).
- Develop the line detection algorithm further, so that it may determine the size of the lane.
- Calculate the centre of the detected lane.
- Develop a lane following behaviour for the robot that will drive along the centre of the lane determined by the previous algorithm.

I arrived at this list of objectives based on what I believe is achievable in the time available for this project.

1.3. Process

You need to describe briefly the life cycle model or research method that you used. You do not need to write about all of the different process models that you are aware of. Focus on the process model that you have used. It is possible that you needed to adapt an existing process model to suit your project; clearly identify what you used and how you adapted it for your needs.

The life cycle model that I used to develop my project was based on an Agile Scrum approach.

2. Design

2.1. Overall Architecture

The image is then passed through a number of OpenCV filters that will allow it to identify the lines on the road that mark the edges of the lane. The image is first cropped, removing unneeded information from the top of the screen. It is then converted to HSV colour space, before being passed through a colour filter that mask off anything in the image that does not match the set filters. The only thing remaining on the screen at this point should be the white lines on the road, the image is then passed through a canny edge detector, which will detect the edges of the lines on the road. Using the information gained from the edge detector, it is then possible to use the HoughLinesP to draw straight lines along the screen and mark their start/end positions.

Using the stored XY coordinates of each of these detected lines I can then determine the most appropriate lines to use to draw the edges of the lane. Once these lines are drawn, I can use the XY coordinates of both left and right lines to determine the centre of the lane which then allows me to decide to if the robot needs to turn left or right, or keep driving straight.

2.2. Detailed Design

2.2.1. Support & Implementation Tools

The programming language selected for this project is C++; I chose this language as it provides an in-depth library that I feel gives me more control over the project that I do not get from other programming languages such as Python. Python in general is a fantastic choice for creating ROS programs, however I have far more practical experience working with C++ compared to Python. Due to this, C++ was the obvious choice for this project.

I made use of the CLion IDE to write the code, which gave me some basic debugging tools which helped to spot some of the more obvious errors before attempting to compile. I made use of Catkin to compile the ROS programs quickly, this would also provide me with fairly detailed error messages if any errors were found in the program which made turnaround times much faster when implementing new code.

This is a ROS based project, designed to run on a live robot using a subscriber/publisher system. ROS allows me to subscribe and publish to specific topics that then control the robot. The camera from the robot will publish its image to the 'camera/rgb/image_raw' topic, my program then subscribes to that topic where it can then get a message containing the raw image direct form the robot. Using the cv_bridge library, I am able to convert that message into an image that can then be used for image processing with OpenCV.

The robot is simulated within Gazebo, this allows me to do repeated tests with the program without needing a physical, real-world robot. Gazebo allowed me to create a test environment that mimicked a real-world road. Due to safety and other limitations with the project, a real-world robot operating on a real road was not possible. The environment model for the road was provided to me by Ben Weatherley, a student who completed a similar project last year entitled *****GET PROJECT NAME***** ***(Ben Weatherley, 2021)***. The road system modelled was that of a short loop, with a number of sharp 90 degree turns, 2 bus stops and 2 give-way junctions.

2.2.2. Image Processing

A subscriber call-back function first takes the raw image message from the ROS topic ‘camera/rgb/image_raw’. The call-back function will then convert this message into a useable image and assign it to a CvImagePtr, before then making use of OpenCV libraries’ image.clone() function and cloning the image onto a global variable which can then be accessed and used by the image processing function.

The first step in the image processing technique for lane detection is to crop the image. The raw image from the robot will contain a lot of information that is not useful, and can actually cause extra issues if not removed. To do this, the image is cropped using a Rectangle of Interest (ROI) which defines the area that I wish to keep. After this, only the lower half of the original image is visible.

The image from the robot is in the Blue, Green, Red (BGR) colour space. This colour space provides a natural looking image and colours however it is not particularly easy to mask. To get around this, the image is converted into the Hue, Saturation, Value (HSV) colour space. The HSV colour space is much simpler to mask specific colours, which makes the next step of the algorithm much easier.

Thirdly, the image has a colour mask applied to it so that only areas that match the set range of HSV values will be seen on the image. In the case of my project, the algorithm will mask off any area that is not white. This leaves only the white road markings visible on the image. Removing the extra unneeded areas from the image allows the edge detection and line drawing in the next steps to be more accurate.

The image is then ran through a canny edge detector, which will create a number of points along the detected edges on the screen. Since the only things on the image at this point should be the road markings, this means the result from the edge detector will be the edges of all the lines on the road.

Lastly, using the information from the canny edge detector it is possible to pass the image through a HoughLinesP function. This uses the edge detection image to calculate and draw straight lines along the screen which will have their start and end XY coordinates stored into a vector. The vector will store all of the detected lines in the scene, but they will still need sorted. An algorithm will then go through the stored lines and try to determine the most appropriate points to use, before calculating the a line to be drawn which will represent the edge of the lane on both left and right sides as best as possible. More information on this algorithm is included in the next section.

2.2.3. Selection of Appropriate Hough Lines

The generated Hough Lines are stored inside a vector, each line has 2 points (start and end) and each point has 2 values (X and Y coordinates). Selecting the most appropriate lines to use as markers for the edge of the lane requires an algorithm that can systematically go through each of the stored lines and decide if the position of the current line is better than the previously checked line.

The diagram below (Fig1) depicts an example image from the robot. The black lines on the image are the detected Hough Lines, representing the road markings visible to the robot. Each of these lines will have start and end XY coordinates that will need to be checked in order to find the best possible points for tracking the edges of the lane.

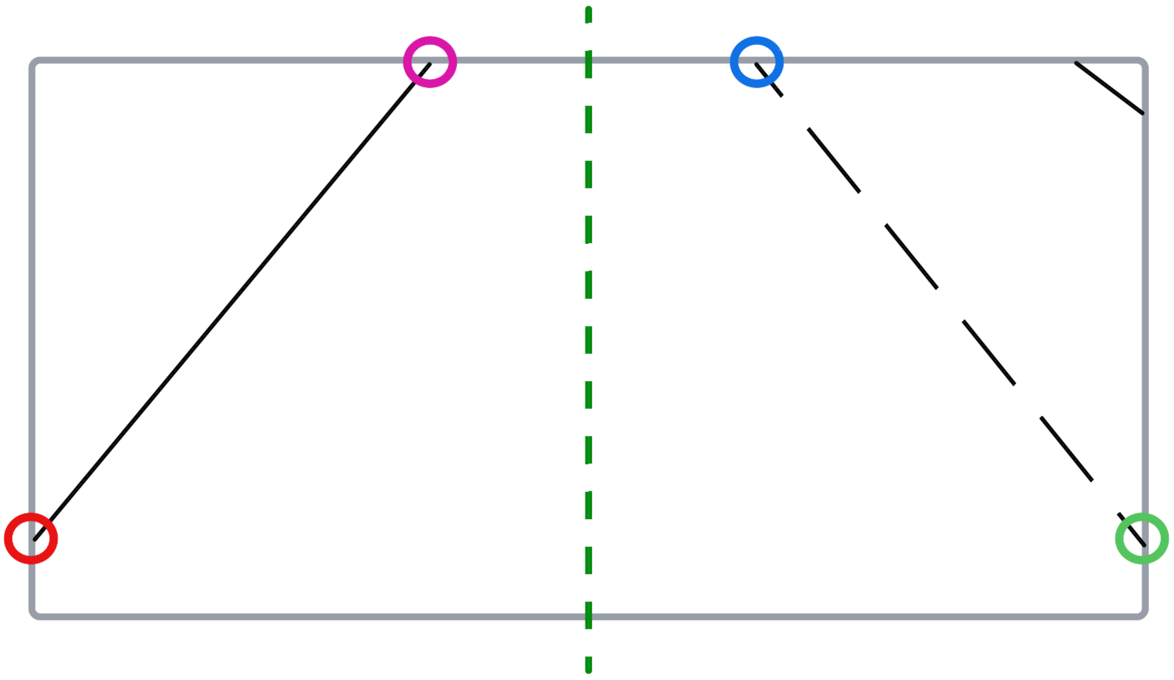

Fig1 (above): Example of Hough Lines selection

The algorithm will select the following:

- The lowest point on the left side of the image (Fig1, RED CIRCLE).
- The lowest point on the right side of the image (Fig1, GREEN CIRCLE).
- The highest point on the left side of the image (Fig1, MAGENTA CIRCLE).
- The highest point on the right side of the image (Fig1, BLUE CIRCLE).

Once the algorithm has selected the 4 most appropriate points, the program will draw a left and right line on the image that will represent the edges of the lane.

2.2.4. Calculating Centre of Detected Lane & Driving

3. Implementation

The implementation should discuss any issues you encountered as you tried to implement your design. During the work, you might have found that elements of your design were unnecessary or overly complex; perhaps third-party libraries were available that simplified some of the functions that you intended to implement. If things were easier in some areas, then how did you adapt your project to take account of your findings?

It is more likely that things were more complex than you first thought. In particular, were there any problems or difficulties that you found during implementation that you had to address? Did such problems simply delay you or were they more significant?

You can conclude this section by reviewing the end of the implementation stage against the planned requirements.

4. Testing

Detailed descriptions of every test case are definitely not what is required in this section; the place for detailed lists of tests cases is in an appendix. In this section, it is more important to show that you adopted a sensible strategy that was, in principle, capable of testing the system adequately even if you did not have the time to test the system fully.

Provide information in the body of your report and the appendix to explain the testing that has been performed. How does this testing address the requirements and design for the project?

How comprehensive is the testing within the constraints of the project? Are you testing the normal working behaviour? Are you testing the exceptional behaviour, e.g. error conditions? Are you testing security issues if they are relevant for your project?

Have you tested your system on “real users”? For example, if your system is supposed to solve a problem for a business, then it would be appropriate to present your approach to involve the users in the testing process and to record the results that you obtained. Depending on the level of detail, it is likely that you would put any detailed results in an appendix.

Whilst testing with “real users” can be useful, don't see it as a way to shortcut detailed testing of your own. Think about issues discussed in the lectures about until testing, integration testing, etc. User testing without sensible testing of your own is not a useful activity.

The following sections indicate some areas you might include. Other sections may be more appropriate to your project.

4.1. Overall Approach to Testing

4.2. Automated Testing

4.2.1. Unit Tests

4.2.2. User Interface Testing

4.2.3. Stress Testing

4.2.4. Other Types of Testing

4.3. Integration Testing

4.4. User Testing

5. Critical Evaluation

Examiners expect to find a section addressing questions such as:

- Were the requirements correctly identified?
- Were the design decisions correct?
- Could a more suitable set of tools have been chosen?
- How well did the software meet the needs of those who were expecting to use it?
- How well were any other project aims achieved?
- If you were starting again, what would you do differently?

Other questions can be addressed as appropriate for a project.

The questions are an indication of issues you should consider. They are not intended as a specification of a list of sections.

The evaluation is regarded as an important part of the project report; it should demonstrate that you are capable not only of carrying out a piece of work but also of thinking critically about how you did it and how you might have done it better. This is seen as an important part of an honours degree.

There will be good things in the work and aspects of the work that could be improved. As you write this section, identify and discuss the parts of the work that went well and also consider ways in which the work could be improved.

In the latter stages of the module, we will discuss the evaluation. That will probably be around week 9, although that differs each year.

6. References

This final section should list all relevant resources that you have consulted in researching your project.

[1] Sylvia Duckworth. A picture of a kitten at Hellifield Peel. http://www.geograph.org.uk/photo/640959, 2007. Copyright Sylvia Duckworth and licensed for reuse under a Creative Commons Attribution-Share Alike 2.0 Generic Licence. Accessed August 2011.

[2] Mark Neal, Jan Feyereisl, Rosario Rascunà, and Xiaolei Wang. Don't touch me, I'm fine: Robot autonomy using an artificial innate immune system. In *Proceedings of the 5th International Conference on Artificial Immune Systems*, pages 349–361. Springer, 2006.

[3] W.H. Press et al. *Numerical recipes in C*. Cambridge University Press Cambridge, 1992.

[4] Various. Fail blog. http://www.failblog.org/, August 2011. Accessed August 2011.

[5] Apache Software Foundation (2014) "*Apache POI - the Java API for Microsoft Documents*" (Online) Available at: http://poi.apache.org Accessed: 14th March 2014.

[6] Apache Software Foundation (2004) "Apache License, Version 2.0" (Online) Available at: http://www.apache.org/licenses/LICENSE-2.0 Accessed: 14th March 2014.

[7] Neil Taylor, "MMP Information", 2021 (Online) Available at: https://teaching.dcs.aber.ac.uk/docs/2022/MMP/information/ Accessed 28th February 2021.

7. Appendices

The appendices are for additional content that is useful to support the discussion in the report. It is material that is not necessarily needed in the body of the report, but its inclusion in the appendices makes it easy to access.

If you have used any 3rd party code, i.e. code that you have not written yourself such as libraries, then you must include Appendix A. In that appendix, you will provide details of the 3rd party code that you have used.

For most other items, it would be better to include them in your technical submission instead of including them as an appendix. For example:

- If you have developed a Design Specification document as part of a plan-driven approach for the project, then it would be appropriate to include that document in the technical work. In this report, you would highlight the most interesting aspects of the design, referring your reader to the full specification for further detail.
- If you have taken an agile approach to developing the project, then you may be less likely to have developed a full requirements specification at the start of the project. Perhaps you used stories to keep track of the functionality and the 'future conversations.' If it isn't relevant to include all those stories in the body of your report, you could detail those stores in a document in the technical work.
- If you have used manual testing, then include a document in the technical work that records the tests that have been done. In this report, you would talk about the use of those tests.

Documents included in the technical work or in the appendices are supporting evidence of the work done. Where you include documents, this report should refer to the documents. You should not be relying on detailed study of those documents in order to understand what is written in this report.

Speak to your supervisor or the module coordinator if you have questions about this.

A. Third-Party Code and Libraries

If you have made use of any third-party code or software libraries, i.e. any code that you have not designed and written yourself, then you must include this appendix.

As has been said in lectures, it is acceptable and likely that you will make use of third-party code and software libraries. If third-party code or libraries are used, your work will build on that to produce notable new work. The key requirement is that we understand what your original work is and what work is based on that of other people.

Therefore, you need to clearly state what you have used and where the original material can be found. Also, if you have made any changes to the original versions, you must explain what you have changed.

The following is an example of what you might say.

Apache POI library – The project has been used to read and write Microsoft Excel files (XLS) as part of the interaction with the client's existing system for processing data. Version 3.10-FINAL was used. The library is open source and it is available from the Apache Software Foundation [5]. The library is released using the Apache License [6]. This library was used without modification.

Include as many declarations as appropriate for your work. The specific wording is less important than the fact that you are declaring the relevant work.

B. Code Samples

This is an example appendix. Include as many appendices as you need. The appendices do not count towards the overall word count for the report.

For some projects, it might be relevant to include some code extracts in an appendix. You are not expected to put all of your code here - the correct place for all of your code is in the technical submission that is made in addition to the Project Report. However, if there are some notable aspects of the code that you discuss, including that in an appendix might be useful to make it easier for your readers to access.

As a general guide, if you are discussing short extracts of code then you are advised to include such code in the body of the report. If there is a longer extract that is relevant, then you might include it as shown in the following section.

Only include code in the appendix if that code is discussed and referred to in the body of the report.

Random Number Generator
The Bayes Durham Shuffle ensures that the pseudo random numbers used in the simulation are further shuffled, ensuring minimal correlation between subsequent random outputs.

```
// Some example code here…
```