The University of Western Australia

School of Electrical, Electronic and Computer Engineering

Final Report

Master of Professional Engineering

Candidate

Ray Barker, 21300048

Title

Interactive Tabletop Simulation System for Testing and Visualisation of Autonomous Vehicle Algorithms

Supervisors

Dr Tim French

Dr Mubashar Ghullam Hassan

Word count: 8410

Contents

List of Figures

List of Tables

1. Abstract

Autonomous vehicles are expected to increase in prevalence over the coming years due to significant advances in autonomous vehicle technology. Software simulation tools exist to simulate and model autonomous vehicle behaviour in ideal conditions, yet tools to simulate their behaviour in realistic real-world environments are limited.

The objective of this project was to address this gap in available tools by developing a tabletop autonomous vehicle simulation system, which allows for the testing and demonstration of autonomous vehicle algorithms and behaviours under real-world conditions. Design requirements were to develop a reproducible, low cost and easily transportable system which can simulate a range of environments. The scope of this project includes the development of the overall system architecture, environment model and closed loop feedback controller for the cars.

By using an iterative development process, a functional system has been designed which includes four ZenWheels microcars, a Raspberry Pi and PiCam, a projector and a custom-built tabletop rig. Testing of the simulator at the 2017 University of Western Australia Open Day showed that the system was engaging and effective at demonstrating the behaviour of an autonomous vehicle.

Further development enabled the system to simulate four cars simultaneously on a visually interesting figure of eight track. Achieving this involved the development of a closed loop control system to control the orientation of the cars and appropriate data structures to represent the environment and objects within the environment. Additionally, interfaces are provided for users to integrate their own agents and vehicle types. Consequently, this sample scenario serves as a platform on which many vehicle behaviours could be demonstrated, including traffic control strategies at intersections and the interaction between human and autonomous drivers.

To allow the system to be reproduced by other researchers, design drawings and all the system's code is provided on an online Git repository. Future development of the simulator is recommended, focussed on improving the system's flexibility and performance, to allow for the simulation of up to eight vehicles.

2. Introduction

This document is a final report summarising the design and build project to develop a tangible simulation and demonstration system for autonomous vehicle algorithms.

The literature review provides an overview of autonomous vehicle technology development and then discusses the details of previous similar projects. Following this, the design process section details how the major system components were selected.

The final design section provides an overview of the two major components of the design, which are the hardware and the software. On the hardware side, the final design is provided, including detailed design drawings. The software design is described by providing an explanation of the software architecture and interfaces. Finally, a discussion section identifies the applications of the developed system and provides future recommendations for improving the system.

3. Literature Review

This literature review commences by providing an overview of the current state of autonomous vehicle technology and the competitions which have fostered their development. Following this, an overview of current state of the art modelling and simulation tools is provided, proceeded by an overview of the issues in traffic system education. The TangiSense table system is analysed, and then a previous similar project, the Miniature Vehicle Testbed, is discussed.

3.1 Autonomous Vehicles

With the increasing numbers of vehicles on public roads, effective traffic simulation is an ever-present concern. Traffic congestion results in increased vehicle emissions and energy consumption, so it is important to have free-flowing traffic systems [1]. Cities need to manage their traffic issues and turn to simulation as a tool to assist designing better transport solutions.

The expected prevalence of autonomous vehicles in the coming years [2] further complicates matters. Autonomous vehicles have different behaviour to human drivers, and the complex interplay between these two types of road users must be considered. Whilst autonomous vehicle technology has progressed rapidly, further work is required in simulating their effects once fully integrated into real-world road systems.

Development Competitions

Considerable research has been done to develop autonomous vehicles over the past decade. Competitions such as the RoboCup serve as a platform to allow artificial intelligence and robotics research to be demonstrated in a competitive environment [3]. Research into areas such as agent architectures, real time recognition, planning and reasoning are promoted by the RoboCup competition [3]. Despite the ability of the RoboCup to further research into these specific areas, the competition does not serve as a platform to demonstrate autonomous transport algorithms, due to the limited end goal of developing robots to play soccer.

The DARPA Grand Challenge is a competition to develop autonomous vehicles. In 2004, the challenge was to develop vehicles capable of completing a 140 mile off-road course [4], and had tremendous involvement from researchers and hobbyists. In the following years, the difficulty of the DARPA challenges increased to ensure a continued challenge against the advances in technology. This competition has been an invaluable driver of autonomous vehicle technology over the last few years.

Competitions such as RoboCup and DARPA Grand Challenge which are focussed on furthering autonomous technologies do not fully consider the impact to the road systems into which they will integrate, as they tend to operate in a stand-alone test environment, with limited and well-defined goals.

The tremendous progress in the research realm, demonstrated at competitions such as RoboCup and DARPA Grand Challenge is now filtering into the real world. Based on the car "Stanley" which won the Grand Challenge, Google has been developing an autonomous vehicle for several years with promising results [5]. Additionally, there is significant interest from other companies including Samsung which has just been given approval to test their self-driving technology [6].

Modelling and Analysis Tools

With the increasing prevalence of autonomous vehicles on real world roads comes the need to understand and characterise their behaviour, particularly the ways in which they will behave in the real world where conditions are non-ideal.

Traditionally, modelling and analysis of transport systems has been done using software simulations on a macroscopic or microscopic level. Microscopic simulations model the behaviour and interactions of each individual road user, whilst macroscopic simulations consider traffic to be a fluid with flow, density and speed [7]. Macroscopic simulations do not provide the level of detail required to model the interactions and behaviours autonomous vehicle systems may have with human drivers.

Microscopic traffic simulation models such as Aimsun have been used to achieve very detailed modelling of intersections and is useful in the design and modifications of road systems. In [8], Aimsun was used to model a complex roundabout by placing roads and setting traffic rules, and whilst challenges were experienced with the gap-acceptance model, the Aimsun software was effective overall to develop improvements for this intersection. One quirk of Aimsun is that the simulations are stochastic in nature, therefore multiple simulation iterations are required to ensure that the simulation output is statistically significant [9]. The flexibility and detail of Aimsun is a strength, however, this adds complexity making the software unsuitable for untrained users. Additionally, Aimsun does not allow for flexible simulation of agent based algorithms.

Software simulation is extremely powerful, yet limited in its ability to communicate results in a meaningful way to the public. The issue with typical traffic simulation tools is that the interpretation of the results tends to be difficult and requires a subject matter expert to derive meaning. Additionally, they are not sufficiently flexible to allow for analysis of the behaviour of custom autonomous vehicle agent software. Furthermore, the simulated software environments are idealised which eliminates the real-world complications that an autonomous vehicle agent would otherwise need to consider.

Simulations and Models in Education

In the education domain, traffic simulation can be an effective tool. The paper [10] demonstrates that simulation enabled experiential learning in the field of transportation engineering, and provides qualitative evidence that students were engaged in the educational content, particularly the simulation elements. The paper [11] recognises the significant barrier to entry posed by the complexity of simulation software commonly available. To resolve this, an intuitive web-based traffic simulation tool was developed to allow students to understand and implement different control strategies and layouts [11]. The ease of use is the key merit of this simulation tool, however, there is no ability to model, demonstrate or simulate the behaviour of autonomous vehicles.

Continuing in education, basic tabletop models of traffic have been used successfully to improve children's understanding of road safety rules [12]. This indicates that a tabletop model can be an intuitive and effective means to demonstrate traffic concepts, even with a simple static model consisting of wooden toy cars, road signs and roads. In this system, there is no method to integrate dynamic simulation into this model, let alone simulation of autonomous vehicle algorithms.

TangiSense Table System

Previous research has recognised the need for more intuitive and tangible simulation systems. A similar project is an interactive tabletop framework using the *TangiSense* table, described in [13]. This project aimed to develop an interactive tabletop simulation system suitable for use by "several people (decision-makers) in a simultaneous and collaborative way" [13], as current computer based simulation tools are not sufficiently intuitive.

Published two years later, [14] demonstrates an updated version of the *TangiSense* system which contains an integrated HD screen, rather than a projector. This paper applies the *TangiSense* system to a traffic simulation scenario with promising results.

The *TangiSense* table measures 1 m x 1 m and consists of an array of Radio Frequency Identification (RFID) antennas. Tangible objects can be fitted with RFID tags which are then detected by the system's antennas when placed on the table. Virtual objects, represented only in software, are displayed on the table using a projector or screen.

Architecture for this system consists of four layers [13]:

1. The system hardware (ie. the *TangiSense* table)
2. Middleware which detects the tangible objects using RFID
3. Multi-agent system (MAS) layer which handles the environment and agent reasoning
4. Human-Computer Interaction (HCI) layer used for communicating with table users

This multi-layer structure appears to be an effective means of separating the different system elements. This system does not have closed feedback loop control of object positions. Instead, the tangible objects are positioned on the table by the users and are not designed to move themselves.

The use of RFID for tracking objects has the distinct advantage that each object can be uniquely identified by the tag, however, the position accuracy provided is limited by the layout and density of the RFID antennas. This makes RFID technology inappropriate for providing closed-loop feedback for this project. Notwithstanding, concepts from the TangiSense table system, particularly the structure of the multi-agent system layer, have informed the design of this research.

Miniature Vehicle Testbed (MVT) Project

An honours research thesis in 2016 attempted to develop a tangible tabletop autonomous vehicle simulator called the Miniature Vehicle Testbed (MVT), however, the project was not completed [15]. The objective was to develop a small-scale system which could be used to model real-life traffic with reasonable accuracy. A system was developed using a machine vision camera, *ZenWheels* cars and a test-rig constructed out of wood. The design of the MVT consists of *ZenWheels* cars controlled via Bluetooth, with position feedback provided from a high-resolution camera mounted above the testbed. The research found that although the project was time-consuming, there was no evidence to believe that the project was technically unfeasible [15].

Hardware Design

The testbed developed had several key design features which will be considered during this research. It was recognised that a reflective surface could create significant noise in the image captured, so a flat finish paint was used to minimise glare on the surface [15]. A fence around the driving surface was added for aesthetics and to prevent the cars from falling off the edge, and the system was designed so that large components could be easily disassembled [15] for portability.

The project used tangible roads constructed from heavy duty card, with lines drawn on manually using pens [15]. A key issue identified with tangible roads is that the testbed is limited to a static configuration of roads. Using a projection system would allow for a multitude of road configurations to be used. One benefit of tangible roads over projection is the ability to build more complex structures such as bridges [15] rather than being limited to the 2D plane.

Two key considerations for the camera mounting frame were to ensure that the structure was adequately sturdy to prevent swaying of the camera, and to be adaptable to different cameras or camera layouts

[15]. The final design used PVC pipes for the camera mounting frame with sturdy camera mount clamps to allow for flexible positioning of the cameras. The required height of the camera was determined by using the camera field of view and simple trigonometry, however it was found that the actual field of view was significantly narrower than specified. This required raising the camera, which caused a reduction in the size of each car in the image to the point where the cars were not able to be detected in the image processing algorithm [15]. This project proposes several solutions to this issue which will be investigated including using a camera with a wider field of view or higher resolution, reducing the physical size of the driving surface or using an array of cameras to increase available resolution.

Software Design

On the software side, the primary challenge faced in the previous project was development of an effective image processing algorithm [15]. The accuracy of the corner detection and K-means clustering method used was not sufficiently accurate nor consistent. Additionally, the algorithm was to run on a Linux computer. To achieve this, the algorithms were developed using MATLAB, and then compiled into C code using MATLAB Coder, however, this approach was not effective as the C code would not run correctly [15]. With large amounts of computer vision algorithms and methods available, it is believed that more suitable algorithms, written directly in C or C++, could be applied to address these issues.

Further issues experienced included being unable to control the focus of the camera, unpredictable focus of the camera, and the output images from the camera feed not updating in real time [15]. It is hypothesised that using a more robust architecture and a more suitable camera will address these issues.

Closing Summary

In summary, the literature provided has demonstrated the need for more interactive simulation tools given the increase in traffic congestion and impending rise in autonomous vehicles used on public roads. Traditional software simulation tools require excessive technical expertise to develop meaningful results, interpretation of the results is often not intuitive and the ability to test autonomous vehicle algorithms in non-ideal conditions is non-existent. Prior research has shown the need for more interactive simulation tools, and tabletop simulation has been proven to be an effective tool for demonstrating traffic concepts.

This research proposes to develop a tabletop simulation system suitable for the demonstration and testing of autonomous vehicle algorithms which provides intuitive results to address the gap in current simulation tools. A previous attempt has been made to develop a similar system, however, the attempt was unsuccessful due to a series of unforeseen issues and time constraints.

4. Design Process

4.1 Purpose of the Design

The literature review identified a gap in the availability of simulation systems for demonstrating autonomous vehicle algorithms to the public. Specifically, software tools such as Aimsun require too much background knowledge to interpret results, and previous attempts at building hardware systems such as the Miniature Vehicle Testbed were unsuccessful.

The purpose of the design is to address this gap by developing an engaging hardware-based simulation system which allows autonomous vehicle algorithms to be easily interpreted and understood by the public.

Achieving this will require developing a system which allows researchers in the field of traffic flow analysis and autonomous vehicles to demonstrate their algorithms. The system should be suitable for display in public-access exhibits, such as the 2017 University of Western Australia Open Day, to interest the general public in the research activities of the university.

The scope of this report covers the conceptualisation, design and building of the simulator from a high-level system perspective. Technical details regarding the operation of the computer vision system are discussed in Aaron Hurst's final report [16].

4.2 Requirements and Constraints

Requirements of the system have been developed in consultation with the project supervisors, Dr Tim French and Dr. Ghulam Mubashar Hassan.

The system must be able to meet the following requirements:

Table 1: Summary of Requirements

ID	Summary	Reasoning
R1	System should operate in a crowded hall or room under typical lighting conditions and Wi-Fi/Bluetooth frequency congestion.	The system will be used at events such as University of Western Australia Open Day, so must be able to operate under these anticipated conditions.
R2	People should be able to surround the testbed without interfering with system performance.	The system must be able to operate with people observing the model when a demonstration is being performed.
R3	System should allow up to eight cars to be simulated simultaneously.	To allow the simulation of complex multi-agent protocols.

ID	Summary	Reasoning
R4	System should be modular for easy disassembly and transport.	To allow the system to be easily taken from a storage or development space to the event where it will be used.
R6	System should be flexible to allow different agents and vehicle types to be simulated.	To support simulation of a range of different algorithms and behaviours.
R7	Cost of the system should be minimised.	To reduce the investment required to develop this system.
R8	System should be visually interesting.	To ensure that the model is interesting for the public to observe.
R9	System should be reproducible.	To allow other researchers to build their own version of the system.

4.3 Methodology

To explain the key decisions made in the design phase, the considerations and justifications for the selection of each system component is described in this section.

Model Cars

The selection of physical model cars to be used was one of the first decisions which needed to be made, as the size of these cars would influence the design of the rest of the system.

Two cars were considered, the "Zenwheels Micro Car" and the "Anki OVERDRIVE".

The Zenwheels Micro Car is designed to be paired via Bluetooth to a smartphone application. The application allows the user to control the throttle and steering of the car, as well as some ancillary features such as lights and a horn. The car can be driven along any arbitrary path on a smooth surface such as a desk or concrete floor.

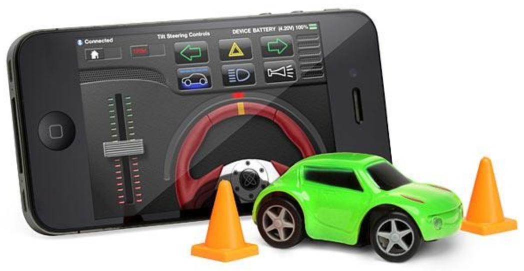


Figure 1: Green Zenwheels Micro Car and iPhone application [17]

The Anki OVERDRIVE is a digital evolution of the traditional "slot cars" made by Scalextric and Carrera. Thin plastic track segments consisting of four lanes are magnetically connected to build a race track. A smartphone application allows users to race cars against computer drivers and other human drivers. Users are given control of the car speed and the car lane, and the system keeps the cars on the track within the chosen lane.

Figure 2: Anki OVERDRIVE track with two cars [18]

The Zenwheels Micro Car was selected over the Anki OVERDRIVE as the former can be driven along any arbitrary path, whereas the Anki cars are limited to driving in predefined lanes on the track segments. Other factors include the lower cost of Zenwheels cars and availability in over eight different colours which would assist the computer vision in identifying each car. Finally, a GitHub repository which has reverse engineered the car control commands was already available [19] for the ZenWheels cars. This simplified the process of controlling the cars from custom software rather than through the closed-source smartphone application.

Overall Design

The overall design of the model consists of a frame which connects all the major parts. By interconnecting the parts, complex alignment issues of the camera and projector can be eliminated, thereby reducing the setup time of the system.

For transportability, the general size of the model was designed to be approximately 1m x 1m. This was later refined to a 1.2m x 0.9m size. This size has an aspect ratio of 4:3 which maximises the utilisation of the camera sensor area. Materials are also commonly available in this size at local hardware stores.

Frame Material

Pine wood and aluminium were considered appropriate materials for building the frame. Both materials are readily available from local hardware stores and have sufficient strength for the design. PVC tubing was also considered, but deemed inappropriate as it is unattractive and was unlikely to support the weight of the projector.

A comparison matrix of the three materials is provided in Table 2.

Table 2: Frame material options comparison matrix

	Pine wood	**Aluminium**	**PVC**
Weight	Relatively light weight	Extremely light weight	Extremely light weight (Hollow)
Aesthetics	Acceptable (especially if painted)	Good (preferable to wood)	Poor
Ease to manufacture	Very simple, using screws and commonly available tools	Cutting and joining more arduous and complicated than wood	Simple to cut and assemble, but design is significantly limited by available joints and fittings
Strength	Sufficient	Sufficient	Insufficient

To maximise the satisfaction of the reproducibility requirement and minimise the time required to build the hardware, pine wood was selected as the material for the design. Additionally, to minimise the complexity of the design and manufacture, it was decided to use wood throughout the design wherever possible, rather than using a mixture of different materials.

Driving Surface

The driving surface is the component on which the cars will drive. The material properties of this surface such as reflectivity and hue influence the ability of the computer vision to identify the cars. During the design phase the team was unsure as to which material would be the most appropriate. Thus, the decision was made to allow the driving surface to be interchangeable. Three materials were selected for trial, each selected due to a particular desirable property, as shown in Table 3.

Table 3: Summary of Selected Trial Materials for the Driving Surface

Material	**Desirable Properties**	**Undesirable Properties**	**Cost**
Plastic cardboard (Corflute)	Extremely light weight	Hollow interior causing variations in white colour	$16.40
White melamine coated MDF sheet	Excellent colour reproduction of projected image	Reflective	$10.82
MDF sheet with ceiling white paint	Non-reflective	Quality of the surface finish influenced by painting skill	$19.27

It was also identified that if the driving surface is not flat, there would be several consequences. Firstly, the projected image would appear distorted, and secondly any slant would add an additional source of error to the vehicle direction when driving. To prevent this, the driving surfaces would be likely to require additional support to prevent sagging.

A fence around the driving surface was added to prevent damage to the cars caused by driving off the edge of the table. The cars are relatively expensive and could break if dropped from table height, so it was important to take reasonable steps to prevent damage.

Overhead Gantry Height

The camera is mounted to the overhead gantry, so the height of the gantry is dictated by the field of view of the camera. Before a camera was selected, the height of the gantry needed to be adjustable to support experimentation with different cameras. This was achieved by omitting two screws and using spring-loaded clamps as a temporary fixing, as shown in Figure 3.

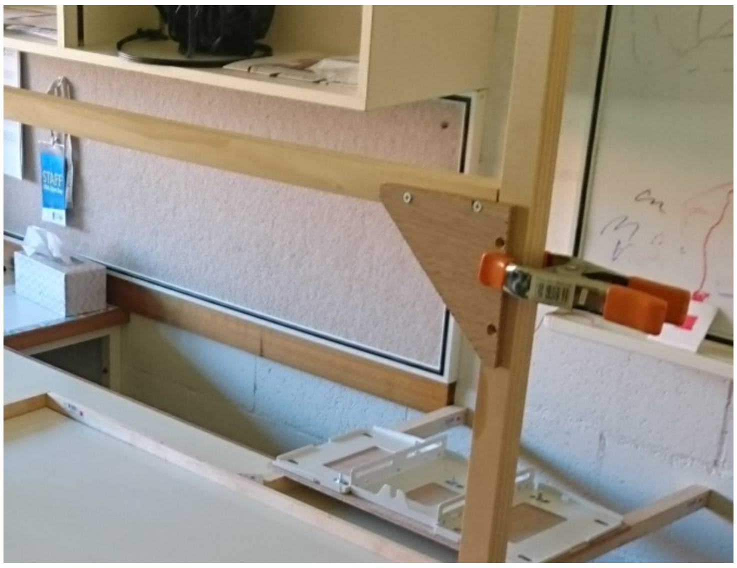


Figure 3: Spring loaded clamp used for adjusting camera height during experimentation

Once the Raspberry Pi Camera V2 was selected, a permanent height for the camera was determined. The selected camera specifications state a viewing angle of 62.2 degrees horizontally and 48.8 degrees vertically [20]. As the aspect ratio of the sensor is consistent with that of the lens and the driving surface, it was only necessary to calculate the height from either the horizontal or vertical viewing angle - either would give the same result.

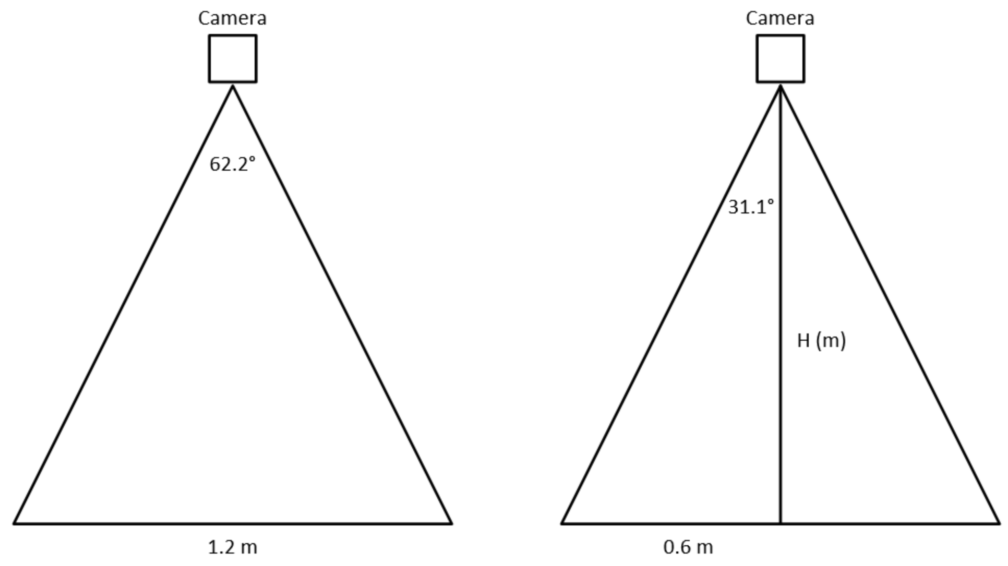


Figure 4: Camera height calculation using the horizontal viewing angle

The required height calculated using trigonometry is shown below.

$$H = \frac{0.6}{\tan(31.1)} = 0.99\ m$$

To fully cover the 1.2 m x 0.9 m driving surface, a minimum camera height of approximately 1 m was required. The camera was mounted at 1.2 m above the driving surface to add a small safety margin, thereby making it easier to correctly position and align the camera.

Raspberry Pi

Raspberry Pi 3 was selected as the computer on which the computer vision and car controller software would run. This is because it is a readily available, low cost, all-in-one integrated solution with built in Bluetooth to communicate with the cars. It is also small and light, and compatible with a wide range of cameras including USB webcams, Pi Cam and Pixy Cam.

The key limitation introduced by selecting the Raspberry Pi is the amount of processing power. With only an ARM Cortex-A53 processor running at 1.2 GHz, efficient code would be required to support the system.

As it was unknown whether a single Raspberry Pi would have sufficient processing power, the architecture was designed to support the option of running the system across multiple Raspberry Pi computers. Specifically, the computer vision could run on a dedicated Raspberry Pi, and the car controller could run on a separate Raspberry Pi, with communication between the programs via a TCP socket over a local network.

Short-throw Projector

The purpose of the projector is to allow the environment on the driving surface to easily change so that different scenarios can be simulated on the testbed. Whilst not a strictly necessary item, it serves to significantly increase the level of engagement and visual interest.

A short-throw projector was a clear choice for this application. A standard projector would be unsuitable as it would need to be mounted far above the camera which would then obscure the image, and would significantly increase the size of the model. Another option considered was to use a flat panel LCD screen as the driving surface itself.

A projector was chosen over a screen as the projector option is a more flexible option. For example, in order to reduce the cost of the design, the projector could simply be omitted from the design entirely, and an environment could be painted directly on the driving surface. This option would not be available if a screen was used as the driving surface, as the screen and screen mount would form an integral part of the design.

The selected projector was the Epson EB580, because it is a good value projector with ultra-short throw capability designed for use in education environments. Other short throw projectors could be used, but this would require redesigning the projector mount plate.

4.4 Design Tools

Autodesk Fusion 360

To allow the final design to be reproduced, a detailed CAD model of the system has been built in Autodesk Fusion 360. This tool was chosen as it has a relatively easy learning curve, in-built version control, and is free for educational use.

Github

Version control software Github has been used during the software development. The Github repository for this project is available at [21].

5. Final Design

5.1 Hardware Design

Summary

The hardware design consists of a two-part design – the base frame and the overhead gantry. Each of these parts can be transported separately.

The following render shows the overall design of the simulation system.

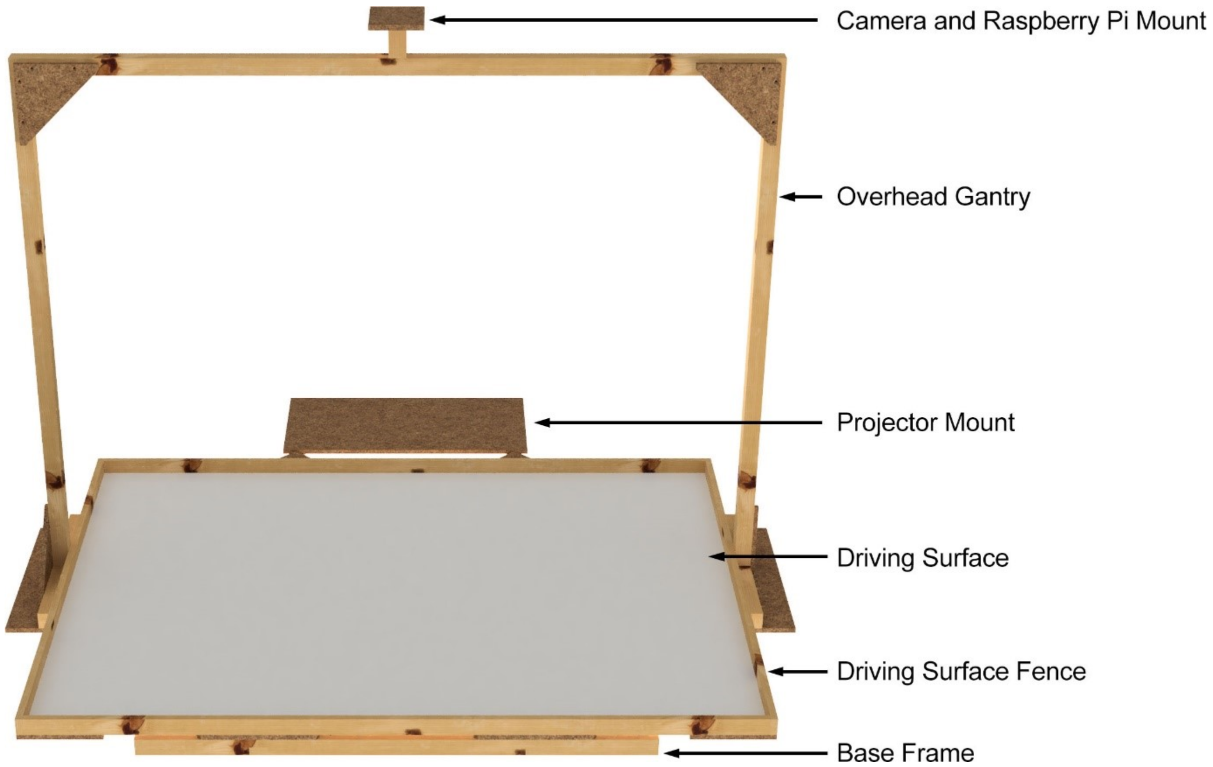


Figure 5: Rendered 3D model of final hardware design

The frame is built entirely from pine wood sections, with plywood plates used to join frame sections for additional strength. The driving surface is 3mm melamine coated MDF and measures 1.2 m x 0.9m. This was found to be an appropriate surface for both the computer vision to detect the cars and accurate colour reproduction of the projected image.

Design Drawings

The detailed design drawings of the hardware are available in the following pages. They are also available on the project GitHub [21].

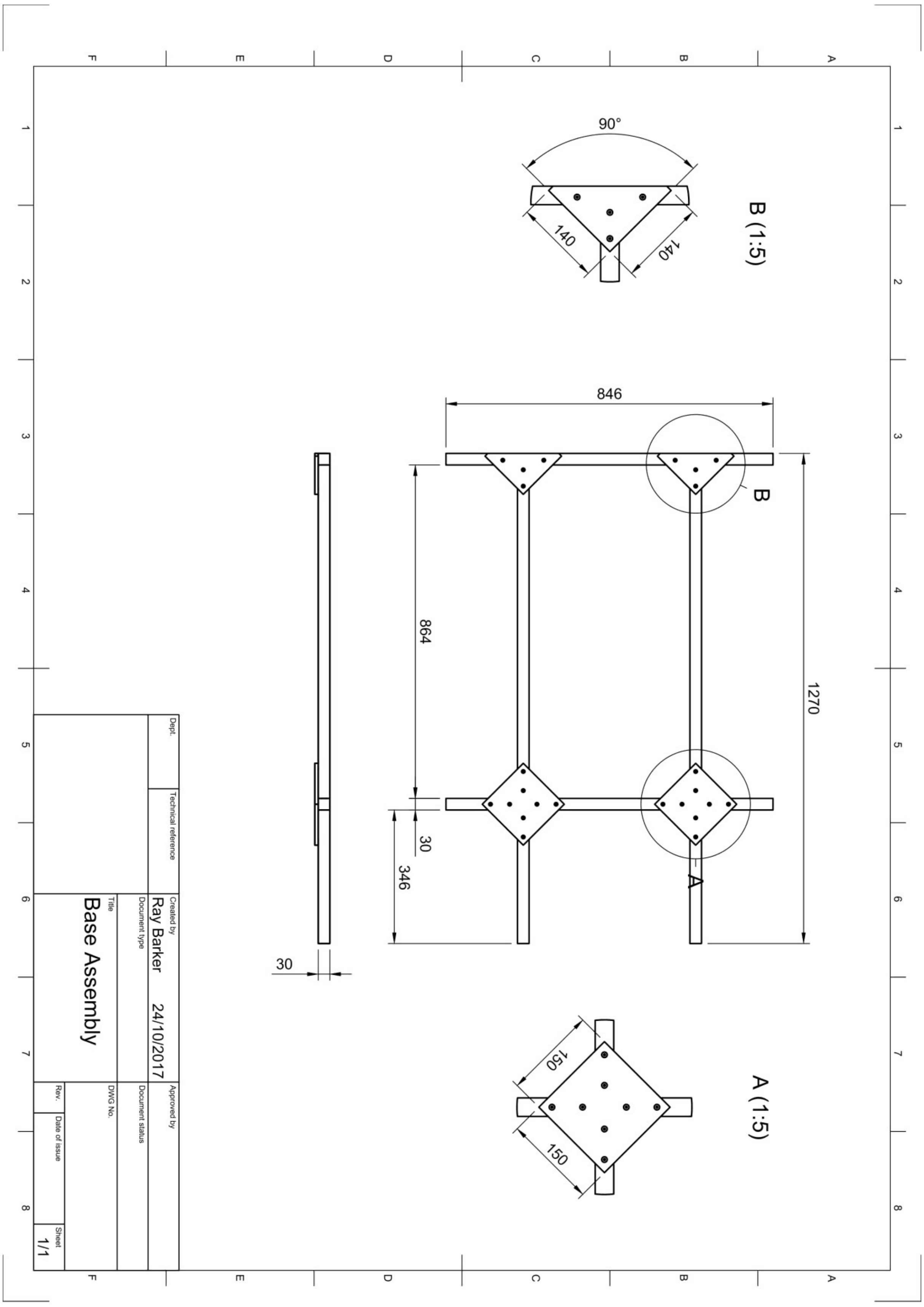


Figure 6: Design drawing of the base assembly

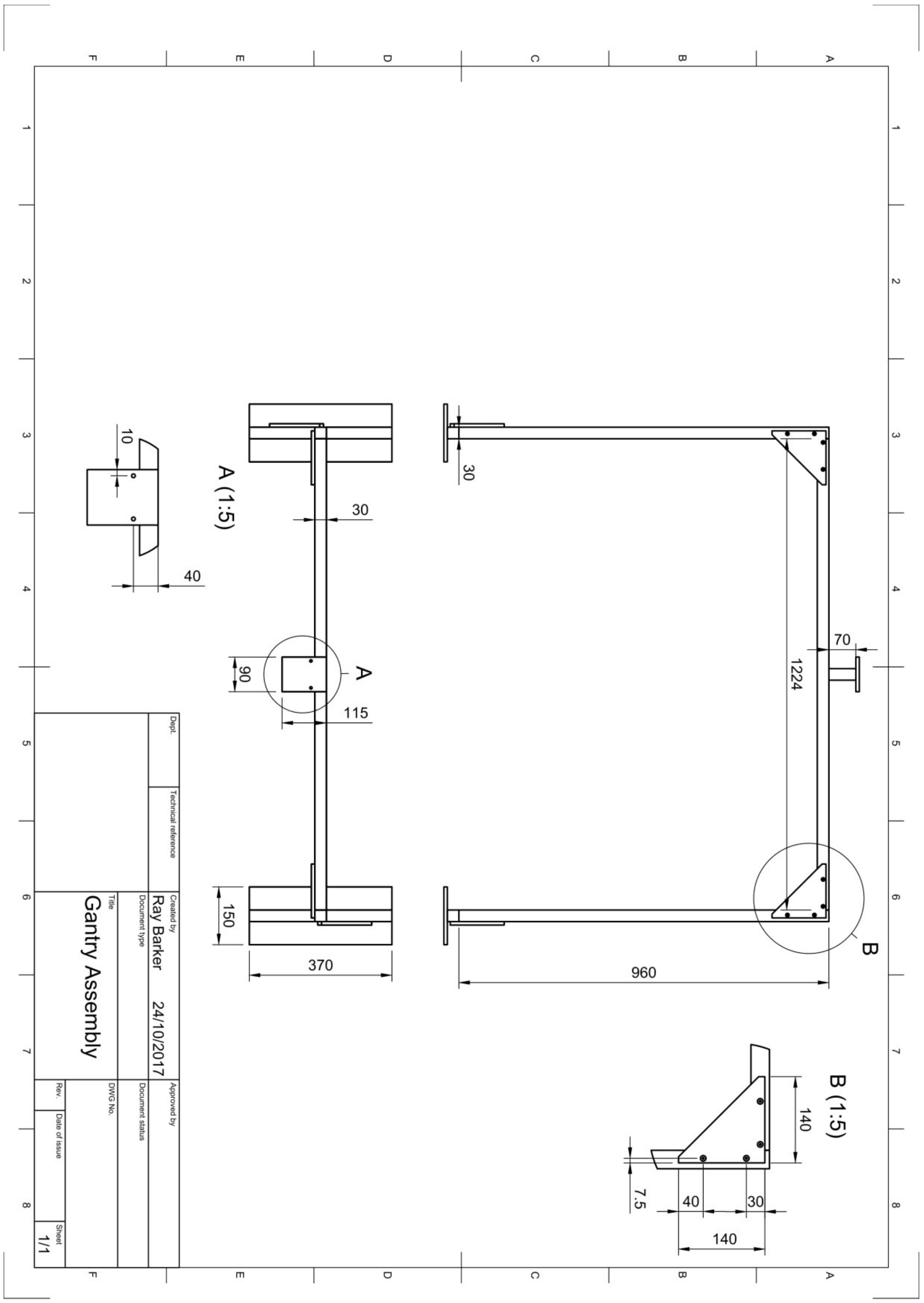


Figure 7: Design drawing of the overhead gantry assembly

5.2 Software Design

Software Architecture

The software for this system consists of two major parts. The computer vision and the car controller software.

The car controller software, written in Python, is the main server for communicating with the cars, accepting computer vision data, controlling car decisions and deciding their actions. The agent and vehicle models are both implemented within the car controller software.

The computer vision software was written in C++ and leverages the OpenCV library. Observations of the cars are sent to the car controller software over a TCP socket frame by frame.

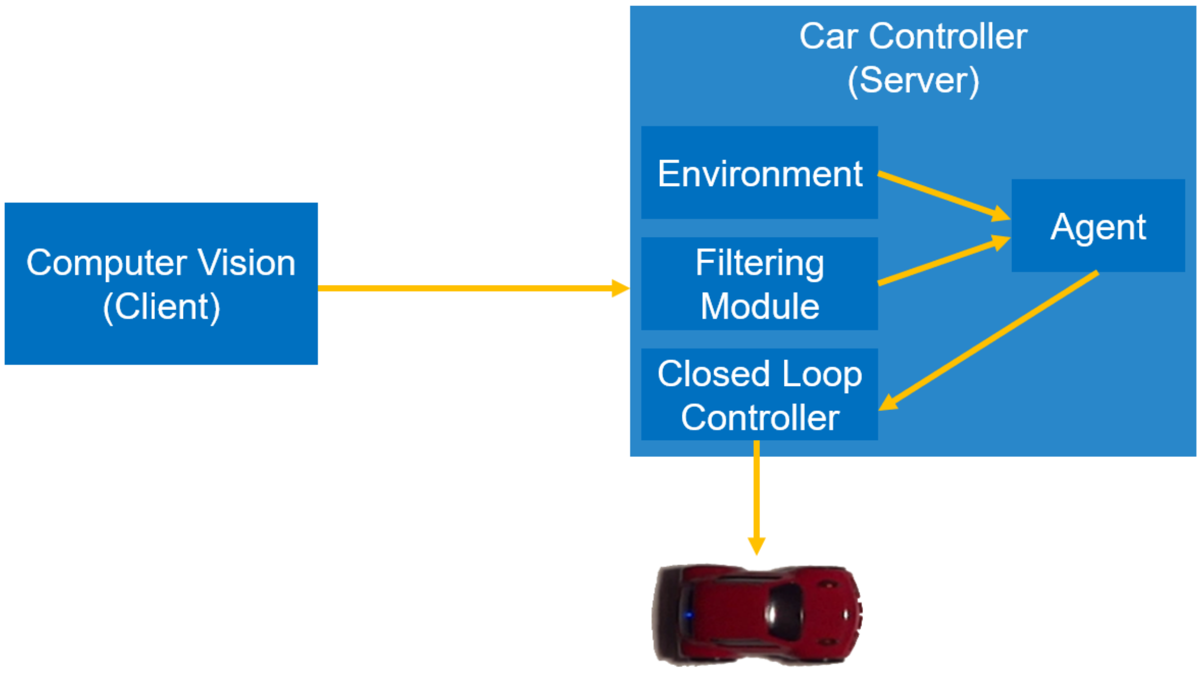


Figure 8: Software architecture overview

Car Controller Software

Environment

The environment model is defined using a Python NetworkX graph. Nodes are used to define driveable points on roads, and edges are used to define the connections between drivable points. A series of nodes defines a road. Each node has the following fields:

Table 4: Parameters required to define graph nodes

Field Name	**Description**
node_id	Unique identifier for the node

Field Name	Description
node_type	Used to indicate any interesting features of the node. Currently allowable values include: ROAD – Standard road point STOP_SIGN – Node has a stop sign INTERSECTION_START – Node is an entrance to an intersection INTERSECTION_END – Node is an exit to an intersection
x	The x coordinate of the node in mm.
y	The y coordinate of the node in mm.

The environment graph is defined in the *cars-controller/config/map_data.csv* file, and is loaded into the software at runtime. The visual representation of the map is stored as *cars-controller/config/map_image.jpg*. The only purpose of this image is to provide an increased level of immersion for observers of the model. Editing these two files allows for custom scenarios to be implemented on the system.

The coordinate system for the system, shown in Figure 9, is defined such that the front left corner is (0,0). Distances are measured in millimetres and angles are measured in degrees. The positive x axis is from the origin and lies on the bottom edge of the model. The positive y axis starts from the origin and lies along the left edge of the model. Angles are measured as a bearing parallel to the positive y axis.

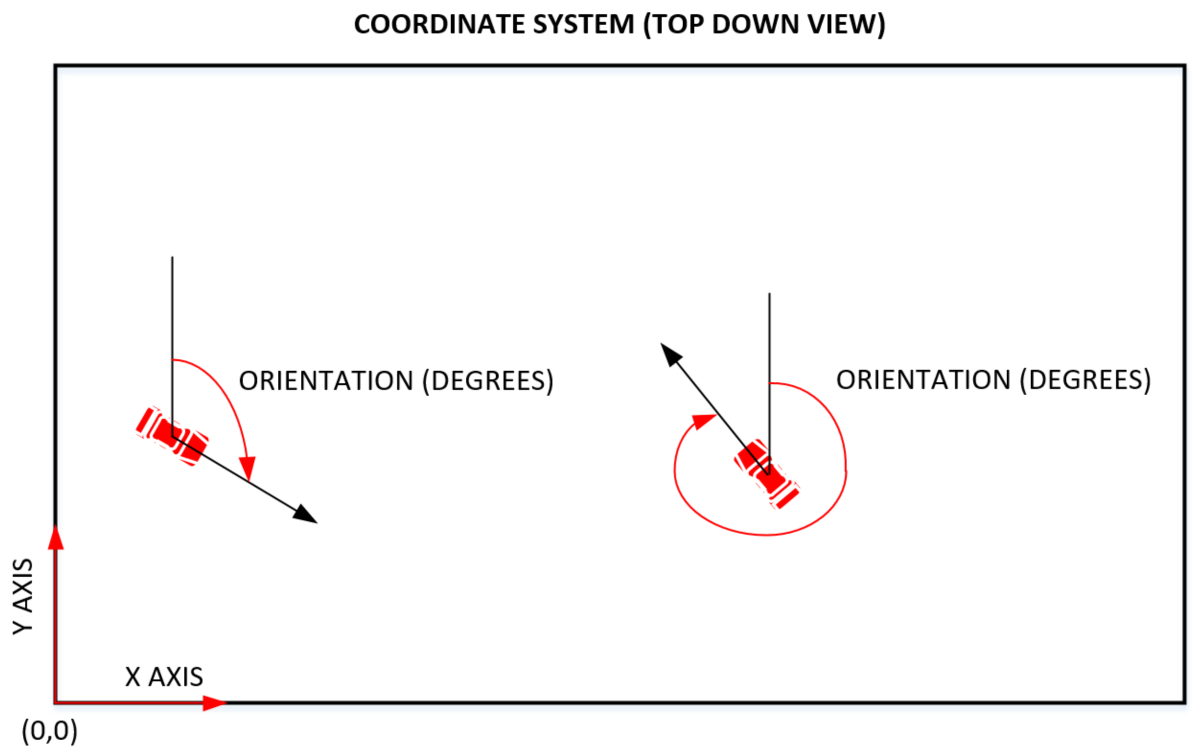


Figure 9: Coordinate system for the model

Filtering Module

The software was also designed with support to simulate different types of vehicles. For example, the allowable speed and acceleration of a race car would be significantly higher than that of a truck. The allowable output of the vehicles is filtered after the agent controller to simulate different vehicle types.

Vehicles are defined in the *cars-controller/vehicles/* folder. The file car.py is an example from which other vehicle types could be defined. The available configuration parameters are:

Table 5: Vehicle filtering parameters

Variable	**Description**
max_speed	The maximum speed which the car can drive, defined on a scale from 0 to 63.
max_acceleration	The maximum allowable increase in speed value per iteration, defined on a scale from 0 to 63.
max_deceleration	The maximum allowable decrease in speed value per iteration, defined on a scale from 0 to 63.
max_turn	The maximum turn permitted, defined on a scale from 0 to 63.
max_turn_change	The maximum permitted change in turn value per iteration, defined on a scale from 0 to 63.

Agent Module

To allow different agents to be simulated on the system to be simulated, an interface to allow agents to be implemented was defined. The *cars-controller/agents/human.py* script contains an example agent, from which others could be developed.

On each loop, the *update_percepts()* function is called by the core thread which updates the observed state of the agent. The *decide_actions()* function is then called. In this function, the logic of the agent should be programmed to use the available percepts and internal state to determine a list of actions which it wishes to perform. The permissible actions which agents may take are given in Table 6.

Table 6: Actions available to agents

Action	**Description**
["STOP",0,0]	Stop the car
["FORWARD",SPEED(0-100%),0]	Drive forward at desired % speed
["REVERSE",SPEED(0-100%),0]	Drive reverse at desired % speed
["LEFT",AMOUNT(0-100%),0]	Turn left at desired % amount
["RIGHT",AMOUNT(0-100%),0]	Turn right at desired % amount

Action	Description
["TARGET",X Coordinate,Y Coordinate]	Target a particular coordinate (x,y)

The list of actions is returned and then sent to the cars via Bluetooth by the core thread at the next available opportunity.

Closed Loop Controller

Closed loop control of the cars is essential as the cars do not drive precisely along the instructed path. A control loop which controls the orientation of the car using computer vision has been developed, and is shown in Figure 10. Essentially, the observed orientation is used to control the steering to make the cars drive in a desired orientation.

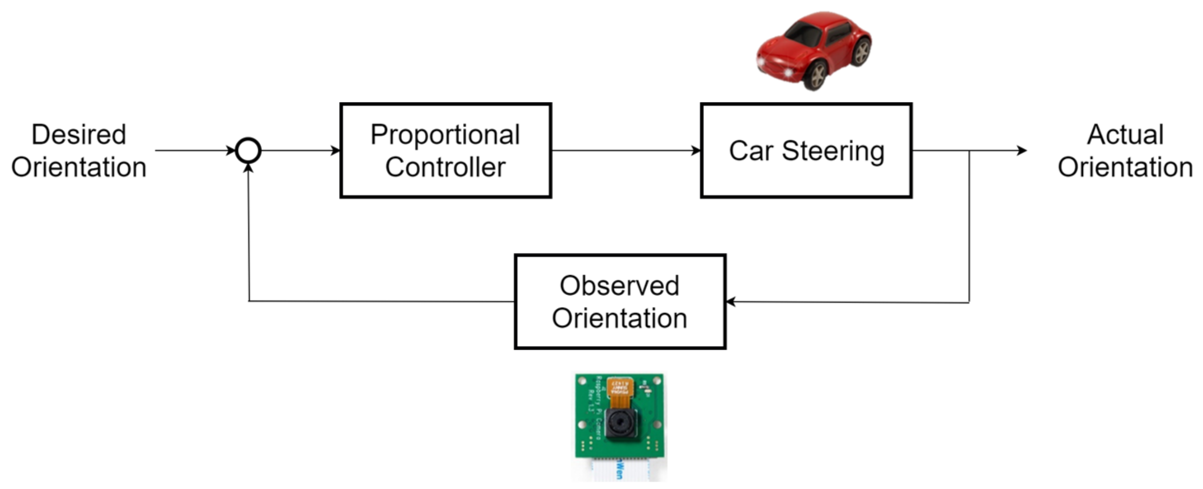


Figure 10: Control system diagram

Proportional control was found to be sufficient for this purpose. It was not necessary to use a PID controller, however, it is likely that this would have slightly reduced the average error and the overshoot of the cars' path tracking. The proportional controller was tuned experimentally, with a sensitivity value of 0.3 found to be most effective.

Driving the cars along arbitrary paths is achieved by targeting points along the track in series. The desired orientation is the orientation from the current position to the target point. The error for the controller is the difference between the current orientation and the required orientation.

Once the car is within 5 cm of the target point, the car will then target the next point along the track. This is so that the cars do not have to hit the exact coordinates of the target point, leading to unwanted circling around the target point.

6. Test Results

6.1 Summary

The success of the final design was evaluated by considering the ability of the system to satisfy the requirements outlined in Table 1: Summary of Requirements.

The following table details how each of the requirements was assessed and summarises the results of the assessment.

Table 7: Summary of requirements, test method and test results

ID	Requirement	Test Method	Test Result
R1	System should operate in a crowded hall or room under typical lighting conditions and Wi-Fi/Bluetooth frequency congestion.	Test the system at the 2017 University of Western Australia Open Day.	Pass
R2	People should be able to surround the testbed without interfering with system performance.	Test the system at the 2017 University of Western Australia Open Day.	Pass
R3	System should allow up to eight cars to be simulated simultaneously.	Test and count the number of cars which can be simulated using test scenario 3.	Conditional
R4	System should be modular for easy disassembly and transport.	Analyse the design to determine its modularity.	Pass
R5	System should be flexible to allow different agents and vehicle types to be simulated.	Demonstrate capability by describing the software architecture.	Pass
R6	Cost of the system should be minimised.	Analyse of the bill of materials.	Pass
R7	System should be visually interesting.	View the design's use of colour and shape.	Pass
R8	System should be reproducible.	Analyse of the state of system documentation (design drawings and code).	Pass

6.2 Test Scenarios

Two test scenarios have been used to assess the design.

Test Scenario 1 – Oval Track

This scenario consists of a basic oval track projected onto the driving surface. There is a black road on which the cars should drive in a loop, and a surrounding white background on which the cars should not drive. An image of the track is shown below.

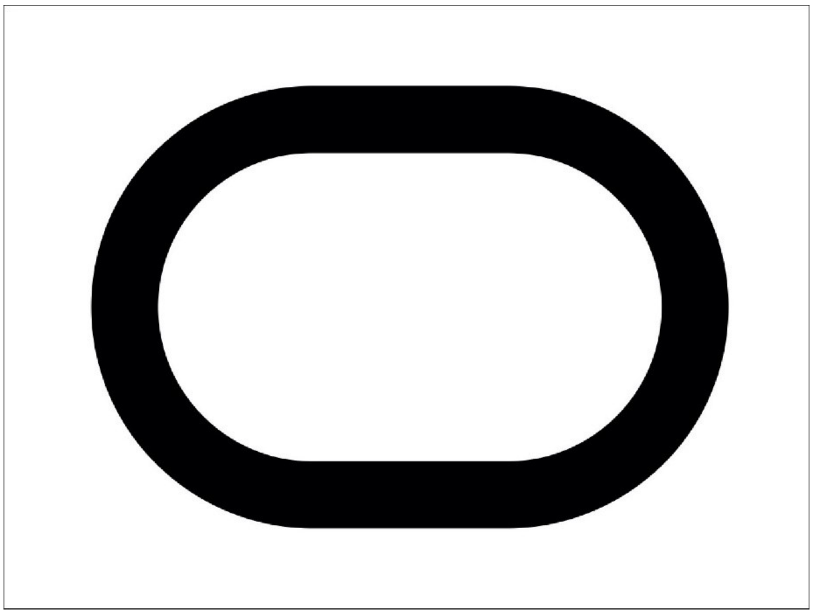


Figure 11: Test Scenario 2 - Oval Track

The scenario presents a challenge for the computer vision as the high-intensity light from the projector will alter the brightness of the vehicles. Additionally, the cars will have a significant shadow, caused by using the short throw projector.

An appropriate data structure will be required to represent the oval shape of the track, and functionality which drives the cars around the track will need to be added to the car controller.

6.3 Test Scenario 2 – Figure of Eight

This scenario, shown in Figure 12, extends the oval track by changing the shape to a figure of eight loop around which the cars should drive. The white arrows on the track indicate the direction of travel. There is a central intersection of two one-way roads at which the behaviour of vehicles under intersection control strategies such as traffic lights or stop signs could be simulated.

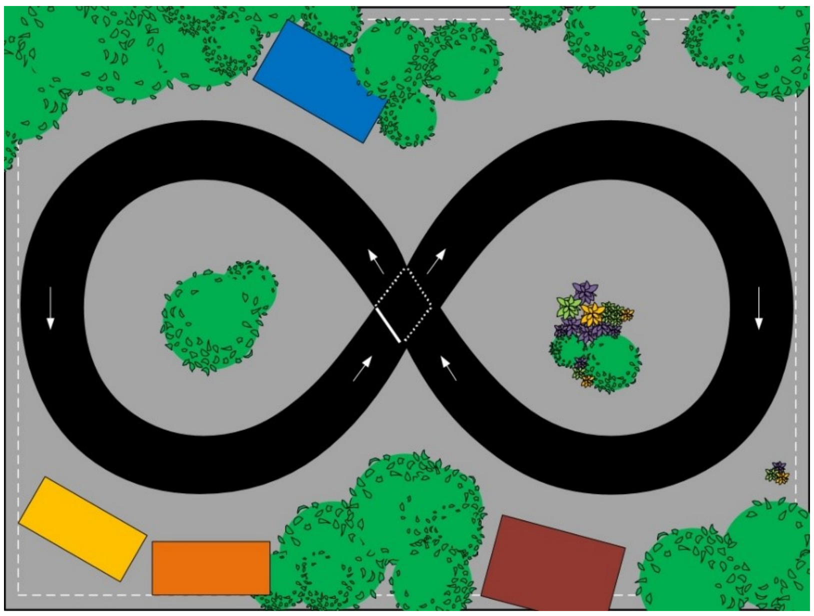


Figure 12: Test Scenario 3 - Figure of Eight

To make the scenario more interesting and visually appealing, additional detail such as trees, shrubs and buildings have been added to the background. This presents a further challenge for the computer vision system as areas of coloured light are added. The additional coloured light must not be spuriously detected as vehicles.

6.4 Detailed Results

This section describes in detail the tests which have been used to verify that the design meets the prescribed requirements.

Detailed Test Description and Results : R1 and R2

Requirements Tested
R1 - System should operate in a crowded hall or room under typical lighting conditions and Wi-Fi/Bluetooth frequency congestion.

R2 - People should be able to surround the testbed without interfering with system performance.

Test Description
Test the system at the 2017 University of Western Australia Open Day and evaluate the performance.

Test Result Analysis
The system was tested at the Open Day using test scenario 1. At this point in time, the system developed only supported a single car. It was able to control this car around the track, despite the presence of frequency congestion and people surrounding the model, so this requirement is considered met.

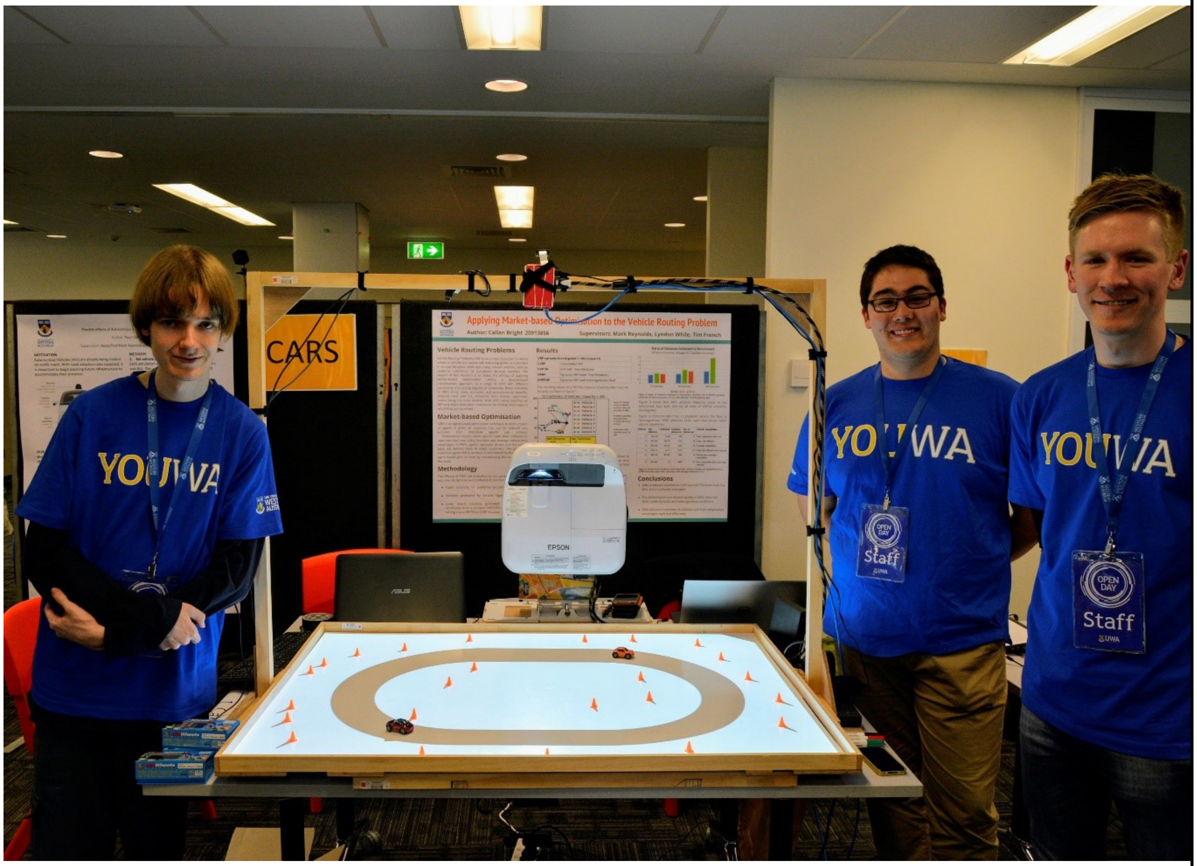
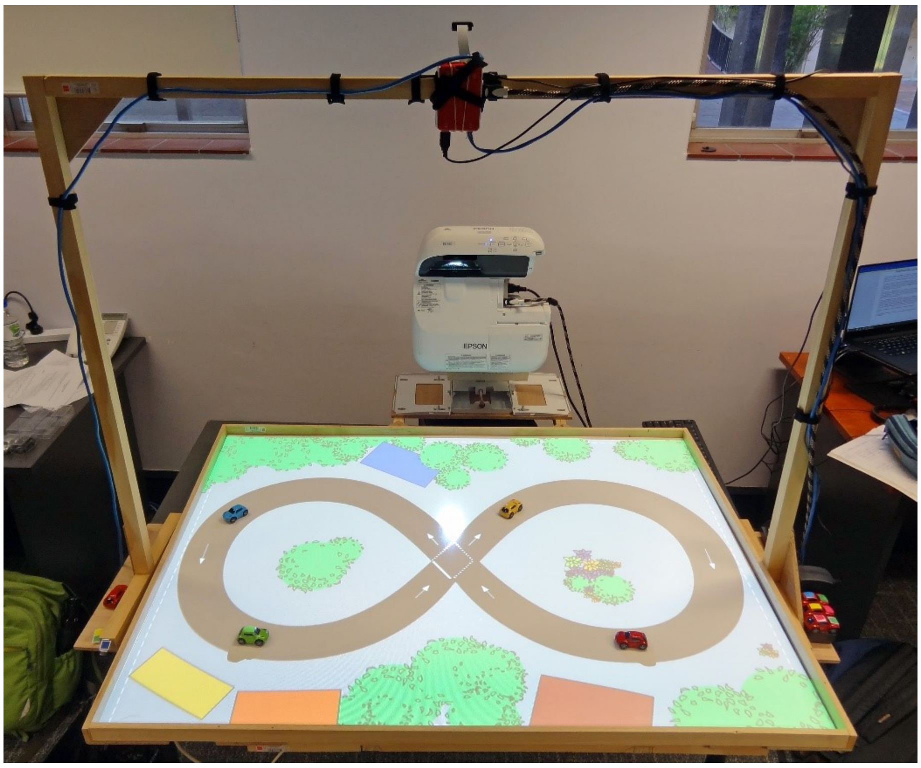

CARS
Applying Market-based Optimisation to the Vehicle Routing Problem
YOUWA
OPEN DAY
Staff

Figure 13: System under test at UWA Open Day

Detailed Test Description and Results : R3
Requirement Tested R3 - System should allow up to eight cars to be simulated simultaneously.
Test Description Test and count the number of cars which can be simulated using test scenario 2.
Test Result Analysis It was found that the system could reliably support four cars. Figure 14 shows four cars being simulated on the testbed.  *Figure 14: Four cars simulated on test scenario 2* It is hypothesised that the design of the system will support the use of the desired eight cars as there is no significant drop in performance when comparing one car to four. The following items will need to be addressed before the system can be tested with eight cars: 1. Purchase or paint coloured cars so that the computer vision can uniquely identify eight cars. 2. Implement an agent which can intelligently avoid and give way to other cars.

Detailed Test Description and Results : R4
Requirement Tested R4 - System should be modular for easy disassembly and transport.
Test Description Analyse the design to determine its modularity.
Test Result Analysis The design is modular as the overhead gantry is detachable from the base frame with only basic tools. The two separate parts can be transported by two people without the need for a trolley or other mechanical assistance, so this requirement is considered met.

Detailed Test Description and Results : R5
Requirement Tested R5 - System should be flexible to allow different agents and vehicle types to be simulated.
Test Description Demonstrate capability by describing the software architecture.
Test Result Analysis This requirement is met as it is straightforward to implement different agents and vehicle types as described in 5.2 Software Design.

Detailed Test Description and Results : R6
Requirement Tested R6 - Cost of the system should be minimised.
Test Description Analyse of the bill of materials
Test Result Analysis The bill of materials is provided in Appendix A – Bill of Materials. Given that the total system cost is only \$2,375.72, or \$475.72 without the projector, this requirement is considered met. There are no unnecessary expenses, and it is especially notable that the system can run on a Raspberry Pi with Pi Cam, which significantly reduces the cost of the system.

Detailed Test Description and Results : R7
Requirement Tested R7 - System should be visually interesting.
Test Description View the design's use of colour and shape.
Test Result Analysis The physical presence of the system makes it immediately visually interesting, especially in a public setting. The use of multi-coloured cars and also the visually interesting background projection lead to this requirement being considered met.

Detailed Test Description and Results : R8
Requirement Tested R8 - System should be reproducible.
Test Description Analyse the state of system documentation (design drawings and code).
Test Result Analysis The code for the system is documented reasonably well using in-line comments, and design drawings have been provided on the GitHub repository [21].

7. Discussion

7.1 Implications of the Work

This project has successfully addressed the gap in available simulation systems by developing a reproducible, low cost system that can be used to demonstrate autonomous vehicle algorithms. It has built on the lessons learnt in the MVT project to design, build and test a functional system.

It is expected that this model will be used at The University to spark public interest in the research activity in autonomous vehicles, real-time optimisation, scheduling and logistics. This may encourage potential students to pursue these topics.

As the design is reproducible, it is possible that other researchers will build their own version of the model to use at their events.

7.2 Areas for Improvement

There are several areas for improvement or continuing work which are described in this section.

Implement PID Control

Currently the cars are controlled using a proportional controller. Use of a proportional-integral-derivative (PID) controller is likely to improve the path-tracking ability of the cars by eliminating the slight overshoot which occurs.

Consolidate Software

The designed system requires the Python central controller program to interface with the C++ computer vision program via a TCP socket. A suggestion for improvement would be to integrate these two pieces of software into a single program. Doing this would eliminate the approximately ten millisecond delay caused by communication whilst increasing the system robustness and reducing complexity.

User Interface

The interface for the system is currently a command-line interface, accessed over SSH. A computer vision based user interface could be implemented, making the system easier to use. For example, hovering a hand over particular parts of the driving surface could trigger actions such as pausing, restarting or changing the simulation.

Percept Filtering

At the current time, all agents receive the same information about their surroundings. A desirable feature would be to implement functionality which limits the percepts depending on the agent type. For example, a self-driving car agent should be able to see all objects within the range of its sensors, whilst a human agent should only be informed about objects visible through the windscreen and mirrors. This would allow the interaction and behaviour of human and autonomous vehicles to be demonstrated more accurately.

8. Conclusion

The purpose of this design was to build a system which allows the demonstration of autonomous vehicle algorithms as there was previously no implemented system of its kind in the literature. Requirements included that the system should be low cost, reproducible and flexible to allow the simulation of different agent and vehicle types.

A system has been developed which uses Zenwheels cars with closed loop computer vision provided by a Rasppbery Pi Cam. A physical rig has also been built to connect all the components and provide a surface for the cars to drive on. Different environments can be displayed using a projector.

Testing of the system has shown that it meets all of the outlined requirements, with the exception of supporting four cars rather than eight. Further testing and development is required to extend the system to support eight cars.

9. References

[1] Bureau of Transport and Regional Economics, “Estimating urban traffic and congestion cost trends for Australian cities,” Canberra, 2007.

[2] A. Weber, “Self-Driving Cars? Never Say Never,” *Assembly,* vol. 57, no. 1, p. 10, 2014.

[3] M. Asada, H. Kitano, I. Noda and M. Veloso, “RoboCup: Today and tomorrow—What we have learned,” *Artificial Intelligence,* vol. 110, no. 2, pp. 193-214, 1999.

[4] R. Kurjanowicz, “The DARPA Grand Challenge: Past and future,” *Computer,* vol. 39, no. 12, pp. 26-29, 2006.

[5] A. Wright, “Automotive Autonomy,” *Communications of the ACM,* vol. 54, no. 7, p. 16, 2011.

[6] T. Martin, “Samsung to Test Self-Driving Cars,” *Wall Street Journal, Eastern edition,* p. B.4, 03 May 2017.

[7] J. Hueper, G. Dervisoglu, A. Muralidharan, G. Gomes, R. Horowitz and P. Varaiya, “Macroscopic Modeling and Simulation of Freeway Traffic Flow,” *IFAC Proceedings Volumes,* vol. 42, no. 15, pp. 112-116, 2009.

[8] K. Zhang and A. Excell, “Modelling a complex give-way situation - AIMSUN vs LINSIG,” *Road & Transport Research: A Journal of Australian and New Zealand Research and Practice,* vol. 22, no. 2, pp. 16-26, 2013.

[9] E. Mintsis, M. Belibassakis, G. Mintsis, S. Basbas and M. Pitsiava-Latinopoulou, “The use of a transport simulation system (AIMSUN) to determine the environmental effects of pedestrianization and traffic management in the center of Thessaloniki,” *European Journal of Environmental Sciences,* vol. 6, no. 1, pp. 25-29, 2016.

[10] C. F. Fang and D. Pines, “Enhancing transportation engineering education with computer simulation,” *International Journal of Engineering Education,* vol. 23, no. 4, pp. 808-815, 2007.

[11] C.-F. Liao, T. Morris and M. Donath, “Development of internet-based traffic simulation framework for transportation education and training,” *Transportation Research Record,* vol. 1956, pp. 184-192, 2006.

[12] A. Fyhri, T. Bjørnskau and P. Ulleberg, "Traffic education for children with a tabletop model," *Transportation Research Part F: Psychology and Behaviour,* vol. 7, no. 4, pp. 197-207, 2004.

[13] S. Kubicki, Y. Lebrun, S. Lepreux, E. Adam, C. Kolski and R. Mandiau, "Simulation in contexts involving an interactive table and tangible objects," *Simulation Modelling Practice and Theory,* vol. 31, pp. 116-131, 2013.

[14] Y. Lebrun, E. Adam, R. Mandiau and C. Kolski, "A model for managing interactions between tangible and virtual agents on an RFID interactive tabletop: Case study in traffic simulation," *Journal of Computer and System Sciences,* vol. 81, no. 3, pp. 585-598, 2015.

[15] J. Suter, "Miniature Vehicle Testbed for Intelligent Transportation Systems," May 2016. [Online]. Available: http://hdl.handle.net/1811/76720. [Accessed 21 May 2017].

[16] A. Hurst, "Computer Vision for a Multiple-Vehicle Traffic Model," 2017.

[17] Gadget Sin, "Zen Wheels Bluetooth Controlled Mini RC Car," 09 10 2012. [Online]. Available: http://gadgetsin.com/uploads/2012/10/zen_wheels_bluetooth_controlled_mini_rc_car_3.jpg. [Accessed 19 10 2017].

[18] Anki, "Anki OVERDRIVE - Getting Started," 2017. [Online]. Available: https://www.anki.com/en-ca/overdrive/getting-started. [Accessed 19 10 2017].

[19] dianageo93, "Github -Android app to control the ZenWheels Microcar," 11 11 2013. [Online]. Available: https://github.com/dianageo93/ZenWheels-Microcar. [Accessed 02 04 2017].

[20] Raspberry Pi Foundation, "DOCUMENTATION > HARDWARE > CAMERA," 2016. [Online]. Available: https://www.raspberrypi.org/documentation/hardware/camera/. [Accessed 12 04 2017].

[21] raybarker95, "Github - cars-controller," 25 10 2017. [Online]. Available: https://github.com/raybarker95/cars-controller. [Accessed 28 10 2017].

[22] University of Western Australia Safety, Health and Wellbeing, "Testing, tagging and workplace inspections," 21 February 2017. [Online]. Available: http://www.safety.uwa.edu.au/topics/electrical-safety/testing-tagging-guidelines. [Accessed 20 May 2017].

[23] University of Western Australia Human Ethics Office, “Welcome to the Human Ethics office at UWA,” 02 March 2017. [Online]. Available: http://www.research.uwa.edu.au/staff/human-research/welcome-to-HREO. [Accessed 20 May 2017].

Appendix A – Bill of Materials

Item	**Description**	**QTY**	**Unit Price**	**Total Price**
Frame Sections				
30mm Square Pine	Standard & Better Pine DAR 30 x 30mm 1.8m Porta	6	$4.88	$29.28
Plywood (Bracketing)	Project Panel 897 x 600 x 12mm BC Plywood	1	$19.90	$19.90
30 x 12mm Pine (Fencing)	Porta 30 x 12mm 1.8m Clear Pine DAR	4	$7.43	$29.72
Fixings				
Bolts and Nuts	Zenith 1 / 4" x 25mm Zinc Plated Round Head Bolt and Nut - 12 Pack	1	$3.95	$3.95
Washers	Zenith 1/4" Zinc Plated Machine Washer - 45 Pack	1	$3.90	$3.90
30mm Screws	Zenith 10G x 30mm Galvanised Countersunk Head Timber Screw - 22 Pack	1	$4.20	$4.20
Driving Surface				
Melamime	CustomWood MDF 1200 x 900 x 3mm White Melamine Sheet	1	$10.82	$10.82
Electrical				
Powerboard	HPM 6 Outlet Surge Protector Powerboard	1	$23.95	$23.95
Projector	Epson EB-580	1	$1,900.00	$1,900.00
Raspberry Pi	Version 3	1	$50.00	$50.00
ZenWheels Cars	Red, Green, Blue, Pink	4	$75.00	$300.00
			Total Estimated Cost:	**$2,375.72**
			Total Estimated Cost (Without Projector Option):	**$475.72**