A Low-Cost Approach to Autonomous Litter Collection

By James Heselden

Completed as part of a third year BSc in Computer Science

School of Computer Science

University of Lincoln

2019

Acknowledgements:

I wish to thank all of those around me who helped to keep me on track, focused, and offered help when I was stuck. I hope I have helped you all as much as you have helped me. I would like to give special thanks to Jack Stevenson and Jisha George, your assistance went above and beyond.

To Jisha, your help to fix my terrible circuiting, soldering my motor controller and keeping me on track and focused on the important things I will always be grateful for.

To Jack, your ability debugging my initial mess of a system, and your knowledge on areas I was unsure of helped an unspeakable amount when it came to work in python, Ubuntu and Raspbian. The time you gave to help me over the year did not go unnoticed and I am grateful.

I am sure there is much more I am missing out.

In all honesty, I don’t believe the quality of this project would be anywhere near what it now is without the help and guidance the two of you offered.

Contents

[Part A: Introduction and Background 4](#_Toc5449385)

[NOT STARTED Chapter 1: Introduction to Problem 4](#_Toc5449386)

[NOT STARTED Chapter 2: Background Domain Research 4](#_Toc5449387)

[Part B: Methodology: 6](#_Toc5449388)

[REVIEW Chapter 3: Project Management 6](#_Toc5449389)

[REVIEW Chapter 4: Software Engineering Methodology 6](#_Toc5449390)

[PROGRESS Chapter 5: Planning, Evaluation & Implementation 8](#_Toc5449391)

[REVIEW Sub-System 1: Foreground Extraction 8](#_Toc5449392)

[REVIEW Sub-System 2: Object Identification and Litter Filter 12](#_Toc5449393)

[REVIEW Sub-System 3: Localisation & Movement System 14](#_Toc5449394)

[REVIEW Sub-System 4: Robot Development/Build 16](#_Toc5449395)

[REVIEW Sub-System 5: Camera Setup 18](#_Toc5449396)

[NOT STARTED Sub-System 6: System-wide Communication 20](#_Toc5449397)

[Part C: Conclusion 21](#_Toc5449398)

[NOT STARTED Chapter 6: Evaluation through Metrics 21](#_Toc5449399)

[NOT STARTED Chapter 7: Achieving the Aim 21](#_Toc5449400)

[NOT STARTED Chapter 8: Changes to Development 21](#_Toc5449401)

[Part D: Reflective Analysis 21](#_Toc5449402)

[NOT STARTED Chapter 9: WWW and EBI 21](#_Toc5449403)

[NOT STARTED Chapter 10: Further Research / Research Limitations 21](#_Toc5449404)

[Part E: References 21](#_Toc5449405)

# Part A: Introduction and Background

## NOT STARTED Chapter 1: Introduction to Problem

The aim of the project is to design and develop an autonomous system to deploy a robot to pick up litter. The project is focused around the requirements of the system being low-cost, low-maintenance, and high-efficacy. To ensure these requirements are met, special emphasis is placed on them during the decision-making processes throughout the development.

|  |
| --- |
| Abstract—Path motion object detection based on video is a fundamental part of intelligent transportation systems, In the aspect of background extraction, this paper compared all existing theories and algorithms, aimed at specific objects (city expressways or high-speed Road), and combined with the virtual loop set method. This paper proposed an extraction and updating algorithm based on the sub-segmentations of invariant background, which greatly increased the time efficiency of the background extraction. It achieved great results of accuracy and real-time of this algorithm under background extraction. |
| With the application of autonomy invading all areas, I say why not anti-littering  What are the biggest hurdles preventing autonomy in different areas?   * Cost? Effectiveness? Maintenance/Complexity? Social/Legal/Ethical?   This project plans to investigate, design and develop an autonomous system to deploy a robot to identify litter within an independent live video feed.  This paper is structured as follows. Section 2 presents an overview and the objectives of current middleware solutions. Some attributes, focusing on the architecture, simulation environment, standards and technologies, support for a distributed environment, security for accessing modules, fault detection and recovery, real-time and behavior coordination capabilities, and open-source and dynamic wiring for the most of the existing robotic middleware frameworks, are then discussed in the following sections. Each section describing an attribute for different middleware structures includes an embedded set of the appropriate bibliographic references to provide researchers with easy access to the current state of the art research in the area. The final section summarizes the survey findings. |

## NOT STARTED Chapter 2: Background Domain Research

**ESSENTIALLY A LIT REVIEW:**

A comprehensive review of literature will provide background to the project, and give weight to the decisions you have made.

This section establishes what you intended to do and shows the reader that what you have done is the result of academic study.

System Structure:

The first thing I need to contextualise, is why I set up the system in an isolated way. So why did I?

*What system structure could I have? Middleware approach? Review literature, and find some which talk about the benefits of a middleware approach to robotics and larger scale systems of interconnected systems…*

*Do a small critical evaluation of what are middleware systems… (e.g. ROS) as a good tool for connecting complex parts of a system together. <https://www.hindawi.com/journals/jr/2012/959013/>*

*Talk about and critically evaluate some basic cost reduction techniques like using cheaper materials and parts like USB webcam, over integrated circuitry on the robot.*

System Components:

Then I need to talk about why I split the project into the sub systems I did.

The specific implementations of each of the sub systems are described below, so the details on them I don’t need to worry about here. But I do need to describe why I chose to make each of these systems isolated.

*List off the individual sub-systems along with the system diagram.*

*Describe each one in turn, and describe what it contains, and why this is considered independent.*

*Try to find some benefits online and in research as to why this system should be independent.*

# REVIEW Part B: Methodology:

## REVIEW Chapter 3: Project Management

During the initial conception of the project, a plan was put forward to lay out the time scales of each of the tasks, so as to get a better perspective of the project. The Gantt chart laid out 4-5 distinct sections of the project, basic image processing, basic robot, advanced image processing, and advanced robot. Each of these sections was given a defined milestone of which the section must be completed by, and smaller milestones which individual components must be completed by.

In actuality, the project deviated from this quite dramatically for a few reasons, the first was the time estimation for building the robot, where the building of the robot took significantly longer than expected due to lack of experience and overestimation of ability. For the implementation of advanced image processing, the aim was to develop a ML approach to identification, however when research was conducted, it was found that existing cloud-based systems could offer much more advanced functionality than could be made with the time and resources available, so this was implemented within a couple days, rather than the 6 weeks planned. The advanced robotics mapping was also removed from the project as for a proof of concept, this feature was far too complex to implement.

Throughout development, goals were set weekly to ensure the development continued smoothly, without much delay. The weekly goals were defined at the start of each week, as small achievable aims such as “Implement a mean and median background construction script & test automatic connection between camera and server”. Weekly goals were used to ensure priority lists were kept up to date for changes which occurred throughout the project, and they proved to be a helpful tool to the project, for instance, after the development of the robot stagnated and delayed the Gantt chart time estimations, the project worked solely off of the weekly aims. These being developed at the start of each week, ensured focus was being placed on the high-priority tasks which working solely off the Gantt chart did not allow.

## REVIEW Chapter 4: Software Engineering Methodology

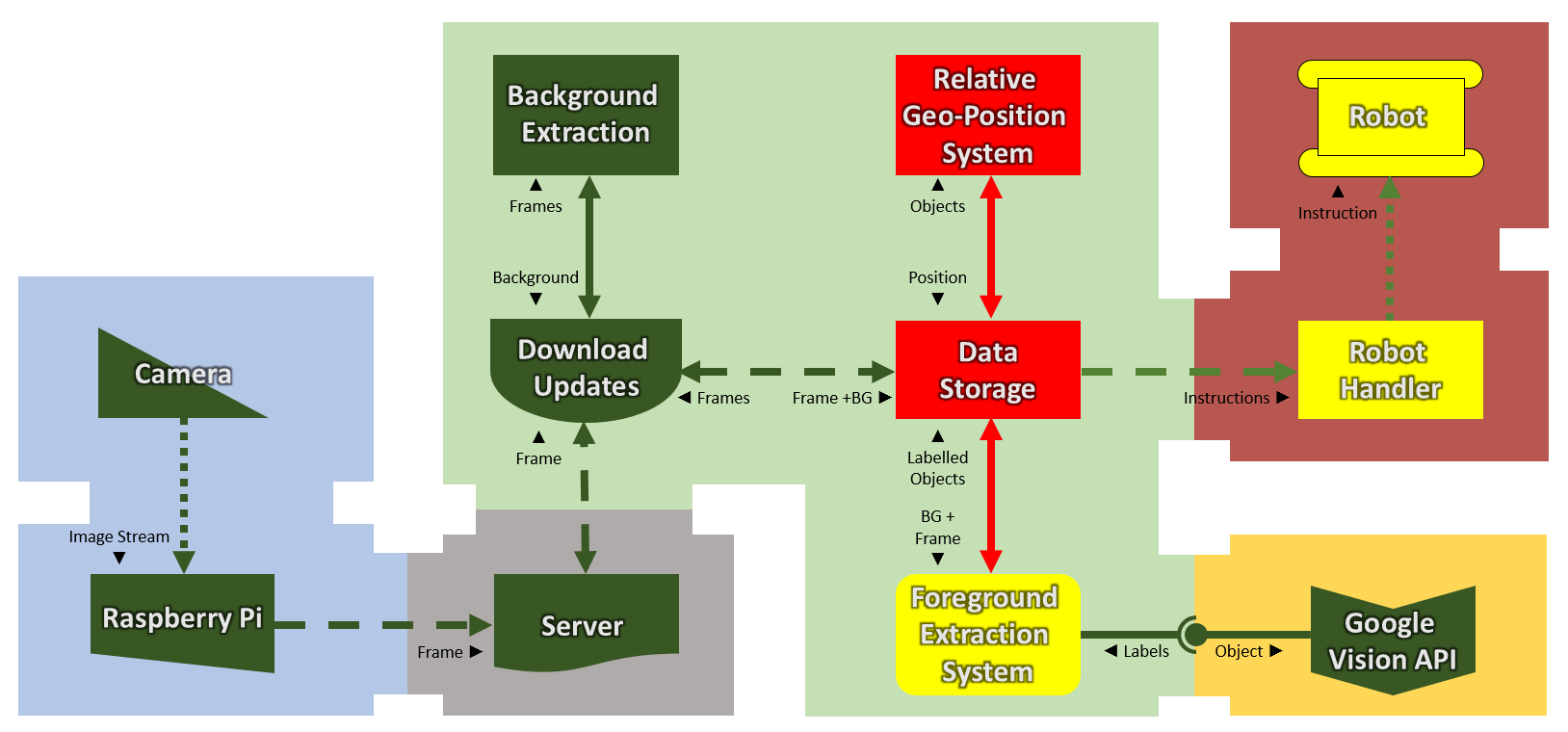
The project initially was aimed as following a waterfall approach. This was due to the structure of the system and the impact of testing the system in an outdoors environment; however as the development continued and new understanding was found on the style and structure of the control system, the project became more of an adaptive waterfall approach, where each sub system in the project was developed under an independent adaptive waterfall methodology to ensure the systems were able to adapt to the growing demands. This was a very adaptive approach to the development of a system with this type of structure, as each individual sub-system was developed to a high quality without too much back-tracking on issues. The systems themselves were all quite small meaning that going back a level of the waterfall did not cause much issue, but together they combined to a strong project, which was well developed to meet the aims set out.

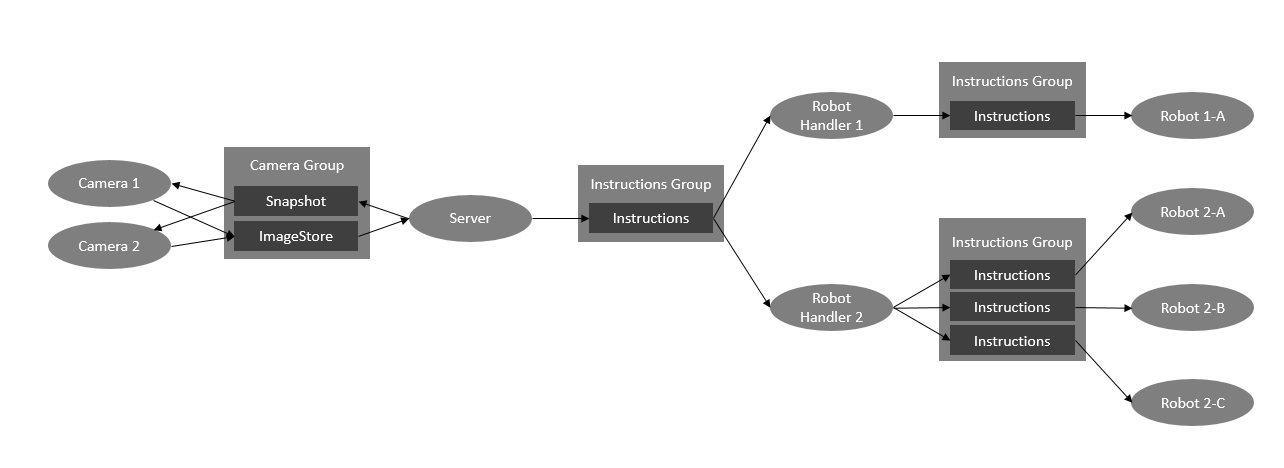
Following the adaptive waterfall approach to developing individual sub-systems, meant each sub-system went through requirements gathering, design, testing and evaluation. Which meant nearly the entire software development lifecycle was met during each stage of the development.

As described previously, the project is structured as a series of independent systems with an intercommunication structure set up to allow data passing, analysis and control. Each system was built independently with specific input and output structures.

The process in which each system was built is described below. For each sub-system, a review of best practices and methods was carried out, followed by an analysis of the best evaluation metric. This is followed by a basic implementation of the most appropriate systems, an evaluation of them and finally the implementation into the full project.

The connection of the sub-systems resembles the following diagram:





**HOW ARE YOU GONNA TEST EACH SUB-SYSTEM?!?!?!**

## REVIEW Chapter 5: Planning, Evaluation & Implementation

### REVIEW Sub-System 1: Foreground Extraction

This sub-system is arguably the most important to the system, as without it, the following sub-systems would have no data to work with. Research into this started with looking into background extraction for use with background subtraction. In this, a background is generated from a single or series of images, then every subsequent image is compared with the background, and the regions of the image which are identical are removed, leaving only the foreground.

Research into background extraction began, and it was found that methods such as image stacking could be used as a very simple and reasonably effective system. Mean, median and mode stacking, were implemented, and all gave different errors. Mode stacking on a relatively small timed cycle was found to be the most effective and allow the smallest impact of changes. Mean is impacted by random small changes, resulting in noise, and when there is a lot of change, it cannot identify the background. Mode is not impacted by random small changes as outliers are irrelevant to the general modal value as it takes what is most static in the frame-set, which should in most cases represent the background.

The biggest issue with applying mode stacking is impulse valued noise occurring. Without, by the least, an identification system for where these problems are occurring, this method is unusable in practice without some sort of smoothing. Attempts to remove IVN mostly revolves around adaptive median filters [https://www.sciencedirect.com/science/article/pii/S1047320314001631], however while smoothing would work to reduce these regions, this comes with its own problems, that being the distortion between the background and new images, which causes errors with the extraction process. The maths behind why mode filtering is so effective at detecting change stems from the lack of maths applied to the values directly, this ensures the comparison with a live image once the background is generated would be accurate, applying median filters would distort this clarity.

Despite its disadvantages, the method worked well, with extraction of foreground objects as shown in figure 1 after a stacking of 20 images, with a temporal spacing of 1 minute; note the noise around the middle of the generated background.



Figure . From top to bottom: (a) Mode stacked image; (b) Input image; (c) Foreground extracted

An adaptive algorithm was also developed to use entropy-based stacking similar to blur detection. In this, regions would be weighted based on their activity, with low active areas being discarded. This process was very resource intensive for the initial generation of background, however it meant the background would work regardless of the state of the environment, weather, and lighting conditions [https://res.mdpi.com/applsci/applsci-08-00807/article\_deploy/applsci-08-00807.pdf?filename=&attachment=1]. The biggest disadvantage to this was the reduced complexity of the resulting image, where due to the blurring performed on the image, the sampling dropped significantly, and there was a strong difference between the background and the new input. Implementation of this was an attempt to rectify a common issue, that being , how trees affect background generation. Where their fixed movement consistently impacts the generated background, and thus the foreground extraction often is filled with many parts of the tree which are different from the background.

Edge Detection was also researched as a method of detecting objects regardless of the lighting and weather conditions, as the edges would not change, however issues did arise with this in practice. In particular the level of detail, where in an outdoor environment which this was designed for the, detail on the ground was often less detailed then the object placed in the image. This difference in detail meant the entire ground around the object would need to be registered in the background for the foreground object to appear; making this method inappropriate for implementation to the system if used primarily outdoors.

**Tools:**

Much consideration was undertaken in choosing the tools used for the image processing system, the requirements for the system in practice is speed, however due to the nature of the project, testing must be done with every choice made; because of this, the choices for language implementation were primarily; MATLAB: an integrated high-level language and IDE for matrix operations and simulation; and Python: a relatively lower level language used often for real-time image processing systems.

Python has an advantage in terms of the processing speed, for a system like this, fast processing allows for a much more reactive architecture in implementation, mode stacking is quite computationally heavy making python a good tool for this.

MATLAB offers something much more fitting to an adaptive framework such as the one being employed. As MATLAB is designed for numerical computation and visualisation, it is much more appropriate for the exploration and evaluation of many different approaches in a shorter amount of time. The inclusion of many complex toolboxes for MATLAB such as the image acquisition toolbox, also allows for many potential implementations to be tested in the context of the project without the requirement of programming; which could include unnoticed bugs worsening the development. The largest caveat to working with MATLAB for the development is cost, where MATLAB requires an expensive licence to use, while Python is free.

As the project focuses around testing many types of implementation, the logical choice would be to use MATLAB, as this would allow much more testing in much less time. However, as the system at release would require a reactive and low-cost solution; further development of the system would require a different language which is lower-level and free to use, such as Python.

### REVIEW Sub-System 2: Object Identification and Litter Filter

The first thing to do was to find out how we could identify litter in the first place once the object is extracted, so what characteristics could be used to differentiate objects, in order to be able to recognise them?

Some listed are, hue, intensity, saturation, glare, shape, sharpness. After some early consideration around the type of things being identified, as it is litter, the shape could change, consider a new packet of crisps and how that same packet would look screwed up into a ball; or the distortion of a crushed can of cola. These understandings meant that any sort of basic approach with regards to object recognition could not be done on shape, thus colour based visual recognition was the most appropriate to go for.

Additional methods were also tested using more complex systems like entropy analysis. The aim for this section of development was to gather data using values identified and test the effectiveness of a handful of clustering algorithm on the data such as Simple K-Means, KNN and Random Forests.

It was found before the clustering was applied to the data, that there was a more effective and time efficient approach to implement for the identification; the application of cloud-based APIs. The Google Vision API was found, through [<https://activewizards.com/blog/comparison-of-the-top-cloud-apis-for-computer-vision/>] to be the most effective, and so was deployed to the system with a simple JavaScript file which formatted and passed the message to the web server.

There is a big disadvantage to using this setup and that is the requirement for the server to be connected to the internet, which adds potential leaks for security in an otherwise enclosed system. There is also an added cost element, where for testing purposes, the account being used is limited to so many requests per day for free, however as time goes on and further development to the system is completed. A more adaptive approach using a custom built ML system as planned may be more appropriate, perhaps using some sort of deep convolutional artificial neural network such as the DCNN described by Sun et al. in [<https://sites.google.com/site/romansbirmingham/>], of which would apply quite well to the problem domain here.

### REVIEW Sub-System 3: Localisation & Movement System

##### Localisation:

Localisation is on the harder spectrum of tasks when it comes to autonomous robotics, with it often requiring an expensive and highly calibrated tools such as laser scanners, Lidar or depth sensors. There has been a recent increase in localisation using cheaper alternatives such as **MONO CAMERA LOCALISATION BY <INSERT NAME HERE>,** using these types of setups allow the cost for building robots to go down, allowing for greater accessibility to the field. These robots often still require cameras which can cost over £100 like the Kinect.

Due to the nature of a small robot designed to pick up litter in an outdoors environment, the robot could be subjected to harsh and unclean conditions which could make a mounted camera unusable for effective localisation and planning. As such, the project has been designed away from including a mounted camera on the robot, and has instead chosen to adopt a style of localisation using external cameras. This method as described by **<** **J-AND-Y >**, **COULD BLAH BLAH BLAH**.

Removing the mounted camera, also leads to other benefits such as an easier way to detect humans approaching the robot, and easier maintenance on the robot itself, as the number of parts is decreased. By removing the camera, the communications with the robot become a one way interface, leading to less demand for the robot handler, the weight of the robot decreases, requiring a less intensive battery, and the impact of a robot being broken or stolen is lass impactful for the client.

The system is not without fault where the communications is concerned, as the robot requires a direct connection to its control hub receive any commands, the system has a larger latency, leading to the robot becoming less responsive to immediate change in the environment around it. This is countered however, by what is arguably the most important benefit of designing the system like this, which is the absence of a strong computation device mounted on the robot itself, as the robot only requires the ability to receive and process communications to send to the motors, an expensive, lightweight computer is not required.

The robot also becomes completely useless with respect to the environment outside the fixed camera’s visibility. The system also has fault with costs relative to the ratio of robots to coverable land, where having a fixed camera on a robot may be cheaper if there is only 1 robot patrolling a large facility compared with many fixed cameras to cover the entire traversable area.

As the project is focused around a proof of concept for the implementation, a more adaptive approach has been used for the robot control, with a simply system of 3 point alignment being implemented. In this localisation system, 3 points are identified, being the litter, the front and the end of the robot. The robot is spun till it is facing the litter at which point the three points are aligned, then the robot is moved forward until it reaches the litter.

### REVIEW Sub-System 4: Robot Development/Build

Many considerations were made when designing the robot and many of the initial decisions were changed due to this overly ambitious design, and the consideration of time and learning constraints applied to the project.

The initial design consisted of 3 major components: the frame and motors; the motor control system; and the computer and communications hardware.

The initial design for the frame and motors was based around applying the robot to an outdoors setting, on pavements and the occasional patch of grass or gravel. As a result, common ground materials became a strong consideration in deciding the type of locomotion, with the first major set of options being tyres or continuous track. In terms of effectives in off road movement, and possible weather implications on the ground tank tracks would be better, as they are designed to spread the weight of the vehicle over a larger area, making it more effective when moving over muddy conditions. The caveat to using continuous tracks is cost, and maintenance, where tracks are built of many smaller pieces connected together, the cost for pieces is much higher then that of tyres around each of the wheels, along with this cost, there is an added cost of maintenance if the tracks slip at all, an issue which does not lie with using wheels. There is also a movement reduction with tracks due to their design nature compared with the full movement of wheeled vehicles.

There were many types of wheeled vehicles which could have been developed cheaply for the given task, as only basic movement was required, the options were front wheel drive, rear wheel drive, opposing wheel drive (front left and rear right or vice versa), and 4-wheel drive. There are many other types of drive systems, however these are the main 4 which were considered for this system. The aim was to go with 4-wheel drive, as this in theory, would give the most power to weight ratio.

Consideration towards the computer and communications hardware was also heavily considered, as there was an abundance of choices for this, more so then with the camera hardware, the choice made here was quite difficult. As the robot would work independently, there was a requirement for the system to be low resource, and as the robot would require mounting the device, it must be light weight. Thus, the most appropriate decisions were between the Arduino UNO and the Raspberry Pi 3B+. Both project boards are able to control motors using a motor driver, and both are able to process information to the level required, and both are able to interact with ROS networks. Due to the reactive nature of an autonomous robot, for example the requirements to stop when something serious occurs such as an interference with the robot, the Raspberry Pi’s faster response time would help a lot with processing the data faster, as it has a 1.6Ghz processor compared with the 16Mhz processor on the Arduino. There are further comparisons which could be made in terms of the Raspberry Pi Zero W which is much cheaper then the alternatives, however for a system with a higher risk, the additional costs to ensure the mechatronics is reliably fast is a worthwhile.

The motor control system itself was also a highly considered system in which quite a lot of research was conducted as this was an areas which was very important to get right. The initial plan before research was conducted was to attempt to wire a breadboard with the L293D motor driver chip, connecting to the raspberry pi, as this would allow an incredibly low price to the development of the additional circuitry to the system. However after careful consideration and research on the internet into the cost of prebuilt motor controller shields, it was chosen to use the MotoZero from ThePiHut.com for £10 as this would ensure a neat and effective solution given the time available for the project.

As cost is one of the more important metrics associated with the aim of the project, cost reduction for the robot itself was high priority; because of this, it was decided to rewire an already mass-produced car as this would be relatively cheaper for prototyping then building the robot from scratch. A remote-controlled car was purchased for £12, and rewired, with a basic low-powered computer and battery pack attached to it for testing.

It was found that the power offered by a conventional battery pack would be to small to power the cheap motors well enough to move the robot effectively. As such, careful consideration on the complexity and size of such a system, it was found that without setting up a gearing system, the robot would have to be quite large in order to space the 4 wheel motors; as including a gearing system would allow more potential points of failure for the system, which could lead to more complex maintenance and such a lower level of autonomy. Using a system of opposing wheels would be much more beneficial, allowing the robot to stay small, and not require gearing.

As the RC car was unable to perform effectively, there was no choice but to get a new car base to use. After some careful research, a car frame with opposing motors was found on Amazon [<https://www.amazon.com/gp/product/1457186039/ref=as_li_qf_sp_asin_il_tl?ie=UTF8&camp=1789&creative=9325&creativeASIN=1457186039&linkCode=as2&tag=therobpod-20&linkId=QHNJA3OMPG5P7T4Z>]. This was purchased for £30 and once it arrived, it was assembled and the remaining components for the robot were put together, recycling the wheels from the RC car into the new frame as the purchased car came with continuous tracks. Not much consideration was taken to the specifics of the purchase, as the priority at this point was to get the high-fi prototype completed. The frame itself consisted of only 2 motors, some bearings, wheel mounts and a basic metal frame.

### REVIEW Sub-System 5: Camera Setup

##### Hardware:

The initial plan for the camera system was to make use of a wall mounted camera, as opposed to a robot mounted one. This decision stemmed around system control, where the system itself would use the robot to as a tool to complete its overall goal, rather than the robot using the camera to complete its own independent task. The server in this instance works as a hub to connect all other devices, be them cameras or robots.

Before considering the specific camera to use, some decisions had to be made, in terms of the type of connection to the server, the cost limitations, and the quality requirements.

For the quality requirements, two cameras were used and tested, with the aim to find if the lower cost camera (a £10 USB camera) was able to perform nearly equally to the more expensive (£30 4k action camera). To ensure the cameras worked for the environment, a few small tests were carried out with the quality of the output image, where a scenario was developed in which litter would be placed haphazardly in a region, and the cameras outputs would be tested in their quality when performing the stacking. It was found through these tests that a basic £12 USB webcam, had high enough quality to detect a piece of paper 50mm wide, from 2.5m away using mode stacking, so the 4k action camera was abandoned. The Pi cam was also reviewed however the module only offered 8MP, which was twice that of the USB webcam, however it was significantly more expensive, costing £24. The benefit of its resolution and neatness are clear, however for the task required; what the USB webcam offers is acceptable, especially since the one of the project aims revolves around cost reduction.

The next stage was choosing a suitable connection from the camera to the server. For simplicity of a project on this scale, cost and time limitations, the most appropriate method would have been to set up the camera on the server computer itself, working with a wireless system meant costs were also reduced in terms of the cabling which would have been required for an upscaled implementation, and less cost in terms of maintenance and installation. So a more scalable and technically challenging approach was taken, which involved connecting the camera to a piece of low-power hardware, in this case a raspberry pi, such as with the robot, and feeding the images through the local network to the server. The Raspberry Pi 3B+ costs £34; due to this expense, and the relatively low amount of processing on the board, testing was also carried out with running the setup on the lower power and cheaper board, the Raspberry Pi ZeroW+, which costs only £10. Testing was done on the effectiveness of the Pi ZeroW for the type and scale of file communications such as would be used in this system. This was **NOT SURE, NO DONE TESTING YET**

**::: INSERT IMAGE OF TESTING RESULTS :::**.

##### Software:

The initial plan for the communications was to reduce the amount of data being sent by processing the images on the camera device, then sending small strings of data to the server with the intention of speeding up the communications and message management from the pi. Once testing began on the speed of the processing, it was found that the pi did not have enough power to process the images in a reasonable time, taking approximately 62 seconds to generate the background image, which is far too long considering the frames used to make up the background image are spaced 60 seconds apart.

It was decided to send the files directly to the server, using ftp [<https://www.raspberrypi.org/documentation/remote-access/ftp.md>], then process the images on the better device. It was found through doing this, that this was much faster then previously. The setup was still quite slow despite the FTP connection being local, leading to the assumption that the Pi was unable to send over FTP very well. **THIS IS TERRIBLY WRITTEN**. For a short period, the system was redesigned to use HTTP passing, over the internet to an independent web server, to then be dragged back down onto the server, and this, despite having much overhead, was near instantaneous. The system was redesigned to work through the ROS with the rest of the network, this resulted in lower latency and more control, while also removing the security risks which come as a result of connecting to the open internet.

The system redesign had settings which required consideration, mainly the data retrieval. It was undecided whether to use a system of timed publishing from the cameras; or a system where the cameras would only send an image if a broadcast with their id was published from the server. The latter was chosen after consideration of management simplicity where each new camera added would not have to be set up individually, the server would only have to be updated to consider the additional camera.

### REVIEW Sub-System 6: System-wide Communication

System-wide communication is the single most important thing for a distributed system, as without it the individual components, regardless of their efficacy, will be unable to communicate and without communication, the system will not be able to achieve its aim.

For this system, a middleware approach has been undertaken in which a software technology is used to manage the complexity of the distributed system and to connect the individual components into a network which spans multiple processors enabling the communication necessary for passing information across devices.

The choice of the middleware available is dependent on a number of factors of weighted importance. To ensure this decision was made most appropriately, the comprehensive review on middleware by Elkady and Sobh was referred to consistently [<https://www.hindawi.com/journals/jr/2012/959013/>].

As this project is working with a system of autonomous robotics, it is important to consider the latency of the middleware communications as the camera, robot, and server are distributed. Without a low latency, the robot will not be able to achieve a reactive nature to the level an autonomous system would require ensuring the minimal amount of disruption to the environment and humans around the robot. The distributed nature of the network must also be factored into the choice of middleware, as it must work across processors on separate devices, allowing for a decentralised network. The platforms the middleware work on is also an important factor as both the camera and the robot work off of Raspbian, a Debian based operating system.

Security was also a major consideration which was brought up by Elkady and Sobh, in which the middleware should offer, for distributed networks especially, a security aspects such as authentication, authorisation, and secure communications to ensure no unwanted access to the robots under control.

Based on the requirements, and some other minor factors such as ease of development, conciseness of documentation, update activity and costs, the list of middleware systems was reduced down. The open source nature of many of potential candidates were focused on, as cost reduction is one of the primary aims for the project.

ROS was found in the end to offer nearly all the functionality required with the only problem being the security aspects. As ROS is a distributed and networked system, it is by design able to receive and communicate shared resources, so security is a large issue (<http://wiki.ros.org/Security>). Security precautions must be added to the system in order to restrict access, implementations such as adding encryption to message passing could be an effective tool and has been developed and tested for protecting private user data in human robot interaction settings (<https://doi.org/10.3389/fict.2018.00002>). Including internal security alone will not prevent flooding attacks such as DNS, which could aim to bring chaos to a network through restricting the resources available for processing, and filling up queues which the processing relies on, there has not been much research into protecting this however the ROS wiki recommends restricting the access to the network and disabling connection to the wider internet as the primary method of protection, with employing firewall rules additionally to protect this. Optionally, tunnelling could be managed for connecting the system over a wider network or even the internet, however this comes with additional overhead.

As the system developed makes use of the Google Cloud API, the host computer must be connected to the internet to manage these communications, meaning the system must be connected to the internet to function. This problem is discussed further in the chapter on research limitations, however for a production-ready system, a new object identification system such as the DCNN described by Sun et al. could be implemented to counter this.

In conclusion, as ROS is open source, the costs are negligible, the benefits of the ROS control model fit the requirements of the project communications very well, and the middleware works on Raspbian technology and over a distributed environment. The biggest benefit of including ROS in the design for this system however is the complexity of the resources, and the scale of its following ensure the active development of the middleware, which ensure the system is continuously evolving and reliable.

# Part C: Conclusion

## NOT STARTED Chapter 6: Evaluation through Metrics

## NOT STARTED Chapter 7: Achieving the Aim

## NOT STARTED Chapter 8: Changes to Development

# Part D: Reflective Analysis

## NOT STARTED Chapter 9: WWW and EBI

As described before, there were limits to what could be achieved considering the scale and complexity of the project along with the time available to achieve the aim. The underestimation on time for building the robot, along with the lack of knowledge in the area delayed the project significantly, this impacted the development…

Breadboarding a motor driver chip to control it was not successful and broke the Pi, diodes, LEDs, and breadboard.

## NOT STARTED Chapter 10: Further Research / Research Limitations

As the system developed makes use of the Google Cloud API, the host computer must be connected to the internet to manage these communications, meaning the system must be connected to the internet to function. This problem is discussed further in the chapter on research limitations, however for a production-ready system, a new object identification system such as the DCNN described by Sun et al. could be implemented to counter this.

# Part E: References