A Cost-Centred Approach to Autonomous Litter Collection

By James Heselden

Completed as part of a third year BSc (Hons) 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

WORKING Part A: Introduction and Background

NOT STARTED Chapter 1: Abstract

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.

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.

REVIEW Chapter 1: Introduction to Problem

The problem of litter is getting worse, as described by (KeepBritainTidy, 2018), and the impact it has on the country, economy, and even the world, is evident; with over £1 billion spent in 2015 alone, on attempts to clean it up (Rowe, 2019). The impact littering has on society is also becoming more prevalent. Recent research into the psychological impacts of littering have highlighted issues and causations which extend beyond the action of littering to understand the mentality of why people litter in the first place, and this research has been put into effect in terms of the 5p carrier bag cost (Gove, 2018); in which the physical worth of the carrier bag results in an emotional acknowledgement to the bags inherent value to retain. This is not enough however, as despite the decrease in carrier bag littering, the amount of littering is still increasing.

Coupled with the rise of autonomy (self-controlled robotic systems), and modern research into the psychology of those who choose to litter; this project aims to develop an adaptive approach to combat this issue. The project will take lessons learnt from the development of systems built by others such as (Nishida et al., 2006) and (Bonnema, 2012), in order to develop a reliable and appropriate implementation to tackle the problem.

The first system to be analysed is the Japanese Outdoor Service Robot (OSR-2). Developed by **Nishida et al.** from 2003 to 2006 it aimed to clean urban areas by collecting discarded trash. The system was effective in identifying and collecting the trash, however it had a large issue which was not discussed as much, and that is the size and cost. For a system like this to manage an area autonomously, it must be able to move into smaller areas in which litter may build up from, and it must be accessible to as many organisations as possible.

Managing the cost of the robot more effectively enables the robot to become faster, as lighter, and smaller components decreases the power consumption of the robot, which in turn decreases the costs further. The benefit to employing these types of autonomous systems is to ensure a large area is kept tidy, however for a large area, a slow bulky robot may not be able to manage. This problem can be solved easily by spreading out more robots and increasing their speed, however for this to be an appropriate choice for an organisation, the robots must be accessible, of which this implementation is not so much.

For industrial use, the benefit to reducing the size and thus the complexity of robots is clear, take for instance the Meca500, an industrial 6-axis arm which has been developed by Mecademic to increase the accuracy of automation, with a 5µm accuracy. The arm was designed small to take advantage of the accuracy which can be achieved, while also reducing unnecessary costs which unnecessarily increase costs (Mecademic, undated).

The second system to be analysed is the result of (Bonnema, 2012), in which a Hako Hamster 700 street cleaner was converted to an autonomous robot to sweep litter. The project, led by Maarten Bonnema, planned to develop a system to tackle litter in public places, and was built as part of an interdisciplinary project by multiple students under supervision from multiple staff members. As the project focused around developing the robot to pick up litter solely, the project did not take into account the organisational practices and costs for producing and deploying the system to industry. This lack of consideration meant the team did not consider tactics to reduce the costs or evaluate the benefits of doing so.

The psychology into why people litter has been researched a lot recently, with many papers being released on why the members of the public choose to litter. The major paper focused on for this project is “Beacons of litter: A social experiment to understand how the presence of certain littered items influences rates of littering." (Tehan et al., 2017), in which it was concluded that the average person is more likely to litter, if they recognise litter already in an environment, for example a person is more likely to litter a branded can of drink, if they can see another of that branded drink on the floor; this can be extended to types of branded food wrappers such as fast food waste, which is in itself designed to be easily recognisable by their colours alone (Howarth, 2017). Understanding this type of psychology can help the efficacy of autonomous systems, as with this, it is possible to develop systems which can target litter in a much more effective manner, and in such a way to quickly support the reduction of litter in an environment while more complex systems work to remove the remainder of litter.

As described, the importance of tackling litter is a growing problem, with an increasing urgency as more of the environment is exposed to its effects. Tackling litter through autonomy is an effective way to handle the growing demand, however more complex tactics must be employed to the autonomy to tackle this problem more effectively.

With all this in mind, the aim for this project is to develop a low-cost, low-maintenance autonomous tool to assist with litter handling.

REVIEW Chapter 2: Problem Exploration

Due to the structuring of the report, the problem exploration and review into literature extends beyond this chapter and into chapters 5 and 6, in which the sub-systems which make up the implementation are designed, tested, and evaluated.

The aim developed for this project faces 2 major considerations, to design the system to be low-cost, and to design the system to be low-maintenance.

As described by Leung and White, maintenance comes as a large cost to the development of solutions (Leung and White, 1991), and a key component to reducing the necessity of maintenance comes from the rigour of the testing an implementation goes through. The ease of maintenance can also reduce the costs, where ensuring ease of access to different parts and systems can ensure the costs and time spent on maintenance are kept low; even in a deployed system (Anandan, 2015).

The costs to robotic systems as described by, [ibid.] decreases as the operating costs decreases, making a simple method to reduce the cost of a robotic system, to reduce the size and complexity of the robot itself. For autonomous mobile robotic systems, this is even more important, where reducing the size of the robot and thus the weight of it, reduces the requirements for the power units of the robot in order for it to move on its own. As described by Henrik Christensen (Christensen, 2014), the general costs for industrial robotics is generally broken down into 25% basic robot system, 25% auxiliary hardware, and 50% software. The cost described here for software is so relatively high due to the complexity and reliability which comes from complex industrial robotic systems, and the testing and maintenance which most go with it. By simplifying the systems and removing as much complexity to the system, the costs to both the basic robot system and the software can be decreased a lot. The emergence of middleware as described [ibid.] can also help to reduce costs by around 30-40%. This is because the software can be integrated in complex fashions, with a much simpler interface and control structure, in a much shorter time, and the long-term maintenance once deployed can also become much simpler. This also allows pipeline infrastructure to be developed, where a complex system is broken down into independently controlled sub-systems of which communication and message passing is placed at a higher importance.

As the use of autonomous robots increases, there must be consideration to the reaction from members of the public who detest the nature of the machines. With multiple attacks on self-driving cars (White, 2018), food delivery robots (Hamilton, 2018), and security patrol robots (McCormick, 2017), the risk of expensive components being damaged and causing the robot to lose control is a serious concern. This risk is escalated by the introduction of children, where research has found that children will not show remorse for attacking or damaging a robot which they cannot perceive as feeling pain (Darling, 2015), despite the implication of damage.

Taking these understandings forward within this project, it is clear that simple, specific decisions can be taken to improve the functionality and deployment of an autonomous system. Decisions such as separating complex and expensive components from the robot, can allow the robot to be cheaper, work to a higher performance, and have less risk of damage.

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.

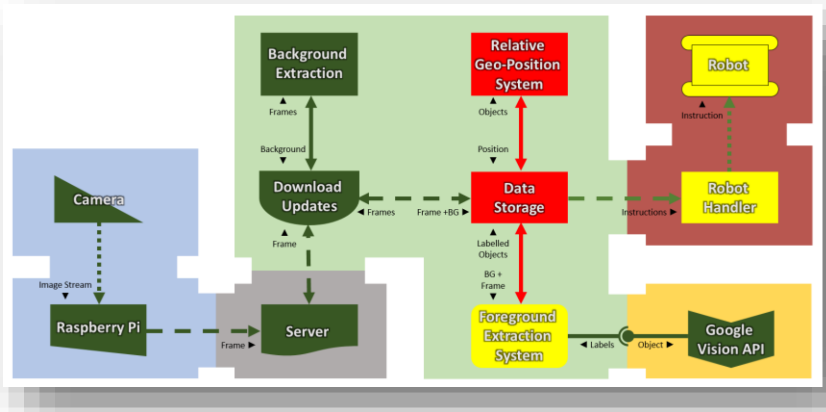
Following the adaptive waterfall approach to developing individual sub-systems, meant each sub-system went through requirements gathering, design, testing and evaluation. Which meant nearly the entire software development lifecycle was met during each stage of the development.

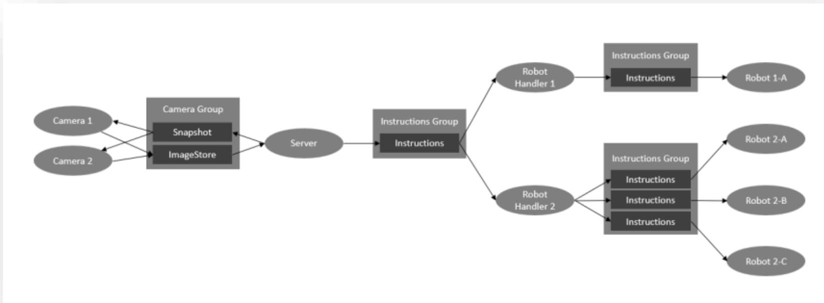
REVIEW Chapter 5: Planning, Evaluation & Implementation

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:





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 (Gupta et al., 2015), 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 1. 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 (Ali, 2018). 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 (Singh et al., 2017), 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 (MATLAB, undated). The inclusion of many complex toolboxes for MATLAB such as the image acquisition toolbox and computer vision 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 (ActiveWizards, 2018) 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 (Sun et al., 2018), 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 the MonoSLAM system developed by (Davison et al., 2007)**,** 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 (Shim and Cho, 2016) allows independent cameras to process and localise a robot which appears within them.

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 (OTTFF, undated). This was purchased for £35 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.

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 (RaspberryPi, undated), 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. 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 (Elkady and Sobh, 2018).

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 (ROS, 2018). 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 (Rodríguez-Lera et al., 2018). 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.

REVIEW Part C: Conclusion

REVIEW Chapter 6: Evaluation through Metrics

The aim of this project was to design, develop and evaluate a low-cost, low-maintenance solution to retrieving litter in an open environment. The evaluation parameters identified through this are quite clear in part, with reductions to cost and the maintenance required on fault occurrence.

Efficacy:

The efficacy of the system was not evaluated as a whole, as the purpose of the project was to define a proof of concept and to prove the feasibility of a system to be implemented by an industry professional. This is not to say that efficacy was not evaluated, but the results of the evaluation were not important to the aim.

The efficacy in this context is not something which can be easily evaluated through metrics alone; as efficacy is built up of the effectiveness of the individual sub-systems to complete their tasks. The project has, through development, aimed to reduce the negative impacts of the sub-systems by using best-practice methods and critiquing the choices made for the implementation in each sub-system. Issues within each of the sub-systems were, through this critical evaluation addressed, such as; the latency of the cameras, the speed of processing the frames, the Google Cloud API response speed , the accuracy of the pathing system, the robot's movement flexibility. Since the failure or inefficacy of a single sub-system within the overall system is a failure of the system as a whole, this was handled very carefully.

Costs:

Evaluation of the cost metric with respect to the aim is quite simple; this section will begin with a breakdown of the costs associated with the individual components of the system, followed by a comparison against existing complex systems for removing litter, and a comparison with the costs associated with small autonomous systems not designed with litter in mind. The cost of the developed system can be broken down into 3 distinct sections; the camera, the robot, and the server. Each of these sections have their own associated costs, with an additional section of costs associated with initial setup and maintenance.

Computer:	Raspberry Pi ZeroW	$9.30
Hard Drive:	8GB SD Card	$4
Camera:	MS LifeCam HD-3000	$24.99
Case:	Pi Zero Case	$6
	Total Cost:	$44.29

Table 1, costs associated with camera

Computer:	Raspberry Pi ZeroW	$9.30
Hard Drive:	8GB SD Card	$4
Case:	Pi Zero Case	$6
Motors:	2x Motors	$11.96
Motor Controller:	MotoZero	$10
Battery:	5V2.5A Power Supply	$29.16
Frame & Wheels:	Metal Sheet & Wheels	$5
	Total Cost:	$75.52

Table 2, costs associated with robot (note this does not include robot handling technology such as remote charging)

The development of the project was aimed primarily at reducing the costs associated with the camera and robot; however due to the nature of the project, not all potential costs could be removed. The MotoZero would not be used in an industry setting as it offers more redundant functionality which is required and building the component itself makes its cost nearly negligible; for this project however, that was unrealistic and attempts to do this were unsuccessful. The biggest cost to the development of the robot was the power supply, as an off the shelf component for this is used, its cost is large.

The development costs for the robot and camera serve to represent the maximum potential cost, which would be received. Within an actual implementation of a system like this, the costs would reduce due to many factors such as mass production, buying in bulk, and simply using elements which are not already inflated due to consumer pricing.

The evaluation of the system with respect to the cost is not so simple as there are no clear comparisons for component costs on the market. The following section aims to compare the implementations of 2 classes of robots, these being; existing litter collection robots, and low-cost autonomous robots. This comparison, will look at the components used for the implementation in this project, evaluating their benefits and drawbacks over the methods employed by the industry implementations.

Many attempts have been made towards building robots to autonomously pick up litter, with one of the most invested approaches coming from Nishida et al., with their OSR-02, a robot designed to clean urban areas by removing litter. The robot itself is quite large as its design allows the robot to essentially carry with it a bin in which to collect the litter found while outdoors. The system itself uses a laser rangefinder and multiple cameras, to scan the environment, along with 2 additional cameras at the end of the robot's arms for more accurate litter retrieval, making the robot itself quite expensive; while not explicitly stated within the reports, the cost of a URG-04LX Laser Rangefinder new is over £1050, making one of the simpler parts of the robot extremely expensive, and dwarfing the costs of the robots designed for this implementation (Nishida et al., 2006).

More analysis and evaluation has been done on sensor-based systems, such as the report by Dresscher 2010, as part of the JaClean project which aimed to develop a system to assist street cleaners with litter collection. The report featured multiple multi-sensor systems, which could be used with litter collecting robots to for localisation and object detection. The least expensive of these came in at €170 (roughly £144 at the time), however each of the sensor systems suggested had many issues such as increased mechanics, leading to more potential failure points (Dresscher, 2010).

Developments by Maarten Bonnema into system design made use of the Hako Hamster 700 Electric Sweeper, which is a petrol operated floor sweeper, which was used by street cleaners and workshop cleaners to pick up litter and dust, the Hako Hamster 700 often retails at over £1000 used. The cleaner, in this report, was upgraded with additional rangefinder sensors, cameras, and Mecanum wheels which cost roughly £750 (Nexus Robot, 2019). The robot, due to the design was also quite slow and very large. This, along with the expensive technology and petrol engine made the system unviable for commercial use (Bonnema, 2012).

Litter picker robots are not the only types of robots which can be compared against to evaluate the effectiveness of cost reduction; this next section critically evaluates the implementations from a number of robotic solutions which chare similar aspects from this implementation, but evaluates their implementations towards their own goals, with respect to litter picking.

Autonomous vacuum cleaners have invaded the average household at an ever increasing rate, with approximately 20% of all vacuum cleaners being autonomous, and Roomba taking 70% of this (with over 14 million sales), according to iRobot CEO Colin Angle in an interview at TechCrunch Beijing 2016 (Angle, 2016). The Roombas themselves can range from $50 to $1500 depending on the level of complexity in the system. Lower cost Roombas generally have very little in terms of localisation and in turn, are quite simple in their operation, following a basic premise of bumping into walls as their main mechanism for turning and exploration. Higher cost Roombas on the other hand are designed to work with effective mapping systems which learn the layout of the room, detect when regions of the room have been visited and learn to focus on areas which are more prone to dirt (Layton, 2005). The basic Roomba offers a small set of components integrated well together, including small IR sensors to act as cliff detection systems (designed to detect the distance to the ground and respond to changes), it also includes a basic geared motor for each wheel, and a bump detection system.

The components included within the low-end Roomba does not change too much as the cost increases, with the only real change to the control board, and its navigation system. The complexity of iRobot's Roomba 400 at $159.95 (£121.78) (iRobot, undated) is very simple compared with the complexity of the low-cost device built for this project at just over £80. The difference in complexity is compared here as an estimation to how cheap the development could potentially be, as by comparing the costs of these systems, a simple evaluation can be performed on the effectiveness of the research carried out into decreasing the price for autonomous systems.

Maintenance:

Evaluation of effectiveness of reducing necessity of maintenance; the IEEE (ISO and IEC and IEEE, 2010) gives the term "Maintenance" three definitions which can be simply described as: Modifying a system to correct faults; Repairing a system to restore its abilities, and; Updating a system to ensure working dependencies. The system design has focused on reducing the impact of these definitions as much as possible. Each of these definitions had been addressed at the start of the system design and had impacted the development in many ways. The benefit of their implementation was clear however in the development.

The first of the definitions (fault correction) was in part managed through ensuring clarity in the code by following the extreme programming practices (Altexsoft, 2018). Practices relating to shared understanding were used to ensure the code itself was easily understood, and easy for maintenance from individuals without direct guidance from the developer, the practices themselves which were followed were simple design, coding standards and system metaphor.

This was only part of the effort to ensure stability and high fault tolerance, with additional measures included specifically for the remote parts of the network, this being the camera and the robot, where updating the scripts on these systems would require retrieval from fixed positions, in order to update the data on the devices. Instead, scripts were set up to enable simple updating remotely through SSH and downloading any new packages or updated files through git. Consideration was made to do this automatically, however without a network tunnelling protocol set up with the system, the devices would have to be connected to the internet, which would severely impact the security of the system. Including remote maintenance in any form, allows the system manager to remotely access, monitor and fix any potential problems which could occur in practice, decreasing complex maintenance costs and the need for developing easy access for what are designed to be permanent fixtures to the environment placed in.

The second definition (functionality restoration), was also a major consideration, not necessarily in terms of lowering the maintenance however; but in terms of simplifying the development process and costs of the system. For instance, developing and building the robot and camera with simple, off the shelf components, decreased the cost and made it simplified the process of replacing parts and testing new fixtures. In an industry implementation this type of setup would allow for a lower level of competence and training with the system to repair and manage the setup, while also enabling the system manager to make quick and cheap repairs using less conventional parts if needed. The code itself, as it is all self-contained, has no points which could require maintenance in this form.

The third definition (dependency correction) within this project did not have much of an impact; the definition mostly relates to managing dependency issues, such as updating the system to work with new libraries and APIs. As the system is fully internalised and there is not much reliance on external libraries, there is no serious impact of using out of date libraries, aside from where libraries are updated due to major bugs found the only libraries which this could affect however is Rospy, where additional security updates could offer advanced security protection; however as long as access to the peripheral's network is managed well, this should never hold any issue. There is only one exception to this, with the Google Cloud API connection, as if there is any changes to the setup of this connection the code will have to be updated to manage this change; however as this is isolated to a single file on the server this is quite well-handled.

REVIEW Chapter 7: Concluding Statements

The aim of the project was clear, to develop a low-cost, low-maintenance autonomous system to retrieve litter. Based on the metric evaluation above, the system design has been effective in ensuring the system is low-maintenance, and low-cost. The system's costs and maintenance systems have been refined through the development and critical evaluation. The system has proven itself throughout the development to achieve the aim.

The design of the system, as shown through the comparisons above, achieves a similar level of quality (with respect to development time and resources) to that of existing research and commercial systems. All in all, the project seems to have met the aim quite well.

//NOT STARTED Chapter 8: Changes to Development

Changes or amendments that may be required to the original delivered artefact should be discussed here, pointing out how and why these changes might have been affected if time or opportunity presented itself.

LAYOUT Part D: Reflective Analysis

Finally, the report should conclude with a critical reflection on the process of completing the project. How did things go?
What might have been done differently, given 20:20 hindsight?
What went well and why?
What went badly, why was that and how were any problems addressed?
What more could have been done, had time and circumstances not been constraints?
Consideration of "theory vs practice" in terms of methodological process requires discussion.

NOT STARTED Chapter 9: Review of Method

What points need to be brought up?

Talk about time management:
Talk about how the Gantt chart at the end was really inaccurate
Talk about how the Gantt chart could have been better
Talk about how using weekly objectives and a weekly review, helped keep the project somewhat on track

Talk about why the time management fucked up; such as:
1. The use of the RC car, along with the breadboarding failures AND how that impacted the decisions in the rest of the project
 (e.g. time became a more important consideration with the implementation after so much was wasted)
2. With hindsight, it may have been better to ask for guidance from colleagues with experience in electronics
3. I want to bring up the time benefit of the cloud API, but also the caveats which come with it such as security and cost

Costs:
The lack of specific planning and the abundance of testing different methods meant there were many costs associated with the research

Redundancy:

The biggest issue with the system design is redundancy, there is none yada yada yada….

Consideration must be made when working on autonomous systems to the wider impact of the work, especially to changes in the structure of society… don't forget to mention the ethics of peeps losing jobs… its is a consideration which must be made in criticism to the development of a system such as this.

WWW?

I'm sure there were some things which went unexpectedly well, and for those things im sure there was a reason why it went well?

ROS implementation? This went very smoothly due to the available guidance in the form of setup tutorials and generally well-detailed information about the systems which govern it.

LAYOUT Chapter 10: Further Research / Research Limitations

What were the limitations on this project?
Further development would remove Google Cloud API for DCNN
Further development would remove MATLAB for Python
Resources made building the robot frame from scratch unrealistic, so more costs; this would be changed
Own knowledge in electronics; with more experience and knowledge, building the MotoZero would be possible, reducing costs

While not implemented fully due to the cost limitations in off-the-shelf components, the development of the robot was planned to use wireless charging capabilities to ensure less of a requirement towards replacing batteries

Stuff:
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.

Cost evaluation metrics reliant on what people should want to pay… this isn't clear enough so data collection needed for autonomous systems appropriate cost, then robot should be designed with this as the target….

LAYOUT Chapter 11: Theory vs Practice in Software Engineering Methodology

How did I plan to follow methodology and how did I actually follow it?

What is adaptive waterfall strict about, and what strict components did I not follow?

Plan was to use an adaptive waterfall, In actuality development more closely followed personal scrum [https://www.infoq.com/news/2015/02/personal-scrum]

As the system was very modular, the development worked as each sub system of development acted as its own sprint. The scrum style daily goals were used as weekly goals as this project was not the only thing which required work each week. For each new module a small plan/list of items which needed to be developed for that module, which acted as a pseudo scrum board.

There was not employed, a very of clear SE methodology structuring for the development as the development was very reliant on the results of research and testing, however the tools which were used were employed to assist when needed, and to give an overview of the development. As this was a personal scrum, and there was little to no outside influence; enabling outsiders to see the development progress was not as important, which contributed to the reasoning of a loosely structured methodology in practice.

Part E: References

ActiveWizards (2018) *Comparison of the Top Cloud APIs for Computer Vision* [blog]. Available from: https://activewizards.com/blog/comparison-of-the-top-cloud-apis-for-computer-vision/ [accessed 7 January 2019].

Ali, U. and Mahmood, M. (2018) Analysis of Blur Measure Operators for Single Image Blur Segmentation. Applied Sciences, 8(5), 807.

Altexsoft (2018) *Extreme Programming: Values, Principles, and Practices.* Atlanta: Altexsoft.com. Available from https://www.altexsoft.com/blog/business/extreme-programming-values-principles-and-practices/ [accessed 17 January 2019].

Anandan, T., M., (2015) *Calculating Your ROI for Robotic Automation Cost vs Cash Flow.* Michigan: Robotic Industries Association. Available from https://www.robotics.org/content-detail.cfm/Industrial-Robotics-Industry-Insights/Calculating-Your-ROI-for-Robotic-Automation-Cost-vs-Cash-Flow/content_id/5285

Angle, C. (2016) *TechCrunch Beijing 2016*. Interviewed by D. Etherington, 18 April. Available from http://tcrn.ch/2fg2ffp [accessed 13 December 2018]

Bonnema, G.M. (2012). System design of a litter collecting robot. Procedia computer science, 8, 479-484.

Christensen, H. (2014) *Confluence of robotics and automation for manufacturing* [lecture]. Independent Study, GeorgiaTech Institute for Robotics and Intelligent Machines, Available from http://cse.umn.edu/x_hosted/mndrive/mndrive_christensen.pdf [accessed 19 September 2018].

Darling, K. (2015) *Children Beating Up Robot Inspires New Escape Manoeuvre System.* IEEE Spectrum, 6 August 2015. Available from https://spectrum.ieee.org/automaton/robotics/artificial-intelligence/children-beating-up-robot [accessed 2 October 2018]

Davison, A.J., Reid, I.D., Molton, N.D. and Stasse, O. (2007). MonoSLAM: Real-time single camera SLAM. IEEE Transactions on Pattern Analysis & Machine Intelligence, (6), 1052-1067.

Dresscher, D. (2010) *An environmental sensor system for an autonomous litter collecting robot.* University of Twente. Available from https://www.ram.ewi.utwente.nl/aigaion/attachments/single/988 [accessed 5 September 2018]

Elkady, A. and Sobh, T., (2018) Robotics middleware: A comprehensive literature survey and attribute-based bibliography. Journal of Robotics, 2012.

Gove, M. (2018) *Plastic bag sales in 'big seven' supermarkets down 86% since 5p charge.* Gov.uk. Available from https://www.gov.uk/government/news/plastic-bag-sales-in-big-seven-supermarkets-down-86-since-5p-charge [accessed 2 January 2019].

Gupta, V., Chaurasia, V. and Shandilya, M. (2015) Random-valued impulse noise removal using adaptive dual threshold median filter. Journal of visual communication and image representation, 26, 296-304

Hamilton, I., A. (2018) *People kicking these food delivery robots is an early insight into how cruel humans could be to robots.* SFGate, 9 June 2018. Available from https://www.sfgate.com/technology/businessinsider/article/People-kicking-these-food-delivery-robots-is-an-12980712.php [accessed 13 January 2018].

Howarth, D. (2016) *Dezeen, McDonald's launches "striking and in-your-face" packaging designed by Boxer*. London: Dezeen. Available from https://www.dezeen.com/2016/01/11/mcdonalds-packaging-rebrand-boxer-fast-food-graphic-design/ [accessed 12 March 2019].

iRobot (undated*) Amazon.com - IROBOT ROOMBA 400 VACUUM CLEANING ROBOT - Household Robotic Vacuums.* London: Amazon UK. Available from https://www.amazon.com/IROBOT-ROOMBA-VACUUM-CLEANING-ROBOT/dp/B000LF6K9Y [accessed 2 December 2018].

ISO and IEC and IEEE (2010) *Systems and software engineering — Vocabulary*. ISO/IEC/IEEE 24765:2010(E). New York, NY, USA: IEEE. Available from https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5733835 [accessed 26 February 2019].

KeepBritainTidy (2018) *KeepBritainTidy, Litter in England: The Local Environmental Quality Survey of England 2017/18.* England: KeepBritainTidy. Available from https://www.keepbritaintidy.org/sites/default/files/resource/National%20Litter%20Survey%20201718_0.pdf [accessed 10 September 2018].

Layton, J. (2005) *How Robotic Vacuums Work.* Atlanta: HowStuffWorks.com. Available from https://electronics.howstuffworks.com/gadgets/home/robotic-vacuum1.htm [accessed 13 October 2018].

Leung, H.K. and White, L., (1991) A cost model to compare regression test strategies. In Proceedings. Conference on Software Maintenance 1991 201-208. IEEE.

MATLAB (undated) *Image Processing and Computer Vision - MATLAB & Simulink Solutions - MATLAB & Simulink*. Cambridge: MATLAB. Available https://uk.mathworks.com/solutions/image-video-processing.html [accessed 16 September 2018].

McCormick, E. (2017) *Big Brother on wheels? Fired security robot divides local homeless people.* The Guardian, 17 December 2017. Available from https://www.theguardian.com/us-news/2017/dec/16/san-francisco-homeless-robot [accessed 21 October 2018]

Mecademic (undated) *Meca500, the world's smallest six-axis industrial robot arm.* Québec: Mecademic. Available from https://www.mecademic.com/products/Meca500-small-robot-arm [accessed 29 October 2018].

Nexus Robot (2019) *254mm Steel Mecanum Wheel Set (2x Left, 2x Right) - RobotShop*. Québec: Robot Shop. Available from https://www.robotshop.com/uk/254mm-steel-mecanum-wheel-set-2x-left-2x-right.html?gclid=CjwKCAjwhbHlBRAMEiwAoDA340sEDOI09c8X745zR-6qReDzmPxJmmoIWXnNgCc5aHyNfUg7upjbFxoCrMEQAvD_BwE [accessed 5 March 2019].

Nishida, T., Takemura, Y., Fuchikawa, Y., Kurogi, S., Ito, S., Obata, M., Hiratsuka, N., Miyagawa, H., Watanabe, Y., Koga, F. and Suehiro, T., (2006) *Development of outdoor service robots.* In 2006 SICE-ICASE International Joint Conference, October. Busan, Korea: IEEE, 2052-2057. Available from https://ieeexplore-ieee-org.proxy.library.lincoln.ac.uk/stamp/stamp.jsp?tp=&arnumber=4109025 [accesses 4 February 2019]

OTTFF (undated) *Amazon.com: OTTFF Robot Tank Chassis Track Arduino Tank Chassis Raspberry DIY STEM - Speed of 20m / min Maximum Load 2kg: Toys & Games*. London: Amazon UK. Available from https://www.amazon.com/OTTFF-Robot-Chassis-Arduino-

Raspberry/dp/B07C2Q63XG/ref=pd_day0_hl_0_4/143-1717844-3250913?_encoding=UTF8&pd_rd_i=B07C2Q63XG&pd_rd_r=646d8fe8-5c33-11e9-b29f-6f56b9d52ae1&pd_rd_w=EsLjM&pd_rd_wg=5mlWO&pf_rd_p=ad07871c-e646-4161-82c7-5ed0d4c85b07&pf_rd_r=17241FMTPKYH04VFS3NJ&psc=1&refRID=17241FMTPKYH04VFS3NJ [accessed 12 September 2018].

RaspberryPi (undated) *FTP - Raspberry Pi Documentation*. Cambridge: RaspberryPi. Available from https://www.raspberrypi.org/documentation/remote-access/ftp.md [accessed 7 October 2018].

Rodríguez-Lera, F.J., Matellán-Olivera, V., Balsa-Comerón, J., Guerrero-Higueras, Á.M. and Fernández-Llamas, C., (2018) Message encryption in robot operating system: Collateral effects of hardening mobile robots. Frontiers in ICT, 5, 2.

ROS (2018) Security – ROS Wiki. Stanford: ROS. Available from http://wiki.ros.org/Security [accessed 15 January 2019].

Rowe, M. (2019) *Britain's growing litter problem: why is it so bad and how to take action.* Countryfile. Available from https://www.countryfile.com/news/britains-growing-litter-problem-why-is-it-so-bad-and-how-to-take-action/ [accessed 26 June 2019].

Shim, J. and Cho, Y., (2016). A mobile robot localization via indoor fixed remote surveillance cameras. Sensors, 16(2), 195.

Singh, S., Prasad, A., Srivastava, K., Bhattacharya, S. (2017) Empirical Evaluation of Edge based Background Subtraction Methods for Object Detection in Video Surveillance System. International Journal of Applied Engineering Research, 12(22) 12036-12043

Sun, L., Zhao, C., Yan, Z., Liu, P., Duckett, T. and Stolkin, R. (2018) A Novel Weakly-supervised approach for RGB-D-based Nuclear Waste Object Detection and Categorization. IEEE Sensors Journal.

Tehan, R., Jackson, L., Jeffers, H., Burns, T. (2015) Beacons of litter: A social experiment to understand how the presence of certain littered items influences rates of littering. Journal of Litter and Environmental Quality, 1(1) 5-15.

White, J., B. (2018) *Self-driving cars attacked by angry San Francisco residents.* Independent, 7 March. Available from https://www.independent.co.uk/news/world/americas/san-francisco-driverless-cars-autonomous-vehicles-attacks-a8243081.html [accessed 15 February 2019].