Implementation and Design of a Multi-Robot System For Wildfire Prevention and Control

Oliver Funk

Prepared for Prof. A. Mishra

Department of Electrical Engineering
University of Cape Town

Submitted to the Department of Electrical Engineering at the University of Cape Town in partial fulfillment of the academic requirements for a Bachelor of Science degree in Mechatronics Engineering.

June 2020

I dedicate the thesis to my mother, for her never ending support and love.

Declaration

1. I know that plagiarism is wrong. Plagiarism is to use another's work and pretend that it is one's own.
2. I have used the IEEE convention for citation and referencing. Each contribution to, and quotation in, this report from the work(s) of other people has been attributed, and has been cited and referenced.
3. This report is my own work.
4. I have not allowed, and will not allow, anyone to copy my work with the intention of passing it off as their own work or part thereof.

Signature:

Oliver Funk

June 2020

Acknowledgements

I would like to thank my thesis supervision, Amit Mishra, for his kind approach to guiding me on my way through the engineering design process and for always be willing to help when I needed it. I would to thank Josh Hewistion for staying up with me all night and helping me film my robots in actions.

Abstract

This thesis served as an investigation into the usefulness of robotics for wildfire control and prevention, with the focus being on a system of multiple robots. The algorithms needed to control Multi-Robot Systems (MRS) were thoroughly explored and implemented on a physical robot, which were designed and built to comply with a specified derived from a user requirement, with the key objectives of the required system being:

- The ability to act automatically, based on its inputs
- The ability to adapt to changes in the fire's position
- The ability to avoid obstacles
- The ability to seek a fire in an environment

The behavior the built and designed robot was partially successful. It showed a robustness to changes in its environment and proved the concept of its operation. However, more work was needed on the control algorithms in order to truly satisfy the user requirement.

The system can been seen as a successful first step toward engineering a solution to the very difficult problem of real-time wildfire monitoring and showed the possibly of MRS in this regard.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

1.1 Background and research motivation

Robotic systems, being completely autonomous or controllable by an operator, are being actively developed by companies such as Siemens, Honda, Lockheed Martin and Boston Dynamics and being are being applied in fields such a process automation, military, remote surgery, remote industrial maintenance, to mention a few. Reliability is one of the most important characteristic of these robotics systems; they must work as intended and perform tasks as expected. Multi-Robot Systems have demonstrated a robustness and reliability in dynamic and unpredictable environments, as they are inherently adaptable and can tolerate failures of one or more robots, unlike single robot systems. MRS generally consist of simple robots, that can perform a small set of tasks well and work together to achieve goals single robot system cannot. MRS have become one of the most researched areas in the robotics field [4] and with this progress, more and more aspects of MRS have been explored, resulting new applications to areas such as autonomous sensor networks, search and rescue missions, environmental surveillance and forest fire detection and control [4]. However, the control and coordination of multiple robots proves to be challenging and is an active area of research, requiring inter-disciplinary approaches from fields such System and Control Theory, Distributed Artificial Intelligence, Biology and Psychology, to mention a few [5, 4].

In the past three to five years, reports have shown an increase in the frequency and severity of wildfire occurrences across the world [6–9]. Many resources cite global warming and global climate change, which result in extreme weather conditions such as prolonged droughts, as one of the key factors that have caused these increases. Although natural to some parts of world, wildfires can be an extremely destructive force, destroying the

habitats of animals and plants and causing millions in damages. New, innovative methods and systems need to be developed to better manage these fires.

The more information firefighters have regarding the location and nature of a fire, the better their decisions will be regarding how and where to control the fire. Systems that give firefighters information about a fire, in real-time, would aid their ability to fight the fire. The adaptability of MRS and the dynamic and unpredictable nature of wildfires play hand in hand together. Therefore, this thesis aims to explore and prototype ways in which MRS can aid firefighters monitor and control wildfires.

1.2 Objective

The objective of this study was phrased as a user requirement:

A system is needed that can find, surround and monitor a fire, in order to report important information about the fire back to central control server.

With the key objectives of the system being:

- The ability to act automatically, based on its inputs
- The ability to adapt to changes in the fire's position
- The ability to avoid obstacles
- The ability to seek a fire in an environment

1.3 Scope and limitations

This thesis includes the an analysis of the needs of firefighters and the design process of Multi-Robot System that could potentially meet those needs, which was implemented and tested

However, due to limited time and budget, only one robots was built. The fire was simulated using light bulbs and were in an enclosure made from planks of wood. The only information that was gathered about the 'fire' was the intensity of the fire as measured by a light dependent resistor (LDR). A major limitation was the inability to implement any kind of self-localisation, or SLAM, algorithm on the robots, due to the time constraints along with the complexity of the implementation.

1.4 Plan of development

Chapter 2 of this thesis presents information from previous works on the topics of wildfires, firefighters and Multi-Robot Systems. **Chapter 3** details the entire engineering design process that was involved in create the robot and the system as a whole. **Chapter 4** gives the implementation details for the designed robot. It also states the verification testing procedures followed, **Chapter 5** containing the results of said tests.

Chapter 2

Literature Review

2.1 The complex problem of Wildfires

Wild (or veld) fires are important ecological events for many of the world's ecosystems and form a natural part of the ecosystem in the Western Cape [10, 11]. Plant life found in environments prone to fires have evolved fire-adaptive traits that help the plant reproduce. These plants are known as pyrophylic, or fire-loving, vegetation and fynbos is one of them [12]. Around 70% of the ecosystems covering South Africa are fire-adapted and therefore need to burn in order to maintain their ecological integrity [13]. Human activity in these fire-adapted ecosystems has created a situation in which natural wildfires pose a risk to property and human life. Therefore, managers of fire-prone ecosystems face the problem of reducing the risk of damaging wildfires, while simultaneously ensuring that fire continues to play its vital role in maintaining healthy ecosystems [14].

Wildfires can cause permanent damage to sensitive habitats - areas in which a species population is limited to – leaving them unable to regenerate to the same extent as before [15]. Additionally, a fire damaged area can experience severe soil erosion, further hindering the ecological regenerative process [11]. Due to the damage caused, "fire and fire suppression" is listed as one of the items on the IUCN Red List of major threats [16]. Studies also show that wildfires are a cause of respiratory and other serious health issue in humans [17–19]. Globally, reports have shown an increase in the number of wildfires occurring year on year [6–8]. Extreme fire events in Australia, Alaska, Indonesia, Canada, California, Spain, Chile and Portugal have occurred over the past three years (from 2015 to present), causing uncountable damage [6]. Locally, there is also evidence to suggest that the frequency of wildfires occurring is increasing [9]. In 2015 and the beginning of 2017, a series of wildfires took hold of the southern Cape region [20], causing an estimated R200

million in damages [21]. More recently, in June 2017, several large fires swept through the Knysna area, causing massive damage to the environment, property and homes, killing four people [22].

Many resources cite global warming and global climate change as a major factor causing the increase in the severity and frequency of wildfires [15, 7, 23, 9]. More frequent and prolonged dry spells, with higher temperatures, increase the likelihood of a fire starting as well as the amount of burnable fuel. The more fuel fires have, the hotter they burn, causing more damage to the environment, which makes it more difficult for the environment to regenerate [15]. More fires cause more pollution and thus compound the effect of global warming, feeding back into the cycle. Invasive alien plants, arson and building properties too close to natural borders also contribute to the increase in the number of wildfires. Prescribed burning reduces the likelihood of wildfires starting, as the dead plant matter is burnt away, but it can damage the environment if the fire becomes uncontrollable or is done as a matter of policy, without scientific reasoning [15].

2.2 The dynamics of fire

A fire, as defined by the NFPA 921, is "a rapid oxidation process, which is a chemical reaction resulting in the evolution of light and heat in varying intensities" [24]. Heat transfer in fires occur via convection (embers etc), conduction and radiation [24]. A fire spreads using conduction and radiation heat energy and can jump across gaps using the convection heat of embers and the wind [24]. If there is an abundance of fuel, fires can spread at rates between 10 to 20 km/hour [25], with most woods begin smouldering at 380 °C and ignite at around 590 °C [25].

Wildfires can be classified into three groups by the ways in which they consume fuel [10]. Surface fires spread by consuming fuels close to the ground, such as grasses or dead leaves and stems. Ground fires burn soils that are rich in organic matter. Crown fires burn in the canopies of trees and shrubs. Most wildfires are a mix of three.

The intensity of a wildfire is classified by measuring the flame length, the rate of spread or the fire line intensity, which is the energy per length of the fire front [10].

Fires spread in a V-shape, from the origin, outwards, heading in the direction of the wind. They generally burn up mountains and through valleys, as the radiant energy from the flames heat the vegetation above it. The hot embers released from a wildfire can damage areas far away that are not currently burning. It is therefore crucial to control the

direction the fire burns, to minimize the damage done to property and land, directly or indirectly [10].

Understanding the dynamics of a fire can help firefighters predict what the fire will do next, giving them better situational awareness of the current fire context.

2.3 The needs of firefighters

Fighting wildfires can be a daunting process, with some fire fronts stretching for hundreds of kilometres and temperatures reaching 800 °C [25]. Therefore, firefighters adhere to a strict, hierarchical chain of command, with captains leading ground teams, reporting to a commander who receives information from a control centre [24]. Any assistive technology would need to fit into this hierarchy. Its behaviour would also need to be understood completely at any point in time and it should be able to shut off if needed.

Firefighters may be exposed to heat stress, fatigue, inhalation of toxic chemicals, carbon monoxide poisoning, burns and physical injuries [26]. They frequently find themselves in situations where they might not be able to see very well and need to plan a course of action. The mental pressure from situations like these often leads to hyper stress and can make the firefighter loss control of a situation [26]. Additionally, heat stress mixed with fatigue can also lead to a firefighter falling unconscious in a dangerous situation, so frequent breaks are taken and firefighters are rotated in and out of duty often [26]. Firefighters therefore carry supplies such as water with them at all times, as well as wear their heavy personal protective equipment, thus they need to be physically and mentally fit and strong [26].

2.4 Firefighting methods

Fighting wildfires differs in each country [27]. In South Africa, when fighting a wild fire, ground teams try to extinguish the fire on its flanks first, gradually moving into the centre, in a pincer movement. Water and fire-retardant chemicals are used to cool and extinguish the fire, while smaller areas are beaten to snuff out the flames [13]. Bulldozers and other heavy equipment can also be used to clear potential fuel away. Helicopters fly over the fire and dump water carried in bucket filled from a nearby water source onto the fire [13]. Helicopters can also use their aerial view to relay information about the fire back to the ground teams, which gives them a better understanding of the fire, but at night, it is not possible to do so [27].

Firefighters also try to push the fire into natural fire breaks, such as roads, streams or rocky fields. The backburning technique is sometimes employed to create a fire break using a controlled fire, downwind of the main fire [13].

After a fire has burnt through an area and it has been extinguished, the brunt area must be ploughed by bulldozers or raked by the firefighters to prevent the fire from flaring up again, as fires can burn the roots of trees and plants and can go underground, smouldering for days. The burnt area must then be plotted, mapped and the data must be recorded [27].

In urban environments (and unlike wild, open environments), firefighters have to deal with people, high density buildings and confined spaces. Finding a way to access the fire in these types of environments can prove to be difficult, especially in an informal settlement where the infrastructure and resources needed to fight the fire are not necessarily present. Large fire trucks are therefore needed but the very confined spaces pose a significant problem [28].

2.5 Existing Fire Prevention and Control Systems

Currently, there are a number of technologies that exist that attempt to predict, control and prevent fires from occurring, a few of them are detailed below.

The Wildland Detection System use a network of digital cameras to monitor an area for signs of smoke and fire. They use image processing techniques to achieve this and are a proven early warning system that helps prevent fires becoming uncontrollable. However, the system is limited. It must operate from a fixed position, only fires that are in line-of-sight of a camera will be detected and only optical information is analysed (WDS).

The work by Pastor et. al. ([29]) details the technology and methodology needed to improve the detection, control and analysis of wildfires, as well as for post-fire hot-spot detection, using helicopters during the day or night. The method includes equipping thermal cameras to helicopters that can precisely detect where heat is being generated from. The technology allows the helicopter to fly over an area quickly, scanning the area and then sending the information to a ground control node, which then processes it. After an analysis is performed, a report is sent to the relevant persons, detailing where hot-spots are. The authors say that an improvement to the speed at which fire can be detected, directly reduces the time it takes to stop the fire.

A prominent drone system that has seen use in the area of firefighting, is the SkyFire UAS service. The drone is used to assess scenes quickly, allowing more time for planning

and eliminating the need to put people in danger to understand the situation at hand. On-board thermal cameras provide additional informational and a unique vantage point to a commander, who can then make a more informed decision about how to approach the situation. They claim it is a cheaper and more efficient way to identify threats and that drones will soon become a stand part of firefighting equipment. Companies such as Lockheed Martin and Parrot also make drones that can be used for firefighting.

Satellites are used to capture images of regions that are exposed to frequent fires. MODIS, SPOT, AVHRR and Landsat are some of the satellite sources that are available. Models are used to try and predict where a fire may start, by analysing Normalized Difference Vegetation Index (NDVI) and other such metrics [30]. Visible and infrared light can provide information relating to the moisture and vegetation content of an area [30]. Images captured of an area before and after a fire has burnt through it allow research to test their fire prediction models, as they know what the end result should be [30].

2.6 Adaptable Systems and Agent-Based Modelling

To solve many of the complex challenges that occur in the real world and outside of the lab, engineered systems need to be designed with adaptability in mind. The assumption must be made that it is not possible (or at least not feasibly possible) to simulate every situation a system may encounter or all contexts and environments in which it may find itself in. Additionally, truly robust systems need to be able to adapt to internal changes in state. If some part of the system breaks down, it should be able to still perform the task at hand, though at some reduced measure of its success [31]. Therefore, the approach of creating systems that are inherently robust to changes in the environment must be taken, in order to begin successfully solving these real world challenges.

Adaptable system need the ability to effectively react to changes (internal or external) without degrading their performance in accomplishing a set of high-level goals [32]. That is, the functional ability of adaptable systems must be invariant to changes, but their functional success (some measure of the 'amount' it has accomplished its goal) should only scale with the measure of system's total ability to accomplish the goal. Reduced ability means reduced total success, but not an inability to perform. Therefore, adaptable systems have a sense of atomicity inherit to them. In that, the whole must be comprised of multiple internal units, such that if some units fail, the whole remains intact, invariant to internal changes.

These types of systems need to be controlled using decentralized approaches. Nature can be a very good source of inspiration when it comes to controlling many units, in a decentralized manner to accomplish a task. Bees hives, ant and termite colonies and even in some ways human social behavior, are some examples of these 'system' [33], which can be can be viewed as a single entity, performing some task. Importantly, the ability to perform the task does not depend on the functioning of any single unit within the system and the entity can perform the same task in many different environments, robustly adapting to the changes.

Agent-Based Modeling (ABM) is a relatively new approach taken to model the dynamics of these complex adaptive systems. These models are composed of many autonomous, interacting agents that are placed into a simulated environment. Each agent has some kind of sensory input and a set of rules, which together, transform the sensory input into a control signal that is then inputted into an actuator. These models are used to observe the collective effects of agent behaviours and interactions and often self-organize themselves, creating emergent order [34].

2.7 Multi-Agent Systems, Multi-Robot Systems and Swarm Systems

Although these three Agent-Based Systems (ABS) relate significantly to an ABM, they are not always the same. ABMs focus on giving explanatory insight into the behaviour of adaptive, collective systems that already exist and ABS focus more on creating new systems that used agents in them, and are used more in the fields of engineering and economics, where an approach or implementation, not an explanation, is sought after.

In this section, an explanation of these three types of ABS is given.

2.7.1 Multi-Agent Systems

Multi-Agent systems (MAS) defines a broad category of systems that involve some use of agents within them, in which the system is considered to be the collective instead of the individual. Literature offers varied definitions of agents, ranging from simplistic to complex ones. The broadest definition found was given by Russel and Norvig in [35]:

> **An agent is anything that can be viewed as perceiving its environment through sensors and acting upon that environment through effectors**

The agents may be virtual (such as a computer programs) or physical (such as a robot), but both inhabit some kind of environment [35]. Agents have sensors, ways of gathering specific information called Percepts, and Effectors, ways of altering or moving in their environment using Actions [35]. Each agent has some domain knowledge of their environment, the rules they must abide by, and generally have a goal they 'want to' achieve. Each agent must attempt to achieve its goal, bound by some rules, by performing some action using its effector, based on the information perceived using sensors [35].

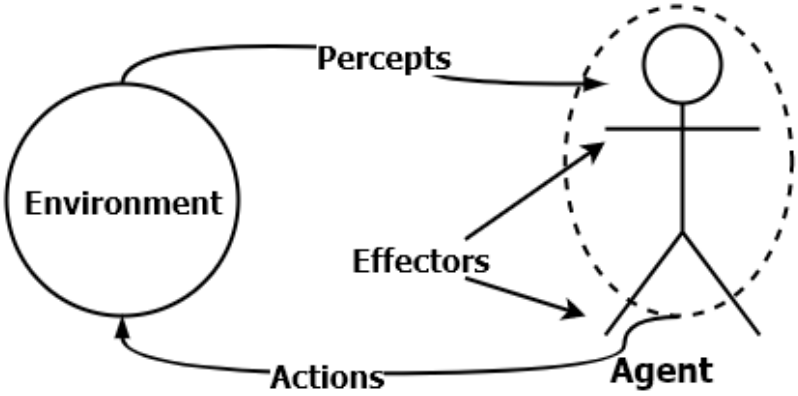


Fig. 2.1 Overview of Multi-Agent Systems, showing an agent interacting with its environment through effectors.

MAS have seen some application in the engineering domain, especially in in the Power Engineering field. Mevludin in [36] models load balancing controllers as agents, that have domain knowledge over the system and can act cooperatively to effectively distribute the load over an area.

MAS are also used in the area of game theory in economics, in which the idea of a rational agent is used. The term describes the idea of an agent that will do whatever action is required to maximize its performance measure, on the basis of what it can currently perceive [35].

MAS can be classified as being either heterogeneous, which include agents that have the same internal structure and domain knowledge (knowledge about the rules), but have different inputs, or homogeneous, which are the same, but every agent has the same input [36].

2.7.2 Multi-Robot Systems

Multi-Robot Systems (MRS) are very similar to heterogeneous MAS, however they cannot simply be considered as a special case of MAS, because of the issues arising when dealing

with a physical environment, such as uncertainty and incompleteness of the information gathered through sampling. MRS need to cope in a far less than ideal world, in which most MAS are designed for, making the experimental evaluation of MRS very challenging [1].

The robots in MRS can individually be complex (be multi-skilled), able to perform a large number of tasks independently and may have more than one sensor array available to them (unlike the simplistic robots in swarm robotic systems) [37]. They always have domain knowledge about their environment and an 'idea' about what the end goal is that they are trying to accomplish. They are *assisted* by the usage of a robot team, but they could (within limitations) enact their desired function and achieve their goal without the help from any other robot.

However, MRS cannot just be seen as a generalization of the single robot case. The approaches taken to design the control algorithms that run on each robot in MRS are characterized by the assumptions made about the environment the system will be placed **and** in terms of the internal organizational architecture of the system that each robot will fit in to (how the robots will interact with each other) [1].

It is generally uncommon for these robots to lack the ability to communicate with an external command node, therefore algorithms designed for MRS can generally utilize this property [37]. There are instances where, possibly due to design constrains or for robustness, these robots do lack external communication abilities, then the algorithms employed tend to be more related to those used in the swarm systems.

Parker in [33] states nearly all the work in cooperative MRS began after the introduction of behaviour based control, based on the model provided by Rodney and Brooks in [2], which detailed the subsumption architecture for robotic systems, which was rooted in biologic inspiration. Many researchers from then on sought to study the social structures of animals and apply their findings to the design of MRS control algorithms. In [38, 3, 1], the authors show the ability of MRS to imitate the flocking of birds, dispersion and aggregation characteristics, search, foraging and following abilities, all using very simple rules. The robustness seen in MRS lines within the simplicity of the rules that control the robots.

The MRS field has seen much growth over the years. This is mostly driven by the technological improvements in both the hardware and the associated software [33]. The availability of complex and accurate sensors as well as the availably of small robotic platforms have resulted in more projects being undertaken, with more complex robots being built, that can achieve a wider verity of tasks more easily. Smaller teams can now afford to equip their robots with laser range finders, cameras, infrared sensors, robotic arms,

gripping devices etc. Techniques for 'basic' behaviours, such as localization, path planning, object transportation, object recognition and tracking, have now been developed and teams can leverage these results to implement MRS and further contribute to the field [39]. The fact that the first IEEE international symposium on Multi-Robot and Multi-Agent systems is being held in December 2017, shows that the field is becoming more influential and important from a global perspective. Therefore, the study and development of MRS applications is particularly relevant and significant at this stage.

2.7.3 Swarm robotics and Swarm systems

Swarm robotics involves controlling a large number of simple robots based on simple rules. It is an approach taken to collective robotics that is inspired from the self-organizational behaviours of social animals, such as bees, termites and ants [40]. Although the rules followed by each robot are very simple, the behaviours that emerge from the collective interactions between each robot in the group can be very complex. An important distinction to make is that a robot in a swarm requires other robots to accomplish the goal or task and it cannot do it alone, unlike robots in MRS. Each robot in the swarm does not know what the end goal or task, but enacts only the simple rules that govern it.

Each robot can only sense and communicate locally; there is no way for them to communicate with a central node or a coordinator. This paradigm is referred to as swarm intelligence and has proven to have very important properties such as robustness, flexibility and the ability to solve complex problems by exploiting the inherent parallelism and self-organization of the group [41].

A well-known implantation of swarm behaviours and intelligence on physical robots is the work by Rubenstein, et. al. [42]. They developed the Kilobot, a small, low cost robot that moves using vibration. The scalability of the Kilobot allowed the team to create 1000 of them and explore swarm robotics in its entirety. The SDASH algorithm was implemented on the swarm as the formation algorithm [43]. It allows a collective of robots to scalability form a shape, without having to know how many robots there in the swarm.

Hamann and Worn in [41] detail many swarm algorithms that can be implemented on physical robotic systems. An interesting method they suggest for random movement, is the use of Brownian motion model and the Langevin equation. This may be a useful approach to take when implementing searching behaviours on robots.

2.8 Comparison between Multi-Robot Systems and a single robot

Lima in [4] suggests that single multi-skilled robots lack robustness and reliability in the face of uncertainty, as robotic systems (being electro-mechanical systems) are bound to fail at some point, for some reason. Since robustness and reliability can often be increased by combining several robots into one system, which is then more robust and reliable than any single robot, they suggest that MRS are better suited to tasks in areas such autonomous sensor networks, building surveillance, transportation of large objects, air and underwater pollution monitoring, forest fire detection, transportation systems, or search and rescue after large-scale disasters. Additionally, Farinelli in [1] suggests that MRS will be used in space and security applications, where robots will need to work without human interaction for a long time, thus requiring a type of system that can handle unpredictable situations and be trusted to execute complex, high level tasks.

2.9 Classification of Multi-Robot Systems

Farinelli in [1] detail a method for classifying MRS based on two metric groups, *Coordination* and *System*. The former is based on the parameters affecting how each robot is coordinated within the system and the latter on is based on system features that influence team development. Table 2.1 shows the two dimensions and the sub-categories within them.

Table 2.1 MRS classification dimensions

Coordination Dimensions	System Dimensions
Cooperation	Communication
Knowledge	Team Composition
Coordination	System Architecture
Organization	Team Size

The *Cooperation* metric is concerned with the ability of the system to cooperate in order to accomplish a specific task. A MRS is cooperative if the robots in the system "operate together to perform some global task". The *Knowledge* metric is a measure of the knowledge that each robot in the system has about its the other robots. Aware robots will

act knowing about the other robots, while unaware robots will act with no such knowledge. The *Coordination* metric is a measure of how strongly the actions performed by each robot take into account the actions perform by their peers, in such a way that the macro behaviour of the system ends up being a coherent and high-performance operation. The robots are strongly coordinated, if they use a communication protocol between them and weakly coordinated if they do not use a communication protocol (only very simple coordination can take place). The *Organization* metric is a measure of how a decision about what action to take next is realised (how coordination takes place). The system is strongly centralized if there is only ever one predefined leader, weakly centralized if leaders can be elected (but there can only be one at any one time) and distributed if there is never a leader.

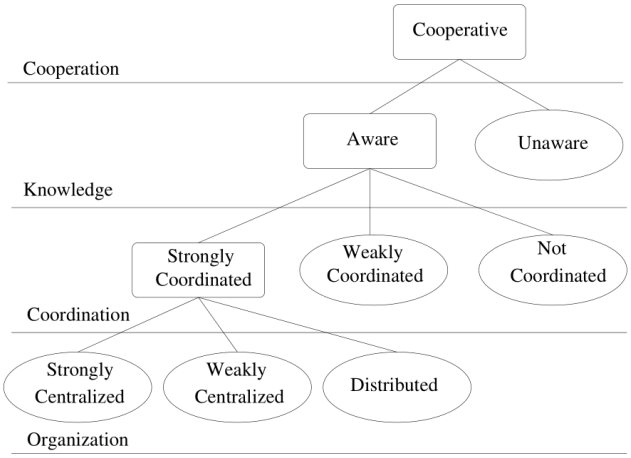


Fig. 2.2 A taxonomy for Multi-Robot Systems based on the work from [1]

It is important to classify different MRS, as the control algorithms that work on one such type may not work on another, therefore one can more successfully apply similar techniques.

2.10 Active Research Problems in the Multi-Robot Systems Field

Some of the most important challenges discussed throughout literature, specific to the MRS field include: ([4, 33, 44])

- The noise and uncertainty introduced during by sensors, pose as a serious challenges to the existing MRS algorithms, as they generally assume the signals that being fed into them are 'perfect'.
- The noisy and limited bandwidth communications among teammates in a cooperative setting, which gets worse as more robots are added .
- The capability of a robot to perceive the surrounding environment correctly and to build models of the environment that are useful for the task at hand.
- Controlling a group of robots using one high-level command issued to them and letting them reach a consensus about how to perform said goal in the most efficient way, automatically without any further instruction.
- Developing distributed controllers for the deployment of mobile sensor networks.

2.11 Multi-Robot control and coordination

MRS are inherently inter-disciplinary, merging the contributions from two fields: Control Theory and Distributed Artificial Intelligence [1]. Distributed Artificial Intelligence focuses on MAS, with a focus on organizational issues, distributed decision making and social relation. Control theory is needed to physically control the actuators on each robot, it serves as the interface from the AI side into the real-world, in which actuators need to be modeled correctly so that they can be controlled properly.

Figure 2.3 shows the well known subsumption architecture as described by Brooks [2]. It is a way to achieve robust control over a robot, using a hierarchical set of layers, in which higher layers take control over the actuators if they start outputting a control signal. Brooks suggested the user of a finite state machine as the main driver of determining which input or outputs should be suppressed at any one time.

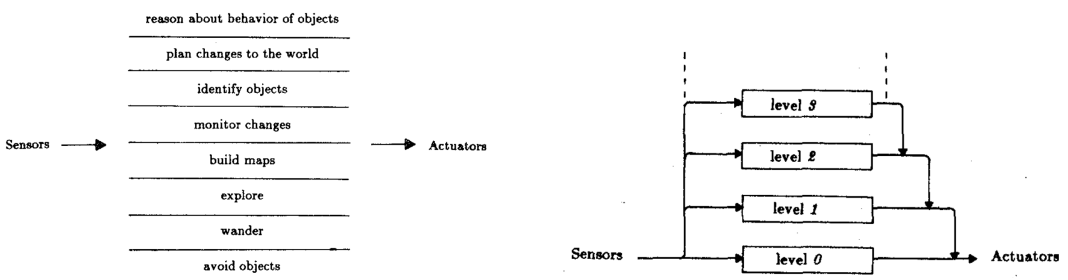


Fig. 2.3 The well known subsumption architecture, as proposed by Brooks. (Source: [2])

An extension to Brooks' idea came in 1998, when a landmark paper [3] published that proposed a fault tolerant method for multi-robot cooperation, dubbed ALLIANCE. It is a software architecture designed for a system of heterogeneous robots that can perform missions, composed of loosely coupled subtasks that may have ordering dependencies (see Figure 2.4). Each team of robots within the system can perform a variety of high-level functions, ALLIANCE then selects the appropriate action each individual robot must do, in order to perform the team level function, which is itself selected by ALLIANCE, in order to optimally accomplish some desired goal. ALLIANCE encodes a mathematical model for motivation. It is a process that governs determines which behaviours are allowed to output their control signal on the actuators (a subsumption architecture). Using the classification framework detailed in section 2.9, ALLIANCE defines a distributed, strongly cooperative, aware MRS.

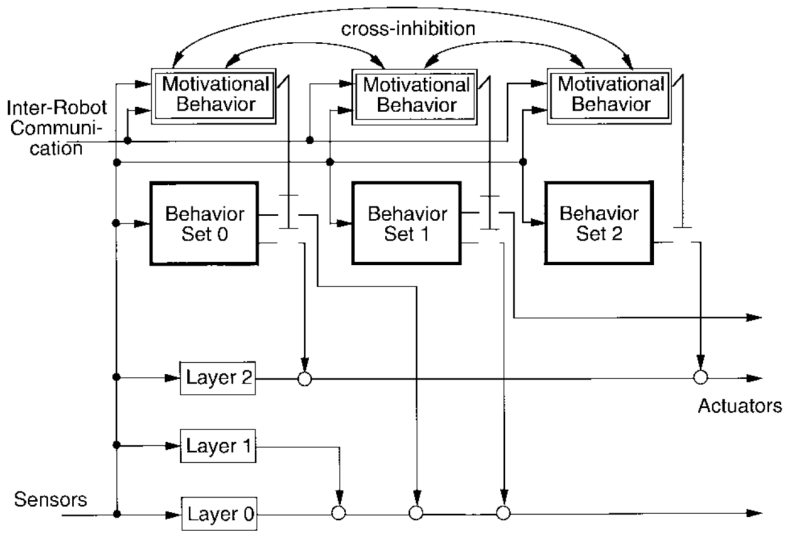


Fig. 2.4 The ALLIANCE software architecture.
(Source: [3])

In [1], the authors propose a sensor fusion method for combining sensory information using Bayesian statistics into useful agreement or disagreement measure, which is then used by control algorithms to enact high-order control on the robots actuators. Uncertainties in the sensor state and observation are modeled by Gaussian distribution. Bayesian observers are used to model the sensors and the observations made N sensors is joined using multiplication of their posterior likelihood. If the joint posterior is greater than all of the senors' posteriors, than an agreement is reached, otherwise there is no agreement.

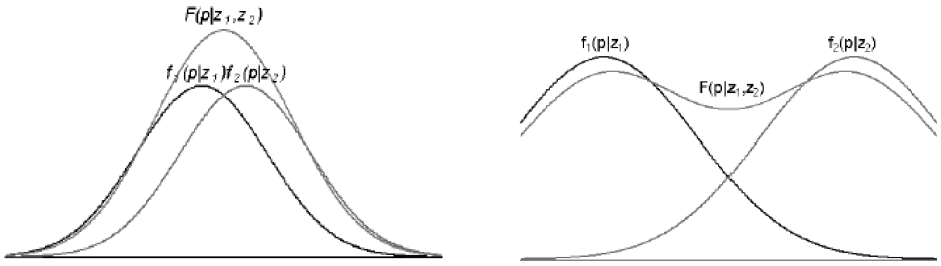


Fig. 2.5 Sensor fusion method using Bayesian observers. *Left* – Showing an agreement measure. *Right* – Showing a disagreement measure
(Source: [1])

In [38], Mataric details the pseudocode for many of the basic MRS algorithms used today, such as collision avoidance, following, dispersion, aggregation, homing and flocking. The criteria for the robots was that they are homogeneous in terms of hardware and software, they do not use explicate peer-to-peer communications, they have no 'hidden' goals (all goals of the system are common to all robots) and they are able to detect other robots.

2.12 Conclusion

Wildfires are costing the world and the country environmentally and economically and it is getting worse every year. They cause the loss of life of civilians, firefighters and animals. Helping control wildfires is a multifaceted problem, with social and economic factors at work. Multi-Robot system have proven to be an effective way to tackle problems that require a system that is robust and adaptable. These traits are desirable when designing a system that must act in such a diverse and challenging environment that wildfires pose. These system would need to help firefighters gain control over a wildfire more rapidly, which if done fewer deaths and economic costs will be incurred, less environmental damage will be done and less pollution will be released into the atmosphere.

Chapter 3

Engineering Design Process

In this thesis, the V-model was used for the engineering design process.

The V-model defines a multi-stage design process. First, the requirements of the system are gathered. They define what the system must do and the specifications it must comply to. A system that can meet the requirements is then proposed, followed by a high level and detailed design of the system. The prosed solution is broken down into subsystems and a system architecture is drawn, which shows how the subsystems connect together. This is followed by a detailing of the hardware (mechanical design, component selection, circuit design) and software (algorithms, flow charts, programs needed, pseudo code) that will be used. At each sub-stage of the V-digram, tests are written that define how the system at that point will be checked for any faults. Each sub-stage is successively more specific than the last, until the system is ready for implementation and testing.

Figure 3.1 shows an overview of the process followed.

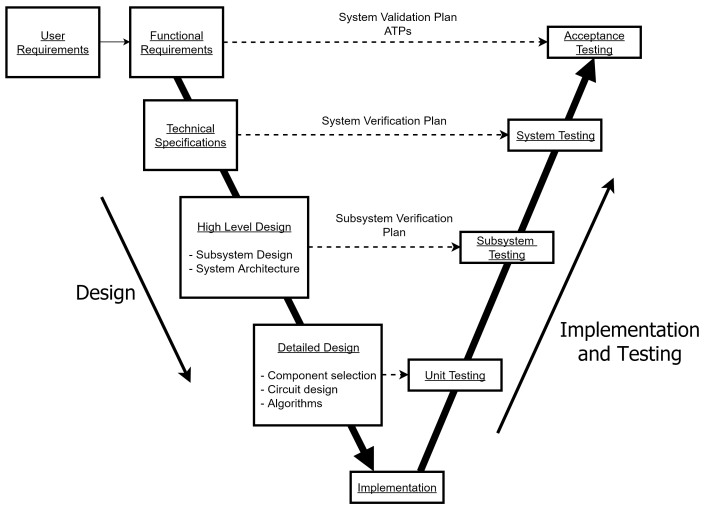


Fig. 3.1 V-diagram Engineering Design Process

3.1 Problem Identification and Requirements Gathering

In this stage of the engineering design process, various requirements the system must meet are identified. The user requirement is stated and analyzed to determine the functions the system must perform. These functions are then translated into technical specifications, which must be measurable and testable, that the built system must comply to. Acceptance tests are then written which define how the system will be tested to check if it satisfies the functional requirements.

3.1.1 User Requirement

The user requirement was stated as:

An autonomous system is needed that can find, surround and monitor a fire, in order to report important information about the fire back to a central control server.

With the key objectives of the system being:

- The ability to act automatically, based on its inputs
- The ability to adapt to changes in the fire's position
- The ability to avoid obstacles
- The ability to seek a fire in an environment

3.1.2 User Requirement Analysis

When dealing with wildfires or urban fires, various challenges are faced by firefighters, most notability having good situational awareness and being able plan effectively under pressure. Situational awareness is the act of knowing what the current dangers and opportunities are in an environment and understand how they relate to the bigger picture or goal at hand. The more information an individual has the better their situational awareness will be, however if too much information is present or the information is not clear, the situational awareness of an individual can be hampered. Relevant and clear information is the most useful. The choice of what to do next is then made by synthesizing all the information available into an actionable plan. Therefore, having better awareness about a situation will directly improve the effectiveness of a plan. Firefighters must have a good sense of their own situation and that of their team. They must know how they relate to the achieving the larger goal.

If the situational awareness of firefighters could be improved, they would be able to better react to environmental changes around them while fighting a fire. This would help reduce the risk of burns or loss of life. At night, the situational awareness of firefighters is dramatically reduced compared to what it is during the day because of reduced light and lack of aerial assistance.

Firefighters are exposed to various hazards and grueling situations. They need to stay alert to remain effective and safe while on duty. Monitoring their physical properties, such as blood sugar levels, oxygen levels, etc, and the properties of their environment, can help a team leader to know when a firefighter needs to take a break.

Early warming systems that detect newly stated fires can provide information, such as the exact location of the fire, where it is most severe and the direction it may be heading, to a central firefighting command center. This could assist firefighters as they could respond to fire emergencies as quickly as possible, knowing exactly where to go to get to the fire and allowing them to create a plan before arriving, giving them time to prepare as a team so they can take action as soon as they arrive.

In urban environments, it can be difficult to know if a water jet is hitting the main parts of a fire, as the fire may be obstructed or it is on top of a tall building, having no clear line-of-sight. In open, wild environments, changing weather conditions can completely change the direction a fire was heading in and dramatically increase its severity. Additionally, knowing the location of other ground teams and where they are in relation to the fire could help in keeping the firefighters safe and effective at fighting the fire. Systems that could

supply local information about the conditions of a fire or provide another vantage point could assist fire firefighting teams and help them overcome these problems.

After a fire has been controlled, the burnt areas are raked and bulldozed to reduce the likelihood of the fire from flaring up again, as fires can burn underground (burning the roots of tree and shrubs) and can smolder for days. However, raking and bulldozing can only do so much, because the heat may be trapped well underground. Potential 'hotspots' in these burnt areas can be difficult to find, given that the area may be very large. Additionally, these burnt areas need to be plotted and mapped, but this process can be slow and inaccurate. An automated system could potentially be used to be improve this process, increasing the speed and accuracy at which it can be done.

Firefighting teams work in strict hierarchical structures, akin to the structure of a military army. Any assistive technology would need to fit into this structure and its behaviour would need to be completely understood and well defined.

Primarily firefighters need to keep people safe. They need to warm them about the dangers of the fire and help them get to safety, but given the unpredictable natural of fire, can make this a challenging task to achieve. A system that can reduce the uncertainty surrounding a fire can reduce the amount of time needed to control the fire, reducing the risk of harm being done to people and property.

3.1.3 Functional Requirements

The following are a summary of the ideal behavioral functions the system must perform to meet needs of the user. These functions, based on the requirement analysis above, are used to derive technical specifications for the system that must be designed and built (see section 3.1.4).

- The system shall autonomously monitor an area for a fire or for conditions that could start a fire
- The system shall surround the fire once found and adapt to changes in its shape
- The system shall sense physical properties about the fire and the environment. These must include sensing the air temperature, the wind speed and direction, the carbon dioxide level and the humidity
- The system shall report this information back to a central station, including the direction the fire is heading in and the severity of the fire at different locations

- The system shall actively avoid situations that could cause it harm
- The system shall shut down and stop quickly when needed

The following are design requirements that must be met by the system, these do not specify any kind of behavior.

- The system shall be robust to heat damage and be energy efficient
- The system shall be designed in a modular way, allowing for easy upgrading and maintenance
- The system shall be made from cheap components to allow for effective scaling
- The system shall be as energy efficient as possible

3.1.4 Technical Specifications

Table 3.1 details the necessary technical specifications for the robotic system.

3.1.5 System Validation Plan

The system validation plan consists of acceptance tests and the procedures needed to perform them, known as Acceptance Test Procedures (ATP's). ATP's are performed once the system is built and ensure that the system satisfies the functional requirements of the user. They prove whether or not the system, as a whole, does what it was designed to do and thus can accepted as being complete.

Acceptance Test 1: Shape of formation

The effectiveness of the system to surround a fire was tested by comparing the centroids of the fire to that of the shape of system's formation (see figure 3.2).

This was done by photographing the formation of the robots while they were surrounding a fire (a group of light bulbs) and computing the centroid of the **Alpha** shape of the formation. The centroid of the fire was computed by identifying where the intensity of the light reaches the threshold value (using a lux meter), doing that multiple times around the fire and creating a closed shape around the fire. Centroids are vectors, thus the magnitude and direction of two can compared.

Table 3.1 Technical Specification for the system

Category		**Description**	**Value**	
Movement	Speed	Movement speed Forwards and backwards	$v \geq 5\,\text{m/s}$	
	Turn Rate	Turn speed Clockwise and counterclockwise	$\omega \geq \pi\,\text{rad/s}$	
External Communications	Range	Communication distance between the central receiver and the robot	$R_{ec} \geq 10\,\text{m}$	
Internal Communications	Range	Communication distance between the robot	$0\,\text{cm} \leq R_{ic} \leq 30\,\text{cm}$	
Light Sensor	Sensing	What must be sensed	$400\,\text{nm} \leq \lambda_{vl} \leq 700\,\text{nm}$	Visable light only
	Sensitivity	Sensor's ability to detect changes	$S = \Delta E_v = 2\,\text{lx}$	Changes in 2 lux must be detectable

Acceptance criteria:

$$\left\| \hat{C}_{fire} - \hat{C}_{form} \right\| \leq 0.5\,\mathrm{m}$$

$$-30° \leq \angle(\hat{C}_{fire}, \hat{C}_{form}) \leq 30°$$

where:

$$\angle(\hat{A}, \hat{B}) = \arccos \frac{\hat{A} \cdot \hat{B}}{\left\| \hat{A} \right\| \cdot \left\| \hat{B} \right\|}$$

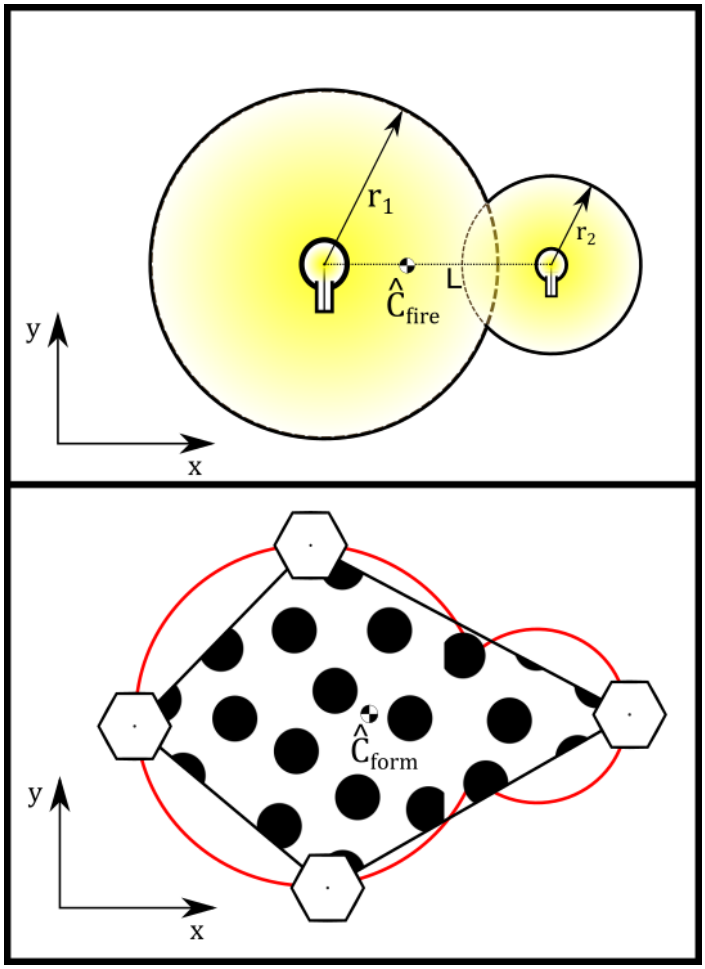


Fig. 3.2 Top, *Centroid of Fire*: The centroid ($\hat{C}_{fire}$) of an illustrative fire made from light bulbs. The black outline shows the intensity radius of each 'fire'. The centroid is closer to the left bulb as it is more intense, i.e. brighter. – Bottom, *Centroid of Formation*: The centroid ($\hat{C}_{form}$) of an example formation Alpha shape, as indicated by the polka dots. The red outline shows the path the robots would follow and that matches the intensity of the fire.

Acceptance Test 2: Speed of Formation

The time it takes for the system to surround a fire was tested by measuring the time taken to find a fire (t_{find}) and the time taken to form around a fire (t_{form}) once found.

The time taken to find (t_{find}) a fire was measured by stating a stopwatch as the system was started and timing how long it took the first robot to search for a find the fire. The moment the first robot begins the surrounding behaviour was the moment the stopwatch was stopped.

The time taken to form (t_{form}) around the fire was measured by putting the robots close enough to the fire such that they can all begin forming and starting a stopwatch as soon as the last robot begins the surround behaviour. The stopwatch was stopped when the criteria for test one was met (see 3.1.5).

Acceptance criteria:

$$t_{find} \leq 5\ minutes$$

$$t_{form} \leq 1\ minute$$

Acceptance Test 3: Reaction to Fire Dynamics

The ability of the system to react to changes in the shape of the fire was tested by measuring the time it took the system to reform around the new fire shape (t_{reform}).

The time taken to surround a new fire shape was measured by starting a stopwatch as soon as a new light bulb is switched on and timing how long it took the system to meet the criteria for test one (see 3.1.5).

Acceptance criteria:

$$t_{reform} \leq 1\ minute$$

3.2 Proposed Solution and Concept of Operation

A high level description

For this thesis, I proposed to use a system of multiple, ground based, wheeled robots, coordinated and controlled using a distributed algorithm (one that can adapt to the number of robots in the system and the shape of the fire), as a first step in developing an understanding of how multi-robot control algorithms and teams of robots could be used to help monitor and control wildfires.

Fire shall be simulated using multiple incandescent light bulbs, with varying brightness, arranged on the floor. The system of robots in this environment and will begin by searching for this 'fire'.

Once a fire is found, the distributed algorithm will begin controlling the robots. There are two simple rules in this algorithm:

1. Each robot must remain a certain distance from the 'fire', whilst moving around it in a certain direction (clockwise or counter-clockwise). This is done by measuring the intensity of the fire by sampling the robot's light sensors and attempting to keep the value at a desired level. If there is only one light bulb (a small 'fire'), the robot shall effectively move in a circle around it, however, if there are multiple light bulbs close enough together, the robot will move in the union of circles of the light bulbs' intensities (see figure 3.2 for a diagrammatic explanation).

2. When two robots meet whilst surrounding a fire, they shall stop and then begin moving in opposite directions from each other (similar to bouncing off one another).

These two rules work together to make the algorithm distributed and scalable. The first rule makes each robot adapt to the shape of the fire, without knowing any global information about it and the second allows the robots to adapt to the number of robots in the system and ensures the fire is always optimally covered by the spread of robots. Additionally, each robot shall spend more time traversing the more intense parts of the fire (indicated by the brightness of the light bulbs), as the radius of the light intensity will be larger than the less intense parts, meaning the robots will take more time to traverse it. This is desired as the more intense parts of the fire need to be monitored for a longer period of time.

The robots shall communicate locally to one another using infrared transmitters (IR LEDs) and receivers with established communication protocols, such as those used by televisions. Infrared light has the advantage that is does not penetrate through solid objects easily and can thus mimic local communication used in natural well. Infrared proximity sensors and communication arrays are also also used commonly in many swarm robotic projects [45–47]. Directional communication is an important trait needed by the control algorithms for multi-robot and swarm systems in order to implement cooperative behaviors and self-organization, as two agents could meet one another at any angle and any obstacle could be approached at any angle. Therefore, multiple IR transmitters and receivers shall be used.

Communication with an central computer shall be achieved using a low power radio transceiver, which could transmit and receive messages over a range sufficient for this project.

Relation to the user requirement

The terms used in the user requirement are qualified against the proposed solution below, to better demonstrate how the requirement will be satisfied:

1. **A system**: The system shall be a system of multiple robots, acting autonomously and collaboratively to achieve a specified task.

2. **Find and surround**: Each robot shall perform two high level tasks, no specific instruction about how to perform the tasks will be given, only the knowledge about what the task is. Each robot shall search their environment for a fire and when found, begin forming around the fire, in a way that effectively surrounds it. The formation algorithm shall adapt to changes in the number of robots forming around the fire and to changes in the fire‘s shape and position. Additionally, each robot shall avoid obstacles that block its path, in other words, avoid objects that stop the robot from performing a task.

3. **Monitor**: Each robot shall gather information pertaining to the state of the fire, in real time.

4. **Important information**: The data gathered by the robots shall be the intensity of and be in a form that is not immediately useful or coherent. Therefore, the data shall be pre-processed into a coherent and useful form.

5. **Central control server**: The pre-processed data shall be sent to a central server that is to use it and present it to a user in a useful form.

3.3 High Level Design

In this stage of the engineering design process, the prosed solution is broken down into various subsystems. A high level description of the function each subsystem performs and a block diagram depicting the components used by each subsystem is given. The overall architectural design of the system is then drawn, showing how the subsystems connect together and the interface used by each. Subsystem verification tests are written as well.

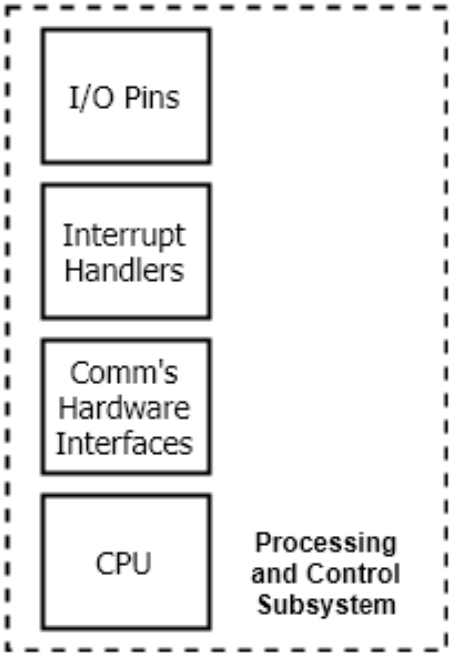


Fig. 3.3 Processing and Control Subsystem

3.3.1 Subsystem Design

Processing and Control Subsystem

The processing and control subsystem will require only one micro-controller. It will be responsible for processing all the sensory information gathered, control the motors of the robot and send and receive messages, locally and externally. It is the central block of the system, through which everything is connected.

See figure 3.3.

Light Sensing Subsystem

The light sensing subsystem will be responsible for converting the illuminance (luminous flux per unit area) incident on the robot into a voltage that can be read by an ADC. This subsystem will be the primary driver of the robot's behaviour, which is based on reactions to statistically significant differences in the current illuminance compared to the average ambient illuminance. A timing circuit will additionally be needed to sample the light sensor(s) at a specific rate.

See figure 3.4.

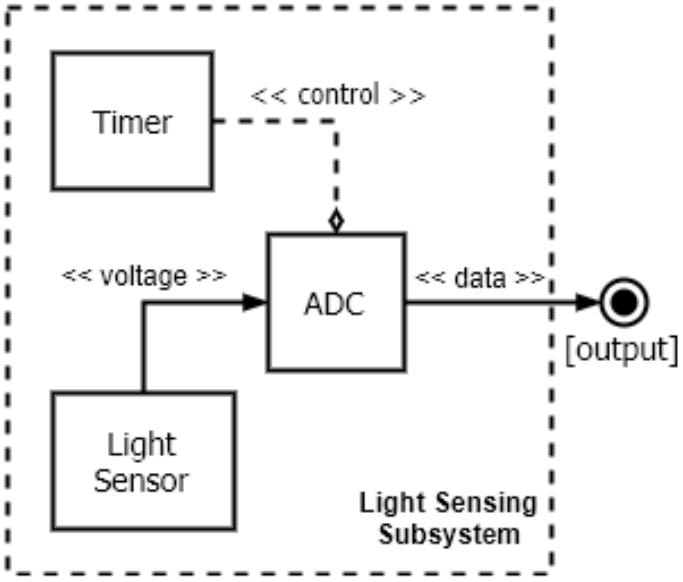


Fig. 3.4 Light Sensing Subsystem

Infrared Receiver Subsystem

The infrared receiver subsystem will be needed to detect signals sent via infrared and convert them into voltages. These voltages will either be a logical true (a high voltage) or logical false (a low voltage). Therefore, edge detection circuitry will be needed to detect the sharp changes in logic levels. A timer will also be needed to decode the signal received.

See figure 3.5.

Infrared Transmitter Subsystem

The infrared transmitter will be needed to transmit a signal via infrared. The signal may contain information that can be used by another robot, or be used as an obstacle detection mechanism, as the reflect signal could be detected. A transmitter selection circuit will be needed to control which transmitters can transmit the signal, which would be necessary to a send relative position to another robot. A timer and signal generator would be needed to create the signal needing to be transmitted.

See figure 3.6.

Locomotion Subsystem

The locomotion subsystem will be responsible for converting control signals sent to it into movement. This would be achieved using a motor driver, that controls the power to the

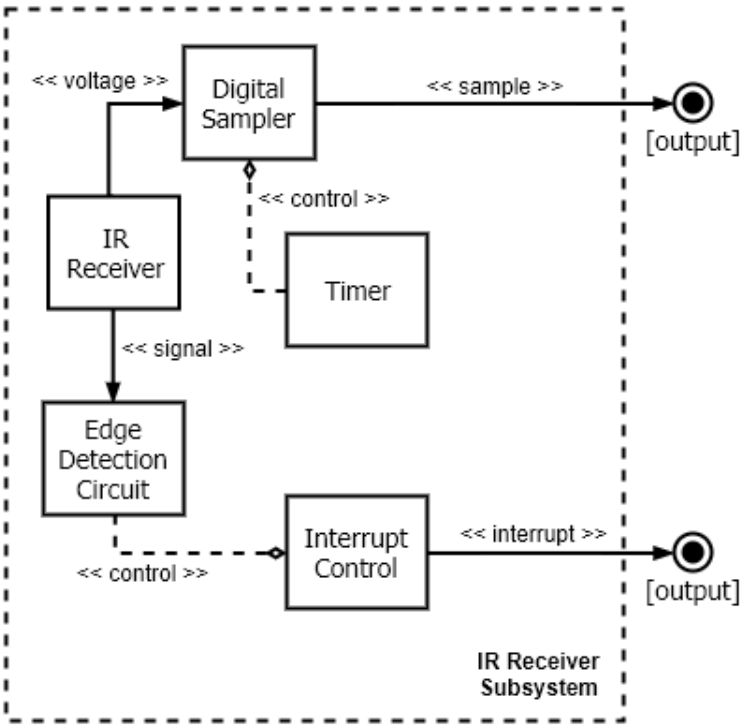


Fig. 3.5 Infrared Receiver Subsystem

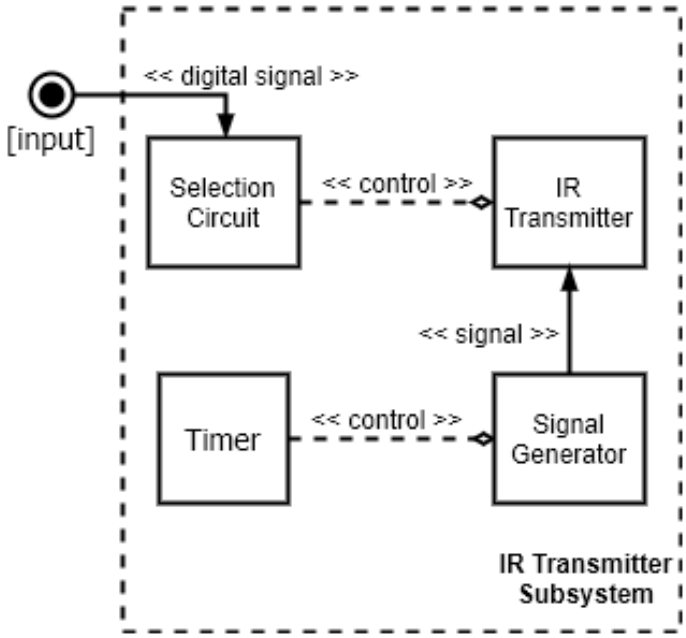


Fig. 3.6 Infrared Transmitter Subsystem

transmission system that drives the wheels. The signal would be a digital output from the micro-controller.

See figure 3.7.

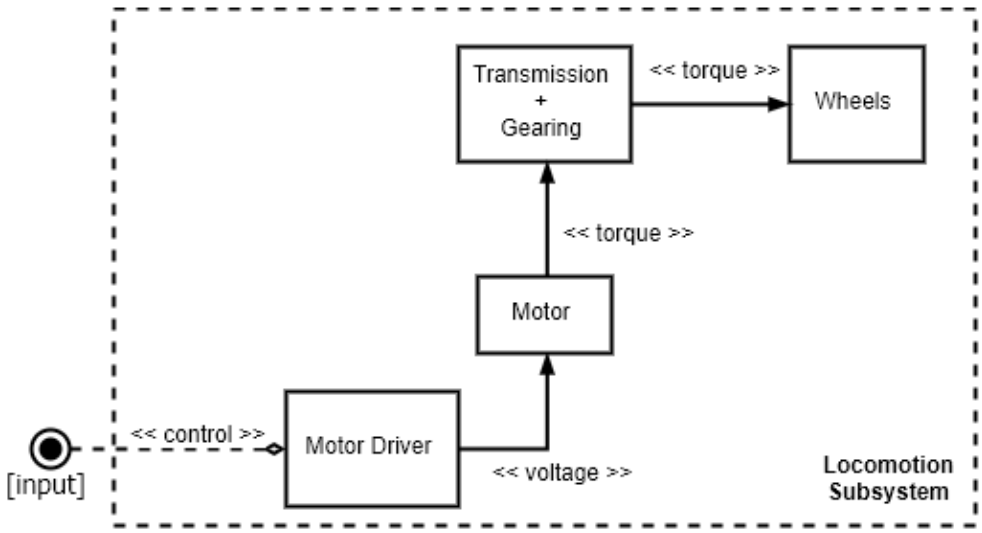


Fig. 3.7 Locomotion Subsystem

Power Supply and Management Subsystem

An on board power supply (a battery) will need to be regulated to a high voltage and a low voltage. The high voltage would power the locomotion and IR transmission subsystems (outputs from the micro-controller). The low voltage would power the micro-controller and the IR receiver and light sensing subsystems (inputs to the micro-controller), as well as the radio transceiver. This would be achieved using two voltage regulators.

See figure 3.8.

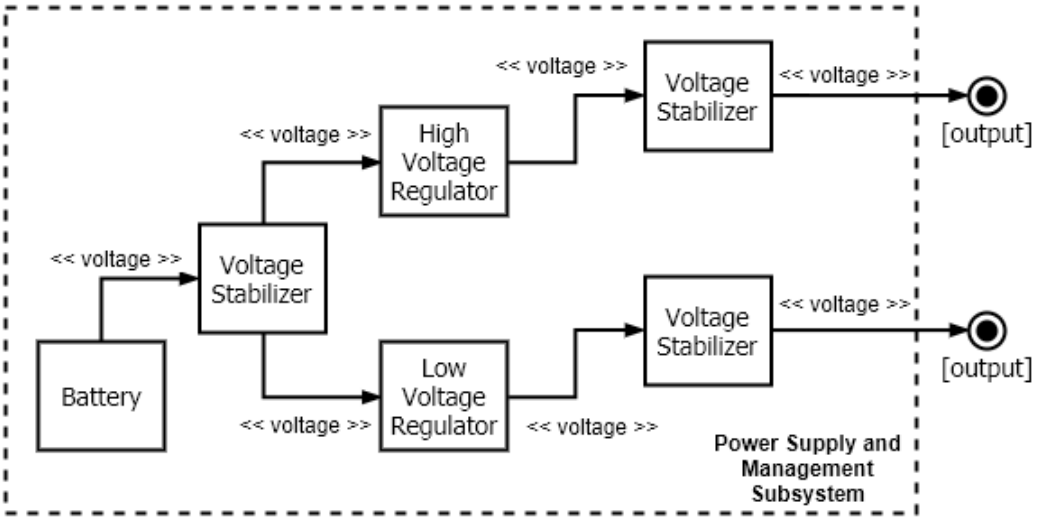


Fig. 3.8 Power Supply and Management Subsystem

3.3.2 Subsystem Verification Plan

Table 3.2 and Table 3.3 define the tests that were performed to check if the built subsystems comply to the specifications given in Table 3.1.

Table 3.2 Subsystem verification hardware tests

Subsystem		Testing Procedure	Figure of Merit	Test. #
Microcontroller	Code execution	Attempt to flash test code onto the microcontroller using a debugger board and verify whether the code is being executed or not	Execution of code	**SVT-HW. 1**
Power supply	Short circuits	Connect the regulators to a voltage supply and verify the current draw does not exceed the threshold amount of 1 A	Current draw	**SVT-HW. 2**
	Voltage stability	Vary the supply voltage to the regulator circuit by ±1 V and verify the output voltage does not change	Output voltage	**SVT-HW. 3**
Locomotion	Speed Control	Measure the speed of the wheels by counting the number of revolutions (N_{rev}) that occur in some time (t). Verify the speed changes in reaction to changing control signals	$\frac{N_{rev}}{t}$	**SVT-HW. 4**
Internal Communications	Range	Verify the IR receiver can detect a signal sent from an IR transmitter placed 30 cm away	Success of signal reception	**SVT-HW. 5**
Light Sensor	Sensitivity	Measure the illuminance of visible light incident on the light sensors using a Lux meter, then increase it by 2 lx using a light source. Do the same with IR light. Verify that the sensor reading **increased** to a change in **visible light** only.	$\Delta V_{sensor} \propto \Delta E_{vis}$	**SVT-HW. 6**

Table 3.3 Subsystem verification software tests

Subsystem		Testing Procedure	Figure of Merit	Test. #
Internal Communications	Message Transmission	Measure the output waveform on the IR LED using an oscilloscope after attempting to send a simple SIRC message using it and verify it matches the specification.	Output waveform shape	**SVT-SW. 1**
	Message Reception	Measure the input waveform of the IR receiver using an oscilloscope during an SIRC message is being transmitted to it and verify it matches the specification.	Input waveform shape	**SVT-SW. 2**
	Message Processing	During the reception of a message, toggle the voltage of a pin while the message processing routing is executing to verify the timing	Timing of pin voltages toggles	**SVT-SW. 3**
Light Sensor	Sampling Behaviour	Sample the light sensors at 100 Hz using a timer and an interrupt. Use a breakpoint in the interrupt handler to note the sample values before and after light is applied to the sensors. Verify the sample values increased after the light was applied.	Increase in sampled values	**SVT-SW. 4**

3.3.3 System Architectural Design

Figure 3.9 shows the architectural design of the system, demonstrating how the subsystems will connect together to form the whole system, needed by each robot.

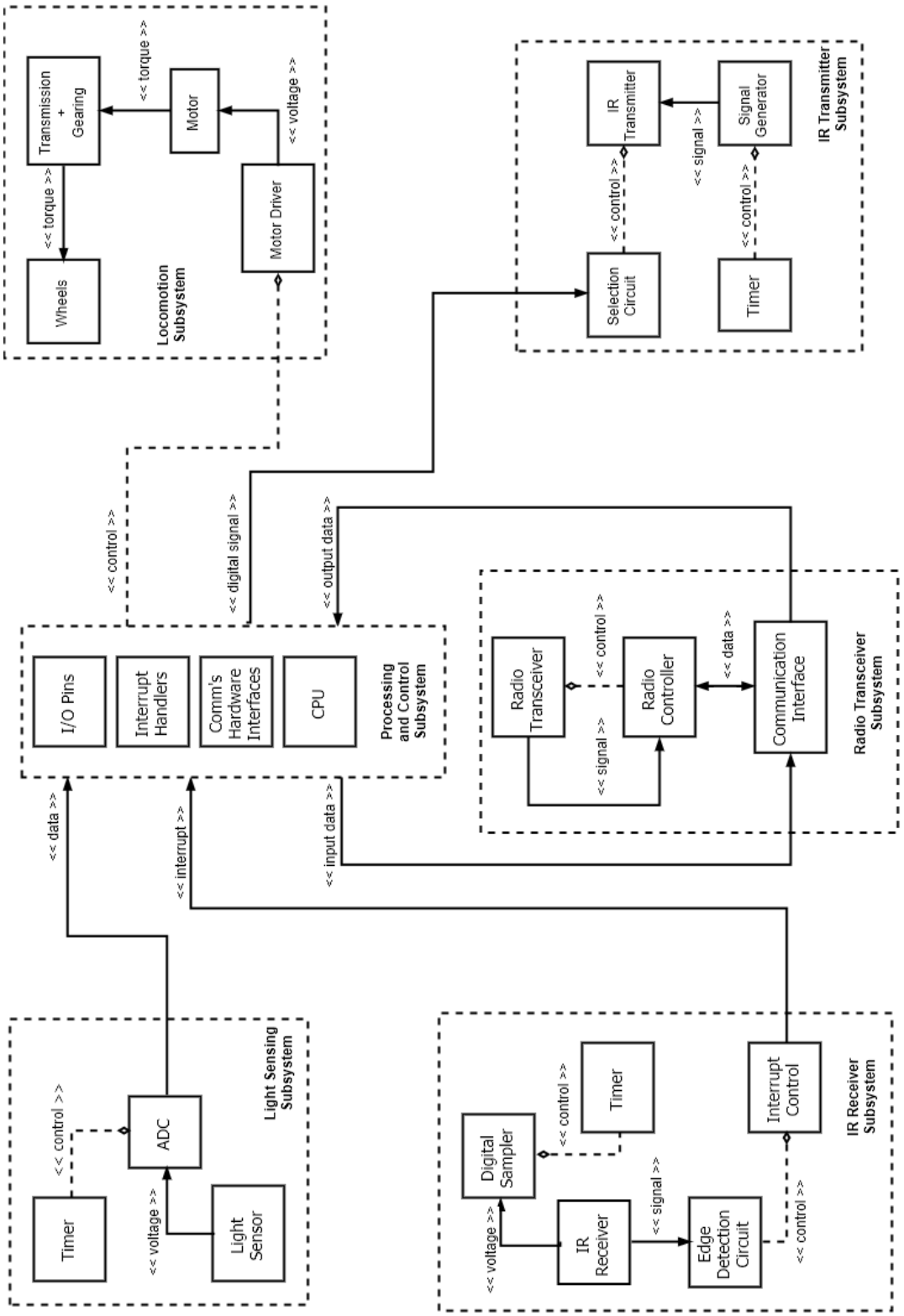


Fig. 3.9 System architecture

3.4 Detailed Design

3.4.1 Hardware

Shape of Robot Body

Tessellation was the main design principle used in when choosing the robot's body shape. Although the robots would never necessarily need to fit together perfectly, in theory they may need to fit together *well*, to effectively cover an area, for example. The robots were only going to move in a 2D plan, thus there were three choices of shape: an equilateral triangle, a regular square or a regular hexagon.

Using a triangle would make mounting the wheels difficult, as they would stick out from either side. A square would work, but distributing sensors around it would be more challenging than using a more 'round' shape, like a hexagon. Thus the hexagon was chosen.

The hexagon is a very natural shape. It is used in the hives of bees, because it tessellates and, out of the three, it has the smallest total perimeter for the largest internal area [48], meaning it is a very compact shape. It has pairs of parallel edges, thus the wheels could be mounted to fit on a pair of the edges and be aligned automatically.

However, other swarm and multi-robot systems do not take this approach. Many of them are circular, such as the Epuck, R-one, Khepera IV and others [45, 49, 47, 46, 42]. The major of these robots have multiple sensors communication receivers and transmitters distributed around them, generally eight of each. Thus, an octagon was considered for the shape of the robots, but having eight of each sensor dramatically increased the number of I/O pins that would be needed, thus it was not chosen.

Motor Drive and Wheels

A differential drive motor system connected to wheels was chosen as the drive system for the robots. It would be the easiest to implement and build, with a large selection of wheels and motors available. It would mean that any control algorithm could not model each robot as a particle that can move freely in a plane, but rather as a non-linear, differential system that must turn (has a heading angle).

The required specifications for the motor drive system were:

- Small size: $A < 40x40$ mm
- Accurate speed control

- Metal gears (better durability compared to plastic gears)
- Stall torque: $T_s > 10\,\text{N}\cdot\text{cm}$ ($\equiv 1\,\text{kg}\cdot\text{cm}$, able to move 1 kilogram, 1 centimeter away)

The required specifications for the wheels were:

- Small diameter: $D \leq 3\,\text{cm}$
- Rubber tier grip

A small standalone DC motor connected to an external gearing system could be used, but gearing systems can take up a lot of space and a motor driver circuit (an H-bridge) would be needed, which would require more space. A servo motor has all three components (DC motor, gearing and motor driver) in a very small form factor, epically if one uses a small servo, but they are generally used for position control. Thus most commercially available servos do not do velocity control.

However, one can hack the servo to make it velocity controlled by solider a resistor divider network between the servo's potentiometer pins. This allows for both forward and backward movement, all in a small form package. Servo motors also come with standard connectors that can fit onto the output head of the servo's shaft, making it simple to attach the wheels onto the servos.

After checking what components were available, the Corona 929MG Metal Gear Servo (see 3.11a) was chosen. It's key specifications were:

- Operating Voltage: 4.8 V to 6.0 V
- Operating Current: 200 mA to 240 mA
- Stall Torque: $2\,\text{kg}\cdot\text{cm}$ to $2.2\,\text{kg}\cdot\text{cm}$
- Size: $22.5x24.6$ mm
- Gearing Material: Metal

[Source: HobbyKing]

These all meet the required specification of the motor drive system.

Servo motors work using position feedback control (given by a potentiometer, coupled to the output shaft). The reference voltage is encoded in the pule length of the PWM wave sent to the controller. The motor controller then drives the motor according the difference

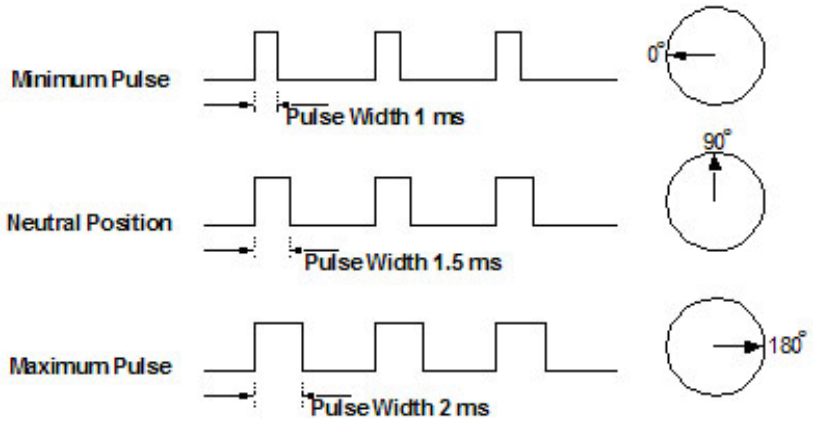


Fig. 3.10 Servo motor position control, showing how pulse length translates into degrees

between the current position (as measured by the pot) and the reference position (reference voltage). The standard pulse length to degrees is the following: 1 ms = 0°, 1.5 ms = 90°, 2 ms = 180° and linear in between (refer to figure 3.10).

Therefore, if you make the position reading constant and you make it half the supple voltage (using a resistor divider), a reference command you give the motor driver will continuously drive the servo based on the deviation away from the 90° reference voltage. Thus, a 1 ms pulse would make the motor turn the fastest in the backward direction (depending on the way the servo is mounted) and a 2 ms pulse would make the motor turn the fastest in the forwards direction. Of course, during the implementation, calibrations to these values were needed.

Refer to figure 3.11b for the selected wheels. Their main specifications were:

- Diameter of 42 mm
- Rubber tire grip

Although these wheels were slightly larger than what was stated in the required specifications for the wheel, the difference was very small and was disregarded.

(a) Selected motor drive and transmission system, the Corona 929MG servo motor
Source: HobbyKing

(b) Selected wheels
Source: MicroRobtics

Fig. 3.11

Microcontroller

The microcontroller was the main component in the system, through which everything was connected. It was responsible for controlling the motor, sending and receiving messages via IR, sampling the light level and generating control signals based on it. The requirements for the chosen microcontroller were:

- Generate PWM signals and accurately control the duty cycle
- Accurately time events and generate interrupts after a certain amount of time
- Sample using an ADC with 10-bit accuracy
- Detect rising and falling edges and generate an interrupt when detected
- Have 20+ I/O ports

The STM32F0 was selected as the microcontroller. It was chosen because it and a development board were readily available, it allowed for low level control of timers and I/O pins and I had experience using it before. The key specifications for the microcontroller were:

- 32 I/O pins

- 1 ADC (8 to 12 bit)
- 9 timers (7 with PWM functionality)
- Can generate interrupts based on external signals (rising, falling edges, pule count etc.)
- Standard operating voltage: $V_{dd,min} = 2\,\text{V}, V_{dd,max} = 3.6\,\text{V}$
- I/O Input Voltage: $V_{in} = V_{dd} \pm 0.3\,\text{V}$
- Maximum output current sourced by any I/O or control pin: $I_{IO,src} = 25\,\text{mA}$
- Maximum input current sunk by any I/O or control pin: $I_{IO,snk} = -25\,\text{mA}$

The microcontroller had two supple (V_{dd}) and GND pins, which (according to the microcontroller's specification) had to be connected to the same supply and ground. A supply voltage of 3.3 V was chosen as it is standard voltage to use and met the supply voltage specification. The $V_{[}bat]$ pin of the microcontroller needed to be connected to ground through a 10 kΩ resistor. Lastly, the V_{ddA} pin needed to be connected to the 3.3 V supply.

Power Supply

Based on the supply voltages needed by the microcontroller and the servo motors, two voltage supplies were needed, one at 5 V and the other at 3.3 V. The 5 V supply would power the servos and IR LEDs and the 3.3 V supply would power the microcontroller, the IR receivers and the light sensors.

An on-board battery was therefore needed. It had to be small enough to fit onto the robot's frame, have a low internal resistance and last long. The required specifications were:

- Output voltage: $V_o \approx 8\,\text{V}$
- Capacity: $E_{cap} > 1000\,\text{mAh}$
- C-rating: $C_{rating} > 10$ (low internal resistance)
- Maximum area: $A \leq 60\,\text{mm}\ x\ 40\,\text{mm}$

Based on availability, the Xcelorin 7.4V Li-Po battery was chosen as the main power source. The key specifications were for the it were:

- Output voltage: $V_o = 8.4\,\text{V}$
- Capacity: $E_{cap} = 3400\,\text{mA h}$
- C-rating: $C_{rating} = 20$
- Area: $A = 55\,\text{mm}\ x\ 30\,\text{mm}$

Lithium Polymer (Li-Po) batteries are commonly used in robotics and consumer products like laptops, electric vehicles, cellphones and the like. They generally have a low internal resistance (as indicated by the C rating of the battery) and can therefore supply the current needed by the system, most notably form the motors.

LP2954 was selected as the 5 V regulator. According to its specification, it can handle a maximum of 1.5 A, which would be enough to supply the motors with the current they require. The LM3940 was selected as the 3.3 V regulator. it is an LDO regulator, which will help prevent 'brown out' and can supply a maximum of 600 mA according to its specification, more than what was required by the microcontroller, the IR receivers and the light sensors.

Light Sensing Subsystem

The light sensing circuit did not need to be complicated. The algorithms that used the sampled light sensor values were designed to be calibrated. However, the light sensors needed to detect changes of 2 lux.

The required specifications for the light sensor were:

- Sensitive to changes of 2 lux
- Sensitive only to visible light
- Output voltage increases with an increase in the light level
- Maximum current draw: $I_{max} < 3\,\text{mA}$

Two potential circuits were considered. The first was based on a photodiode as the sensor and an op-amp in negative feedback (see figure 3.12a), the second was based on a Light Dependent Resistor (LDR), in a resistor divider network (refer to figure 3.12b).

Photodiodes can commutate between ON state and OFF state currents in nanoseconds. They are commonly used in applications such as light meters, laser scanners, remote controls and positioning systems. Photodiodes can be modeled as current sources, with

the output current being dependent on the light intensity. An analysis of the circuit shown in figure 3.12a is performed below:

$$\frac{V_{out} - V_{-}}{R_{fb}} = I_{fb}$$

Due to negative feedback, the inverting input (V_{-}) is at (virtual) ground, implying the voltage across the photodiode is close to 0 V:

$$V_{-} = 0\,\text{V}$$
$$I_{fb} = I_{-} + I_{p}$$

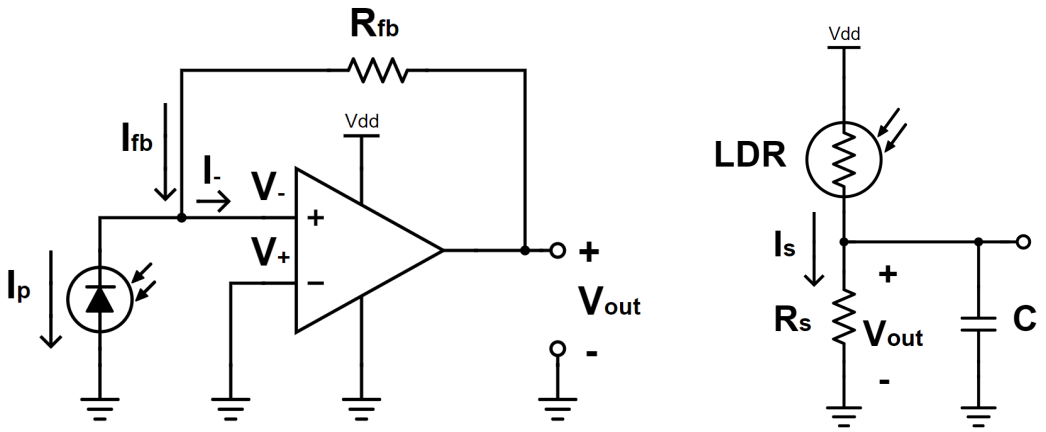
The terminals of op-amps have very high impudences, thus the inverting current (I_{-}) will be close to 0 A:

$$I_{-} = 0\,\text{A}$$
$$\therefore I_{fb} = I_{p}$$
$$\therefore \frac{V_{out}}{R_{fb}} = I_{p} \rightarrow V_{out} = I_{p} \cdot R_{fb}$$

Showing that the relationship between the output voltage (V_{out}) and the current through the diode (I_p) is linear with a gradient of R_{fb}. Figure 3.13a shows the near linear relationship between lux and current through the diode (at 0 V), therefore the relationship between the output voltage and the illuminance (lux) would also be nearly linear.

LDR's are cheap, commonly used sensors. The resistance of an LDR decreases from a few mega ohms to a few hundred ohms as more light falls upon it. They generally have a long response time to changes in illuminance (within a few milliseconds). An analysis of the circuit shown in figure 3.12a is performed below:

$$V_{out} = V_{dd} - I_{t} \cdot R_{LDR}$$
$$I_{t} = I_{s} + I_{ADC}$$
$$I_{s} = \frac{V_{out}}{R_{s}}$$

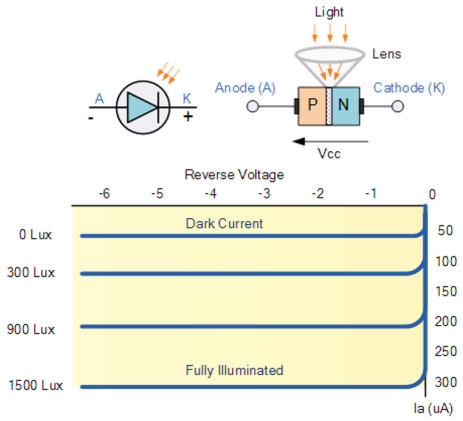


(a) Photodiode light sensing circuit **(b)** LDR light sensing circuit

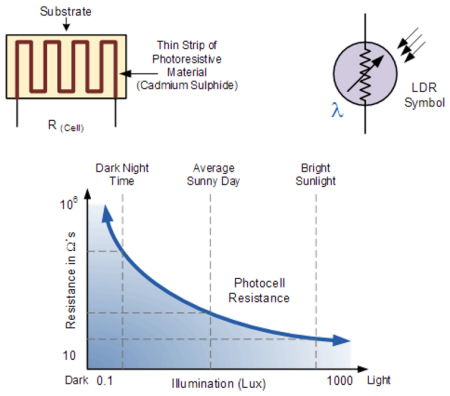
It is reasonable to assume the current draw from the ADC (I_{ADC}) is 0 A because when it attempts to sample the output voltage (V_{out}), the current it requires will be supplied by the buffering capacitor and thus will not load the sensor and therefore not distort the output voltage:

$$I_{ADC} = 0\,\text{A}$$
$$\therefore I_t = I_s$$
$$\therefore V_{out} = V_{dd} - (\frac{V_{out}}{R_s}) \cdot R_{LDR}$$
$$\rightarrow V_{out} = \frac{R_s}{R_s + R_{LDR}} \cdot V_{dd}$$

Showing that the relationship between the output voltage (V_out) and LDR's resistance (R_{LDR}) is non-linear and hyperbolic. Additionally, figure 3.13b shows that the relationship between the LDR's resistance and the lux incidence on it is also non-linear and exponential. Thus the relationship between the output voltage and the light level detected by the LDR is non-linear, but crucially, they are positively proportional. The output voltage will increase if the light level increases (because the LDR's resistance decreases).



(a) Relationship between the photodiode current (I_p) and the illuminance incident on it
Source: Electroincs Tutorials



(b) Relationship between the LDR's resistance (R_{LDR}) and the illuminance incident on it
Source: Electroincs Tutorials

Due to the ease of procurement, the cost and relative simplicity of the circuit, the LDR was chosen as the light sensor (see3.14). Although the linearity of the photodiode was very compelling and the current commutation times far exceeded that of an LDR, both features were not needed. The algorithms that used the light sensors readings could easily be calibrated to account for the exponential natural of the LDR and the sample rates were slow enough to handle its slow response.

The key specifications of the LDR were:

- Maximum Voltage: 100 V
- Maximum current: 5 mA
- Maximal response: 610 nm $\leq \lambda \leq$ 530 nm (see figure 3.15)
- Cell resistance at 10 lux: Min = 20 kΩ, Max = 100 kΩ
- Cell resistance at 100 lux: Typical = 5 kΩ

Source: RS data sheet

The LDR responds only to visible light and not to IR light (1 mm $\leq \lambda_{IR} \leq$ 700 nm). This is an important characteristic, as the IR communications must not interfere with the light level readings.

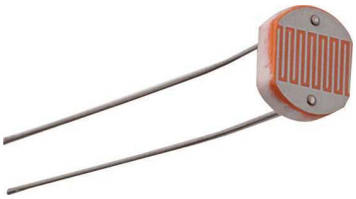


Fig. 3.14 An LDR, used for light sensing
Source: Robotistan

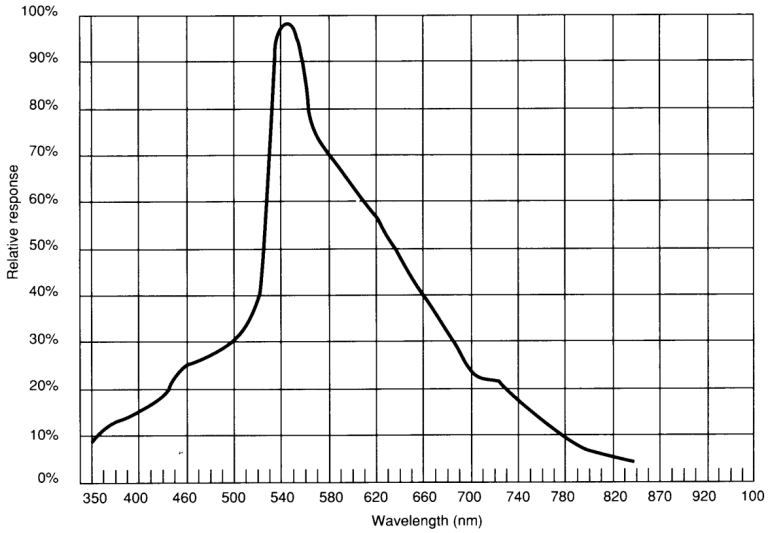


Fig. 3.15 Spectral response of the LDR
Source: Datasheet

Now the resistor R_s for the LDR circuit had be chosen. It had to maximize the sensitivity of the circuit to changes in light, whilst minimizing its current draw, as there would be continual current flow, which wastes power. The light sensor would be supplied with 3.3 V (V_{dd}), because the microcontroller could only sample up to it's supply voltage of 3.3 V. The procedure detailed below was followed:

The sensitivity of the circuit was defined as the change in the output voltage for a change in the LDR's resistance:

$$\mathbf{S}_{V_{out}/R_{LDR}} = \frac{\partial V_{out}}{\partial R_{LDR}} = \frac{0 \cdot (R_{LDR} + R_s) - 1 \cdot R_s \cdot V_{dd}}{(R_s + R_{LDR})^2} = -\frac{R_s \cdot V_{dd}}{(R_s + R_{LDR})^2}$$

The sensitivity for $R_{LDR} \gg R_S$ is given by:

$$\mathbf{S}_{V_{out}/R_{LDR}} = -\frac{1}{R_{LDR}^2}$$

However, only the sensitivity for lower values of R_{LDR} are of interest, that is for $R_s \gg R_{LDR}$ the sensitivity is given by:

$$\mathbf{S}_{V_{out}/R_{LDR}} = -\frac{V_{dd}}{R_s}$$

This shows as R_s decreases the sensitivity increases, but the current draw for the circuit is given by:

$$I_t = \frac{V_{dd}}{R_{LDR} + R_s}$$

Ignoring R_{LDR}, this shows as R_s decreases, the current draw increases

Therefore a compromise must be made. Based on the current draw specification, the minimum resistance of R_s was:

$$R_{s\ min} = \frac{V_{dd}}{I_{max}} \rightarrow \frac{3.3\,\mathrm{V}}{3\,\mathrm{mA}} = 1.1\,\mathrm{k\Omega} \approx 1\,\mathrm{k\Omega}$$

However, this would make the sensor too sensitive in lighter areas and not sensitive enough in darker areas. The half point (when $R_{LDR} = R_s$) would be too low (in terms of resistance), the roughly 100 lx would need to be incident on the to make its resistance low enough, only to be at the half way point. The sensor would not be able to pick up on small changes in the light level in dark areas. Therefore choosing $R_s = 10\,\mathrm{k\Omega}$ would alleviate this, make the sensor sensitive enough in darker areas and not saturate in lighter areas. The maximum current draw is given as:

$$I_{max} = \frac{3.3\,\mathrm{V}}{10\,\mathrm{k\Omega}} = 0.33\,\mathrm{mA}$$

This meets the specification of $I_{max} < 3\,\mathrm{mA}$.

IR receiver

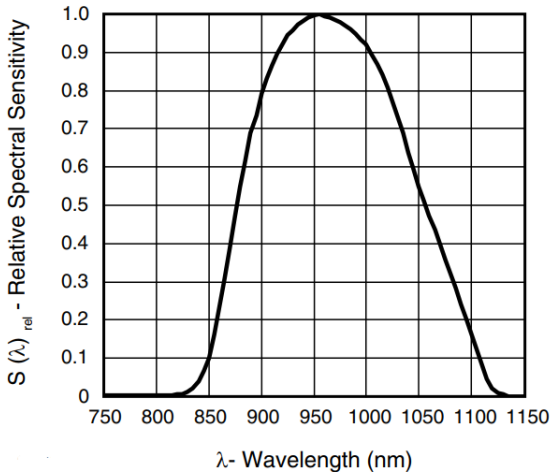
The IR receiver needed to filter out ambient IR light, as well as detect signals sent to it. It therefore would need a physical filter and implement some kind of phase lock loop to be receptive to a carrier wave that encoded the data signal.

After searching for available components, the Vishay TSOP22 3-pin IR Receiver was found (see Figure 3.16a). The key specifications of the receiver were:

- Supple voltage: Min = 2.5 V, Max = 5.5 V
- Carrier wave frequency: $f_o = 38\,\text{kHz}$
- Minimum irradiance: $E_{e,min} = 0.25\,\text{mW/m}^2$
- Receive angle: $\phi \pm 40°$
- Receiver area: $A_{rec} = 6.0\,\text{mm}\ x\ 5.3\,\text{mm}$

[Source: Datasheet]

(a) The TSOP22 IR Receiver
Source: Lees Electronics



(b) The spectral sensitivity of the TSOP22 IR Receiver, showing that it maximally responds to IR light only
Source: Datasheet

The receiver works in the following way, the output signal is normally high (V_{cc}) and when it detects an PWM signal of IR light, with a frequency of 38 kHz, it goes low (active low) (GND). Therefore, the sent signal will be inverted when received. Figure 3.17 shows the internal block diagram of the receiver.

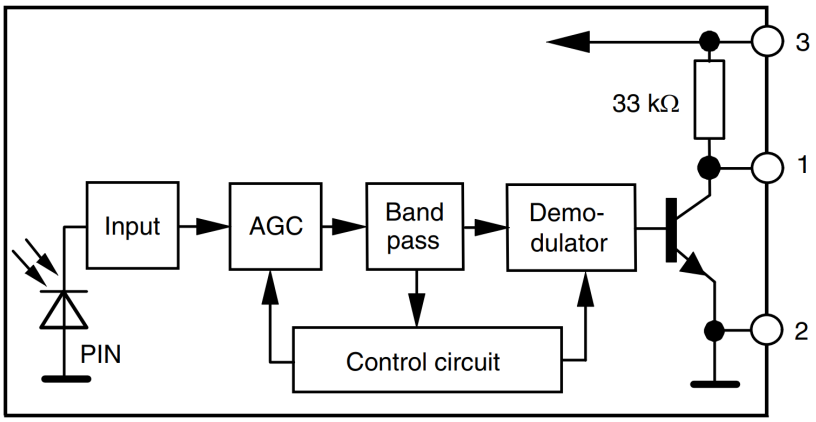


Fig. 3.17 Internal block diagram of TSOP22 IR Receiver
Source: Datasheet

The circuit used to drive the IR receiver (shown in Figure 3.18) is based on the recommended circuit shown in the datasheet. It has a pull-up resistor to make sure the signal is properly high when not active (even though receiver has an internal pull-up resistor, it might not have a high enough resistance, thus the datasheet recommends the use of an external one). One small resistor is used to limit the supply current a little and two filtering capacitors are also used to maintain a constant supply voltage.

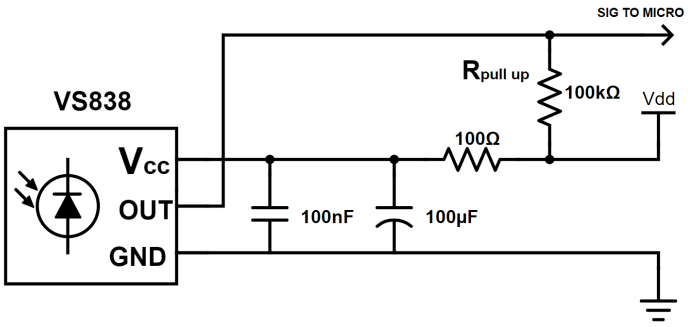


Fig. 3.18 IR receiver driver circuit

IR transmitter

The IR transmitter needed to output IR light and be able to do at around 38 kHz, with minimal distortion. Because there were going to be more than one transmitter on the robot at one time (for directional communication), a circuit was needed that could route the signal needing to be sent to any IR transmitter independently. This proved to be the main design challenge for this system.

The required specifications for the transmitter system were:

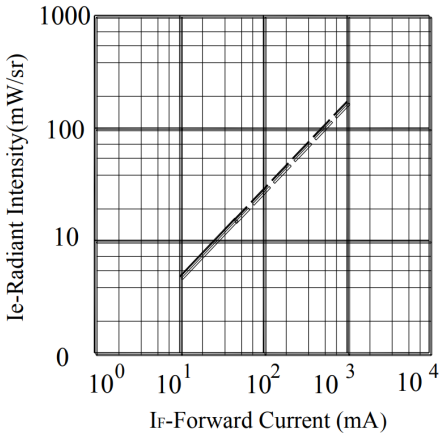
- Transmission medium: IR light
- Transmission range: 0 to 30cm
- Transmission frequency: 38 kHz
- Independent selection of transmitter
- Must handle a supply of 5 V (V_{dd})

After searching for available IR transmitters, the Everlight 5mm IR LED (SIR333) was found (see Figure 3.19a). The key specifications for the IR LED were:

- Maximum Continuous Forward Current: $I_F = 100\,\text{mA}$
- Peak Wavelength: $\lambda_p = 875\,\text{nm}$
- Spectral Bandwidth: $\Delta\lambda = 80\,\text{nm}$
- Forward Voltage: $V_F = 1.65\,\text{V}$ at $I_F = 20\,\text{mA}$ and $V_F = 1.8\,\text{V}$ at $I_F = 100\,\text{mA}$
- View angle: ±20°

[Source: Datasheet]

(a) The IR LED (SIR333), used for transmitting
Source: Tayda Electronics



(b) The radiant intensity of the IR LED ($I_{e,\gamma}$) versus the forward current (I_f) through it
Source: Datasheet

Fig. 3.19

A forward current (I_f) of 14 mA was chosen to be the driving current for the IR LEDs. This generated enough radiant intensity, in order to meet the minimum irradiance of

0.25 mW/m^2 needed by the receiver. This value of forward current was validated in the following way:

Using the graph shown in Figure 3.19b, a forward current of 14 mA generated a radiant intensity of around 10 mW/sr. The irradiance (E_e) was found from the radiant intensity ($I_{e,\gamma}$) using the following equation:

$$E_e = I_{e,\gamma} \cdot \frac{\Omega}{A_{rec}}$$
$$\Omega = 2\pi(1 - \cos(\frac{\alpha}{2}))$$

Where:
E_e is the irradiance in W/m^2
$I_{e,\gamma}$ is the radiant intensity in W/sr
A_{rec} is the area of the receiver in m^2
Ω is the soil angle of the transmitter of the receiver in sr
α is the apex angle of the transmitter in rad

The apex angle of the transmitter (α) is 20° (= $\frac{\pi}{9}$rad) as it has a relative intensity of 50% between ±10°. Thus:

$$\Omega = 2\pi(1 - \cos(\frac{\frac{\pi}{9}}{2})) = 0.095\,\text{sr}$$
$$E_e = 10 \cdot \frac{0.095}{(6.00\ x\ 5.30) * 10^{-6}} = 0.029\,87\,\text{W/m}^2 = 29.87\,\text{mW/m}^2$$

This was above the specification for the minimum irradiance needed by the receiver:

$$E_e > E_{e,min} = 0.25\,\text{mW/m}^2$$

To control the LED's independently, a few ideas were tried. The first was the use of a shift register to control whether or not the LED was grounded by sending a 0 or 1 to that pin of the shift register and simply applying the PWM signal to the anode of the LED. The premise of this idea was that LED's are current driven devices, thus if the LED's cathode was pulled high and the signal was high, no current would flow though it because there would not be no voltage across it and if the signal was low, there would

still be no current flow because the diode would be reverse biased and therefore block the current. Conversely, cathode was pulled low, the signal would simply be output on the LED, with current flowing when the signal was high and not flowing when the signal was low. Therefore the output of the LED would be directly correspond to the signal. The circuit was built and tested, but it failed to work, mostly due to the poor current sinking ability of the shift register, as it was not designed for such as task.

The second idea was to apply the PWM signal to the Output Enable (OE) pin of the shift register and have the LED's driven by transistors, with the base of the transistor connected to the output of a shift register pin. It was based on the first idea and was thought that a transistor could control the current far better than the shift register could. The premise of the idea was that an IR LED could be selected by sending a 1 to that pin on the shift register, then the OE pin would pull the pin high and low corresponding to the signal, turning the transistor and in turn the IR LED on and off. If a 0 was sent to the pin on the shift register, the signal would never propagate through the transistor and therefore the IR LED would not transmit the signal. The circuit was built and test, but again did not work. This was because the OE pin did not pull the shift pins high or low, rather when OE was low, it put the pins into a high impedance state (because they were tri-state buffered) and did not ground them. Thus, the transistors went from being on to being in an undefined state, as the base pin was effectively floating.

The third circuit was based on the second. It was realised that the shift register was not needed and that the LED's could be selected in the same way, by simply using the microcontroller's I/O pins, as there were enough of them. Additionally, another transistor could control the power line going to the LED's, the base of which would receive the PWM data signal. Two circuits were designed and tested, the first being the circuit shown in Figure 3.20 and second shown in Figure 3.21.

N-Channel MOSEFET's were used as the transistors controlling the selection of the LEDs for both circuits. MOSFET's were chosen as they are simple to use, have fast switching times and dissipate very little power, only requiring a small resistor to connect through. The 2N7000 N-Channel MOSFET was chosen due to its availability. The key specifications for the MOSEFET were:

- Maximum continuous drain current: $I_{D,max} = 200\,\text{mA}$
- Gate Threshold Voltage: Maximum of $V_{GS(th)} = 3\,\text{V}$
- Drain-Source On-Voltage: $V_{DS(on)} = 0.45\,\text{V}$ at $I_D = 75\,\text{mA}$

- Turn-On Delay Time: $t_{on} = 10\,\text{ns}$
- Turn-Off Delay Time: $t_{off} = 10\,\text{ns}$

A small resistor was needed to connect the gate of the MOSFET to the I/O pin of the microcontroller. This was because the MOSFET could modeled as capacitor connected to ground when the pin of the microcontroller goes from 0 V to 3.3 V, therefore the resistor was needed to limit the in-rush current.

$$R_m = 100\,\Omega$$

BJT's were chosen as the transistors controlling the power to the IR LED's. BJTs were selected because LEDs are current controlled devices, therefore a current amplifier was required and hence the choice. a MOSFET would need to be driven to a voltage high than 3.3 V, to allow enough current through it to drive 6 IR LEDs at one time.

The most important distinction between the two circuits is where the BJTs sit relative to the LEDs. The circuit shown in Figure 3.20 (from now on refered to as circuit A) uses an NPN BJT to drive a PNP BJT, with the PWM signal being send to the base of the NPN BJT. The NPN BJT is needed because the output voltage from the microcontroller is only3.3 V, which if connected straight to the base of the PNP BJT, would never be able to turn it off. To do so would require a voltage > $V_{be(sat)}$, which is around 4.3 V for a V_{dd} of 5 V. However, to drive the NPN transistor into a saturated ON state requires a voltage > $V_{be(sat)}$, which is around 0.7 V and into an OFF state, a voltage of 0 V, both easily achievable using the microcontroller. This type of switch is known as a High Side Switch. The circuit shown in Figure 3.21 (from now on referred to as circuit B) used only a NPN transistor to control whether the selected LEDs were grounded or not. The PWM signal was attached to the base of the transistor, thus the current flowing through the active LED would be an amplified version of the data signal, therefore outputting the signal as IR light via the LEDs.

The 2N2222 NPN transistor and the 2N2906 PNP transistor were selected, based on availability. The key specifications for the 2N2222 NPN transistor were:

- Maximum Continuous Collector Current: $I_{c,max} = 600\,\text{mA}$
- Maximum Emitter-Base Voltage: $V_{be,max} = 5\,\text{V}$
- Base-Emitter Saturation Voltage: $V_{be(sat),min} = 0.6\,\text{V}$, $V_{be(sat),max} = 1.2\,\text{V}$ at $I_c = 150\,\text{mA}$, $I_b = 15\,\text{mA}$

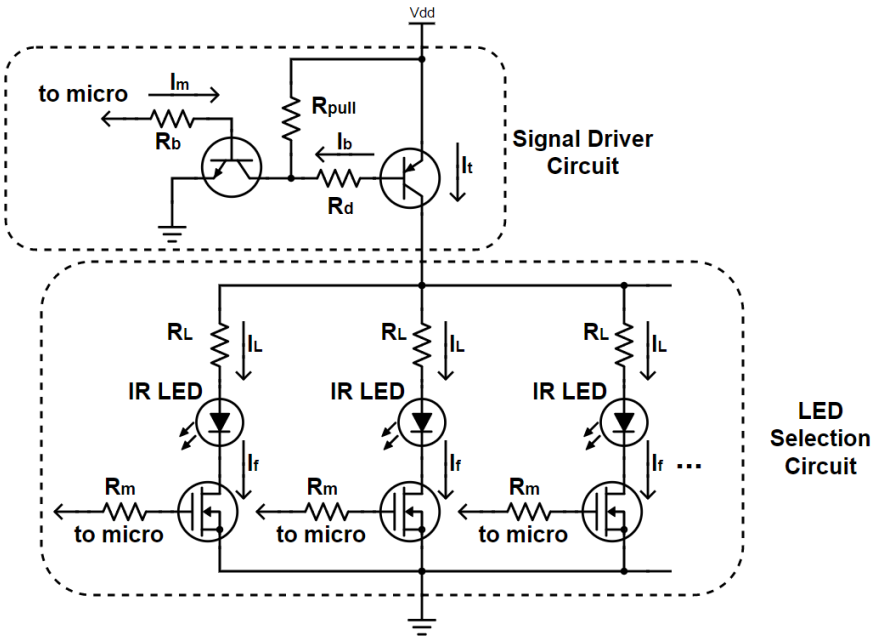


Fig. 3.20 (**Circuit A**) IR LED driver circuit, using a PNP BJT driven by an NPN BJT to control the signal going to the IR LED's and N-Channel MOSFET's to control which IR LED can transmit the signal

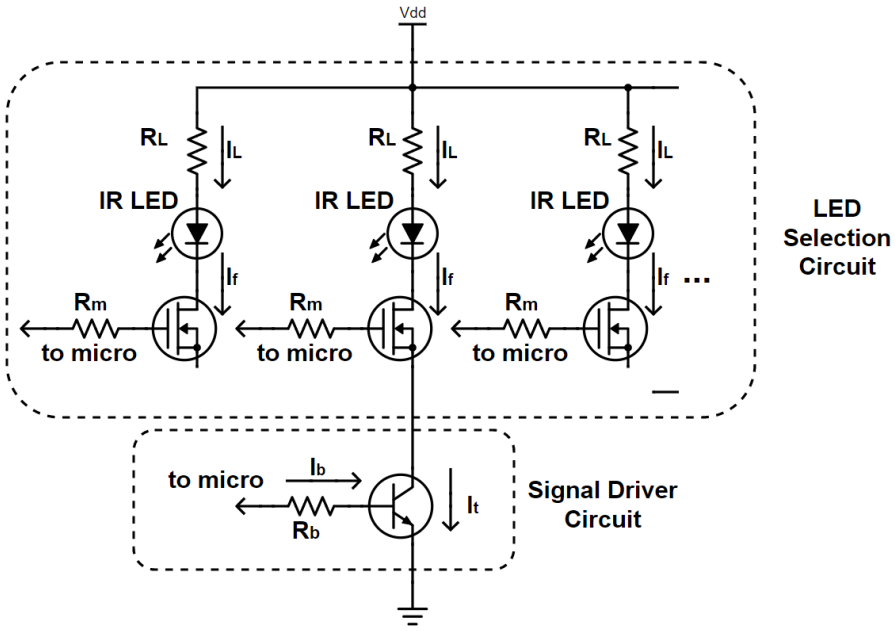


Fig. 3.21 (**Circuit B**) IR LED driver circuit, using an NPN BJT to control the signal going to the IR LED's and N-Channel MOSFET's to control which IR LED can transmit the signal

- Collector-Emitter Saturation Voltage: $V_{ce(sat),max} = 0.4\,\text{V}$ at $I_c = 150\,\text{mA}$, $I_b = 15\,\text{mA}$
- DC Current Gain: $50 \leq h_{FE} \leq 300$
- Turn-On Delay Time: $t_{on} = 25\,\text{ns}$
- Turn-Off Delay Time: $t_{off} = 60\,\text{ns}$

[Source: Datasheet]

The key specifications for the 2N2906 PNP transistor were:

- Maximum Continuous Collector Current: $I_{c,max} = -600\,\text{mA}$
- Maximum Emitter-Base Voltage: $V_{be,max} = -5\,\text{V}$
- Base-Emitter Saturation Voltage: $V_{be(sat),max} = -1.3\,\text{V}$ at $I_c = -150\,\text{mA}$, $I_b = -15\,\text{mA}$
- Collector-Emitter Saturation Voltage: $V_{ce(sat),max} = -0.4\,\text{V}$ at $I_c = -150\,\text{mA}$, $I_b = -15\,\text{mA}$
- DC Current Gain: $40 \leq h_{FE} \leq 120$
- Turn-On Delay Time: $t_{on} = 45\,\text{ns}$
- Turn-Off Delay Time: $t_{off} = 100\,\text{ns}$

*some values are negative as they are measured relative to the emitter
[Source: Datasheet]

Circuit A is analyzed below, using worst case values, to find the required values of R_d, R_{pull}, R_b and R_L:

There are 6 IR LEDs, each requiring 10 mA, for a total current of 60 mA (worst case, all LEDs are on). Make it 100 mA for safety:

$$I_t = 100\,\text{mA}$$

The worst case DC current gain (call it β_{min}) for the PNP BJT is 40 A/A, use half of this value (precautionary) to find the base current needed to make sure the transistor is saturated:

$$I_{b,needed} = \frac{I_t}{0.5 * \beta_{min}} \rightarrow \frac{100\,\text{mA}}{0.5 * 40} = 5\,\text{mA}$$

$V_{be(sat),max}$ for the PNP transistor is around 1.3 V and $V_{ce(sat),max}$ is around 0.4 V for the NPN transistor (absolute worst case). Therefore, R_d was determined to be:

$$R_d = \frac{V_{dd} - V_{be(sat),max,PNP} - V_{ce(sat),max,NPN}}{I_{b,needed}} \rightarrow \frac{5\,\text{V} - 1.3\,\text{V} - 0.4\,\text{V}}{5\,\text{mA}} = 660\,\Omega$$

Choose 620 Ω, as it is a standard value (a lower value errs on the side of caution):

$$R_d = 620\,\Omega$$

Choose R_{pull} to be around 10 or 20 times R_d as it only need to act as a pull up resistor:

$$R_{pull} = 10\,\text{k}\Omega$$

The NPN transistor needs to sink more than 5 mA of current, therefore design for 10 mA. The worst case DC current gain (β_{min}) for the NPN transistor is 50 A/A, use half of this value (precautionary) to find the current the microcontroller needed to source to make sure the transistor was saturated:

$$I_m = \frac{I_{b,needed} * 2}{0.5 * \beta_{min}} \rightarrow \frac{10\,\text{mA}}{0.5 * 50} = 0.4\,\text{mA}$$

Make this value 1 mA, to make sure the transistor saturates:

$$I_{m,needed} = 1\,\text{mA}$$

Find the R_b, with $V_{be(sat),max}$ for the NPN transistor at around 1.2 V:

$$R_b = \frac{V_{micro} - V_{be(sat),max}}{I_{m,needed}} \rightarrow \frac{3.3\,\text{V} - 1.2\,\text{V}}{1\,\text{mA}} = 2100\,\Omega$$

Make this 2 kΩ, as it is a standard value:

$$R_b = 2\,\text{k}\Omega$$

Now the current limiting resistor (R_L) for each IR LED must be designed. Each LED requires 10 mA, thus design for 15 mA ($I_{L,needed}$). The assumption that both the MOSFET and PNP BJT are in a saturated state must be made, with $V_{ce(sat),max}$ at around 0.4 V for the PNP BJT, $V_{DS(on)}$ at around 0.45 V for the MOSFET and the forward voltage of the IR LED (V_F) at 1.65 V:

$$R_L = \frac{V_{dd} - V_{ce(sat),max} - V_F - V_{DS(on)}}{I_{L,needed}} \rightarrow \frac{5\,\text{V} - 0.4\,\text{V} - 1.65\,\text{V} - 0.45\,\text{V}}{15\,\text{mA}} = 166.66\,\Omega$$

Choose 150 Ω, as it is a standard value (a lower value errs on the side of caution):

$$R_L = 150\,\Omega$$

In summary, the following resistor values were calculated:

$$R_d = 620\,\Omega$$
$$R_{pull} = 10\,\text{k}\Omega$$
$$R_b = 2\,\text{k}\Omega$$
$$R_L = 150\,\Omega$$

Circuit B is analyzed below, using worst case values, to find the required values of R_b and R_L:

There are 6 IR LEDs, each requiring 10 mA, for a total current of 60 mA (worst case, all LEDs are on). Make it 100 mA for safety:

$$I_t = 100\,\text{mA}$$

The worst case DC current gain (β_{min}) for the NPN BJT is 50 A/A, use half of this value (precautionary) to find the base current needed to make sure the transistor is saturated:

$$I_{b,needed} = \frac{I_t}{0.5 * \beta_{min}} \rightarrow \frac{100\,\text{mA}}{0.5 * 50} = 4\,\text{mA}$$

$V_{be(sat),max}$ for the NPN transistor is around 1.2 V. Therefore, R_d was determined to be:

$$R_b = \frac{V_{micro} - V_{be(sat),max}}{I_{b,needed}} \rightarrow \frac{3.3\,\text{V} - 1.2\,\text{V}}{4\,\text{mA}} = 525\,\Omega$$

Choose 510 Ω, as it is a standard value (a lower value errs on the side of caution):

$$R_b = 510\,\Omega$$

The current limiting resistor (R_L) for each IR LED was then determined. Each LED requires 10 mA, thus design for 15 mA ($I_{L,needed}$). The assumption that both the MOSFET and NPN BJT are in a saturated state must be made, with $V_{ce(sat),max}$ at around 0.4 V for the NPN BJT, $V_{DS(on)}$ at around 0.45 V for the MOSFET and the forward voltage of the IR LED (V_F) at 1.65 V:

$$R_L = \frac{V_{dd} - V_{ce(sat),max} - V_F - V_{DS(on)}}{I_{L,needed}} \rightarrow \frac{5\,\text{V} - 0.4\,\text{V} - 1.65\,\text{V} - 0.45\,\text{V}}{15\,\text{mA}} = 166.66\,\Omega$$

Choose 150 Ω, as it is a standard value (a lower value errs on the side of caution):

$$R_L = 150\,\Omega$$

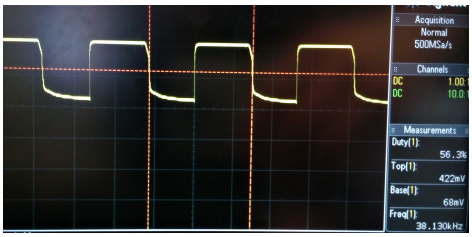
In summary, the following resistor values were calculated:

$$R_b = 510\,\Omega$$
$$R_L = 150\,\Omega$$

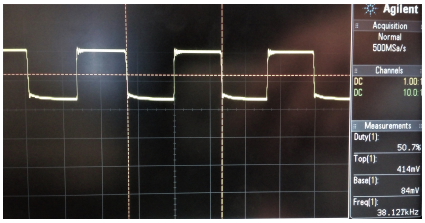
Both circuits were built and tested, using a 38 kHz 50% duty cycle PWM wave as the test signal (see Figure 3.22a and Figure 3.22b). Both circuits worked as expect, with the only difference being the shape of the output waveform and the current draw. The test results for circuit A show a slightly more distorted wave compared to the other results. This was due to additive parasitic properties of both transistors, each contributing a time on and time off delay, as well as both having parasitic capacitances. The current draw for circuit B was around 180 mA and was slight higher than that for circuit A, which was around 150 mA. Due to less distortion of output waveform and simpler circuit, circuit B was chosen (see Figure 3.23 for the implementation of circuit B used for testing).

Pin and function definitions, circuit schematic for the system

After finalizing the circuit design for each subsystem, knowing the number of ports and power requirements for each, the pins and functions (ADC, timers etc.) that were going to be used by each needed to be defined. Certain pins on the STM32F0 have specific alternate functions associated to them, which could be found on the STM32F0 datasheet,



(a) Test results of circuit A



(b) Test results of circuit B

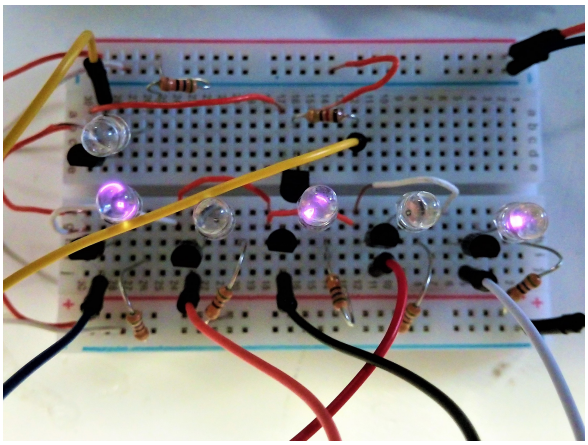


Fig. 3.23 Implementation of circuit B used for testing, additionally showing the LED selection functionality (some IR LEDs are on and some are off)

tables 14 and 15. An important consideration that had to be taken into account was that only GPIOA pins could be sampled by the ADC. Table 3.4 details the pins that were used and Table 3.5 details the microcontroller functions needed by each subsystem.

Table 3.4 Pin definitions for the system using the STM32F0

Pin Number	Pin Name	Mode	Description
		Light Sensor	
10 → 15	PA0 → 5	Analog Input	Analog voltage input for the six light sensors. Only GPIOA pins can be sampled by the ADC.
		Motor Control	
21	PB10	Alternate Function 2	Alternate Function 2 maps to TIM2 CH3 for pin PB10, which was used to generate the PWM signal needed to control the servos.
22	PB11	Alternate Function 2	Alternate Function 2 maps to TIM2 CH4 for pin PB11, which was used to generate the PWM signal needed to control the servos.
		IR Receiver	
18 → 20, 39 → 41	PB0 → 5	Digital Input	Digital input for the six IR receivers. A pull up resistor was used with the pins to make the input normally high.
		IR Transmitter	
46	PB9	Alternate Function 2	Alternate Function 2 maps to TIM17 CH1 for PB9. Used to output the PWM IR data signal.
29, 30, 25 → 28	PA8, PA9, PB12 → 15	Digital Output	Digital output to control the selection of the IR transmitters (IR LEDs).

Table 3.5 System functions needed from the STM32F0 by the system

Pin Name	System Functions	Module
	Light Sensor	
PA0 → 5	Timer in interrupt mode, Analog to Digital Converter	TIM6, ADC
	Motor Control	
PB10, PB11	Timer in PWM output mode	TIM2
	IR Receiver	
PB0 → 5	External interrupt on falling edges	EXTI
	IR Transmitter	
PB9	Timer in interrupt mode and another in PWM output mode	TIM16, TIM17

With the pins and functions defined, the complete electrical design of the system could take place (refer to Figure **??**).

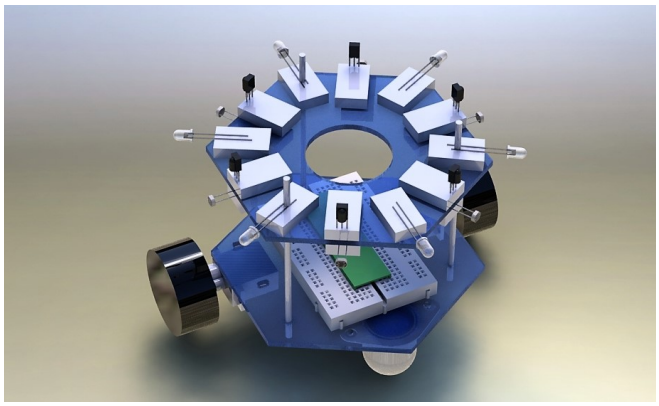
Mechanical design of the robot body

After finalizing selection of components needed by the system and completing the design of the circuitry, the mechanical design could take place, as the dimensions of each component were known (or roughly known).

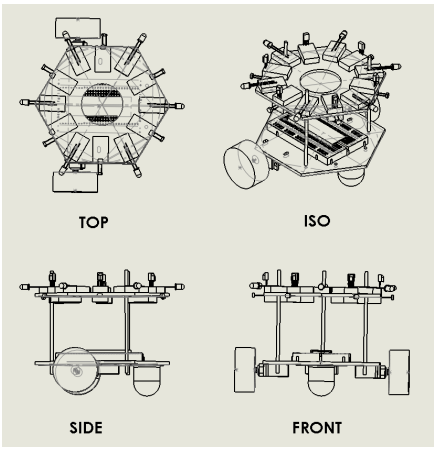
A few considerations needed to be kept in mind while doing the mechanical design. The routing of the wries needed to be as easy as possible and no PCB was going to be used (due to time restraints). The robot had to hexagonal in shape and leave enough room for sensors and actuators. The design had to be modular, making it as easy as possible to upgrade or repair components.

Figure 3.24a shows a render of the final design of the robot and Figure 3.24b shows the mechanical drawing of it. The large whole in the middle of the top layer was designed for routing of cables and the three rods going from the bottom to the top layer hold the whole design together. They were also conceptually a hardware interface for extensions to the robot, more hexagon layers with their own subset of functionality could be added by simply bolting it onto the rods and connecting the wires into the microcontroller. The top layer contained all the light sensors, IR transmitters and receivers, spread out evenly. The battery, the two driven wheels and the omni-wheel (not driven) were attached to the bottom of the bottom layer. Conceptually, the battery, the light sensors, transmitters and

receivers were going to be attached using Velcro with double sided tape (to make it easy to add or remove them), the mini-breadboard that held the microcontroller had double sided tape already and thus could simply be stuck onto the bottom layer, finally the motors were attached using zip-ties and the omni-wheel using bolts. It was necessary to model each component in the system, as the size of the frame could then be determined being 10 cm x 10 cm (point to point) and the volume of the entire design at just over 10 cm x 10 cm x 10 cm.



(a) Render of the final robot design



(b) Mechanical drawing of the full robot design

Fig. 3.24

Figure 3.25 shows the mechanical drawing for the top and bottom layers of the robot's frame. These were the final output of the mechanical designed and used by the laser cutter to cut the frame out of perspecs. Perspecs was chosen as it is non-conductive, does not erode and does not warp easily (unlike chipboard).

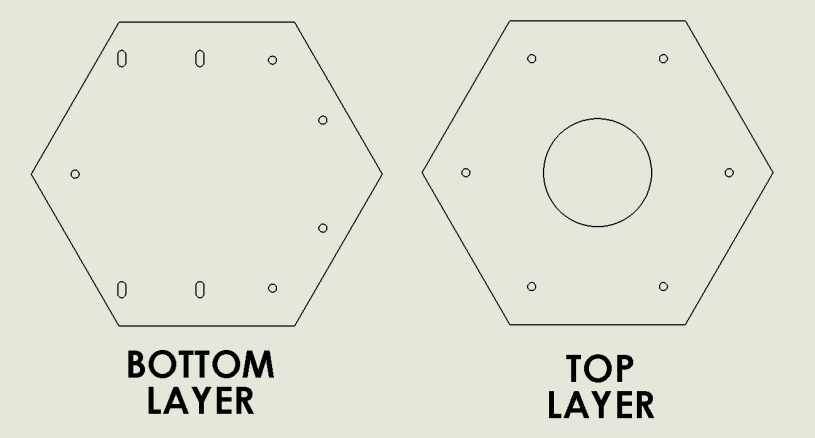


Fig. 3.25 Mechanical drawing of the frame for the robot

3.4.2 Software

The design philosophy chosen for the software was based on a subsumption architecture, which is a behavioral model for robotic system detailed by Brooks in [2]. The architecture involves having hierarchical set of layers that act asynchronously and independently from each other. Each layer has some kind of sensory input and attempts to control (or not control) some relevant actuator. Higher level layers can always subsume (take over) control of the actuators from lower levels, if it is outputting some control signal. Addi tonally, the system was designed to always tend to a stable state over time. That is even if everything fails, the system should fail to known, defined state, which it can then operate from (begin seeking its goal, exploring and so on).

The system had only two types of sensory input that could supply the model with information, one was the light sensor and the other was the IR receiver. Most of the control action was based on the light sensor input, as it related to the main goal of the system. Avoiding obstacles and other robots was based on the input from the IR receiver. Transmitting messages was done completely independently from controlling the motors. Layers communicated to each other using modifiable global states, that could be checked in different loops.

There were four main modules that needed to be designed: message sending, message receiving, light sensing and motor control. The sections below detail the design of the algorithms needed by each module, how they were arrived at and why they will work.

Message transmission and reception

The system needed a way of being able to send messages from any point in it's execution. Therefore some kind of queue was needed that would be checked in the main loop (therefore it is guaranteed the queue would be checked) for any unsent, queued message. Thus, any function could post a message to the queue and be sure it would *eventually* be sent.

An IR communication protocol was needed. It had to be simple to implement, transmission times had to be quick and it did not need much error checking, as the messages were going to be simple. There were a few possibilities including the NEC protocol, Phillips RC-5 and Sony SIRC. The NEC protocol takes around 67.5 ms to transmit a message. Each message contains a start phase, a command and an address. An inverted and non-inverted version of the command and address are transmitted, which is done for error checking purposes. The RC-5 takes around 25 ms to transmit a message. Each message contains two start bits that define the start as well as the layout of the message. The data in the message is encoded using Manchester Coding, with a 1 being a high for 889 µs and low for 889 µs and a 0 being the opposite. The Sony SIRC protocol takes between 17 ms to 25 ms to transmit a message. The data is encoded in a similar way to the RC-5 protocol, using a Manchester Encoding, with a 1 being high for 1.2 ms and low for 600 µs and a 0 being high for 600 µs and low for 600 µs. The Sony SIRC protocol was chosen, due to it simplicity and lack of error checking bits. An example message transmission is shown in Figure 3.26.

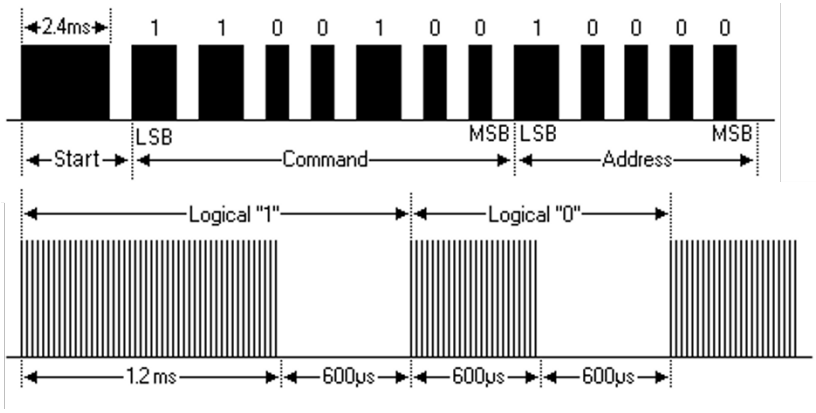


Fig. 3.26 *Top* – Example Sony SIRC message. *Bottom* – The manner in which a logic 1 and logic 0 are encoded as defined by the SIRC specification. The period $T = 600\,\mu s$ is used for the data signal, a 1 is encoded as 2T high, 1T low and a 0 as 1T high, 1T low. The lower frequency data signal is modulated using a 40 kHz carrier wave
Source: SB Projects

The process of transmitting a message was derived from an explanation given by STM, which can be found in *Implementation of transmitters and receivers for infrared remote control protocols with MCUs of the STM32F0 and STM32F3 Series*

Messages needed to be received asynchronously at any time and thus necessitated the use of external interrupts. The IR receiver were active low, so when a message was first detected, its logic level would go from high to low. Therefore, the external interrupts had to be generated on falling edges. A timer was needed to measure the amount of time between falling edges, which would indicate what logical bit had just be transmitted. [50] details a routine that can be implemented to receive SIRC messages.

Light Sensing

The light sensors were the main input to the motor control algorithms. They needed to sense where the source of light was relative to the robot. They were designed to be sampled at 100 Hz and a 10-point backward moving average was used to filter out noise, updating the reading value every 10th sample:

$$S_{avg}[n] = \frac{1}{10} \cdot \sum_{k=0}^{9} S_{sample}[n-k] \tag{3.1}$$

Motor Control

The fundamental goal of the control algorithms needed by the system were to, search and find a fire based on sensory input (from the light sensors) and surround the fire once found. Therefore, two movement states were defined. The first being a searching behaviour, based on a Brownian motion model of random movement as explained in [41], in which the robot either moves towards the goal (a source of light) or makes a random move (refer to Figure 3.27). The second being the surrounding behavior, in which the robot moves around the source of light. This behaviour had to be entered into after the robot 'realised' it had found a fire.

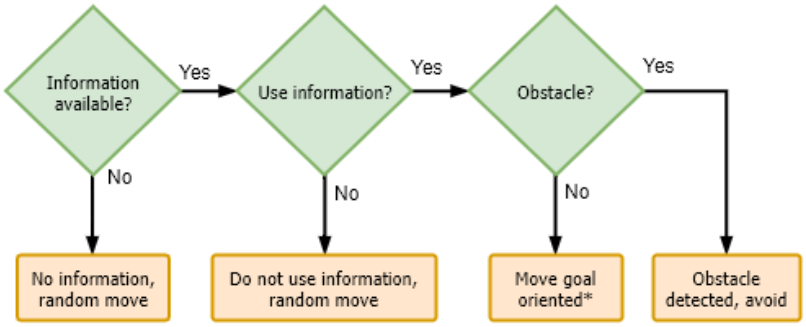


Fig. 3.27 Simplified searching routine. The move-goal orientated control algorithm is detailed else where.

Both control algorithms were designed using the ideas of Braitenberg vehicles and linear differential control [51], in which robots are modeled as having sensors and actuators, with control signals being mapped from the sensory input to the actuators using only a composition of linear functions. The sensors either directly drive or inversely drive the actuator based on the amount of the quantity sensed. In this case, there were six light sensors, but they are simplified to being two light sensors (by averaging the values from the left and right sides) and the actuators were the two wheels. A topology Braitenberg dubbed 'Love', inherently makes the robot move towards the thing being sensed, in this case a light bulb (see Figure 3.28). A modified version of this topology was used as the control algorithm for the move goal orientated behaviour and the surrounding behaviour.

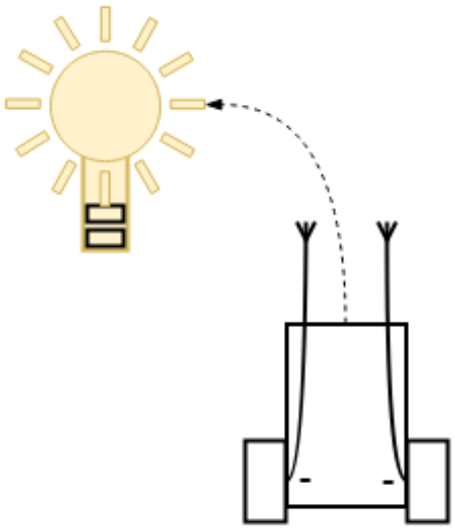


Fig. 3.28 'Love' Braitenberg vehicle topology. The negative signs show the inverted relationship between the amount of light sensed and the speed of the wheels. The robot will always move towards the light, because if there is a difference in the sensed light between the two side, the side with less light on it will be driven faster than the other, therefore self-correct. Eventually the robot will get close enough and stop.

Searching control algorithm

While the robot searches its environment, it had to move towards a light source if the signal was strong enough, otherwise it would make a random move. Additionally, the robot had to stop a certain distance away from the light and then begin surrounding it. Using the idea of the Braitenberg vehicle, the algorithm is based on a piecewise linear graph, scaled between 100 and -100 (representing the percentage of speed for the motor). The six light sensors were broken in two groups, the right sensors controlled the right motor and the left sensors controlled the left motor.

Figure 4.6 shows the control law, which was outputted by the controller and dependent on the input from the average of the three light sensors. The point C is proportional to $\frac{1}{D^2_{light}}$ and defined the point at which the robot would settle from the light source. If the robot got too close to the light, it would reverse away from it, to 'avoid damage'. The simple controller (**P**) defined in Equation 3.2 is shown to always make the robot track a light source and settle a certain distance from it (Figure 4.6 shows the equation graphically).

$$\mathbf{P}(L_{avg}) = \begin{cases} -\frac{100}{C} \cdot L_{avg} + 100 & \text{if } L_{avg} \leq C \\ 100 \cdot (\frac{M-1}{M-C} \cdot L_{avg} + 1) & \text{if } L_{avg} > C \end{cases} \tag{3.2}$$

Where: L_{avg} is the average from the three light sensors on side of the robot

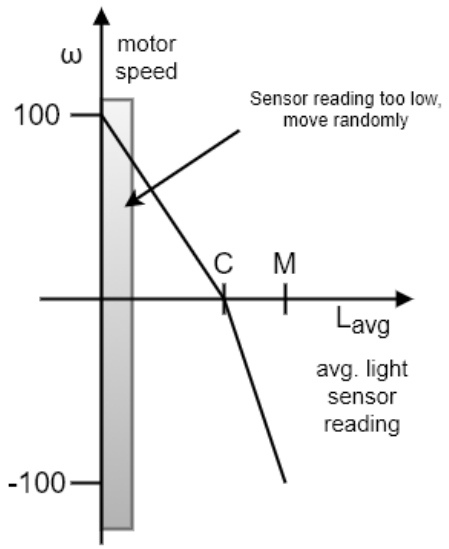


Fig. 3.29 The control law for the light finding control algorithm, showing how the motor speed is controlled based on an average of the light sensor readings, as defined in Equation 3.2. This control law is only applied if the average light sensor reading is large enough, otherwise random movement is used.

The following is a detailed explanation as to why the control algorithm would work. First a model of the robot was needed. Figure 3.30 shows the robot at a certain position, a distance (r) and angle (α) away from the goal (as the origin). The heading angle (θ), the speed (v) and the speeds (V_l and V_r) and angular velocities (ω_l and ω_r) of the wheels of the robot are shown, the robot always moves in the heading direction at a speed determined by the relative speeds of the wheels. The controller needed to make the robot move in the direction of the goal by adjusting only the speeds of the wheels and having only the light readings as an input. The controller used the control law given in equation 3.2.

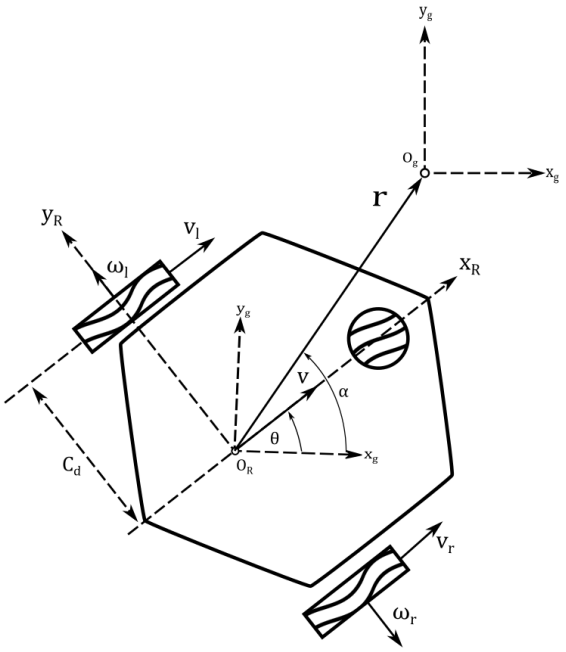


Fig. 3.30 Model for the robot, searching for and moving towards its goal at position O_g

The angular velocity, rate of change of the heading angle ($\dot{\theta}$) and speed of the robot (v) was related to the angular velocities of the wheels (ω_l and ω_r) by:

$$V_l = r_{wheel} \cdot \omega_l,\ V_r = r_{wheel} \cdot \omega_r \tag{3.3}$$

$$\dot{\theta} = \frac{V_r}{C_d} - \frac{V_l}{C_d} \rightarrow \frac{r_{wheel}}{C_d} \cdot \omega_r - \frac{r_{wheel}}{C_d} \cdot \omega_l \tag{3.4}$$

$$v = V_l + V_r \rightarrow r_{wheel} \cdot \omega_l + r_{wheel} \cdot \omega_r \tag{3.5}$$

The average of the light sensor readings for each side of the robot can be approximated using the following equation (γ, β are a proportionality constants):

$$L_{avg,l} = \frac{\gamma}{r^2} + (-\beta \sin(\alpha - \theta) + \beta)$$

$$L_{avg,r} = \frac{\gamma}{r^2} + (\beta \sin(\alpha - \theta) + \beta)$$

The sin of $\alpha - \theta$ part of the equations are used to model the difference in light readings due to one side of the robot being illuminated more than the other. For example if $\alpha - \theta = \pi/2$ (the right side is directly facing the light), then the value contributed to the

right side reading will be 2β and the left side will be 0, as it is completely not facing the light.

Now, to explain how the control law works, consider one side of the robot being illuminated more than the other (say it is the right side):

$$L_{avg,r} > L_{avg,l}$$

The controller **P** will make the right wheel slower than the left, based on the control law given in Equation 3.2:

$$\omega_l > \omega_r$$

Therefore $\dot{\theta}$ from Equation 3.4 will be negative:

$$\dot{\theta} < 0$$

Meaning the robot will turn clockwise, self-correcting the fact that the right side is illuminated more than the left

Surrounding control algorithm

The surrounding behaviour was designed to be entered into if the 10-point backwards moving average of the average light level is in a range around C, mathematically:

$$\left| C - \frac{1}{10} \cdot \sum_{k=0}^{9} L_{avg}[n-k] \right| < \epsilon \tag{3.6}$$

If the above condition was met, a state change in the robot would occur. ϵ was a value determined during implementation and testing, as it required calibration.

After the state change has occurred, a new controller (**G** and **S**) would be used to control the robots movements, defined in Equation 3.7 and Equation 3.8. At a high level, the surrounding controller worked by first checking which side of the robot was more illuminated, then setting the opposite side's speed to 50% and controlling its own side in proportion to the amount of light being sensed. Figure 3.31 shows the Braitenberg Vehicle-like topology for this.

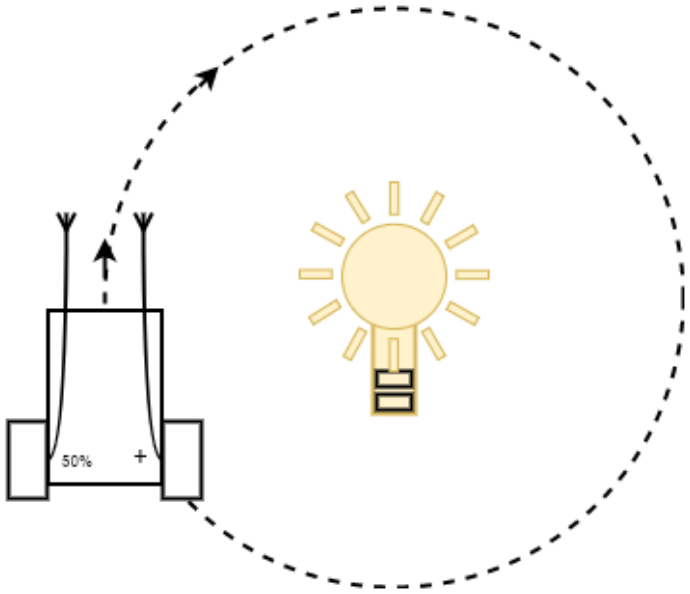


Fig. 3.31 The Braitenberg Vehicle for the surrounding control algorithm.

Say the right side was more illuminated, so the speed of the left side would be set to 50% and the right side became the controlled side. The speed of the right would be set to 50% if the average light sensor reading for the right equals C (as defined in the controller given in Equation 3.2), meaning the robot will move straight. As the robot moves away from the light source, the sensors reading will drop, therefore the right would slow down (less than 50%) and the robot would begin self-correcting turning into the light, which would then start increasing the light reading and making the right speed up (more than 50%). This oscillatory behaviour would make the robot surround any shape of light, a certain distance (C) away, on average.

More mathematically speaking, **G** and **S** are defined in Equation 3.7 and Equation 3.8. **G** was defined from 0 onwards (it was never negative) and **S** required as its inputs the average light readings, from the left and right sides discretely.

$$\mathbf{G}(L_{avg}) = \frac{50}{C} * L_{avg} \tag{3.7}$$

$$\mathbf{S}(L_{avg,l}, L_{avg,r}) = \begin{cases} \omega_l = \mathbf{G}(L_{avg}, l),\ \omega_r = 50\% & \text{if } L_{avg,l} > L_{avg,r} \\ \omega_l = 50\%,\ \omega_r = \mathbf{G}(L_{avg}, r) & \text{if } L_{avg,l} < L_{avg,r} \end{cases} \tag{3.8}$$

The interactions between robots in Multi-Robot Systems is key to the success of the system as a whole. This interactions are always based on a very simple set of rules, but

the macro scale behaviour that emerges because it, always seems complicated. For this system to work, the interaction had to make the robots move around a source of light as optimally as possible.

The interaction that was designed worked in the following way, if two robots met one another whilst surrounding a source of light, they would begin surrounding the source of light in the opposite direction. The was called the "bounce effect". This simple rule would allow the system to adapt to any number of robots in it, whilst always covering the source better if more robots were added.

If for example three robot were surrounding the light, due to the difference in the speeds at which the robots moved (inherent to mechanical systems), they were eventually 'bump' into one another (if even they all went around the light in the same direction) and begin moving the opposite direction. The state of the system (at a macro scale) over time would tend to the robots always covering the light area as well as effectively as possible, given the number of robots.

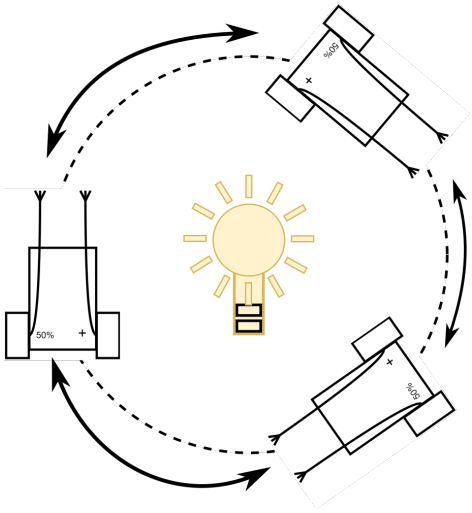


Fig. 3.32 The interaction between robots, showing how three of them would move around a single source of light.

The backward movement controller was then defined as:

$$\mathbf{S}_{bwd}(L_{avg,l}, L_{avg,r}) = \begin{cases} \omega_l = -\mathbf{G}(L_{avg}, l),\ \omega_r = -50\% & \text{if } L_{avg,l} > L_{avg,r} \\ \omega_l = -50\%,\ \omega_r = -\mathbf{G}(L_{avg}, r) & \text{if } L_{avg,l} < L_{avg,r} \end{cases} \tag{3.9}$$

3.5 Bill of Materials for The Final Designed System

Table 3.6 Bill of Materials needed to build one robot

Element	Part	Description	Quantity
Body	Perspecs	Hexagon 10 cm tip to tip	2
Motion	Corona 929MG	Servo motor	2
	Wheel	Rubber wheel	2
Microcontroller	STM32F051C6	The microcontroller	1
	Mini-bread board	The breadboard used to hold the micro	1
Voltage Regulator	LP2954	5V regulator	1
	LM3940	3.3V regulator	1
Power	Excelon 7.4V	7.4V battery	1
Light sensor	NORP12 LDR	The LDR	6
IR receiver	TSOP22	The IR receiver	6
IR transmitter	SIR333 IR LED	The IR LED	6
	2N7000	N-Channel MOSFET	6
	2N2222	NPN BJT	6
Misc	Veroboard	Used to solider components into	*
	Resistors		*
	Capacitors		*

Chapter 4

Implementation and Testing

This chapter focuses on the implementation of the designed system and detailing the procedures followed in order to verify it. Beginning with an explanation of the simulation that was written for the system, followed by an explanation about how each hardware subsystem was built and the implementation of the algorithms needed to run them. The tests conducted to verify each subsystem are then detailed. An explanation is given about how each subsystem was put together to create the system as a whole, followed by stating how the behavioural and acceptance tests were conducted.

4.1 System Simulation

A simulation was written using the Unity Game engine. Each robot was modeled as a hexagon with 6 light sensors. The lights had some radius, in which the robots could sense them. Only the surrounding and searching behaviours are demonstrated as the peer detection algorithms did not work as expect within the demo.

Run the demo.exe in the folder demo/ to view the simulation.

4.2 Subsystem Implementation

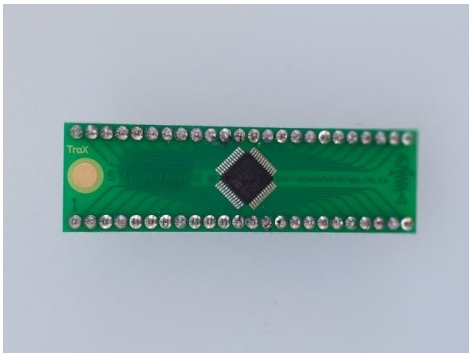
This section deals with the implementation details of building each subsystem.

4.2.1 Hardware Implementation

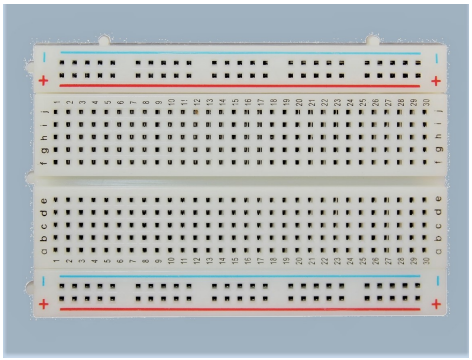
Processing and Control Subsystem

The chosen microcontroller, the STM32F0, came in a breakout board and had no pins attached to it, thus they had to be soldiered on. A breadboard was used to hold the microcontroller and provide a breakout for its pins. The breadboard had doubled sided tape attached to it, which would be used to stick it onto the frame of the robot.

The microcontroller had a few special pins that needed to be connected. It had two power V_{DD} and GND pins, both needed to be connected to the supply (3.3 V) and ground rails, the BOOT pin needed to grounded through a 10 kΩ resistor and the V_{DDa} pin had to be connected to the 3.3 V supply. The way these pins had to be connected was defined in the microcontroller's datasheet.



(a) The STM32F0 microcontroller in a breakout board, with pins soldiered to it.



(b) The bread board used to hold the STM32F0 microcontroller breakout board

Fig. 4.1

Power Supply and Management Subsystem

The LP2954 5 V and LM3940 3.3 V regulators were soldiered to Veroboard. The Veroboard had to be cut into a shape that would fit into the body of the robot. Special care had to be taken when solider the 3.3 V regulator, as it had a relatively low soldiering temperature.

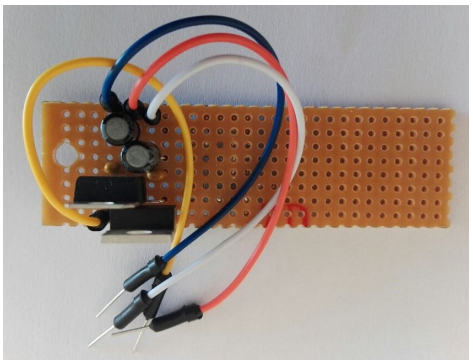


Fig. 4.2 The built power regulation circuit, with 3.3 V and 5 V outputs

Locomotion Subsystem

Both servo motors selected had to first be 'hacked' using a voltage-divider network of resistors. Due to the very compact space of the servo motor's housing, surface mount (SMD) resistors had to be used. This proved to be a very challenging exercise, as two SMD resistors had to be soldiered between three floating wires, in a confined space. The first few attempts at this caused the motor driver boards to break, but after the sixth one, each one was nearly perfect. Contact adhesive was used to glue an attachment to the wheels, which was needed to connect them to the output shaft of servo motors securely.

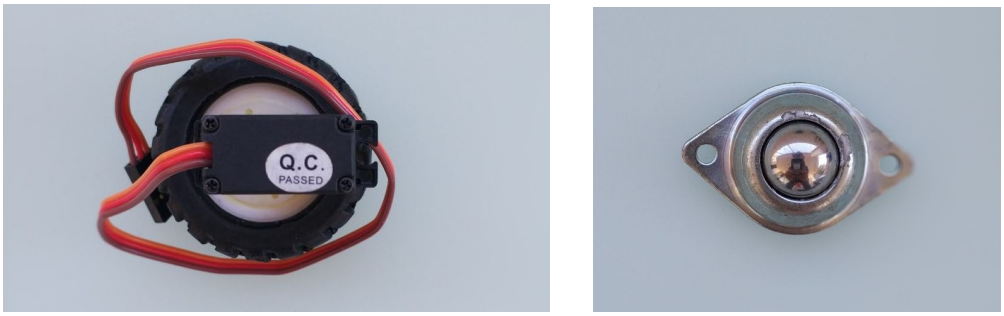


Fig. 4.3 *Left* – The servo motor connected to the wheel. *Right* – The omni-wheel used.

IR Communications Subsystem

The IR receiver was implemented using the circuit shown in Figure 3.18. Tulip pins were used to connect the IR receiver to the circuit so that it could be easily replaced if it got damaged or became faulty. Six IR receivers needed to be built in total.

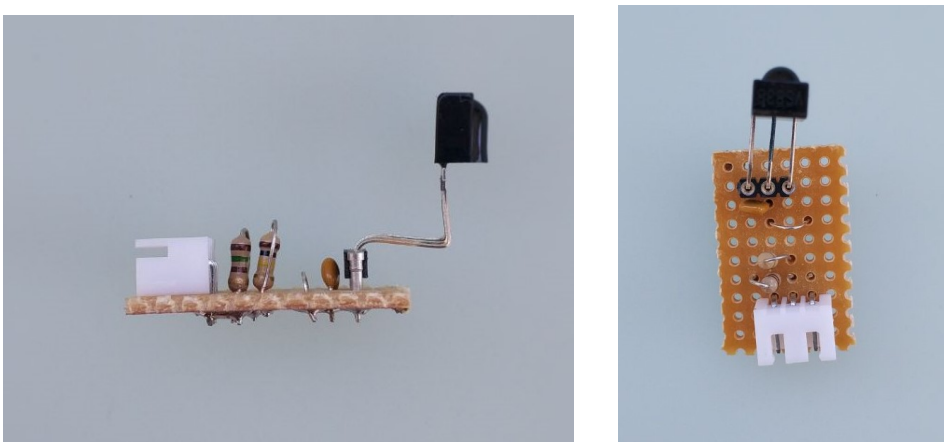


Fig. 4.4 The TSOP22 IR receiver, connected to the driver circuit

The IR transmitter was implemented using the circuit shown in Figure **??**. The signal driver part of the circuit was implemented on the breadboard of the microcontroller, for easy access to it and convenience. Six IR transmitters needed to be built in total.

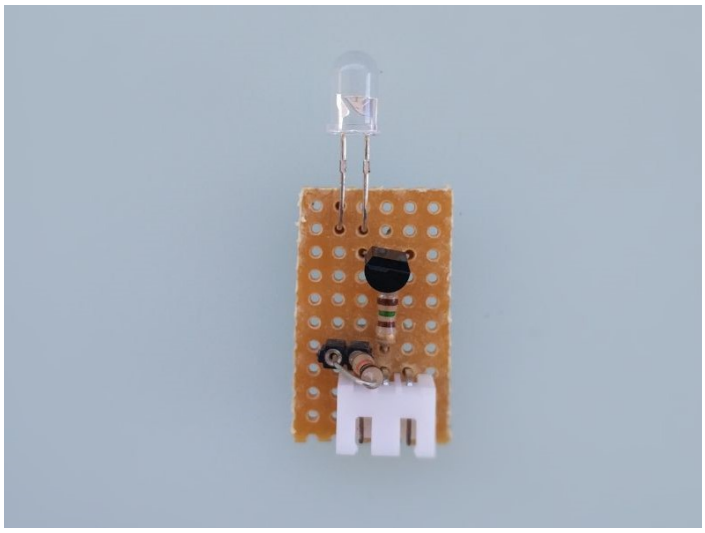


Fig. 4.5 The IR transmitter implementation, using an IR LED. The transistor was used for selecting the whether the transmitter could transmit the signal or not.

Light Sensing Subsystem

The light sensors were built using the circuit shown in Figure 3.12b. Tulip pins were used to hold the LDR so that if it was faulty, it could be easily replaced. The boards were designed to be as small as possible, as size was always a constrain on the robot's body frame. Six light sensors needed to be built in total.

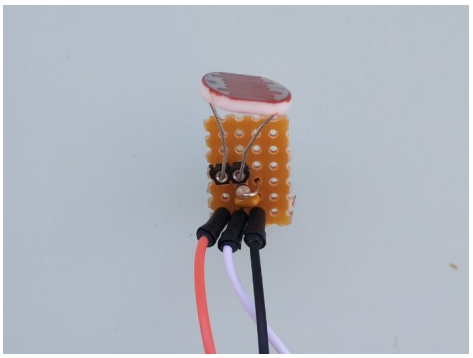


Fig. 4.6 The built light sensor using an LDR in a voltage-divider network

4.2.2 Software Implementation

Message transmission

Figure 4.7 and Figure 4.8 show the implementation of the message transmission routines, in which three stages were needed. A queue that any function could add to was used and the message at the state of the message at the top of the queue was continuously checked if it was ready to be transmitted. If this was the case, the message data (the command and address) was encoded using a Manchester encoding, where a 1 becomes a 0b011 and a 0 becomes a 0b01 and the IR LED that needed to be sent transmitted on were selected (the others were deselected0, by outputting a digital high on their pins. If the message was found to be in an error state, it would simply be discarded.

Once the message data was properly encoded and the appropriate IR LEDs selected, the timer interrupt was activate. Timer 16 and 17 were used in conjunction for transmission. Timer 17 was responsible for outputting the high frequency PWM wave at 40 kHz, which was achieved by setting the ARR value to 199, PSC value 0 and duty cycle to 25% (as specified in SIRC protocol) in the timer's settings, calculated using:

$$f_{out} = \frac{f_{clk}}{(PSC+1)\cdot(ARR+1)} \tag{4.1}$$

Where: $f_{clk} = 8\,\text{MHz}$

Timer 16 generated an interrupt every 600 µs. The encoded message data was then sent by enabling, if the current encoded bit was a 1, or disabling, if the current encoded bit was a 0, timer 17 using interrupt handler of timer 16. This effectively modulated the encoded data bitstream at 40 kHz at periods of 600 µs.

Messaging reception

Figure 4.9 and Figure 4.10 show the implementation of the message reception routine, which needed two routines. The one handled message receiving the message while it was being transmitted and the other processed the message once it had been successfully received. It was decide that only one message could be received at any one time. This was chosen to reduce the complexity of the implementation.

In the main loop, the state of the message object was checked continuously. If the received state was detected, it meant a message had successfully received. It would then be decoded by passing into a function that converted the received binary data into a command and address. The command was just a number, the all robots knew about. If the command was "Hey" it would mean that another robot was close by and control over the motors had to be subsumed. If the robot was searching, it would have to turn to avoid colliding with the other robot, or if it was surrounding the light, it would need start doing so in the opposite direction (the 'bounce' effect as defined in the surrounding control algorithm). If the message was found to be in an error state, it would simply be discarded. This routine is shown in .

An external interrupt generator (EXTI), using falling edge detection and Timer 14 were used for processes the message while it was being received. As a falling edge was detect, the pin that detected it was noted and the timer was setup to generate an interrupt after 2500 ms, to wait for the start bit. If a falling edge was detected before the timer interrupt fired, then there was some kind of error in transmission. If the timer fired, it would mean that the start bit had been successfully received. The timer was then restarted for 1200 ms. This would allow for enough time for the logic bit to be sent. When the timer fired, the pin would be sampled and the inverse of the logic level detected (as the receiver was active low) would be bit shifted into the received message data bitstream.

Light sensing

Figure 4.11 shows the implemented routine used to sample the light sensors. Timer 6 was used, set to a frequency of 100 Hz, with its ARR = 7999 and PSC = 9 (found using Equation 4.1) and the ADC, setup in a one-shot mode using 10-bit resolution.

Because there were six light sensors, an array of floats was used to store their value. Each time they were sampled, the value was added to the float array and after 10 samples the average was taken. Once the averages had been calculated, the motor control function

was called. All control signals were based on the light readings, therefore it made sense to couple them.

Motor control

The motor controller implementation is shown as pseudo-code, as it does not fit well into a flow diagram. First, a combination operator was applied to the 6 light sensor readings, which combined them into 2 values corresponding to the amount of light being sensed on left and right side ($L_{avg,l}$, $L_{avg,r}$). Then the two motor control signals were calculated based on these values.

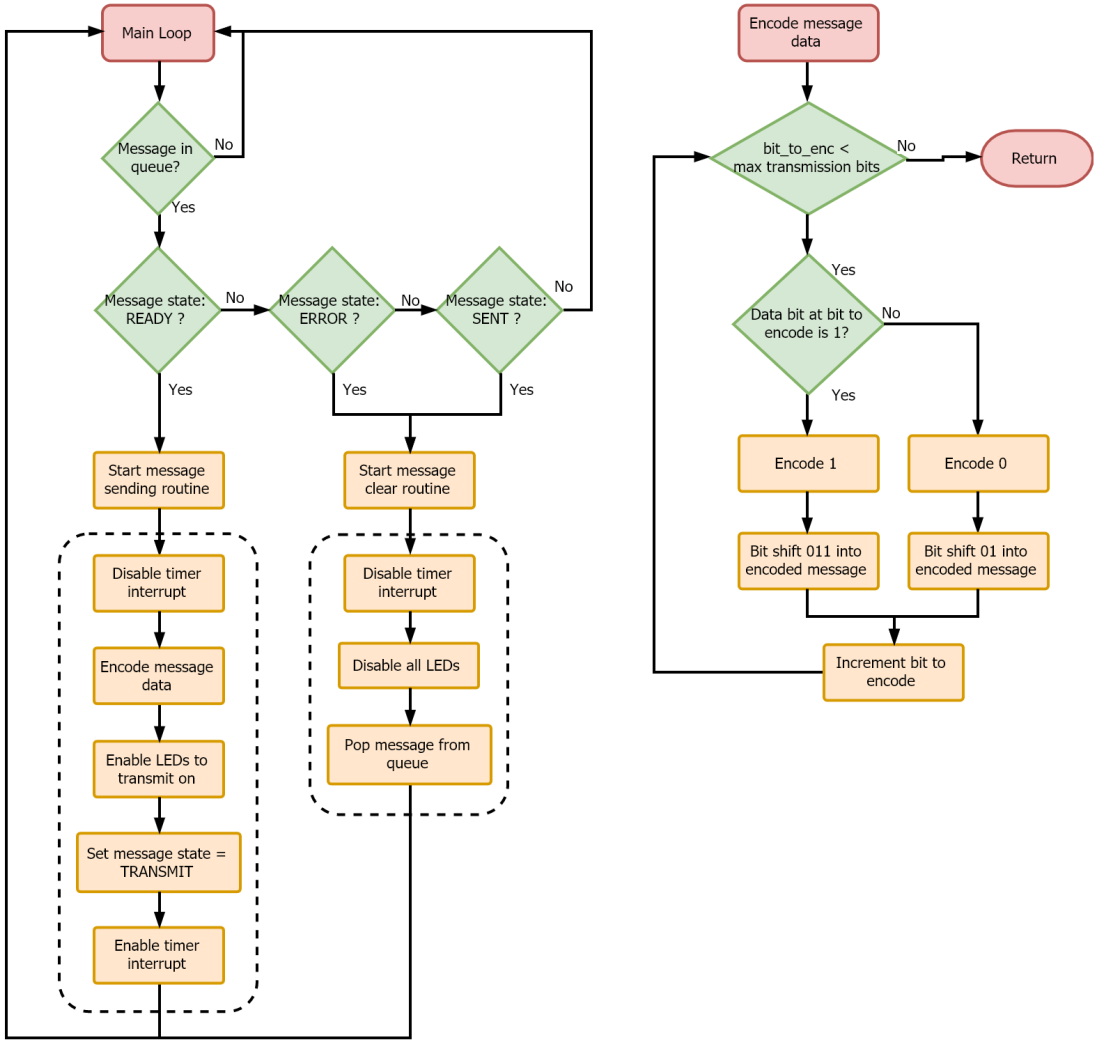


Fig. 4.7 *Left* – The message transmission routine in the main loop, used to get the message ready for transmission or clear it if in error or it has been transmitted. *Right* – Message encoding routine using Manchester encoding

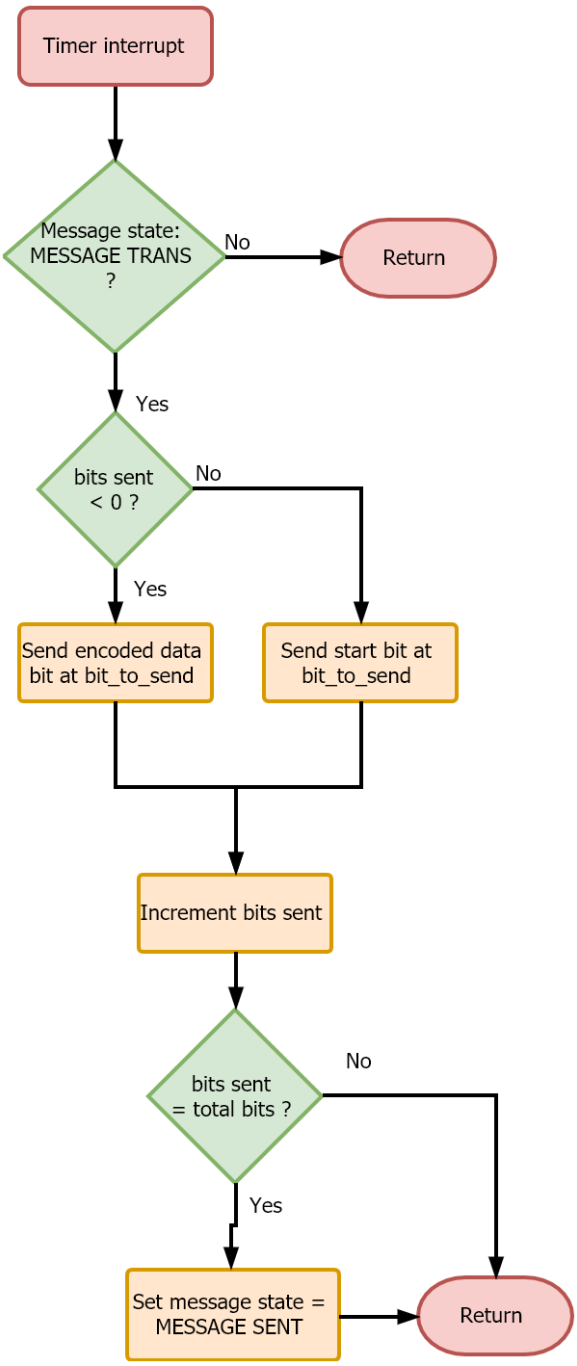


Fig. 4.8 The timer interrupt routine used to output the signal onto the LEDs.

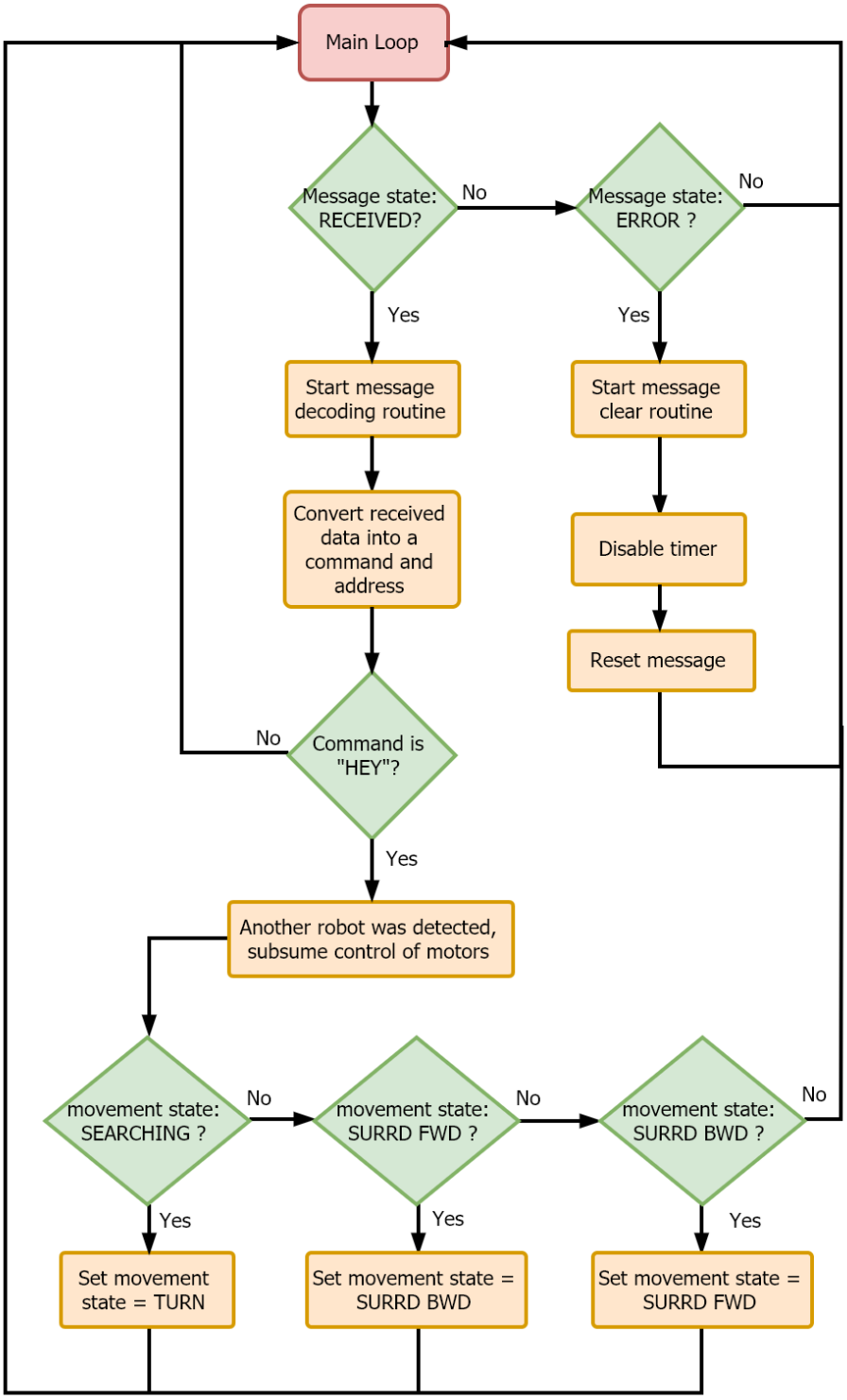


Fig. 4.9 The routines followed when a message is received. If the message is a "Hey", then another robot sent it and thus the robot must either turn to move away from it or begin surrounding in the opposite direction (which is the "bounce" behaviour in the light surrounding control algorithm).

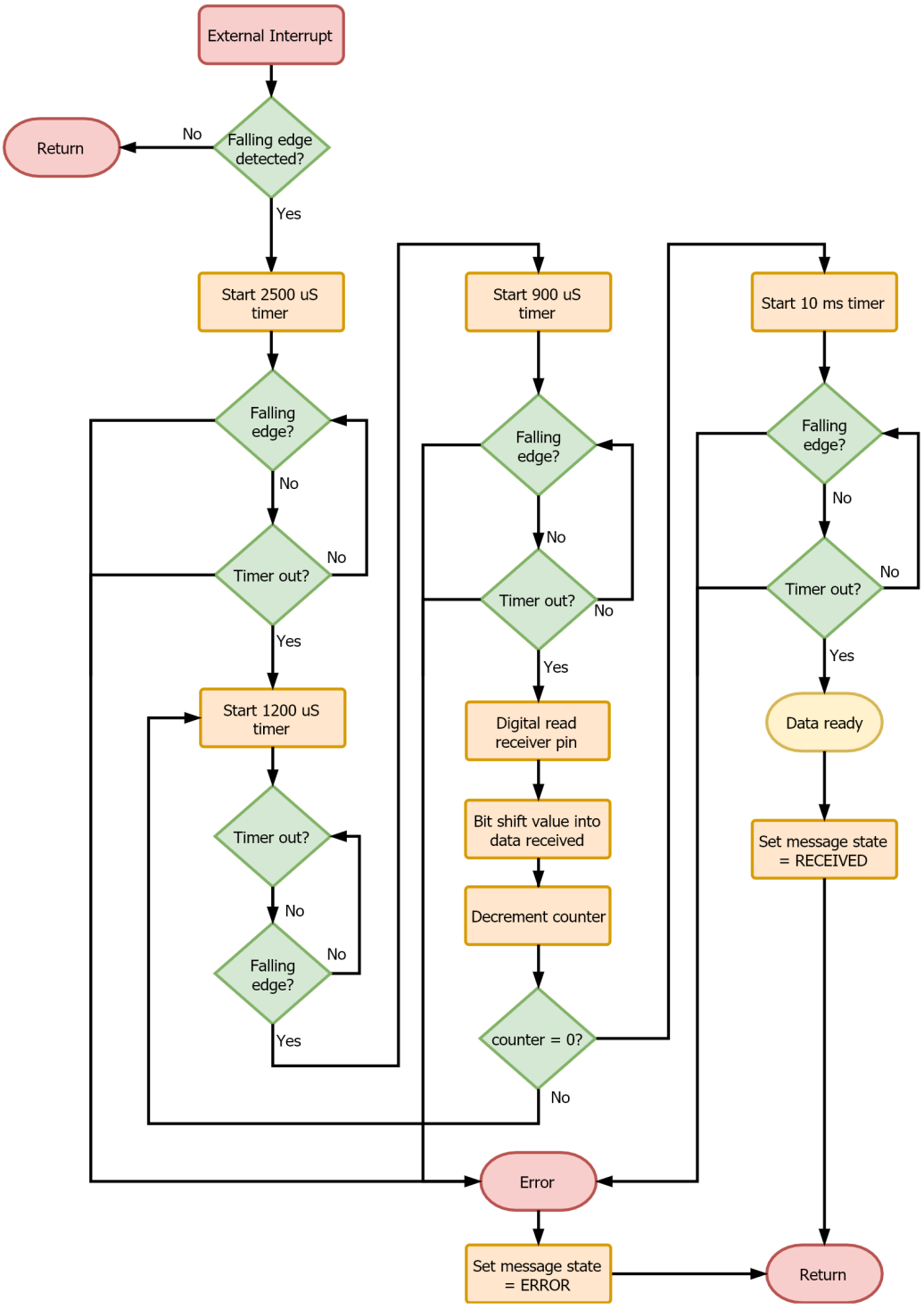


Fig. 4.10 IR receiver routine, involving the use of falling edge interrupts and a timer

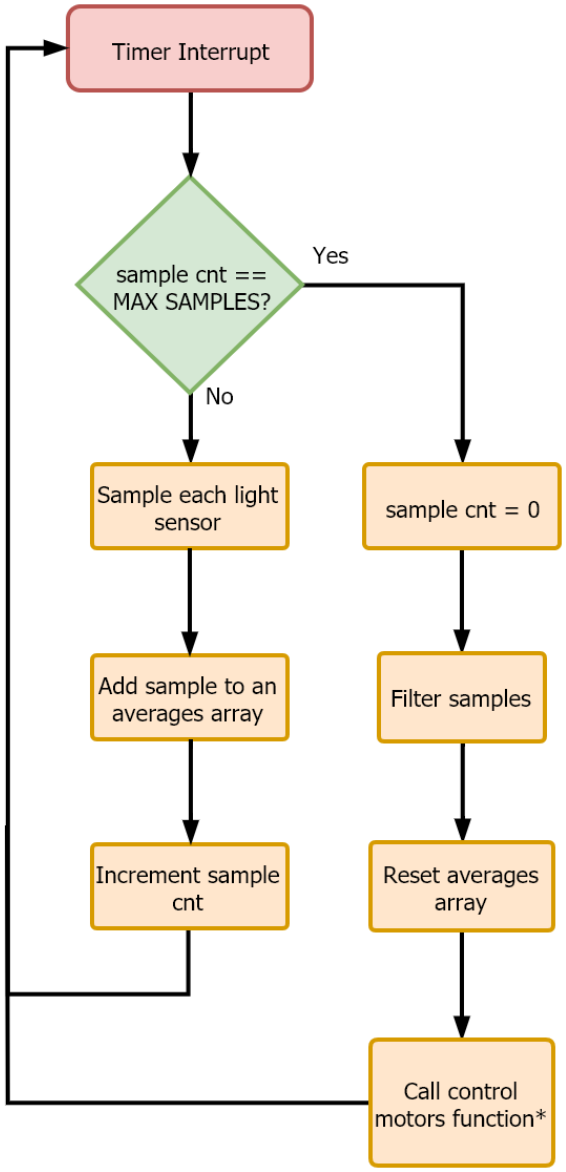


Fig. 4.11 Light sensing routine

input : The averaged light sensor readings array S_{avg} [n], where n is the index corresponding to each light sensor reading. n between 0 and 2 are the right sensor values and between 3 and 5 are the left sensor values

output : The combined average of the light sensor readings for the left and right sides ($L_{avg,l}$, $L_{avg,r}$)

avg_r = (S_{avg} [0] + S_{avg} [1] + S_{avg} [2]) / 3 ;
avg_l = (S_{avg} [3] + S_{avg} [4] + S_{avg} [5]) / 3 ;
w_r = 0 ;
w_l = 0 ;
`/* Right sensor values */`
for $i \in [0,2]$ **do**
    devi = S_{avg} [i] - avg_r;
    weight = 1 ;
    if *devi* > *0* **then**
        weight += devi ;
    end
    w_r += weight ;
    $L_{avg,r}$ += w_r * S_{avg} [i]
end
$L_{avg,r} = \frac{L_{avg,r}}{w_r}$;
`/* Left sensor values */`
for $i \in [3,5]$ **do**
    devi = S_{avg} [i] - avg_l;
    weight = 1 ;
    if *devi* > *0* **then**
        weight += devi ;
    end
    w_l += weight ;
    $L_{avg,l}$ += w_l * S_{avg} [i]
end
$L_{avg,l} = \frac{L_{avg,l}}{w_l}$;

input : The deviation weighted average of the light sensors from the left ($L_{avg,l}$) and right ($L_{avg,r}$) sides

output : The speed of the left (SpeedPercentRight) and right (SpeedPercentLeft) motors as a percentage

```
/* Control action is based on the current movement state of the
   robot                                                        */
if MovementState is SEARCHING then
    if L_avg,r < min sig and L_avg,l < min sig then
        SpeedPercentRight ← RandomInRange(SpeedPercentRight- 5,
         SpeedPercentRight + 5) ;
        SpeedPercentLeft ← RandomInRange(SpeedPercentLeft- 5,
         SpeedPercentLeft + 5) ;
    else
        if L_avg,r < C then
            SpeedPercentRight ← Lerp(100, 0, 0, C, L_avg,r) ;
        else
            SpeedPercentRight ← Lerp(-100, MAX, 0, C, L_avg,r) ;
        end
        if L_avg,l < C then
            SpeedPercentLeft ← Lerp(100, 0, 0, C, L_avg,l) ;
        else
            SpeedPercentLeft ← Lerp(-100, MAX, 0, C, L_avg,l) ;
        end
    end
else if MovementState is SURROUNDING_FWD then
    if L_avg,r > L_avg,l then
        /* Move clockwise around the light, forwards            */
        SpeedPercentRight ← Lerp(0, 0, 50, C, L_avg,r) ;
        SpeedPercentLeft ← 50 ;
    else
        /* Move counter-clockwise around the light, forwards    */
        SpeedPercentRight ← 50 ;
        SpeedPercentLeft ← Lerp(0, 0, 50, C, L_avg,l) ;
    end
```

```
else if MovementState is SURROUNDING_BWD then
    if L_avg,r < L_avg,l then
        /* Move clockwise around the light, backwards              */
        SpeedPercentRight ← -50 ;
        SpeedPercentLeft ← Lerp(0, 0, -50, C, L_avg,l) ;
    else
        /* Move counter-clockwise around the light, backwards      */
        SpeedPercentRight ← Lerp(0, 0, -50, C, L_avg,r) ;
        SpeedPercentLeft ← -50 ;
    end
else if MovementState is TURNING then
    if CNT < 10 then
        SpeedPercentRight ← 50 ;
        SpeedPercentLeft ← -50 ;
        CNT += 1 ;
    else
        CNT = 0 ;
        MovementState ← SEARCHING;
    end
end
```

Algorithm 1: Motor Controller

4.3 Subsystem Verification Tests

In this section, the verification testing procedure followed for each subsystem is described. The results of the tests are given the Subsystem Test Results section5.2 of the chapter 5.

4.3.1 Hardware Verification Tests

Table 4.1 shows the verification tests defined in the subsystem design phase of the engineering design chapter (chapter 3, 3.2) for the hardware needed by each subsystem. They state what experimental procedure was followed in order to verify the hardware was working correctly for each subsystem.

Table 4.1 Subsystem verification hardware tests

Subsystem		Testing Procedure	Figure of Merit	Test. #
Microcontroller	Code execution	Attempt to flash test code onto the microcontroller using a debugger board and verify whether the code is being executed or not	Execution of code	**SVT-HW. 1**
Power supply	Short circuits	Connect the regulators to a voltage supply and verify the current draw does not exceed the threshold amount of 1 A	Current draw	**SVT-HW. 2**
	Voltage stability	Vary the supply voltage to the regulator circuit by ±1 V and verify the output voltage does not change	Output voltage	**SVT-HW. 3**
Motion	Speed Control	Measure the speed of the wheels by counting the number of revolutions (N_{rev}) that occur in some time (t). Verify the speed changes in reaction to changing control signals	$\frac{N_{rev}}{t}$	**SVT-HW. 4**
Internal Communications	Range	Verify the IR receiver can detect a signal sent from an IR transmitter placed 30 cm away	Success of signal reception	**SVT-HW. 5**
Light Sensor	Sensitivity	Measure the illuminance of visible light incident on the light sensors using a Lux meter, then increase it by 2 lx using a light source. Do the same with IR light. Verify that the sensor reading **increased** to a change in **visible light** only.	$\Delta V_{sensor} \propto \Delta E_{vis}$	**SVT-HW. 6**

Testing the microcontroller

In accordance with **SVT-HW. 1**, the microcontroller was tested by connecting an STM JTAG debugger chip to the relevant pins on it, being: NRST to pin 7 (PF1), T_SWCLK to

pin 37 (PA14) and T_SWDIO to pin 34 (PA13). Using Atollic TrueSTUDIO for ARM, version 8.0.0, test code was written and an attempt was made to flash the code onto the micro.

Testing the Power Supply subsystem

In accordance with **SVT-HW. 2** and **SVT-HW. 3**, both voltage regulator circuits were supplied with 8 V, which is the same as the output voltage for the battery. The first thing that was check was the current draw for both regulators. If it exceeded 1 A then there must be a short in the circuit. The output voltage of each regulator was then checked using a voltmeter to see if it matched the rated output voltage. The input voltages to both regulators was then varied between 7 V and 9 V and stability of the respective regulator's output voltage was checked.

Testing the Motion Subsystem

After the servo motors were converted to velocity controlled devices and the wheels were attached to the motors, the motors were tested in accordance with **SVT-HW. 4**. The speeds of the wheel in reaction to changing control signals as well as the current draw was recorded. The test results were for calibrating the motor control algorithms.

The motors were powered using 5 V and a microcontroller was used to output the control signals needed. Both the power supply and microcontroller had to be grounded together. The speed as well as the current draw of the motors was recorded, for each 100 ms increase in the pulse width of the PWM control signal.

Testing IR communications (internal communications)

As per **SVT-HW. 5** the range of the IR communications implementation was tested by putting an IR receiver 30 cm away from the transmitter. A 100 Hz with a duty cycle of 25% was modulated with the 40 kHz carrier wave, as a simulation of the real transmission protocol. Figure 4.14 shows the experimental setup used, with a ruler showing the distance.

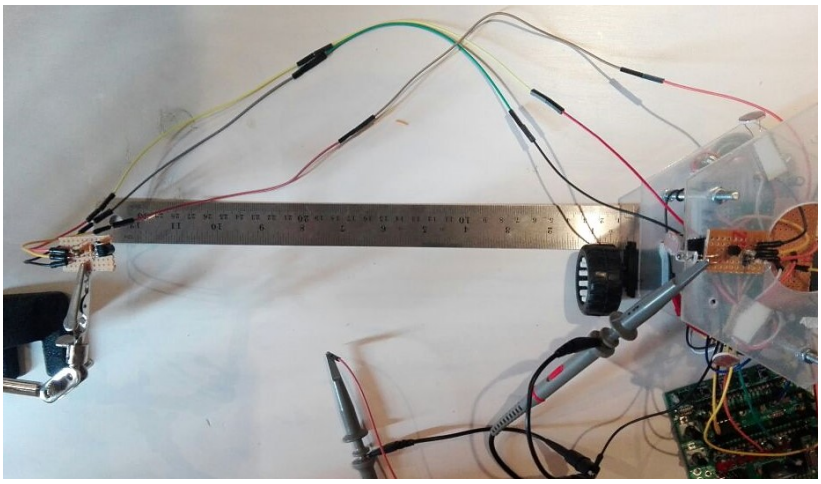


Fig. 4.12 IR communications subsystem range test

Testing the light sensors

The experiments conducted on the light sensors were in-line with the procedure detailed in **SVT-HW. 6**.

An IR LED was supplied with 5 V through a 240 Ω resistor. The sensors output voltage was recorded before and after the IR LED was made to shine onto the LDR of the light sensor. An average change of change was recorded. This test was needed because if light sensors did react to IR light, the control algorithms running on the robots would work incorrectly, therefore it needed to be checked.

The light from a mobile phone was used as the visible light source and the HS1010 Digital Lux Meter, borrowed for the lab was used as the Lux. An initial reading of the light level by the light sensors was taken and then the light source was used to increase this by 2 lx. If no noticeable change was detected, the light source was brought closer, increasing the Lux Meter reading until a stable, noticeable change was detected.

4.3.2 Software Verification Tests

Table 4.2 shows a summary of the verification procedures followed to test the behaviour of each subsystem.

Table 4.2 Subsystem verification software tests

Subsystem		**Testing Procedure**	**Figure of Merit**	**Test. #**
Internal Communications	Message Transmission	Measure the output waveform on the IR LED using an oscilloscope after attempting to send a simple SIRC message using it and verify it matches the specification.	Output waveform shape	**SVT-SW. 1**
	Message Reception	Measure the input waveform of the IR receiver using an oscilloscope during an SIRC message is being transmitted to it and verify it matches the specification.	Input waveform shape	**SVT-SW. 2**
	Message Processing	During the reception of a message, toggle the voltage of a pin while the message processing routing is executing to verify the timing	Timing of pin voltages toggles	**SVT-SW. 3**
Light Sensor	Sampling Behaviour	Sample the light sensors at 100 Hz using a timer and an interrupt. Use a breakpoint in the interrupt handler to note the sample values before and after light is applied to the sensors. Verify the sample values increased after the light was applied.	Increase in sampled values	**SVT-SW. 4**

Testing internal communications

Tests **SVT-SW. 1** and **SVT-SW. 2** were tested using the simplest SIRC message, which is the command = 0 and address = 0, with the resulting message in binary being 0b000000 0000 and the encoded messaging being 0b010101010101 01010101. The PWM signal was

outputted on PB9 (pin 46) and the LED selection signal was outputted on PA8 (pin 29). Figure 4.14 shows the experimental setup.

The SIRC message used for test **SVT-SW. 3** was command = 6 and address = 0, with the resulting message in binary being 0b000110 0000 and the encoded messaging being 0b01010101101101 01010101. 6 was chosen for the command because it has two 1's next to it binary and thus tested all cases reception (00, 01, 11, 10). PA9 (pin 30) was used to output the toggle voltage. The voltage on the pin was toggled during the reception of data phase the receiver routine. The same experimental setup was used before, as shown in Figure 4.14.

Testing the light sensor behaviour

The light from a mobile phone was used to change the light incident on the light sensors. A breakpoint was set in the interrupt handler of the timers used to sample the light sensors. The value of the samples array was recorded. The phone light was then switched on, the execution hit the breakpoint and the value of the samples array was recorded again. Atollic TrueSTUDIO for ARM, version 8.0.0 was used during this debugging process.

4.4 System Implementation

After each subsystem module was built and tested throughly, they could all be connected together. First, the frame of the robot had to be laser cut. A few iterations were made, mostly due to the size. 3 10 cm long 3 mm bolts were used to hold the frame together. The mini-bread board holding the microcontroller had doubled sided tap on it and could simply be stuck onto the robot's bottom layer.

Double-sided tape and Velcro was use on the light sensors, IR transmitters and receivers to attach them to the robots frame. The wires from each module were the connected to the bread bread. The power board was attach to the frame by bolting it to the 3 mm bolts. A hole needed to be drilled through it in order for this to work.

Figure 4.13 shows the finial implementation for the robot. Its behaviour in reaction to light and its peers needed to then be tested.

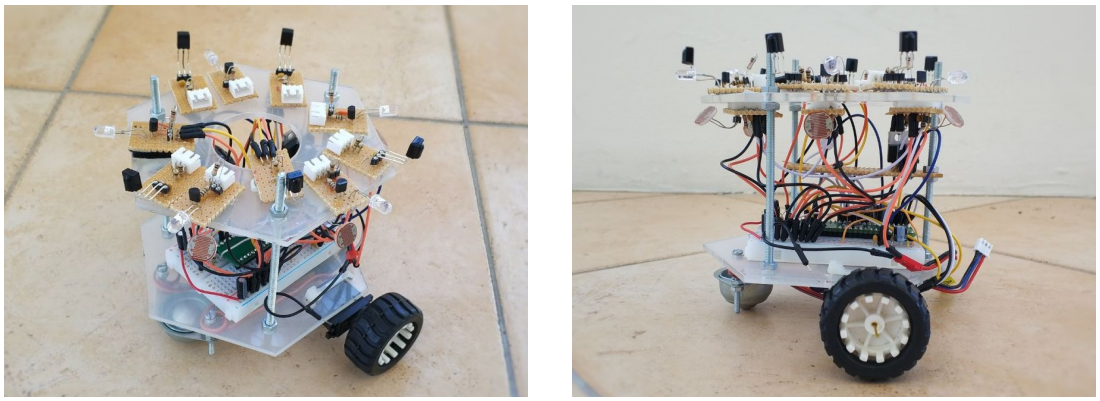


Fig. 4.13 The full implementation of the robot. It was dubbed "0xBot", because of its hexagonal shape and 0x is used as the designator for hexadecimals in C.

4.4.1 Calibration

The light sensor and the motor drive systems had to be calibrated before any system test could be conducted.

Each light sensor was placed 30 cm away from the light source and the sampled value was recorded. This was found by putting a breakpoint in the sampling method and investigating the float array.

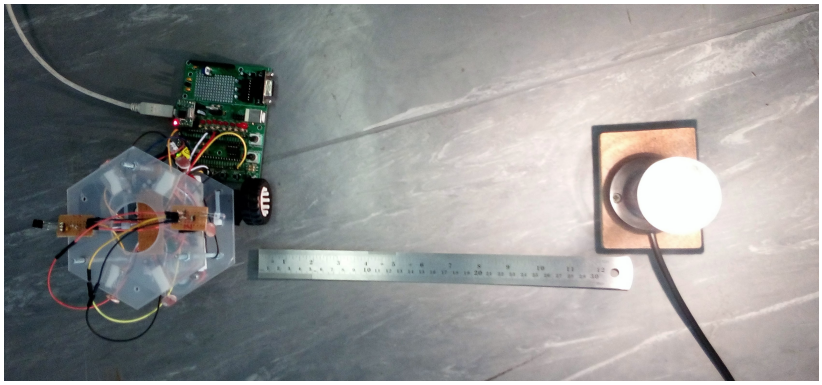


Fig. 4.14 Light sensor calibration test

The speeds of the motors were calibrated using trail and error. Speeds of 50%, 0% and -100% were used as the test values. The initial values for these speeds were determined from the motor drive subsystem hardware tests and then adjusted until the speed of the each side of the robot were equal.

4.5 System Verification and Validation

In this section high level tests are conducted on the whole system, verifying the behaviour of the system and performing the acceptance tests as defined in System Validation Plan in section 3.1.

4.5.1 System Behavioural Tests

Light Based Searching and Surrounding Behaviour

The searching behaviour of the robot was tested by placing it in a darken environment with an incandescent light bulb turned on placed on the floor. The robot was made to point away from the light. It was then left in the environment and observed.

Once the robot had found the light and began to settle around it, the surrounding behaviour was then tested. The first action required by the robot in this situation was it had to 'realise' it had found the fire and transition states into the surrounding behaviour. The robot was observed for its ability to track the light as it moved around it, which determined how well it performed this task.

The robustness of the robot's response to a change in the shape of the 'fire' was tested by moving the light as it began moving towards it and while it was surrounding it.

Fig. 4.15 The incandescent light used to test the robot's searching and surrounding behaviour.

Communication Based Peer Detection

The ability of the robots to detect a peer (another robot) was tested by holding an IR transmitter, which transmitted SIRC messages, to the robots receiver and observing if it reacted to the signal. The robot had to successfully receive the message and then act on it accordingly, based on the command sent.

Due to that fact that only one robot could be built, the peer detection behaviour could not be properly tested, but as a substitute the IR transmitter worked well. It could not, however, accurately reproduce the randomness of the robot's movement, as it was pointed straight at the other robot.

4.5.2 System Acceptance Tests

No acceptance tests could be conducted as the system did not reliably perform as intended.

Chapter 5

Results and Discussion

In this chapter the results from each test conducted during the implementation and testing phase in Chapter 4 are given. A discussion is given in each section, giving an interpretation of what the result means in terms of this thesis.

5.1 Results of System Simulation

The simulation can be found in the simulation/ folder. The robots were placed in two different environments, one with only one light and one with two. They successfully search for and surround the lights.

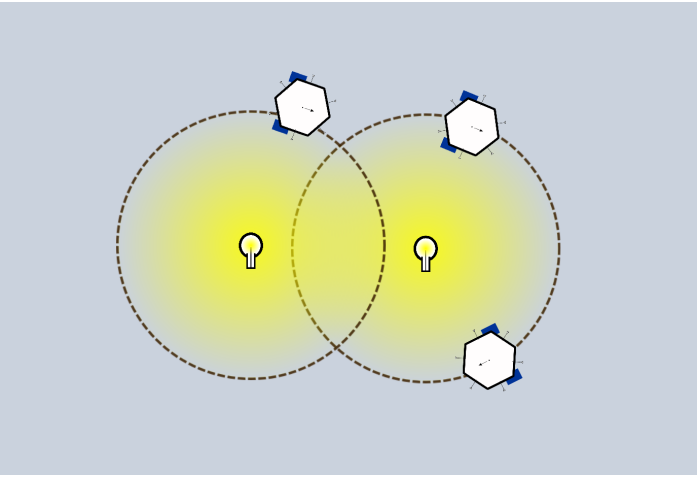


Fig. 5.1 Simulation of the robots in an environment with two lights

5.1.1 Discussion

The simulation proved the concept behind the control algorithms for the robots. Within the simulation, they robustly tracked the lights source whilst searching for it and they demonstrated good surrounding capabilities. However, the simulation also showed that more worked needed to be done in order to improve the robot's ability to surround any configuration of light. The robots could not smoothly track two light sources and began rotating sharply in the regions between the lights.

5.2 Results of Subsystem Verification Tests

5.2.1 Hardware

Microcontroller Tests

The microcontroller executed the test code that was flashed onto it and thus was deemed to be working. The execution of code was shown by toggling a GPIO pin voltage and making an LED flash.

Voltage Regulator Tests

After the 8 V power supply was connected to both both regulators, the current draw was a consistent 0.014 mA, meaning there were no shorts. After varying the power supply by 1 Volt, the outputs of the regulator remained constant within ±0.01 V.

Motor Drive Tests

Figure 5.2 shows the test results after performing the motors speed tests.

The trend shows a clear linear decrease in speed for an increase the pulse width of the PWM control signal, in the range from 1100 µs to 1600 µs. The current draw graph shows the amount of power being used by the system and correlates to the speed. After stress testing the motor by driving it as fast as it would go (around 5 rev/s) and stalling the motor, the maximum current recorded was just below 600 mA.

The fact that the drive system could vary speeds in both directions meant the subsystem was working correctly.

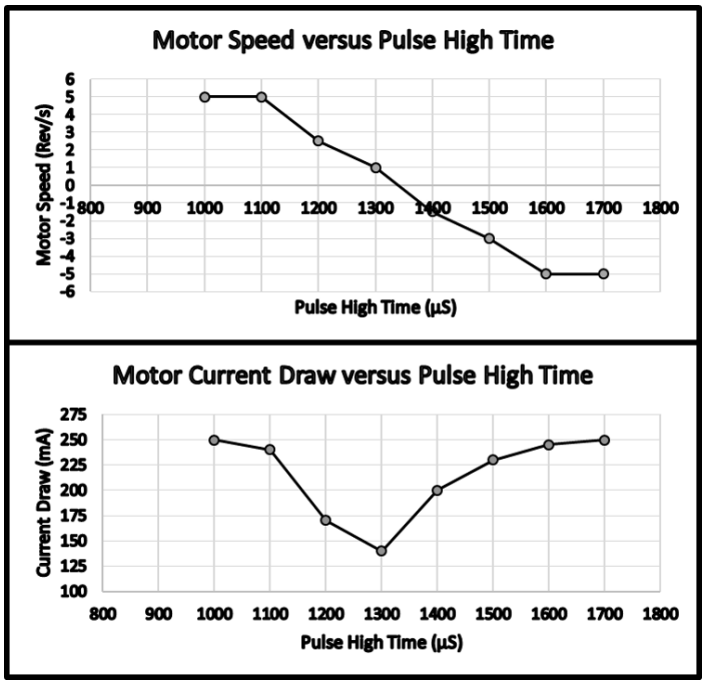


Fig. 5.2 Results of the motors tests (negative was defined as anti-clockwise). *Top* – The motor speed in rev/s for a given pulse high time. *Bottom* – The current draw from the motor for a given pulse high time.

Results of IR communication Hardware Tests

Figure 5.3 shows the results of the 30 cm test performed. The result matched the transmitted data signal exactly, with slight phase lag, but that would make no difference when interpreting the received signal.

It was found that the IR receiver was too sensitive to the signals being sent. The receiver still reacted to signals at a distance of 3 m away, even when turned to face the opposite direction. Therefore, the way to limit the range was to increase the resistance of the IR transmitter, reducing its current draw and the power of the signal. After a number of trail and error tests, a resistor of 1 kΩ was found to be optimal. The signal was not properly received at a distance of more than 40 cm.

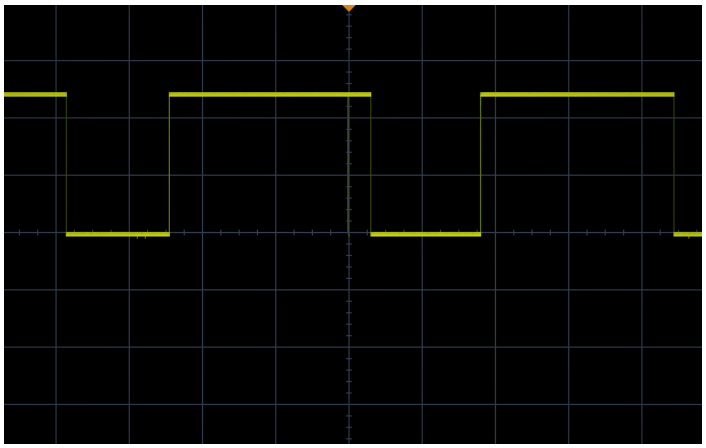


Fig. 5.3 The results of testing the IR receiver 30 cm away from the transmitter

Results of Light Sensor Tests

The initial lighting conditions of the room came to around 486 lx (measured using the Lux meter). This was increased to 489 lx (it proved to be very difficult to get a constant change of 2 lx) using the phone light. A noisy average voltage of 0.07 V was recored and was deemed to be insignificant. Only after a change of 6 lx was there a significant (and relatively constant) change in the output voltage of the light sensor, with an average of 0.17 V being recorded. Additionally, after shining an IR LED (which drew 140 mA directly onto the LDR of light sensor, an recorded average voltage change of 0.02 V, which was insignificant.

Therefore, the light senor was less sensitive than what was hoped for, but the sensitivity would be sufficient for the project. Additionally, the light sensors did not react at all to IR light, which was a results that was hoped.

5.2.2 Software

Results of IR communication Software Tests

Figure 5.4 shows a SIRC message being transmitted (bottom), with the high frequency carrier wave, and the received signal (top), going low when transmitted signal is 'high'. There was a slight phase lag in the received signal, due to the distance between the transmitter and receiver and the slew rate of the receiver. The time between the pulses was verified to be 600 µs, using the oscilloscope's width measuring tool.

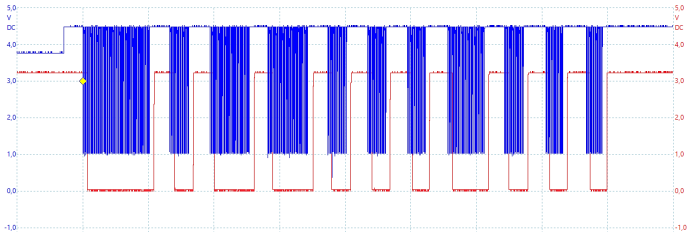


Fig. 5.4 The sent (blue) and received (read) SIRC message signals

Figure 5.5 shows the results of the test performed to verify the IR reception routine. It proved to be a very useful way to debug the IR communication implementation, as one cannot simply set a breakpoint in the code's execution, as the signal exist in the real, analog world and only last a few milliseconds.

Wherever the red line in the figure changes logic stage, shows the point at which the routine sampled the incoming signal. The red line always toggled after 900 µs after a falling edge in the blue line was detected, after the start bit was received. If the routine sampled a logical HIGH, it meant the logic value transmitted was a 0, if it sampled a logic LOW, it meant the logic value transmitted was a 1.

Using this technique shows the routine in action and that it worked as intended (after many hours of debugging). The interpreted command was a 6 with an address 0, which matched the command sent.

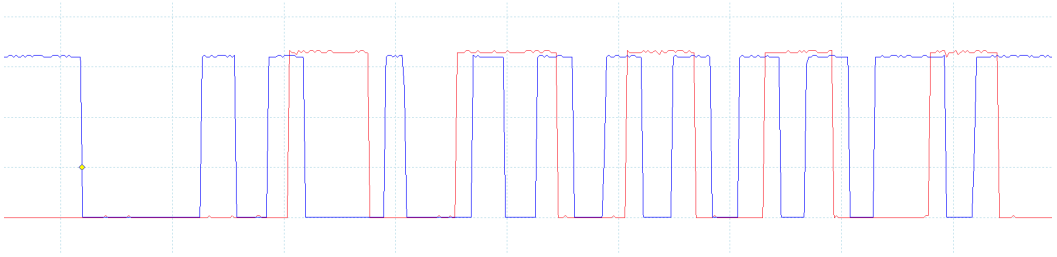


Fig. 5.5 The received signal and the voltage toggling signal, which shows when the message's logic level was being sampled. Using this technique proved to be the best way to debug the .

Result of Light Sensor Behaviour Test

The sampled light sensors values did indeed increase with an increase in the light level. The light sensor software module was therefore verified to be working correctly.

5.2.3 Discussion of Subsystem Verification

All subsystems passed their verification tests. The results of some tests lead to a modification of the implementation for the subsystem, especially for the IR communication and light sensing subsystems.

5.3 Results of System Calibration

The result of the light sensor calibration test is shown in Table 5.1.

Table 5.1 Light sensors calibration test results for the sensors placed 30 cm away from a light source

Light Sensor	Sample Value
1	719
2	770
3	869
4	800
5	770
6	865

The result of the motor speed calibration test is shown in Table 5.2.

Table 5.2 The pulse high time (t_{on}) for different speeds of the motors

Speed (%)	Left Motor t_{on} (µs)	Right Motor t_{on} (µs)
50	1165	1185
0	1330	1350
-100	1460	1483

5.3.1 Discussion of System Calibration

These calibration results show the drastic differences between physical systems and sensors. These results were used in the control algorithm to attempt to mitigate these differences and bring about coherence in the control signals.

5.4 Results of System Verification

5.4.1 Results of System Behavioural Tests

The videos in the folder video/system_behavioural_tests show the results of the final behavioural tests conducted.

Light Based Searching and Surrounding Behaviour Verification

It was found during initial testing (that wasn't filmed) that some way of mitigating the ambient light from the control algorithm was needed, as the system reacted sluggishly when there was too much external light. It was realised that the control algorithm's lower bound should be based on an measurement of the ambient light level, instead of absolute 0. This meant that an ambient measure had to take place when the robot started and then the all sampled averages would be subtracted from the ambient measure. Effectively, the formula for averaging the light sensors readings (from Equation 3.1) was changed to:

$$S_{avg}[n] = \frac{1}{10} \cdot \sum_{k=0}^{9} S_{sample}[n-k] - S_{ambi}[n] \tag{5.1}$$

Videos "searching_1" show the robot moving in a straight line, until it senses the light and moves towards it. Videos "surrounding_1", "surrounding_2", "surrounding_3" show the robot making the correct state change to the surrounding behaviour. The robot then surrounds the light source by moving around it. Although, the shape the robot makes is not a perfect circle, it still showed the main concept behind the control algorithms.

Video "light_tracking" shows the robot robustly tracking changes in the light's position, while in its searching state. Video .. shows the robot robot robustly tracking changes in the light's position, while in its surrounding state.

Although the surrounding behaviour did not work as well as expected, it did work and therefore both the search and surrounding behaviors can be marked as being verified.

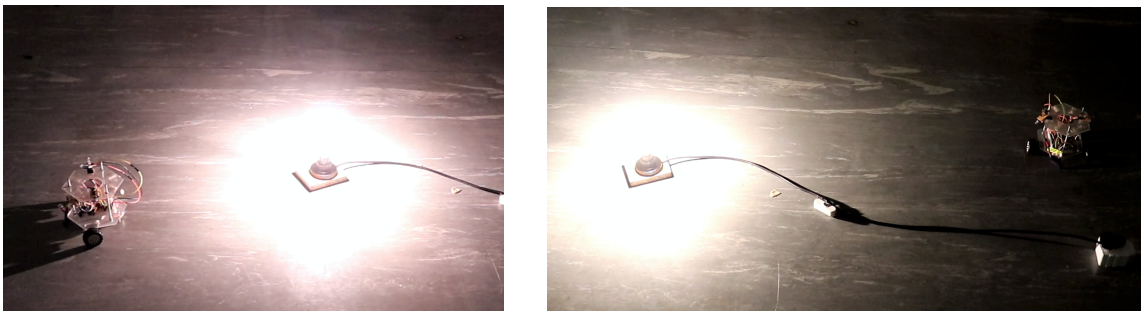


Fig. 5.6 The robot during a behavioural test, with the light source being the key feature in the environment. *Left* – The robot the searching and finding the light. *Right* – The robot surrounding the light.

Communication Based Peer Detection

Videos "peer_searching" and shows the robot turning in reaction to a signal sent to it via IR, whilst it was in its searching state. Video "peer_detected_surounding_backwards" shows the robot entering the backwards surrounding state in reaction to a signal sent to it via IR, whilst it was in its surrounding forward state.

In both cases the robot does not react very quickly, but it does show that the algorithms in theory work and thus the behaviour can marked as being verified.

After measuring the signal whilst the robot was moving, it was discovered that the motor drive system introduced a significant amount of noise into the system. Figure 5.7 shows a message that was received, but corrupted due to the noise.

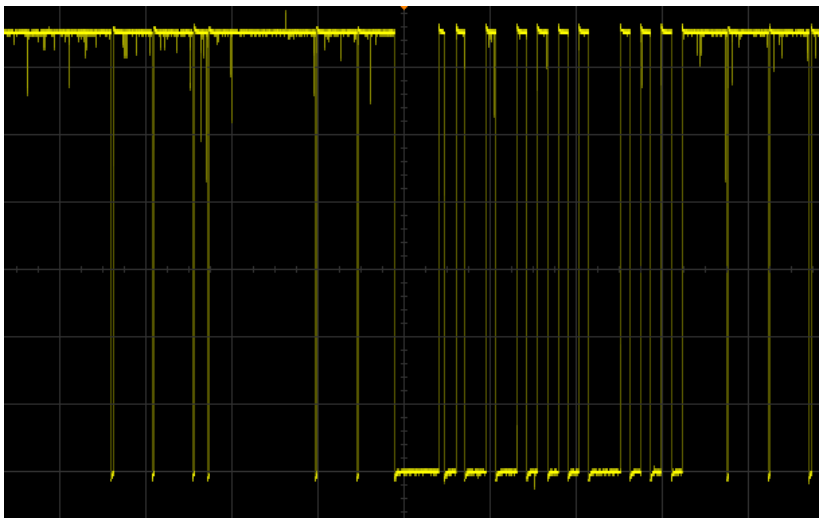


Fig. 5.7 A received message signal being corrupted by noise

5.4.2 Discussion of System Behavioural Tests

Searching Behaviour

The searching behaviour as demonstrated in the videos, clearly shows that once the robot had sensed the light, it quickly moved towards and settled around the C point. The searching behaviour worked worked well and it robustly tracked the light if it was moved, therefore proving the concept behind the robot's light searching algorithm. This means that if more lights were added to the system, it would be reasonable assume the robot would respond as expected, adapting to the changes in the lighting shape. However, the robot was not sensitive enough to light. It failed to react to the light if it was placed more than 2 m away from it.

Surrounding Behaviour

The surrounding behaviour as demonstrated in the videos, shows the robot was indeed able to move around the light, both in the backward and forward direction. However, the robot could not do so as robustly as was hoped for. There were two main issues identified with the surrounding algorithm.

The first being that it did not take into account the non-linearities of the light intensity (being a proportional to one over the distance squared) and that of the inherit exponential nature of LDR's. Therefore, the controller would over compensate in the extremes (when the robot was very far away from or very close to the light) and did not react enough in the middle regions. Slowing the robot down while it surrounded the fire did mitigate the issue, but fundamentally the issue was still there. These issues were verified in the simulated written and only after applying a significant amount of gain to the angular rotation factor made the behaviour work as intended. The second being difficulty in keeping the motor speeds calibrated. The controller may have attempted to set the speeds to the same value (for example), but the end result was a difference between the two. Even after calibrating the controller, over time recalibration was always needed.

Peer Detection

The implementation of the IR communications worked well before. SIRC messages were successfully sent and received, interpreted and processed. However, as the robot began moving, the noise introduced by the motor drivers completely corrupted the signal, making the it almost impossible to interpret. This was not planned for and no noise mitigation

circuits were added to the design, expect for capacitors between the ground and power rails.

Chapter 6

Conclusions and Recommendations

6.1 Conclusion

This thesis proved to be a successful first step towards designing and building an adaptable Multi-Robot System. Although the robot did not work completely and the surrounding control algorithms did not robustly track the light intensity radius, the system still demonstrated an ability to adapt to changes and could, theoretically, surround a light source in an efficient manner.

The results from the behavioural tests show the possibility of using simple control algorithms to create complex emergent behaviour. These types of behaviours could be applied to solve some of the most demanding challenges currently facing engineering.

Unfortunately, the system did not work sufficiently well to meet the user requirement. The acceptance tests could not be performed and therefore the project failed in that regard. This was mostly due to the unmodeled non-linearities in the robot's light sensors and in light intensity versus distance relationship. However, it did meet the key objects of the required system, which had to have:

- The ability to act automatically, based on its inputs
- The ability to adapt to changes in the light's position
- The ability to detect peers in its environment (partially met)
- The ability to seek a light source, which simulated a fire

The coordination and control of multiple robots is main focus in the Multi-Robot Systems field. This thesis attempted to implement various distributed behaviours to

demonstrate the possibly of using a MRS to help firefighters monitor and control wildfires in the future. From this point of view, the thesis was successful, with the system, in simulation and in the physical environment, being able to adapt to changes in the lighting conditions.

MRS are very difficult to create by nature, far more work needs to be done in order to produce a system that could potentially satisfy the user requirement as detailed in the beginning of Chapter 3.

6.2 Recommendations and Future Work

This project has the scope to be taken further. The robots would need to be rebuilt using better sensors, possibly photo-diodes instead of LDRs, due to their more linear-relationship to light intensity. IR receivers that can filter noise better would lead to more success in the peer based behavior. The noise in the system could be reduced by using two power supples, better regulators and filter circuits. The motor drive system should be implemented using more robust components, with a better H-bridge, which would reduce the noise generate by the motors and allow for fine grained control over the motors.

The idea of using a system of drones that could find and monitor wildfires is highly compelling. These drones would run MRS control algorithms to adapt to the changes in the fire's shape and could give firefighters accurate predictions of where the fire may be heading.

There exists the possibility of attempting to use machine learning algorithms to create the controllers needed by the system to perform some task. The robots could be trained on what they need to do and then they figure out how to perform the task. This would allow for even greater generalizations of systems like such as the one presented in this thesis, to perform not just one task in an adaptable manner, but many. The brains of each of these robots could be stored on some central computer which is accessible through a high bandwidth connection and the training could happening 'on-line', in real time.

References

[1] A. Farinelli, L. Iocchi, and D. Nardi, "Multirobot systems: a classification focused on coordination," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 34, no. 5, pp. 2015–2028, 2004.

[2] R. Brooks, "A robust layered control system for a mobile robot," *IEEE journal on robotics and automation*, vol. 2, no. 1, pp. 14–23, 1986.

[3] L. E. Parker, "Alliance: An architecture for fault tolerant multirobot cooperation," *IEEE transactions on robotics and automation*, vol. 14, no. 2, pp. 220–240, 1998.

[4] P. U. Lima and L. M. Custódio, *Multi-Robot Systems*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 1–64. [Online]. Available: https://doi.org/10.1007/10992388_1

[5] T. Arai, E. Pagello, and L. E. Parker, "Advances in multi-robot systems," *IEEE Transactions on robotics and automation*, vol. 18, no. 5, pp. 655–661, 2002.

[6] A.-S. Brändlin, "Wildfires by region: Observations and future prospects," http://p.dw.com/p/1JfrW, accessed: 2017-08-07.

[7] A. L. Westerling, "Increasing western us forest wildfire activity: sensitivity to changes in the timing of spring," *Phil. Trans. R. Soc. B*, vol. 371, no. 1696, p. 20150178, 2016.

[8] UCSUSA, "Infographic: Western wildfires and climate change," http://www.ucsusa.org/global_warming/science_and_impacts/impacts/infographic-wildfires-climate-change.html#.WYi8Noh96Um, accessed: 2017-08-07.

[9] S. Strydom and M. J. Savage, "A spatio-temporal analysis of fires in south africa," *South African Journal of Science*, vol. 112, no. 11-12, pp. 1–8, 2016.

[10] W. J. Bond and J. E. Keeley, "Fire as a global 'herbivore': the ecology and evolution of flammable ecosystems," *Trends in ecology & evolution*, vol. 20, no. 7, pp. 387–394, 2005.

[11] G. Forsyth, F. Kruger, and D. Le Maitre, "National veldfire risk assessment: Analysis of exposure of social, economic and environmental assets to veldfire hazards in south africa," *Unpublished report*, 2010.

[12] F. Kruger and R. Bigalke, "Fire in fynbos," in *Ecological effects of fire in South African ecosystems*. Springer, 1984, pp. 67–114.

[13] Working With Fire, "Fire in south africa," https://workingonfire.org/fire-in-the-south-african-landscape/, accessed: 2017-08-07.

[14] B. W. Van Wilgen, G. G. Forsyth, H. De Klerk, S. Das, S. Khuluse, and P. Schmitz, "Fire management in mediterranean-climate shrublands: a case study from the cape fynbos, south africa," *Journal of Applied Ecology*, vol. 47, no. 3, pp. 631–638, 2010. [Online]. Available: http://dx.doi.org/10.1111/j.1365-2664.2010.01800.x

[15] T. Doherty, E. Burgess, M. Maron, and R. Davis, "Bushfires are pushing species towards extinction," http://theconversation.com/bushfires-are-pushing-species-towards-extinction-54109, accessed: 2017-08-07.

[16] IUCN, "Major threats," http://www.iucnredlist.org/initiatives/mammals/analysis/major-threats, accessed: 2017-08-07.

[17] D. J. Tenenbaum, "A burning question: do farmer-set fires endanger health?" *Environmental health perspectives*, vol. 108, no. 3, p. A116, 2000.

[18] E. Frankenberg, D. McKee, and D. Thomas, "Health consequences of forest fires in indonesia," *Demography*, vol. 42, no. 1, pp. 109–129, Feb 2005. [Online]. Available: https://doi.org/10.1353/dem.2005.0004

[19] S. C. Emmanuel, "Impact to lung health of haze from forest fires: The singapore experience," *Respirology*, vol. 5, no. 2, pp. 175–182, 2000. [Online]. Available: http://dx.doi.org/10.1046/j.1440-1843.2000.00247.x

[20] South African Press Association, "Crews battling as cape town fires continue to rage," http://www.enca.com/south-africa/crews-battling-cape-town-fires-continue-rage, accessed: 2017-08-07.

[21] SAnews.gov.za-TLM, "Western cape tallies agriculture fire damages," http://www.sanews.gov.za/south-africa/western-cape-tallies-agriculture-fire-damages, accessed: 2017-08-07.

[22] I.-M. Le Roux, "Knysna fire death toll climbs to 4," http://ewn.co.za/2017/06/08/knysna-fire-death-toll-climbs-to-4, accessed: 2017-08-07.

[23] Earth Policy Institute, "Wildfires by region: Observations and future prospects," http://www.earth-policy.org/images/uploads/graphs_tables/fire.htm, accessed: 2017-08-07.

[24] N. F. P. Association *et al.*, *NFPA 921: Guide for Fire & Explosion Investigations*. Technical Committee on Fire Investigations, 2013.

[25] M. G. Rollins, "Landfire: a nationally consistent vegetation, wildland fire, and fuel assessment," *International Journal of Wildland Fire*, vol. 18, no. 3, pp. 235–249, 2009.

[26] C. Campbell and L. Dalsey, "Wildland fire fighting safety and health," https://blogs.cdc.gov/niosh-science-blog/2012/07/13/wildlandfire/, accessed: 2017-08-08.

[27] "Conversation with Robin Verrinder, UCT," personal communication, performed on: 2017-07-21.

[28] "Conversation with Samuel Ginsberg, UCT," personal communication, performed on: 2017-07-26.

[29] E. Pastor, M. Sole, J. Lopez, P. Royo, and C. Barrado, "Helicopter-based wildfire monitoring system software architecture," in *Aerospace Conference, 2010 IEEE*. IEEE, 2010, pp. 1–18.

[30] N. Pettorelli, J. O. Vik, A. Mysterud, J.-M. Gaillard, C. J. Tucker, and N. C. Stenseth, "Using the satellite-derived ndvi to assess ecological responses to environmental change," *Trends in ecology & evolution*, vol. 20, no. 9, pp. 503–510, 2005.

[31] B. H. Cheng, R. De Lemos, H. Giese, P. Inverardi, J. Magee, J. Andersson, B. Becker, N. Bencomo, Y. Brun, B. Cukic *et al.*, "Software engineering for self-adaptive systems: A research roadmap," in *Software engineering for self-adaptive systems*. Springer, 2009, pp. 1–26.

[32] P. Inverardi, P. Pelliccione, and M. Tivoli, "Towards an assume-guarantee theory for adaptable systems," in *Software Engineering for Adaptive and Self-Managing Systems, 2009. SEAMS'09. ICSE Workshop on*. IEEE, 2009, pp. 106–115.

[33] L. Parker, "Current state of the art in multi-robot teams, in distributed autonomous robotic systems," 2000.

[34] C. M. Macal and M. J. North, "Tutorial on agent-based modelling and simulation," *Journal of simulation*, vol. 4, no. 3, pp. 151–162, 2010.

[35] S. Russell, P. Norvig, and A. Intelligence, "A modern approach," *Artificial Intelligence. Prentice-Hall, Egnlewood Cliffs*, vol. 25, p. 27, 1995.

[36] M. Glavic, "Agents and multi-agent systems: a short introduction for power engineers," 2006.

[37] L. Iocchi, D. Nardi, and M. Salerno, "Reactivity and deliberation: a survey on multi-robot systems," in *Workshop on Balancing Reactivity and Social Deliberation in Multi-Agent Systems*. Springer, 2000, pp. 9–32.

[38] M. J. Mataric, "Designing emergent behaviors: From local interactions to collective intelligence," in *Proceedings of the Second International Conference on Simulation of Adaptive Behavior*, 1993, pp. 432–441.

[39] T. Balch and L. E. Parker, *Robot teams: from diversity to polymorphism*. AK Peters, Ltd., 2002.

[40] M. Brambilla, E. Ferrante, M. Birattari, and M. Dorigo, "Swarm robotics: a review from the swarm engineering perspective," *Swarm Intelligence*, vol. 7, no. 1, pp. 1–41, 2013.

[41] H. Hamann and H. Wörn, "A framework of space–time continuous models for algorithm design in swarm robotics," *Swarm Intelligence*, vol. 2, no. 2, pp. 209–239, 2008.

[42] M. Rubenstein, C. Ahler, and R. Nagpal, "Kilobot: A low cost scalable robot system for collective behaviors," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 2012, pp. 3293–3298.

[43] M. Rubenstein and W.-M. Shen, "Automatic scalable size selection for the shape of a distributed robotic collective," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*. IEEE, 2010, pp. 508–513.

[44] C. Virágh, G. Vásárhelyi, N. Tarcai, T. Szörényi, G. Somorjai, T. Nepusz, and T. Vicsek, "Flocking algorithm for autonomous flying robots," *Bioinspiration & biomimetics*, vol. 9, no. 2, p. 025012, 2014.

[45] J. McLurkin, A. McMullen, N. Robbins, G. Habibi, A. Becker, A. Chou, H. Li, M. John, N. Okeke, J. Rykowski, S. Kim, W. Xie, T. Vaughn, Y. Zhou, J. Shen, N. Chen, Q. Kaseman, L. Langford, J. Hunt, A. Boone, and K. Koch, "A robot system design for low-cost multi-robot manipulation," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, Chicago, IL, USA, September 14-18, 2014*, 2014, pp. 912–918. [Online]. Available: http://dx.doi.org/10.1109/IROS.2014.6942668

[46] M. Dorigo, E. Tuci, V. Trianni, R. Groß, S. Nouyan, C. Ampatzis, T. H. Labella, M. Bonani, F. Mondada *et al.*, "Swarm-bot: Design and implementation of colonies of self-assembling robots," IEEE Computational Intelligence Society, Tech. Rep., 2006.

[47] F. Mondada, M. Bonani, X. Raemy, J. Pugh, C. Cianci, A. Klaptocz, S. Magnenat, J.-C. Zufferey, D. Floreano, and A. Martinoli, "The e-puck, a robot designed for education in engineering," in *Proceedings of the 9th conference on autonomous robot systems and competitions*, vol. 1, no. LIS-CONF-2009-004. IPCB: Instituto Politécnico de Castelo Branco, 2009, pp. 59–65.

[48] "What is it about bees and hexagons?" May 2013, [online] http://www.npr.org/sections/krulwich/2013/05/13/183704091/what-is-it-about-bees-and-hexagons. [Online]. Available: http://www.npr.org/sections/krulwich/2013/05/13/183704091/what-is-it-about-bees-and-hexagons

[49] F. Mondada, E. Franzi, and A. Guignard, "The development of khepera," in *Experiments with the Mini-Robot Khepera, Proceedings of the First International Khepera Workshop*, no. LSRO-CONF-2006-060, 1999, pp. 7–14.

[50] PIC Projects, "Sony sirc infrared protocol," http://picprojects.org.uk/projects/sirc/sonysirc.pdf, accessed: 2017-11-07.

[51] V. Braitenberg, *Vehicles: Experiments in synthetic psychology*. MIT press, 1986.