Department of Computer Science

The Design and Evaluation of a Cooperative Handheld Robot for Alleviating Cognitive Workload

Christopher Daniel Meehan

A dissertation submitted to the University of Bristol in accordance with the requirements of the degree of Master of Science in the Faculty of Engineering

October 2016 | CSMSC-16

Declaration:

This dissertation is submitted to the University of Bristol in accordance with the requirements of the degree of Master of Science in the Faculty of Engineering. It has not been submitted for any other degree or diploma of any examining body. Except where specifically acknowledged, it is all the work of the Author.

Christopher Daniel Meehan, October 2016

Executive Summary

Typically, the field of robotics can be categorised by two established areas; external robots and wearable technology. However, in recent years, a new field has emerged which bridges the gap between these domains, namely *Handheld Robotics*. The primary focus of this field is the augmentation of human-oriented tasks, combining the intuitiveness offered by humans with the precision and processing power of robotics. Research into this area has been conducted at the University of Bristol, where devices have been developed and early investigations carried out into the cooperation with the user.

It is thought that through pre-acquired task knowledge and effective cooperation with the user, such devices may facilitate improved performance in users who lack skill or training in the operation at hand. This project aimed to develop the field further, by investigating a simplified method of designing a novel handheld robot and evaluating the potential for such a device to alleviate cognitive workload in mentally demanding tasks. This is a specific area that has not received much focus in the previous investigations. As such, the project can be classified as 50% Type I (software/hardware development) and 50% Type II (investigation). In order to achieve the aim, a number of steps were undertaken:

- A novel device was designed and built, incorporating a highly flexible cable-driven continuum structure for added versatility in unstructured environments and withstanding collisions.
- A control system was implemented through a developed program in order to actuate the various components of the device using an external PC and on-board microcontroller.
- An algorithm was formed around obtaining an appropriate input to a kernel regression method in order to obtain a solution for the servomotor angles to manoeuvre the tip of the robot towards a target in real-time.
- A number of experiments were designed and undertaken in order to evaluate the device and whether it could successfully cooperate with the user to provide cognitive assistance in an otherwise mentally taxing task, potentially improving performance.

These steps were carried out successively, with the program being developed and upgraded in the background, leading up to the final evaluation. This was performed to test the theory that handheld robotics possess the potential to reduce perceived mental workload of the user and improve task performance. The evaluation itself took the form of a user study in which ten participants undertook two tasks; one with mental assistance and one without. Prior to this, a number of performance investigations were carried out in order to assess the robot's suitability to operation in practice. A number of key findings were made throughout this project, contributing to existing research into the viability of handheld robotics in general:

- The robot proved suitably responsive for the required application of use with a human operator.
- The device and the algorithm it employed proved capable of effectively gesturing a desired target location to a user, within the limits required for the user study.
- The completion time of the tasks was shown to exhibit a statistically significant reduction through receiving mental assistance from the robot, with a t-test result of $\mathbf{5.02}(> 2.10)$.
- While a small number of user errors were observed in the non-assisted task, none occurred when mental assistance was provided.
- The overall perception of workload from the users was shown to significantly reduce through using the device in an otherwise cognitively demanding task, with the difference in combined score yielding a t-test result of $\mathbf{6.71}(> 2.10)$. In particular, the mental effort was shown to exhibit the largest reduction.

Given these findings it was concluded that handheld robots do possess a strong potential to improve task performance and reduce subjective workload through the provision of cognitive assistance. This further highlighted the usefulness of such devices, warranting deeper investigation into their potential real-world applications.

Acknowledgements

Firstly, I would like to thank my supervisor, Professor Walterio Mayol-Cuevas, for his patience and guidance regarding the field of robotics in general and the associated experimental methods.

I would also like to thank Austin Gregg-Smith, a former PhD student of a similar project, for his invaluable advice in both designing and controlling the robot.

I would like to thank Richard Grafton for providing me with so much direction regarding the electrical components of the design, through which I feel I have learned a considerable amount.

Finally, I would like to thank all of my family for their continued support throughout the project.

Contents

1 Introduction

1.1 Contemporary Robotics

Advancements in the field of robotics have been plentiful over recent years, stemming from increased knowledge of the subject, improved processing power and developments in mechanics. As such, the capabilities afforded by robots have also risen, enabling a range of applications that, until recently, would not have been possible. As the area has progressed since its inception, it has naturally split into two distinct categories; external robotics and wearable technology, as illustrated in Figure 1.1. They are each characterised by the respective workspace in which they operate. This branching has appeared to come about due to typical robotic task requirements lending themselves to one of the two forms.

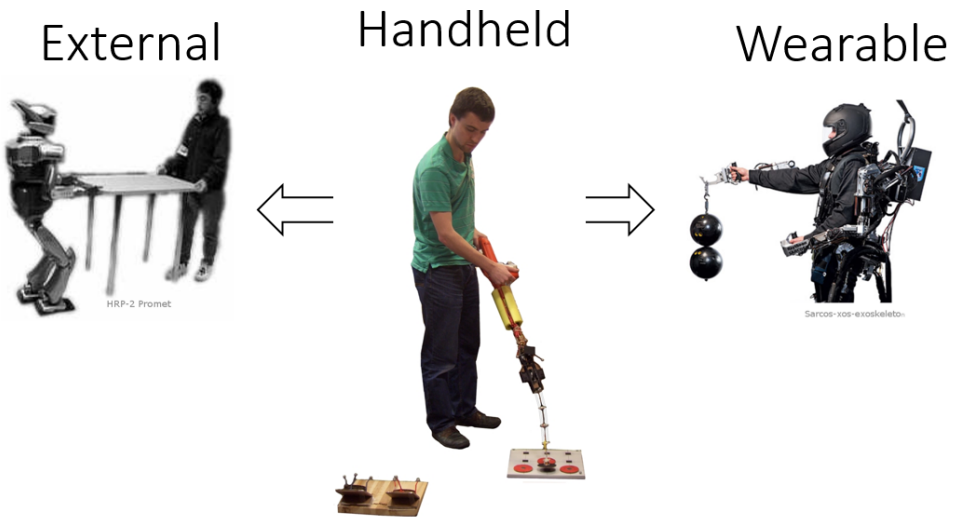


Figure 1.1: Two established forms of robotics bridged by new 'Handheld' domain, image adapted from [1].

External robotics are distinctly separate from humans in terms of physical proximity. They are becoming increasingly common in large-scale industrial spaces due to their ability to provide improved efficiency and task precision through autonomy. A typical application is the automotive industry, in which production has been accelerated through the introduction of external robotics, performing specific assembly and painting operations, mostly in detached workspaces from the human workers. The other discrete area of robotics is referred to as 'wearable technology', which, as implied by its name, is concerned with robotics that operate within the immediate personal space of the human user. This domain is typically associated with the augmentation of human actions, such as increasing physical strength or permitting certain motions that those with disabilities have been rendered incapable of. A well known use of this technology is the 'exoskeleton', which can be tailored to accomplish either of these two functions through structural support and/or actuation.

However, both of these forms of robotics comprise inherent limitations, reducing their suitability to certain situations. As such, certain physical tasks exist that cater to neither domain. One case where this is evident is the *tool space*. This area encompasses the typical range of tasks that humans have long since employed passive tools to accomplish. Typically, this space is closely associated with flexibility, where unique tools can be used to deal with the same general task in various environments, or with completely different tasks altogether. Conversely, external robotics are mostly limited to a rigidly defined workspace, performing very specific and repetitive tasks that they were designed for. Whilst wearable technology is more suited to this flexibility, it suffers from its limited adaptability to different users. Devices such as exoskeletons are very costly, and are uniquely tailored to individual body dimensions. This lack of generality equates to reduced applicability to the tool space, in which it is essential that other users are able to operate the tool when a certain individual is absent, or where a tool is marketed to a wide variety of operators.

1.2 Motivation

A proposed solution to bridge the gap between the two established forms of robotics, providing performance improvements in areas that neither are suited to, is *Handheld Robotics* [1]. This area aims to combine the key benefits of robotic efficiency and augmentation of human performance afforded by both fields, whilst maintaining the flexibility from human intuition in traditional tool use. Early work in this area has hinted at a wide range of potential applications where handheld robotics may excel, particularly in the fields of assembly, construction and maintenance/quality control. The fundamentals of their operation is that they cooperate with a human user, who in turn, positions the robot in a general location in which it can perform its desired operation, leading to improved task performance. This sequence of events however, relies heavily on a key aspect of the device; its *task knowledge*.

Task knowledge refers to the robot being context aware, that is, it 'knows' what operations need to be performed as well as the current state of progression through the task. This can act to share or alleviate the cognitive burden on the user, so that they may focus on other aspects of the task. Most importantly, it enables a degree of cooperation between the operator and device, in which the robot indicates its preference of actions to be performed by the user, such as simply repositioning it. In turn, the more precise or demanding operations are carried out by the device. Such a cooperation can lead to reductions in perceived workload by the user, be it physically or mentally, as well as improved task performance. This amalgamation of human intuitiveness and robotic efficiency through task knowledge may lead to unprecedented flexibility in the field of robotics.

The *Handheld Robotics Project* at the University of Bristol [2] has begun to lay the groundwork for the development of the field, investigating the characteristics of two initial prototypes, namely the degree of cooperation with the user and the kinematics in general. This early work has warranted further investigation into the field, highlighting the potential for handheld robotics in general. As such, the consideration of new device configurations, whether in terms of cost, lead time, or general mechanical design, is worthwhile. In addition, the way in which people interact with such robotic devices in 3D space under various cognitive loads is still open to investigation, and is crucial for the assessment of handheld robots and their viability in real-world applications. It was therefore decided to design and build an alternative robot, in part due to lack of available devices in which to perform such work, and attempt to evaluate its performance with regard to providing mental assistance through cooperation with the user; a relatively unexplored aspect. It is hoped this will shed more light on the potential of handheld robotics, as well as highlight areas of strengths and future research.

1.3 Aims and Objectives

The overarching **aim** of this project is to investigate a simplified method of developing a research oriented handheld robot, and evaluate the potential for such a device to provide mental assistance to a cognitively demanding task. As such, a new robot must be designed, assembled, and integrated with a control system so that it is fully operable. The functionality must be such that it enables a degree of performance evaluation. In order to implement this aim, a number of **objectives** must be met:

1. Design a new handheld robot for 3D printing and the facilitation of open platform release via handheldrobotics.org. The exact design will be achieved through an iterative prototyping process comprising physical testing. Once fully designed and dimensioned, the robot should be constructed and assembled to connect to an off-the-shelf handle.
2. Develop a control and calibration system so that the actuation of the robot may be executed with minimal delay via an external PC. This will entail employing an on-board microcontroller to receive commands and actuate the device accordingly.
3. Design and implement a general algorithm so that the robot may move to its intended target, or indicate the general position to the user when it is out of reach.
4. Perform a user study to assess the robot's ability to indicate a desired position, cooperating with an operator to alleviate cognitive workload in an otherwise mentally demanding environment.

1.4 Added Value

The design of this device, while taking inspiration from existing work, will focus on incorporating novel features to offer potential improvements over previous applications. Specifically, this will involve improving the range of motion over comparable designs, and adding extra versatility in terms of operating in unstructured environments, where withstanding collisions is essential. This will take the form of employing an ultra flexible and impact resistant continuum structure as the actuated component of the device. Although work has been undertaken regarding the benefits of such structures in robotics, there has been little with regard to their application in the handheld domain.

The specific application of this new device is focused around cooperating with the user to alleviate cognitive workload during mentally demanding tasks. This aspect has not been a key focus in existing work pertaining to handheld robotics. Instead, past studies have concentrated on the cooperation between the user and device with an emphasis on the physical component. The notion that such devices hold the potential to assist and improve task performance, particularly in users who are comparatively unskilled or untrained, can be investigated more thoroughly if the mental factor is also considered. This is due to the range of potential applications for handheld robots not being limited to operations based solely around physical activity. Rather, it is thought that mentally taxing tasks, also in a 3D environment, could benefit from such a cooperative device.

1.5 Document Overview

This thesis serves to cover the work accomplished throughout the project. Firstly, a background and context will be provided for existing areas of robotics in which the handheld domain may incorporate ideas, or provide a potential to improve in the future. On top of this, a review will be given of the current status of handheld robotics in general, with a focus on the work undertaken at the University of Bristol, which pertains most closely to this project.

The process associated with the design of the robot will be covered, including the control system, interfacing with an optical tracking system and the algorithm central to the robot's ability to perform its required operations. An overview will be given of the program that was developed to run the robot in practice. The stages involved with evaluating the performance of the device, and its capacity to cooperate with the user to alleviate mental effort, will be also be detailed. Finally, the corresponding results of the experiments will be presented and discussed to ascertain the capability and potential of such a device with regard to the intended purpose. Given these unexplored aspects, and considering the relatively small existing body of work concerned with cooperative handheld robotics, it is hoped this project will make a significant contribution to the breadth and depth of the field in general.

2 Background and Context

2.1 Robotics with Humans In-The-Loop

Although robots have become vastly more capable over recent years with technological improvements, many tasks still require some form of human assistance in order for the objectives to be adequately met. Robotic devices can be perfectly equipped to handle operations which involve precise motion or repetitive tasks, which may be present physical challenges to humans [3]. However, they still lack ability in areas which require the high-level cognitive reasoning that comes naturally to humans. As such, there exists many potential applications in which humans can be in-the-loop for robotic operations. This applies to both forms of robotics that the handheld area sits between; external robots and wearable technology.

2.1.1 Safety

Perhaps the most important factor in these situations is how the robot receives and processes user intent. An input from the user should be analysed by the robot in terms of importance and the choice of subsequent actions should be made appropriately. A prime example of where this is of paramount concern is in vehicular operations, where potential accidents could be catastrophic. Anderson et al [4] investigated a semi-autonomous hazard avoidance framework and tested it in a real-life driving environment, where the human driver received full control during non-threat situations. However, in times of threat detection, the control system became fully autonomous and guided the car to the safety. In this situation, the robot assumed the expertise of hazard assessment and reaction time, whereas the human provided general navigation that was accepted by the system under normal circumstances. This is one extreme, where the human input is essentially ignored under certain conditions, to preserve the safety of the user and those nearby.

At the other end of the spectrum, human input must be immediately responded to in order to act safely, where it is assumed the human possesses greater threat detection. Thus safety to both the human and robot must be positively ensured [5]. In industrial applications, the robot often operates in an isolated workspace from the human, where an extreme human input would constitute an emergency button or entering the workspace via a safety door/light gate, which the robot must respond to by halting all physical actions for human safety [6]. However, it is becoming increasingly common for the robot and human to share the same workspace, occasionally cooperating in a physical manner, such as the humanoid robot presented by Ott et al [7] and illustrated in Figure 2.1, where parts can be passed between the robot and human.

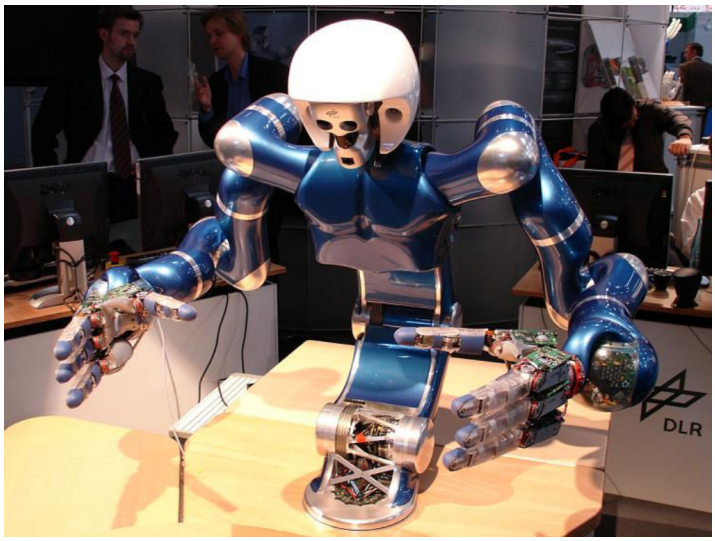


Figure 2.1: Two-Arm Humanoid for Dexterous Manipulation, image from [7].

Existing operational requirements for collaborative robots have been specified in IS0-10218, which focus on defining the upper limit of either the tool centre point velocity, the dynamic power, or the static force. However, as discussed by Haddadin et al [5], these values are merely based on heuristics. As such, more specific severity indices were introduced, taking from biomechanically motivated quantities. In the case of this project, the robot was designed with the consideration of limiting the required dynamic power to suitable levels to be deemed safe as a collaborative robot. This included lightweight and flexible material design choices for moving parts, greatly reducing any risk to a human user.

2.1.2 Autonomy Study

The reliability offered by human controllers can rarely be matched by autonomous agents. Similarly, the efficiency afforded by robots is highly attractive in physical tasks. Finding a strong balance between the two is essential in successfully accomplishing a task. Heger et al [8] investigate a mode of shared control, referred to as "Sliding Autonomy". Fundamentally, this is an analysis to find the optimal flow of control between autonomous agents and a human operator for a given task, where the former observes the operation and intervenes when necessary. A scenario was performed based on this model where three robots worked cooperatively with an operator to assemble a physical structure, with numerous failure modes deliberately implemented. The results indicated that the cooperation compromise resulted in task efficiency similar to a fully autonomous mode, while maintaining a level of reliability associated with teleoperation systems (remote human control). The constant attention of the human, without being cognitively demanding, improved the success rate, while acting as a safety net for failures that the robot alone cannot recover from.

There are plenty of other investigations into the study of variable autonomy, such as in [9], which looks into variable interactions in space missions, and in [10], which concentrates on the costs (time delays and other effects) of such transfer-of-control systems. Although this area of interchangeable control is mostly concerned with external robotics, it is still relevant to the handheld domain. As covered in [1], examining the varying degrees of autonomy that a handheld device may exhibit for a given task is necessary to locate the control set-up to give optimal performance. A different approach was taken to shared control in [11], where the efficiency offered by both members in the cooperation was measured at each time instant from the point of view of reactive strategy. The experiment in this paper focused on the shared navigation of a Pioneer AT robot, where the commands of both the human and the robot were weighted and linearly combined, resulting in a single motory operation.

The results of this experiment showed that the weighting between the two inputs led to performance which, although not matching that offered by each member alone, did seem to combine the benefits of both when taking a range of scenarios into consideration. This is a rare situation in which the task at hand could be performed with equal skill by both human and robot, and caters to the speciality of neither. However, the approach of taking a weighting between the two inputs is interesting.

It may be worth investigating such a method with handheld robots in the future. This would entail combining the two signals of human intention and task planning, and deciding the action based on the weighting for the given scenario. This could add a degree of flexibility to the robot, adapting to new changes in the task, as identified by the human, or compromising between the two signals. With such a case, different settings may be explored by tuning the weighting given to each signal on a scale with a range of values, as opposed to just three discrete levels of autonomy as was the case in [1]. However, such an investigation would require an extensive user study, with participants performing tasks repeatedly for a large variety of weightings. This is outside the scope of what is hoped the current project will achieve.

2.1.3 Collaboration with Assistive Robotics

Rather than replacing human operations, robotics can be employed to assist in them, maintaining human flexibility while increasing productivity. One such case where this is applicable is in the manufacturing industry, particularly assembly. Robots with highly specific purposes have long been used on assembly lines to increase output, performing simple and repetitive tasks at a faster rate than is possible with human workers. However, humans are frequently used for more complicated procedures that require motion in multiple degrees of freedom. The adaptability and range of motion of human workers is often unrivalled compared with traditional robotics, as well as being much more available.

A sensible compromise is investigated in [12], where robotics are used to provide assistance to a human-oriented assembly task. A Hidden Markov Model (HMM) based analysis was used so that the robot may predict the next stages of the task, lending assistance by passing the required components to the user at the appropriate time, so that they may continue with the operation. As a pre-requisite for any useful interaction between the human and the robot, the robot must first be trained. In order to accomplish this, the assembly tasks were performed solely by the human whilst sensors were used at various hand and body positions, allowing task motions to be adequately tracked. A similar study, focused primarily on the timing of assistance, contains an in-depth description of this preliminary set-up [13]. The task comprised the assembly of a six-block tower, bolting the blocks together in varying manners. Once the data was collected for the human-performed task, the results from the sensors were used to form the model.

The model generated in [12] could then be transferred to the robotic platform JAHIR (Joint-Action for Humans and Industrial Robots). The hybrid assembly platform for the context outlined in the paper is illustrated in Figure 2.2. Both the robot and the human share a designated workspace, whereas the storage areas for the parts are only accessible to the robot. An interaction area exists which overlaps with the human workspace, in which the robot may assist by transporting required parts to within human reach. There are numerous options for estimating the progress of human-performed tasks, one such technique is that used for the preliminary experiment; strategically placed accelerometers and microphones. This method is outlined in detail in [14], where most workshop activities could be identified at 84.4% accuracy. In the case of the preliminary experiment in [12], pertaining solely to the assembly task, greater accuracies were achieved.

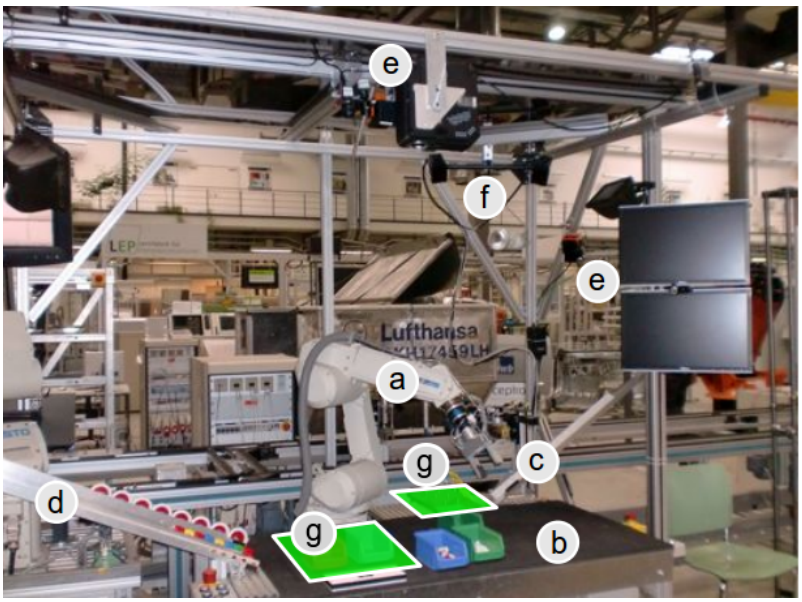


Figure 2.2: JAHIR assembly demonstration platform - a) Industrial Robot, b) Shared Workspace, c) Slide for Assembly blocks, d) Slides for Other Parts, e) Sensor Devices for Input/Output, f) ARTrack System g) Storage Space for Robot. Image from [12].

However, in reality, it is not feasible for assembly workers to wear a plethora of wired sensors while performing tasks. The physical and mental comfort suffers greatly under such conditions, with the sensors creating a feeling of intrusiveness. A viable solution to this issue is the use of occupancy maps, which are commonly employed in robotics to determine position and motion, providing a robust approach to spatial perception and navigation problems. This technique, covered in depth in [15], was implemented for the robot assisted assembly task in [12], using the three-dimensional velocity, acceleration and jerk as input to the individual HMMs. This idea of measuring movement in physical space without excessive wiring is an essential component to the viability of handheld robotics, which have previously employed optical tracking from an external device as a solution. Other sensing methods will be explored in the future, with the aim of improved practicality in real-world scenarios.

This occupancy map method allowed for a recognition accuracy of 92.26% for the right hand, comparable to the recognition rate of the reference experiment that relied on sensors. Therefore, for the vast majority of the time, the system could accurately predict the task progress, and respond with the correct actions accordingly. The key here is that the robot must be context-aware. This enables much greater cognitive capabilities and a higher standard of interaction with the user. Cognitive Technical Systems (CoTeSys) is a research group with a focus on sensing and actuation in the physical world. As explained in [16], this allows the robot to exhibit behaviours that suggest it can 'learn'. The foundations of cognition in humans and animals are used to develop cognitive models, which can then be applied when designing control systems for more specific technical applications.

The group uses a cognition-based perception-action closed loop as a basis for investigating cognition in technical systems, as illustrated in Figure 2.3. This architecture focuses on five main areas; perception, action, knowledge, learning and reasoning. The model has potential to be directly applicable to the field of handheld robotics. Task knowledge and the way in which the device receives and processes data from both the environment and the user, responding accordingly, bears many parallels to the purpose of handheld robotics in general. Thus, such an architecture may be considered as a reference when designing applications for handheld robotics.

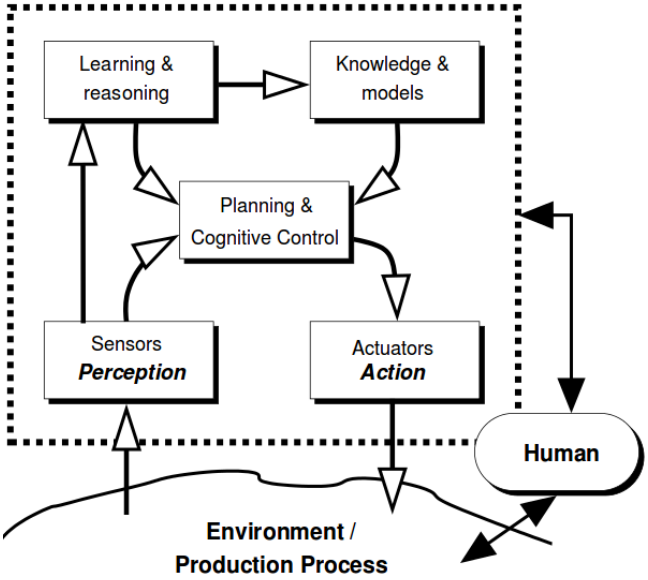


Figure 2.3: Cognitive System Architecture: Perception-Action Closed Loop, image from [16].

2.2 Robotics in Building Environments

From a fully autonomous perspective, using robots in construction environments is becoming an increasingly attractive proposition. Although machines have been utilised to assist in construction since the industrial revolution, there has typically been little in the way of automation. With advancements in robotics, they may be incorporated into building applications to reach unprecedented levels of speed and efficiency. Numerous investigations and demonstrations have been carried out for robots performing assembly operations from a set of known requirements, corresponding to 'task knowledge'. However, most of these examples are based around two-dimensional systems, such as manipulating blocks to construct flat structures, as in [17], not considering height.

The building and construction industry has been identified as an area of strong potential for handheld robotics. It is thus used as background to highlight examples where the introduction of a new domain between the two established forms of robotics could prove worthwhile and beneficial. External robots, although able to possess task knowledge, are limited in this respect by their ability to deal with changing requirements, as well as their on-site mobility. Of course, the limitations of wearable technology remain their lack of adaptability to different users. These issues in particular may have a solution in the form of handheld robotics. On top of this, the notion of unskilled workers being able to cooperate with such a device to accomplish a task may also have applicability to the construction space.

2.2.1 Mobile Robots

Petersen et al [18] presented a hardware system in which a mobile robot manipulates blocks in three-dimensional space to match desired structures. As well as this, they also produced a high-level control scheme that could enable multiple robots to autonomously share the workload of a user-specified building task. The results were successful and the prototype robot assembled structures that were greater in height than itself, doing so via climbing, navigation and manipulation. This was accomplished without the need for complex sensing and control mechanisms through the design of the simplified interconnecting structural blocks and the way in which the robot interfaces with them, as illustrated in Figure 2.4. It has been suggested that there exists a future opportunity for a series of handheld robots and respective human operators collaborating to accomplish tasks of much higher complexity than possible from just a single device. This would bear parallels to the work in [18], in which the task knowledge is shared and updated between multiple devices accordingly.

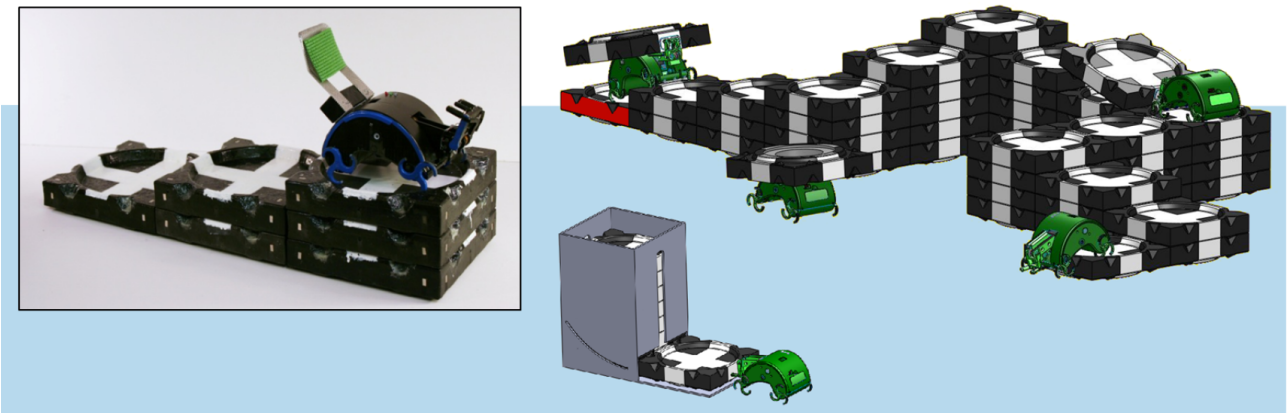


Figure 2.4: Hardware system in which robot builds specified structures. Inset: Physical image of prototype atop six-block stricture. Outer image: Concept for system goal, in which multiple robots act together to distribute workload of a much larger structure. Image taken from [18].

This idea of designing the structure blocks in a passive manner so that they naturally fit together, reducing sophistication in orientation, provides a neat solution to overcoming positional errors. In this project, the physical nature of the tasks not of primary concern, as the main focus is the evaluation of potential for mental assistance. However, physical motion will always be an element for robotic devices, and future work will likely address this to a greater degree. In such case, more physically oriented research will likely take strong inspiration from such self-aligning building blocks as observed here. A typical feature that is simpler to incorporate is the use of magnets for physical interaction with targets. Target blocks in this project will be of a much larger scale due to the realistic accuracy expectations of the simplified design, so cost reduction in terms of material and manufacture will also be a large consideration. Although this project is based around cognitive workload alleviation, it is likely that the experiments will still incorporate abstractions from a building scenario. This is due to the nature of building tasks, requiring working memory and logical sequencing to complete, both of which possess strong potential to benefit from mental assistance.

2.2.2 Scene Knowledge

One of the key considerations in robots performing construction tasks is how they navigate around the environment. The careful pre-planning of the construction environment is often necessary, so that autonomous agents can be informed of valid paths to avoid undesired collisions. Whole studies have been dedicated to this area, such as in [19]. In this paper, the autonomous agents are given prior knowledge of the environment as a whole and not just specific paths. This behaviour-oriented approach encourages vision/ultrasonic sensing to detect obstacles, reducing reliance on dead-set routes. Being more than fifteen years old, such techniques have become more established now, as in [18]. In many situations however, there is still a relatively strong dependency on a well defined environment, beyond just the relative locations of source/targets.

This reliance on environmental knowledge, as well as pre-planning of paths, is one of the areas in which handheld robotics can excel, providing an elegant solution to the guidance issues experienced by separate and autonomous robotics. As an operator can transport and orientate a handheld device in any way they intuitively see fit, much greater flexibility is provided. The need for a rigidly defined environment and careful pre-planning, which is imperative to external robots, is vastly reduced with handheld robotic tools. Of course, fixed obstacles can still be made known to the device in order to provide the most efficient movement. However, as the user is in full control of the general location of the robot, the user actively recognises and avoids obstacles with minimal cognitive effort. Hence this is an element of construction based robots that is of far less concern when designing new applications for handheld robotics, further outlining their potential as a whole.

2.3 Handheld Robotics

2.3.1 Medical Devices and Other Examples

The field of handheld robotics is largely unexplored in comparison to the established areas of robotics in which it sits between; fully external and wearable technology. The vast majority of the existing work in the handheld domain is focused on the medical industry. Numerous surgical devices exist within this area, supplementing the skill and performance of the surgeon during precise operations. Payne and Yang [20] review the existing and emerging trends in handheld medical robots, stating that the technical challenges differ from grounded robots in that miniaturising the devices is essential, whilst having multiple degrees of freedom is often less important.

An example of such a device is the handheld Master-Slave Combined Manipulator (MCM), developed by Matsuhira et al [21], which offers added functionality to laparoscopic surgery that is not possible with conventional instrumentation. The design entails the addition of two degrees of freedom to conventional forceps, forming a cable-driven slave hand controlled by a master grip mechanism that houses two servo motors, as illustrated in Figure 2.5. The added yaw axes and the gripper *end effector* (the component at the tip that interacts with the environment) then enable internal suturing tasks, where potentiometers on the master grip interface with a notebook computer for controlling the motors. Other handheld robotics in the medical field include cutting implements that incorporate haptic feedback (indicating force exertion) [22]. There is also a significant amount of research regarding self-stabilising devices to compensate for physiological tremors, such as the "ITrem" [23], which utilises sensing, filtering and manipulation to enhance surgical precision, or the mechatronic cup holder that alleviates the symptoms of Parkinson's patients [24].

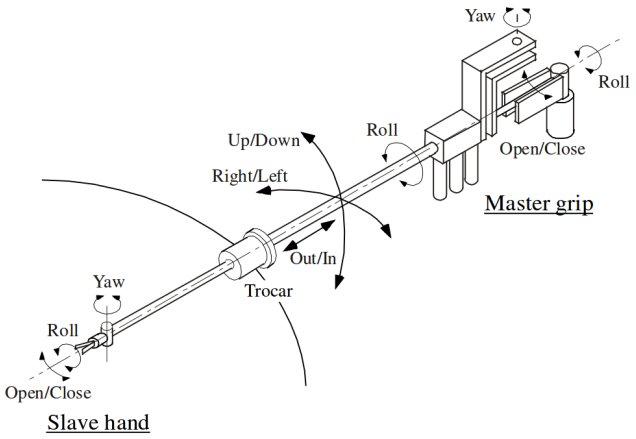


Figure 2.5: Master-Slave Manipulator for laparascopic surgery, image from [21]

However, up until 2015, there had been little research in the way of handheld robotics in the *toolspace*. Tools have become a fundamental part of the way humans interact with the physical world, enabling the completion of tasks which would otherwise propose physical or mental challenges. They often allow operations to be performed that would otherwise be impossible, or simply reduce the task duration and workload, thus leading to an improved quality of life. One such example of work in this area is a handheld automated welding device, which enables welding to occur only when positioned correctly [25]. Although this constitutes an application of handheld robotics in the tool space, it is constrained by similar limitations to those present in the examples cited above when looking to classify this as a generalised handheld robotic tool that cooperates with the user. The aforementioned novel devices were designed with relatively few degrees of freedom and with very specific tasks in mind. This means that they are limited to single tasks, and cannot easily be adapted to alternative applications. The other key factor associated with the existing devices is their limited 'task knowledge', reducing the potential for cooperation between the device and the user.

2.3.2 University of Bristol Handheld Robotics Project

Recent work undertaken at the University of Bristol has formulated the 'Bristol Handheld Robotics Project' [2]. This has focused on the potential of multi-degree-of-freedom handheld robotic devices operating in the tool space. A critical component of the research has been the capability of the robot to assume task knowledge for the desired procedure. The foundation for this branch of work was laid by A.Gregg-Smith and W.W.Mayol-Cuevas in [1], where a handheld robot was constructed and evaluated in terms of user cooperation. The underlying motivation of the paper was the assertion that there exists a much greater potential for handheld robotic applications than just the medical industry.

The work in [1] covers the development of a four degree-of-freedom (4-DoF) prototype handheld robot, in which the aim was not to design a robot which can surpass human capability, but rather to evaluate the cooperation between the user and robot for various degrees of autonomy. This design and evaluation here has certainly laid the groundwork for future development of the field. The arm comprised a carbon-fibre backbone, and is cable-driven by two pairs of motors and pulleys located in the base; one opposing pair providing a degree of freedom each to the midpoint and the tip, with the other pair providing a second degree to each point, as illustrated in Figure 2.6.

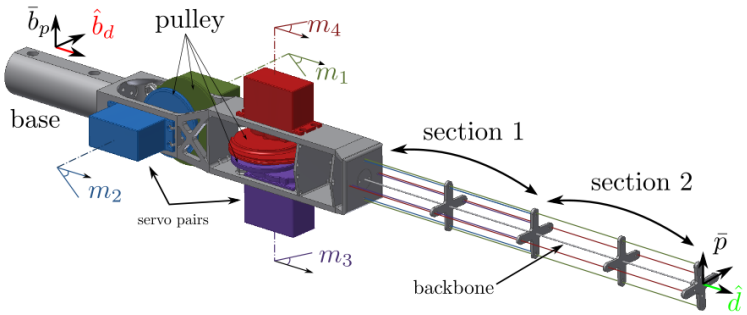


Figure 2.6: CAD model of the Four Degree-of-Freedom Mark I robot, image from [1].

The design takes inspiration from [31] which, unlike typical cable-driven designs, employs one motor to provide tension to a pair of opposing wires, which reduces the motor count to DoF ratio from 3:2 to 1:1. An external computer is used to interface with the servomotors and the trigger, responding to external cues or program instructions by providing the necessary motion. In order to locate the robot spatially, an OptiTrack system made up of six strategically mounted cameras is used to provide motion tracking, identifying the retro-reflective markers mounted on the robot. This provides an accuracy of 0.2mm, although it was stated that the future aim would be to employ on-board sensors for portability and practicality.

In order to calibrate the robot, it was necessary to perform kinematic modelling. Conventional forward kinematics methods were not applicable to this design due to the non-linear interaction between sections 1 and 2 of the carbon-fibre backbone, shown in Figure 2.6. As such, a regression based method was employed. This entailed sending a wide range of commands to the arm and recording the resulting pose with the motion tracking equipment. In total, 14641 vectors of motor angles were sent, with the corresponding poses comprising a relative tip-to-base position and direction. A regression technique was then used on the results to interpolate between the points to acquire estimated poses for unrecorded motor vectors. It is important to note that due to the limits of the carbon-fibre backbone, the arm cannot self-intersect, therefore it is safe to interpolate between the motor angles.

Inspiration was taken from this Mark I design in the development of the new device, and naturally, similarities will be observed between the set-up, control processes and external infrastructure to interface with the robot and provide motion. The key differences originate from the intended change in arm material/structure, having a chain reaction of required design changes. Also, with the benefit of hindsight, design and material choices that hindered this Mark I design may be avoided, while strong points could be adapted. An example of this is the issue of the friction from the cables wearing through the base material. This led to the choice of an ultra low-friction cable material, as covered

in Section 3.1.2, and a stronger base material, using PLA over ABS. Similarly, the approach taken to evaluate the cooperation between the user and robot was to be based around a user study which entailed a Task Load Index questionnaire to assess the perceived effort, as detailed in Section 2.4.4.

Subsequently, a more advanced six degrees-of-freedom device was constructed over a longer period of time, as covered extensively in [26]. This design comprised distinct solid sections connected by links, as opposed to a continuous backbone, to allow for a greater range of motion. The details of the Mark II device are less relevant than the first design, based on the lack of similarities between them, and will therefore not be covered in detail. Both the Mark I and II devices are made openly available at [2].

2.3.3 New Design Inspiration

Some initial design decisions were made at the start of the project. The first was that the robot was to be cable driven, due to already possessing four high torque servomotors that have been proven to be reliable and effective on an existing prototype. Another was for the arm to be of a singular solid piece of material, mitigating the design complexity associated with numerous links. As such, the design began to take shape as a four degrees-of-freedom cable-driven device.

There are numerous existing designs that incorporate a single structure, cable driven arm, such as those covered in [27] and [28]. However, these designs relied on a method in which each connecting cable was driven by a single dedicated motor. This was improved upon in [29], where a design was introduced that uses a single motor to drive a pair of opposing cables. This design was the inspiration for the first handheld robot prototype, as covered in [1]. Given the similarities in the general design principles and handheld nature, inspiration was taken from the first prototype in the development of the new device.

However, one area identified for potential improvement associated with the designs in [27][28][1] was the possible range of motion. All of these designs comprise a solid carbon-fibre backbone, which, whilst offering rigidity and a degree of flexibility, has natural limits on its deflection before snapping. Thus a maximum range of motion in the first prototype was found to be about $\pm 20^{\circ}$of each motor from the starting straight-arm position. This value was determined during the development stage as the carbon-fibre rods were found to snap with larger angles. As such, an alternative material for the arm of the new device was to be used, giving it discernible characteristic differences from the first prototype. It was hoped that a greater range of motion could be achieved whilst reducing vulnerability to damage on sudden impacts.

In [30], an investigation was undertaken into a continuous elastic spine, actuated by tendons. This serves as an alternative to typical spinal structures which are stabilized by, or the equivalent of, intervertebral discs, muscles, and ligaments. The variation between structures with a finite number of links and elastic continuums is illustrated in Figure 2.7. In this paper ([30]), a multiple DoF prototype is presented to demonstrate the possibility of different tendon routing configurations and elastic continuum shapes. One of the key benefits highlighted here was the fact that such structures address the problem of collisions. Robots that interact with humans and the physical environment must be able to withstand intended or accidental impacts without damaging the environment or the robot itself. Such impacts are damped with a passive elastic element, enabling work in unstructured environments. This robustness to impact was verified in [30], with external impact tests to both rigid and the elastic continuum structures proving that rigid mechanisms experience two times the force under similar collision conditions.

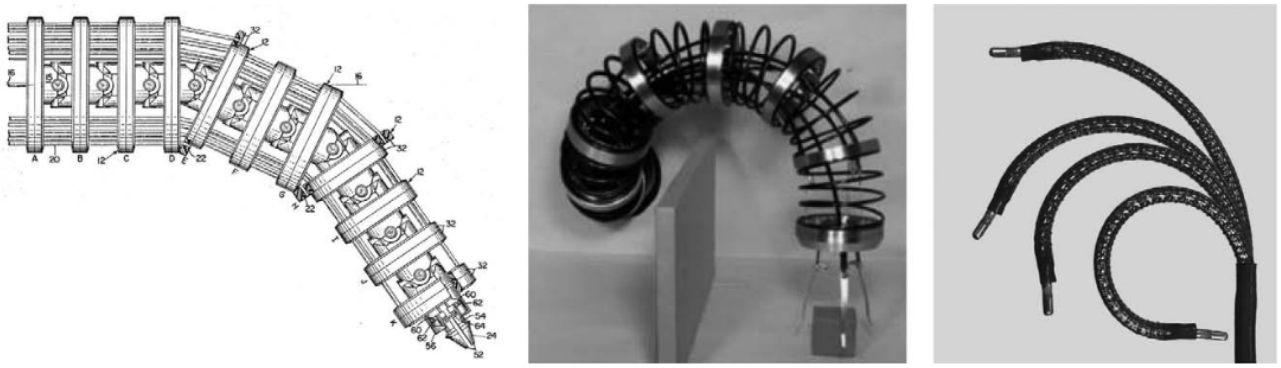


Figure 2.7: Left: The hyperredundant Tensor Arm of Anderson and Horn [31], having a large number of linkages to increase flexibility. Centre: The first sustained research program in continuum and hyperredundant robots, image from [32]. Right: Modern continuum robot used in medical applications, image from [33]. Note that all images were adapted from [34], which reviewed existing robots in this space.

The other key advantage offered by elastic continuum arm structures is the greater range of motion as a result of their continuously bending nature. Further investigation into the design and modelling of such structures in a robotic context is provided in [34]. Inspiration, like much of robotics, has been taken from nature, where extreme manipulation and dexterity capabilities are exhibited by the likes of snakes and octopus tentacles. In this paper, multiple mechanical continuum robot architectures were reviewed, outlining the advantages over robots with a finite number of rigid links. At present, continuum robotics are relatively unexplored in comparison to more established rigidly linked structures, though it is predicted that their accelerating progress will lead to increasingly practical solutions in the future. For these reasons, which are highly applicable to handheld robotics (human/environment interaction and increased motion), it was decided to employ a continuum elastic structure for the arm of the new design. This offers a fundamental difference between the first prototype, and provides added breadth into the field of handheld robotics in general.

2.4 Designing Experiments for Handheld Robotics

At such an early stage in the development of handheld robotics, it is essential to carry out experiments that reveal as many details as possible, complementing existing research that may be used as reference as the field progresses. Focus must be placed on important factors such as how well the user and device cooperate, how the desired actions of the robot are communicated to to the user and the benefits of using the device in terms of perceived effort and performance. The earliest experiments undertaken focused on evaluating the cooperation between the human and robot. This took the form of investigating the effectiveness of different feedback methods for the robot in communicating desired positions, while other experiments were concerned with the kinematics and motion performance in general. In the first two of the following example tasks, the cooperation was highlighted by the performance associated with three different levels of robot autonomy.

2.4.1 Painting Task

The painting task from [1] entailed participants being shown a template pattern on a screen, and assigned with tracing the pattern with ‘virtual paint’. The motion capture system would register the position and orientation of the tip of the arm to establish the absolute direction of the painting. Then, a ‘key action’ (user or program oriented action associated with progressing the task) would result in virtual painting, changing the colour of the pixels that the tip is pointing to.

In both manual and semi-autonomous mode, the key action was a user oriented cue, in this case the pressing of a trigger to initiate painting. The manual mode involves a straight arm with the paint

turned on with the trigger, while the semi-autonomous mode draws upon the pre-programmed task knowledge to facilitate movement along paths of contiguous pixels in the same target region while the trigger is held. However, it cannot move across gaps to unpainted regions by itself, requiring the user to reposition the tip. Alternatively, the fully autonomous mode involves the robot initiating and halting the painting action where it deems appropriate, providing the key action itself. This task set-up is illustrated in Figure 2.8.

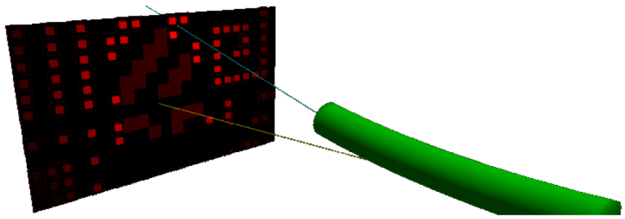


Figure 2.8: Model of painting task set-up in autonomous mode (lights indicate preference of subsequent motion), image from [1]

2.4.2 Tiling Task

The tiling task, also from [1], involved fitting the end effector (component to interact with the environment) with an electromagnet, picking up red or black tiles from designated hoppers before placing them in a predefined chequerboard pattern, as illustrated in Figure 2.9. The magnet is activated/deactivated by pressing the trigger, except in the case of full autonomy in which the robot fully provides the key action, controlling the magnet as necessary. In manual mode, the absence of task knowledge means the arm remains straight, responding only to the trigger press with the magnet activation. In this case, it is possible for the user to provide an incorrect pattern, therefore a template was made visible so that all users performed the task correctly.

In the two modes that comprise task knowledge, the arm moves towards the nearest hopper that it knows will provide a suitable colour, before moving towards the nearest suitable empty space on the board. In both cases, the robot will refuse to perform an action that conflicts with the task specification. In the case of full autonomy, the electromagnet is also controlled by the robot. Thus the tool will keep seeking to perform necessary actions, gesturing to the user where it should be positioned to do so, until the task is complete.

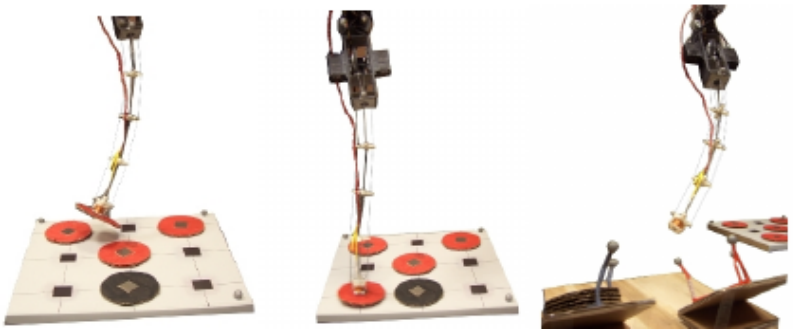


Figure 2.9: Pick and Place Tiling Task with tip of first prototype, image from [1]

The results from the previous two tasks indicated that the completion time was significantly improved with increasing degree of autonomy. The accuracy of the tasks however, showed little difference among the different modes. This was to be expected as the robot end effector was designed with an accuracy comparable to that of a human. The combined Task Load Index (TLX) scores also showed improvement with increasing autonomy, indicating that introducing cooperation with the robot reduces the perceived task effort. The individual TLX scores for various factors are shown for both the painting and tiling tasks in Figure 2.10.

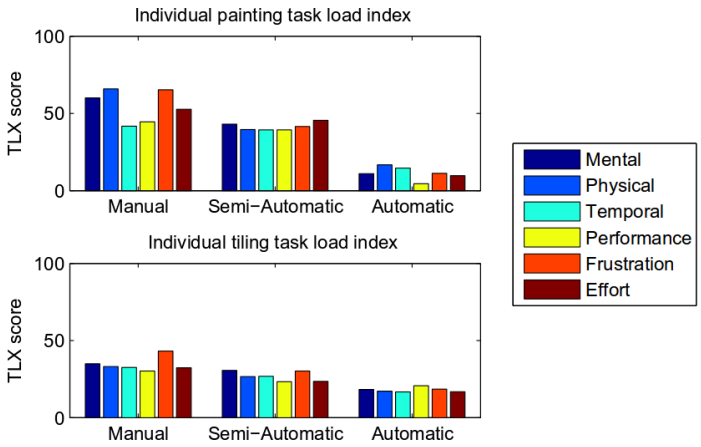


Figure 2.10: Individual TLX scores for both the tasks, image from [1]

As the task accuracy exhibited little difference among the modes, and the completion time and perceived effort showed improvement with task knowledge, it was concluded that the users cooperated well with a task-aware handheld robot, where introducing levels of automation had a positive impact on essential elements of task performance. Although sharing elements of the task between the user and robot did not negatively affect performance, maximum overall benefit (accounting for perceived effort) was attained when the robot was given full control. Of course, there is still a strong element of cooperation in this scenario, as the user must move the robot to its desired location, responding to its gestures accordingly.

2.4.3 Spatial Guidance Task

Manoeuvring the robot to a specific location so that it may perform its operations is an essential element of strong cooperation. Without effectively conveying desired locations in order to proceed with the task, the usefulness of the robot is severely hindered. Thus, some form of guidance is necessitated. Fitt's law is an established method of estimating the time to reach a known destination [35], and although was originally created with a 2D perspective, it has since been extended to a 3D environment, such as in pointing tasks [36]. In either case, the way in which information is presented to the user from a human interaction point-of-view has often been considered to affect performance [37] [38].

As covered by A.Gregg-Smith and W.W.Mayol-Cuevas in [39], various methods were investigated in which the robot provided spatial feedback to the user during a 3D guidance task, using the more advanced 6-DoF device. The task comprised positioning the robot so that the end effector could be aligned to within 5mm and 5 degrees of a target pose for at least 200ms. The target pose took place at a range of locations on a small side table, and was also performed with a stand-alone handheld wand, in order to evaluate the benefit of using the robot in a performance and feedback sense.

The visual conditions investigated include a mounted screen, a handheld screen, an augmented reality headset and a virtual reality headset, with the set-up for the latter illustrated in Figure 2.11. The other feedback options were from gesturing motions of the arm and using the wand with the aforementioned headsets. In each case, an infra-red optical tracking system was used with retro-reflective markers attached to the key objects; the arm, wand, table, augmented/virtual reality headsets and the 7 inch screen. This allowed the pose of the robot to be calculated.

Figure 2.11: Spatial guidance task being performed with the handheld robot where feedback is provided via a virtual reality headset, image taken from [39].

The robot itself possessed a 6-DoF pose in the real-world space, while the joint angles constitute a further six degrees of freedom, resulting in an overall 12-DoF pose. Solving from one of these fully defined poses to another is computationally expensive, requiring minutes to be calculated where sub-second times are required for a responsive tool. In order to overcome this issue, the path planning ability of the human was exploited, reducing the robot path-planning problem to three degrees of freedom. A variant of informed Rapidly Exploring Random Trees (RRT*) [40] was then produced in [39] to calculate the shortest possible route from robot to target, using the frame of reference of the target as the root of the tree. Once such a trajectory is known, the robot can use it to gesture along the absolute line as one of the feedback methods, as well as searching for a valid solution, as explained in [26]. This algorithm was always run during robot use, regardless of the feedback method.

The results of the completion time and the TLX combined scores in [39] indicate that all forms of feedback (augmented reality, virtual reality, mounted screen and gesturing) are shown to improve performance compared to the handheld wand. Although the gesturing mode itself does not show as great a performance difference as the visual methods, it does show an improvement nonetheless, justifying it as a viable option. As such, feedback through gesturing was the chosen option for this project. The visual feedback methods themselves exhibited little difference between them. These strong results demonstrate the effectiveness of the robot in assisted tasks, outlining its potential for use in the tool space.

2.4.4 Task Load Index (TLX) and Perceived Effort

When investigating task effectiveness, it is essential to quantify the perceived workload of the subject. This is especially true when researching methods that may improve the performance. The established means of this quantification is the use of the NASA TLX (Task Load Index). A multi-dimensional scaling was introduced in [41] under the belief that subjective workload is in fact measurable, and it has since become widely accepted. Subjective responses can thus be reliably evaluated to meaningful ratings. The six key rating scales associated with the index are mental demand, physical demand, temporal demand, performance, effort and frustration level.

This TLX methodology was employed on each of the experiments that have been carried out for handheld robotics thus far [1] [39], yielding valuable information regarding the subjective effort of cooperative tasks with the robot. As such, it is logical that a similar technique will be used on the new evaluation. An ethics approval agreement is already in place that will span the duration of the project. This will allow the undertaking of studies with human participants, concluding with them filling out a tailored TLX form.

3 Design of New Handheld Robotic Device

3.1 Physical Design Process

3.1.1 General Shape Concepts

The initial design of the device was influenced by the availability of four HiTec HS-MS7980TH servo-motors (Appendix A). These servos, having already been proven in terms of reliability and sufficient torque provision, were thus selected to provide the necessary driving force in a cable-driven design. As a result, a number of configurations were initially explored, with the servos housed in various physical arrangements. Initial inspiration was taken from the general shape of the design in [1], in which the servos are positioned in two opposing pairs.

As covered in Section 2.3.3, the key novel feature of the new design, compared with previous handheld robotic devices, was to be the addition of a highly flexible continuum arm. This must meet the requirements of being a singular structure, with minimal material limititations on deflection, as well as being highly tolerant to collisions. These features are essential in the device providing a more versatile solution than in [1], especially in unstructured evironments where impacts are common, such as a construction site. Moreover, the device had to remain comparatively simple and cheap, with a much shorter lead time than the 6 degrees of freedom device in [26], in order to be feasibly produced within the time-frame of the project. Taking these factors into consideration, the arm structure was selected as a flexible foam tube, with an internal and external diameter of 17mm and 42mm respectively. It was found to exhibit no signs of breaking when bent to extreme positions, even when folding in on itself.

With these components in place, prototypes were constructed for the base of the robot, which has the primary purpose of housing the servos and one end of the arm. The initial concept comprised a single structure body, in which the servos were mounted in two opposing pairs, each a conservative distance apart so that the cables could be connected directly from the servos to the arm, eliminating the need to account for friction. The arm was then located in a tubular mounting, with a smaller axially located cylinder to sit within its bore, providing added structural support. This general concept is illustrated in Figure 3.1.

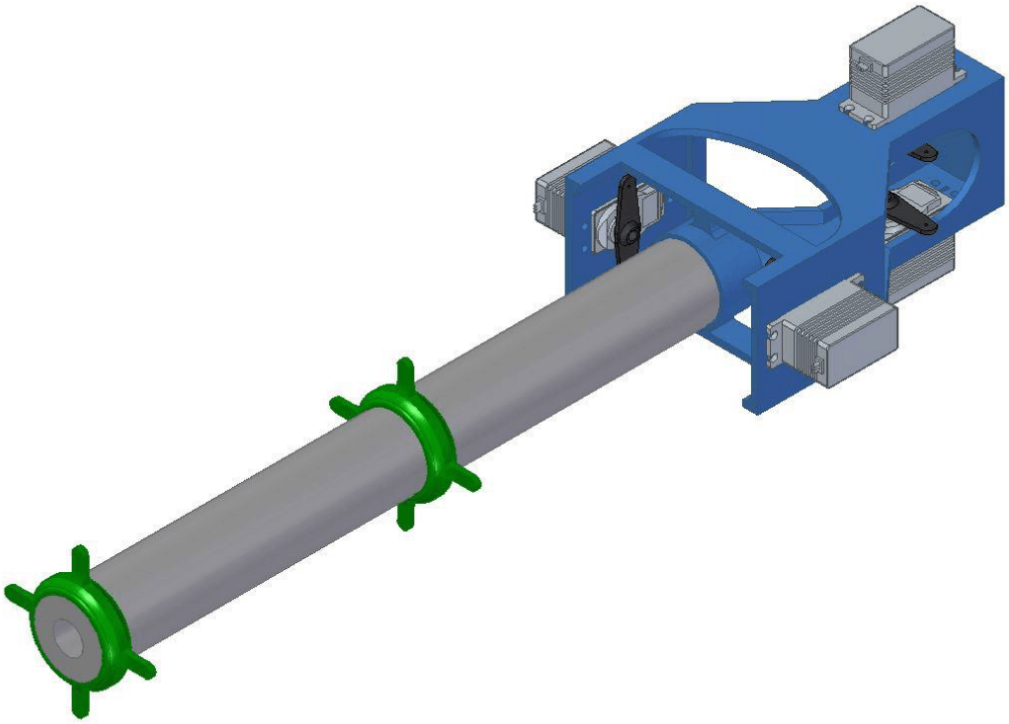


Figure 3.1: Initial concept for the base of the robot, to house the servos and the arm.

At this point, the linkages to connect the arm to the servo cables were assumed to be circular discs with tabs (arms) for attaching the cables, which would be fitted to the midpoint and tip of the arm to provide two degrees of freedom at each point. This structure was then 3D printed at its initial dimensions, using the printer given in Appendix G, in order to gain an insight into appropriate sizing and shape. The first issue was the physical finish of the printed part from the extrusion type 3D printer. As the design was a single structure with overhanging elements, a support scaffolding was necessary to prevent collapse of the material during the printing process. Upon removing this excess material, abrasive edges were still present. This hampered the quality of the fittings, such as the mounting bolts of the servos, as well as scratching the foam when locating it within the holder, while also resulting in a generally unprofessional appearance. As such, it was decided to ensure future designs do not require support material during printing, and thus don't consist of overhanging elements.

When the servos were mounted to the base, the structure appeared unnecessarily large, where the makeshift wire reached the linkages without any points of near-intersection. Furthermore, due to the wide gap between the two horizontally configured servos, potential issues were discovered with stabilising the device, found through holding and manoeuvring it. Turning the device about the axis of the arm whilst these servos are spaced particularly far apart gives a much larger angular momentum than if they were close, making it more difficult for the user to turn the device in such a manner, as well as slowing it once started. Therefore, for any subsequent designs with the servos are in a twin-pair opposing configuration, the distance between the horizontal servos was to be minimised.

At this point, an alternative design was also considered, with the intention of minimising the distance of the servos from the user, requiring less counter-moment force to hold. This design focused on being as compact as possible, and in such a manner that it could be printed in one piece without the complications associated with the first print. This concept thus took on a 'cubic' shape, as depicted in Figure 3.2.

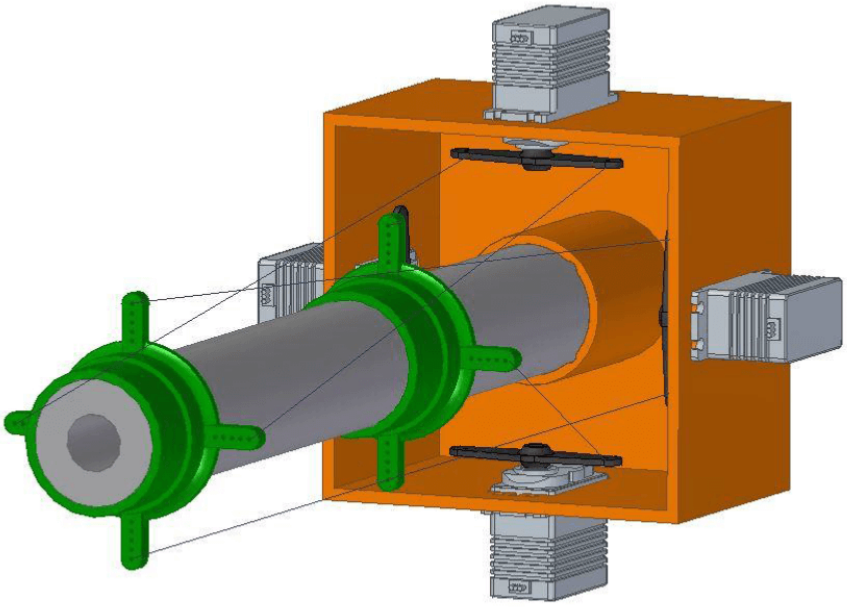


Figure 3.2: Cubic concept for the base of the robot, with the intention of being compact in length.

In conjunction with the formation of this concept, the earlier prototype from Figure 3.1 was assembled using generic servo horns and spare wire to drive a pair of printed rings acting as the linkages on the arm. The servos were then connected to an Arduino UNO (as covered in more detail in Section 3.2) in order to issue simple positional commands to test the motion of the arm. This revealed a few traits that warranted further design changes. The most notable aspect was the way in which the servos were connected to the linkages; whilst the 'direct line' approach as employed in both of the early concepts had the advantage of being virtually frictionless, issues were found with achieving predictable motion response.

When a servo actuated, the response direction of the arm was not constant with increasing servo motion. That is, the arm initially moves in the direction parallel to the connecting cable, and as the servo moves further, so does the starting position of the cable, changing the direction of motion of the arm. This response is made more unpredictable by the fact that the linkage in question could be in different starting positions depending on the angles of the other servos at the time. Hence, the actuation of a single servo was not constrained to an arm response in a single degree of freedom. This was predicted to cause added complexity further down the design and control process. Put simply, it would be much more logical and easier to account for if the cables were connected in such a manner that the actuation of a single servo could be ensured to result in arm motion in only one degree of freedom.

Other problems were also identified with the early implementations of linkages. Firstly, as the linkages were pulled in one direction and the arm deflected in response, gaps appeared between the arm and linkage, causing it to shift its position somewhat on the arm. Hence, the linkages were re-designed to be longer axially, interacting with a larger portion of the arm so that there are no gaps between the linkage and arm. Also, wrapping the cable around the tabs of the linkages was found to be a non-viable solution, as they constantly became unravelled, losing their tension almost instantaneously. Therefore, a series of holes along the tabs were added to interweave the wire between, creating a stronger grip through friction before fastening them by other means. This design evolution of the linkages is illustrated in Figure 3.3.

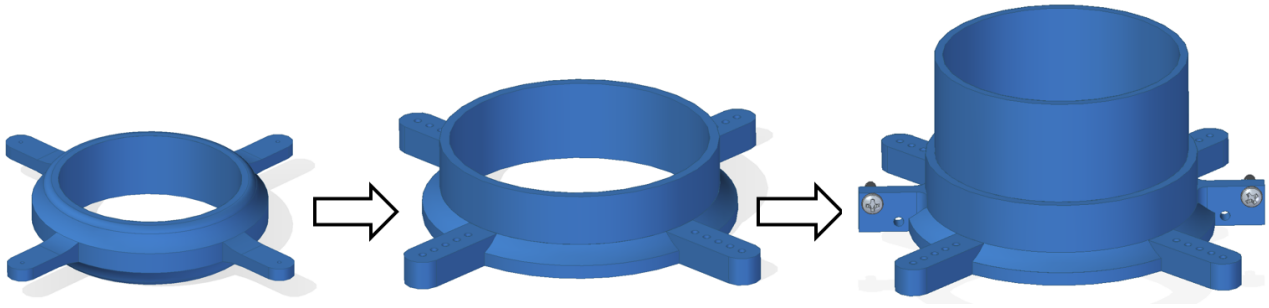


Figure 3.3: Design evolution of the linkages for driving the arm via the cables. Left: early, flat like design, Centre: taller design to minimise the gap during arm deflection, Right: Taller still, with holes to grip the cables via nut and bolt to reduce loss of tension over time.

The height of the linkage was experimented with over time to find the smallest size that appeared to suitably minimise the gap between the arm during deflection. If the linkage is too tall however, the flexibility of the arm will suffer. The tabs on the linkages have small 1mm diameter holes to feed the cable through, of which there are five in total on each tab, which the cable can be weaved through and pulled taut to gain a strong grip without immediate slippage. For the central linkage, the outermost holes on the tabs are used only to feed the outer cable through for the linkage at the tip, so that the cables all run parallel to each other. Therefore the tip linkage has one extra hole that the cable can be weaved through.

Originally, the cables were weaved through then wrapped around the tabs, held in place by knots, tape and glue over time. However, noticeable tension loss was still observed over a short period of time. As such, with the final design iteration, an extra face was added in order to incorporate a nut and bolt, with a neighbouring spare hole. The excess cable coming out of the weaving could then be passed through this larger 3mm diameter hole, and wrapped around the bolt, so that two cables are constrained per bolt. Washers and nuts are then applied and tightened so that the cables are firmly gripped with no slippage. Ultra high strength tape was then used to connect the linkages to the arm. Although loss of tension was still present over time, due to gradual stretching of the cables and slight slippage of the tape, the effect was greatly reduced, requiring less frequent re-tensioning operations.

3.1.2 Cable Routing

In order to limit the motion of one servo to response of the arm in one degree of freedom, it was necessary to redesign the base structure in a manner that routed the cables to run parallel to the arm. In order to have sufficient space for this routing, it was decided that the staggered twin-pair servo arrangement would be better suited, allowing the servos to be positioned alternatively such that less extreme angle changes of the cables would be required. This can be accomplished by positioning two of the servos as close together as possible to pre-align the cable with the linkages to a degree, which would not be possible with a cubic arrangement. Of course, the design was arranged in two halves, so that it may be 3D printed without any overhanging elements and need for support material. As a result, the centralised tube holder (to locate the base of the arm) was also a separate part, located in a groove between the two halves of the base when they are secured together by four threaded rods. This arrangement is illustrated in Figure 3.4.

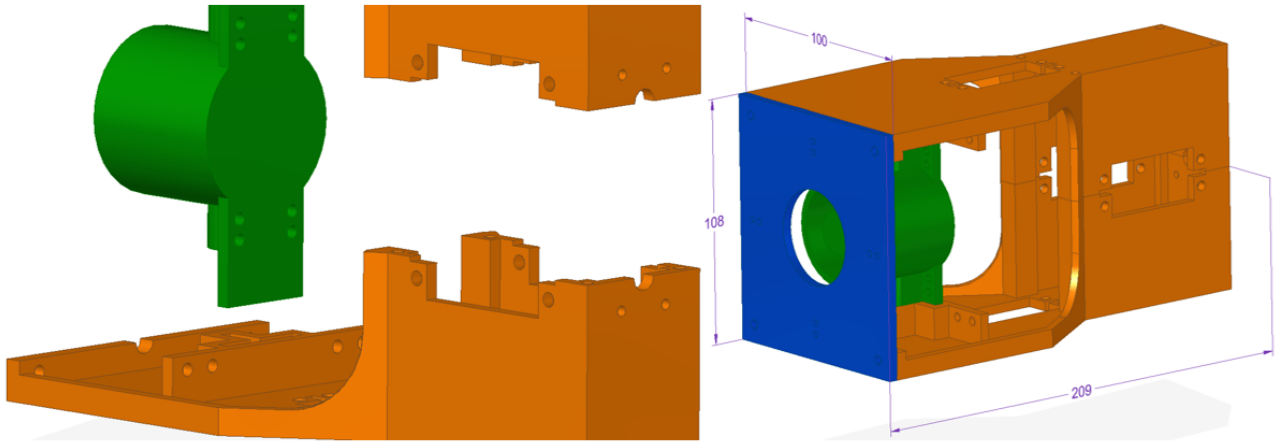


Figure 3.4: Identical halves for the base structure, with a separate centralised part to locate the arm.

In this arrangement, the two rear servos are positioned close together while the front servos are further apart, creating space between the front two in which to run the cables from the rear pair without intersection. A series of guidance holes were then put in place so that the cables can run parallel to the plane of their corresponding servo horns, tangentially to their starting position. These holes were positioned in the body of the tube holder itself, as well as the slot groove structure on each half of the base that was used to locate it. Multiple holes were added so that different radius servo horns could be fitted if desired. In order to shift the direction of the cables one more time so that they ran parallel to the axis of the arm, a simple square faceplate was to be fitted to the front of the base. This simply comprises a central hole for the arm to run through, and four smaller holes that each locate a pair cables, the latter of which aligned with the holes on the linkage tabs when the arm is in its neutral (straight) position.

As the cables sharply change direction at distinct points, namely the guidance holes, a reaction force will be felt at the contact point on the circumference of the holes. This force will increase linearly with the tension of the cables, which must be sufficient enough to stabilise the arm and manipulate it during servo actuation. Hence wear of the cable on the rim of the hole was a concern, as this would cause added resistance to motion or, in the worst-case, shear through the material so the cable is no longer routed. This was an issue apparent on the Mark I design in [1], which had to be addressed with extra attachments and rings to strengthen it against further breakages. As such, an ultra low friction cable was chosen from the start, using a high-strength fishing wire (Appendix B). This exhibited no signs of wear during the phase of prototype testing, even when deliberately over-tensioned. Therefore no extra support fixtures were required and the cable ran freely over the PLA material of the base. With this, the dimensions were tweaked to move the front two servos closer together and to the rear servos, making the design more compact, and the dimensions were selected for the final design of the base, which overall were 209 x 100 x 108mm as illustrated in Figure 3.4.

3.1.3 Pulleys

With the design for the base structure in place, final parts were 3D printed and assembled so that an underlying foundation of the device could be tested at the real dimensions of the base and servo locations. When actuating the servos whilst moving and orienting the device, issues were found with the way in which the servo horn drives the cables. Initially, the servos were fitted with the supplied standard two-sided servo horn (arm), that had an overall diameter (furthest hole from the centre on one side to the other) of 50mm, as shown in Figure 3.5. The cable was connected by interweaving it through the holes on the arm, giving a friction grip, and knotting the loose end so it cannot pass through the innermost hole. However, this option of driving the cables was limited in terms of available motion. At small servo movements from the starting position, a large response of the arm occurs due to the servo horn being perpendicular to the cable. However, at similar servo movements at higher angles, much smaller responses of the arm are observed. This is due to the cable being at a different angle relative to the servo horn compared with the perpendicular starting position, thus the servo motion does not 'pull' the cable as much. This effect is illustrated in Figure 3.5.

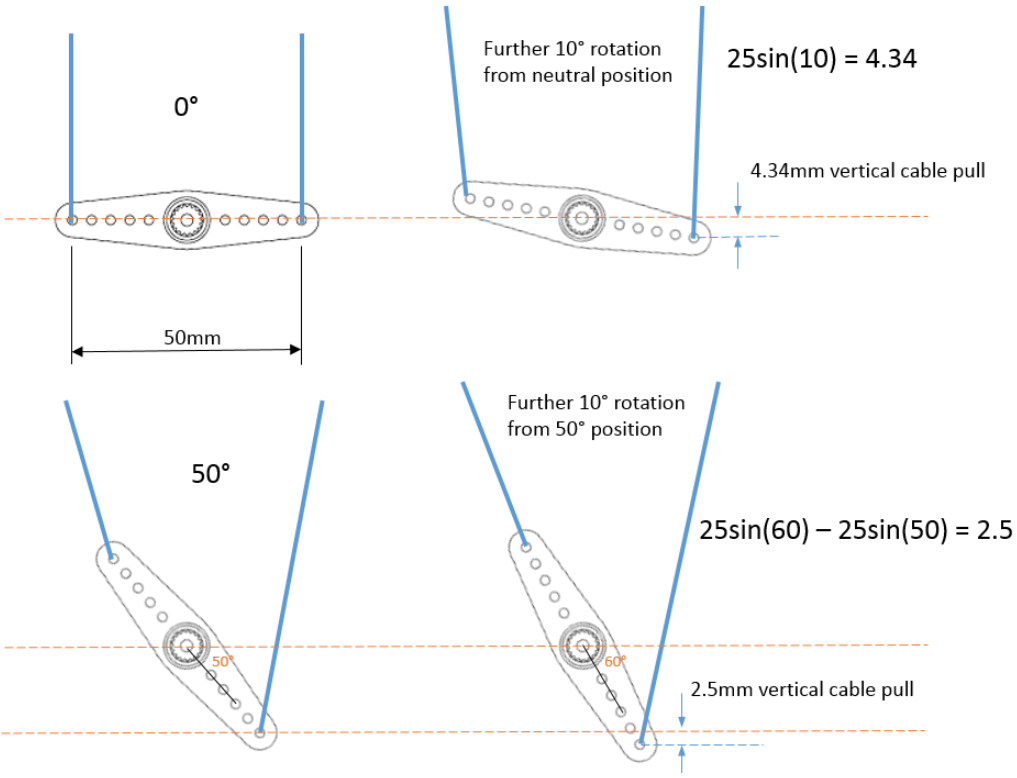


Figure 3.5: Vertical cable displacement ('pull') at a constant servo increment of 10 degrees, compared at different starting positions. Note that the horizontal cable displacement is ignored for simplicity as it has minimal effect on the pulling of the linkages.

This effectively limits the potential range of motion of the arm in that only small responses can be achieved when the servos are already at high relative angles. On top of this, the image also shows that the angles of the cables going into the first set of guidance holes will also change with servo motion. The general effect of this servo horn type is a non-constant, and limited, arm response to servo actuation. As such, the arm motion can appear 'jerked' and uncontrolled. In order to resolve this problem, it is necessary to employ a circular driving mechanism on the cables, so that the amount of pull is constant for a set servo increment, no matter what the position of the servo relative to its starting position, this effect is illustrated in Figure 3.6.

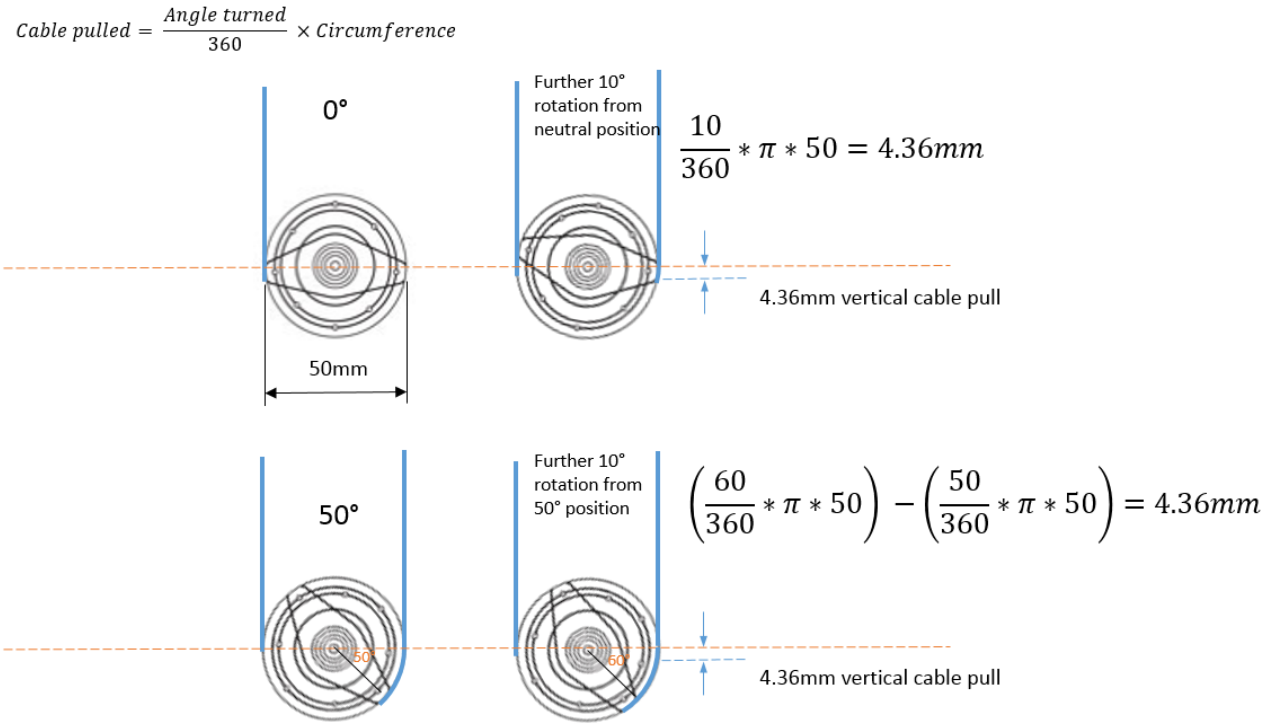


Figure 3.6: Vertical cable displacement ('pull') at a constant servo increment of 10 degrees using a circular attachment, compared at different starting positions. Assumed the cable is wrapped around the pulley before starting so unravels on side opposite to tension.

As the image depicts, the cables also remain at a constant angle going off to the first set of guidance holes. This circular approach leads to generally smoother motion of the arm, a more constant torque requirement, and an improved maximum range of motion. As such, pulleys were designed to be fitted to the servos. The servos were also supplied with a circular horn of about 45mm in diameter, so this was to be used as the central hub of the pulley to attach to the servo. The diameter was to be stepped out to 50mm so the cables could run parallel to each other, tangentially to the pulley, and straight through the existing guidance holes. The holes in the circular horn were used to pass one continuous cable through, so the length of free cable on each side, to be used for driving the linkages, is equal. The pulley then, was designed at two separate parts to be glued together, with a groove for the circular horn to be glued into. A small slot between the two halves could then be used to pass both ends of the cable through, so that it exits straight into the pulley groove to be wrapped around. This design is illustrated in Figure 3.7.

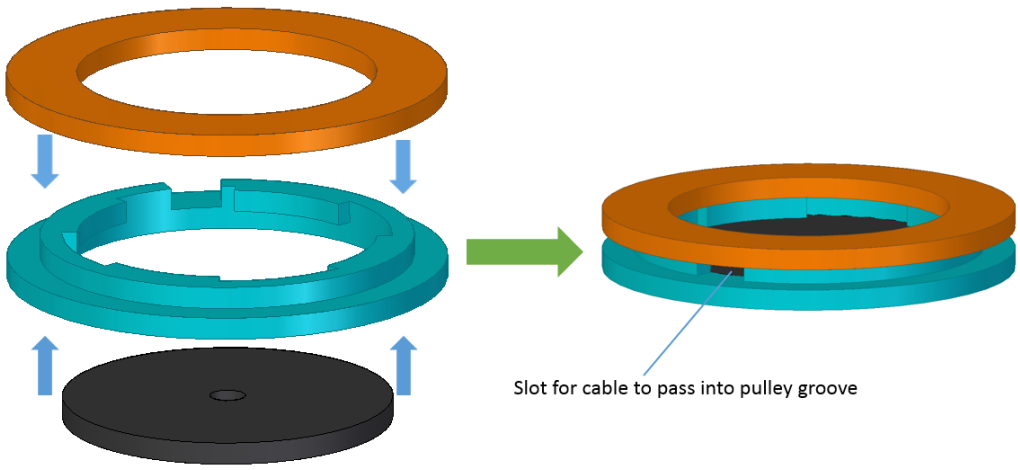


Figure 3.7: Three separate parts that make up the pulley design, including the supplied circular servo arm to interweave the cable into to constrain it.

3.1.4 Final Design

Once the pulleys were constructed and fitted to the servomotors, the motion of the arm was much improved in terms of smoothness and maximum range. However, when the arm was moved from its neutral position, an issue was identified with its stability. Although the cables that were actively pulling the arm were taut, as expected, the opposing cables had become particularly slack. The implication of this was that the opposing cables offered no resistance against further arm motion in the direction it was being pulled. This issue was exacerbated when the device was held parallel to the ground, as the arm appeared to 'sag', due to the added force of gravity and no tension in the upper cables to resist this motion. This comes about as a result of the path of the cable on the non-taut side of the arm. As the arm deflects, the amount of cable let out by the pulley is greater than the increase in length required to follow the arm motion, hence the loss of tension. Specifically, this is a result of the cable being connected in discrete, straight-line segments to the linkages, whereas the arm itself curves naturally when deflected. The effects of this problem can be reduced by adding extra linkages to the arm, purely to guide the cable and increase the total distance it covers during arm deflection. This problem, along with the solution, is illustrated in Figure 3.8.

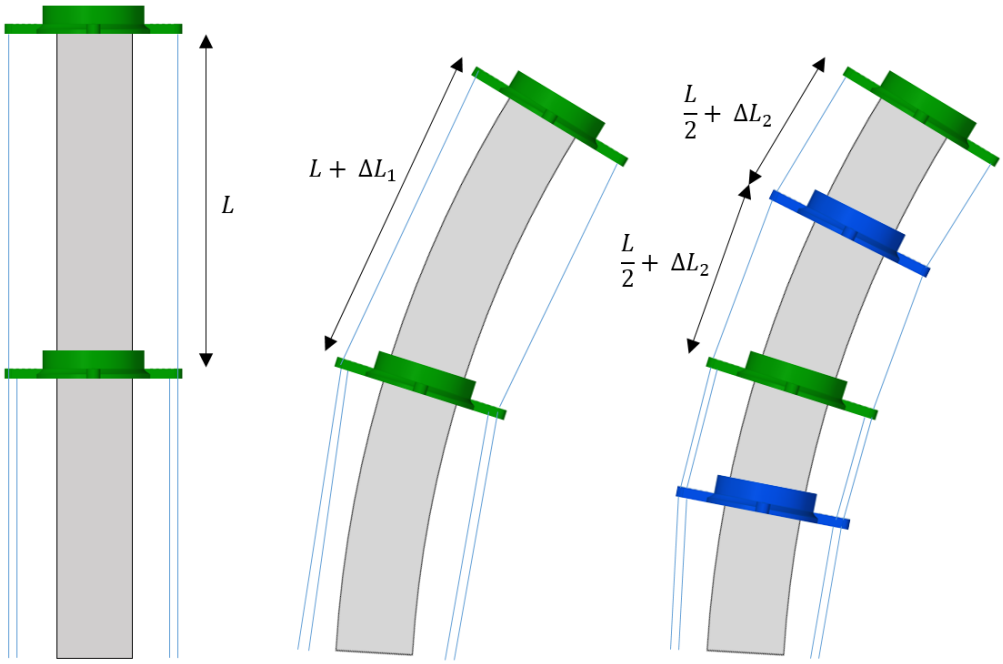


Figure 3.8: Varying length of the cable on the side opposite the pulling (taut) cable during arm deflection. Adding guidance linkages increases the cable extension during deflection (assuming constant arm curvature), reducing excess cable released from the pulley. Left: neutral position, Centre: deflection without guide linkages, Right: deflection with two added guide linkages.

As more guidance linkages are added to the arm, the distance the cable must travel from the central linkage to the tip linkage during deflection is increased. This is because the guidance linkages maintain the cable at a set perpendicular distance away from the surface of the arm as it passes through their guide holes. Without this, as shown in the central image of Figure 3.8, the cable goes from the mid-point to the tip of the arm in a straight line, coming much closer to the surface of the arm. Hence, this length comparison can be expressed as:

$$2(\frac{L}{2} + \Delta L_2) > L + \Delta L_1 \tag{1}$$

Therefore, there is a smaller difference between the amount of cable let out by the pulley and the extra amount taken up to follow the deflection of the arm. This means that there is a reduced loss of tension, and thus more stability of the arm and resistance to further motion in the direction of the deflection. As such, two guidance linkages were added to the arm, at the mid-points between the two

sections of the arm. A simple mounting surface for the magnetic end effector was then used as a cap and glued to the tip linkage, giving a total arm length from base to tip of 0.34m. This completed the physical design of the mechanical aspects of the robot, including the base, the arm and the cable driving mechanism, as illustrated in Figure 3.9.

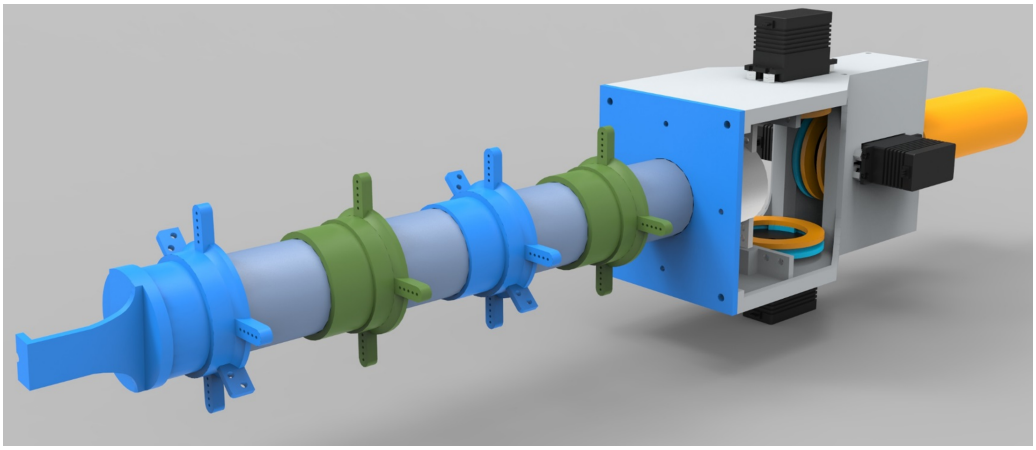


Figure 3.9: Rendering of the design of the base and arm of the robot

3.1.5 Ancillaries

For the end effector, it was decided to employ an electromagnet of some form, and design the physical aspect of manipulating objects around magnetic interaction, i.e metallic strips facilitating their collection. The chosen option for employing an electromagnet was to use the copper coil and metal cylinder from the central core of a small solenoid (Appendix C). This solenoid was disassembled to obtain just the necessary components for the electromagnet, as illustrated in Figure 3.10, and was then fitted to the surface mount at the tip of the arm. A small groove in the metallic cylinder of the solenoid slotted onto a similar sized semi-circular bore at the front face of the surface mount (visible in Figure 3.9, to prevent the cylinder from moving axially when the current flows through the coil and induces magnetic force. The exposed end of this cylinder is then, in essence, the 'grabbing' part of the robot, as it is magnetised and put in contact with metallic targets.

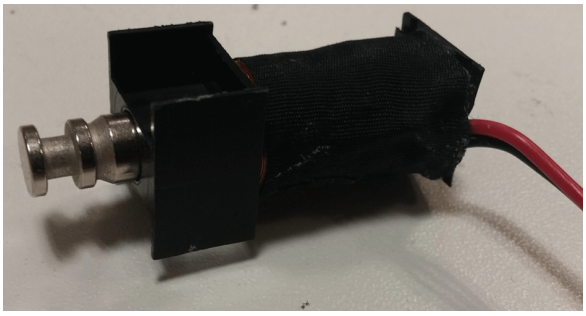


Figure 3.10: Electromagnet formed from the disassembly of a solenoid valve

.

In order to use the robot in an intuitive, user friendly manner, it was decided to modify a conventional handheld power tool to provide the handheld aspect. This must be ergonomic in terms of handling the robot, while also providing a trigger or switch so that the user may send signals to the device to instigate a required action. As a result, a standard outdoor grass strimmer was procured (Appendix D) and modified to remove everything but its base (handle, trigger and shaft). A sleeve was then designed to connect the robot with this base. This was a near-cylindrical cross-section tube which was simply bolted to the bases of the robot and the strimmer, completing the handheld component of the device.

3.2 Controlling The Device

In order to properly control the device, it was necessary to implement a microcontroller to interface with both the actual control program and the robot. As such, an Arduino UNO (Appendix E.1) was selected to sit between the PC that runs the program and the various electrical components of the robot (the servomotors and electromagnet). The servomotors themselves (Appendix A) require a 6 Volt power source, thus two 4 x AA (1.5 Volt) battery boxes were connected in series. The position of each servo is determined by the value from the corresponding PWM (Pulse Width Modulation) pin on the Arduino.

The electromagnet requires 9 Volts for generating adequate magnetic force. So in order to safely control this, enabling the magnet to be switched on and off as required, it was necessary to use an Arduino Motor Shield (Appendix E.2). This allows a 9 Volt battery to be connected via the V_{in} and *Gnd* terminals, with the output power and ground wires connecting to the electromagnet. A pin on the Arduino is then designated for turning this output power on and off. There are two options for controlling this magnet, by having the trigger of the strimmer base hooked up directly on the power wire to the electromagnet, or by having it connected to a pin on the Arduino, that can respond to signals by setting the pin that controls the power output. The former option would result in the magnet being on whenever the trigger is pressed, whereas the latter allows for much more flexibility. These include responding in other manners to the trigger press, counting the number of trigger presses and only activating the magnet if the device is 'in range', saving battery. Hence, the second option was chosen. A schematic of this set-up is illustrated in Figure 3.11.

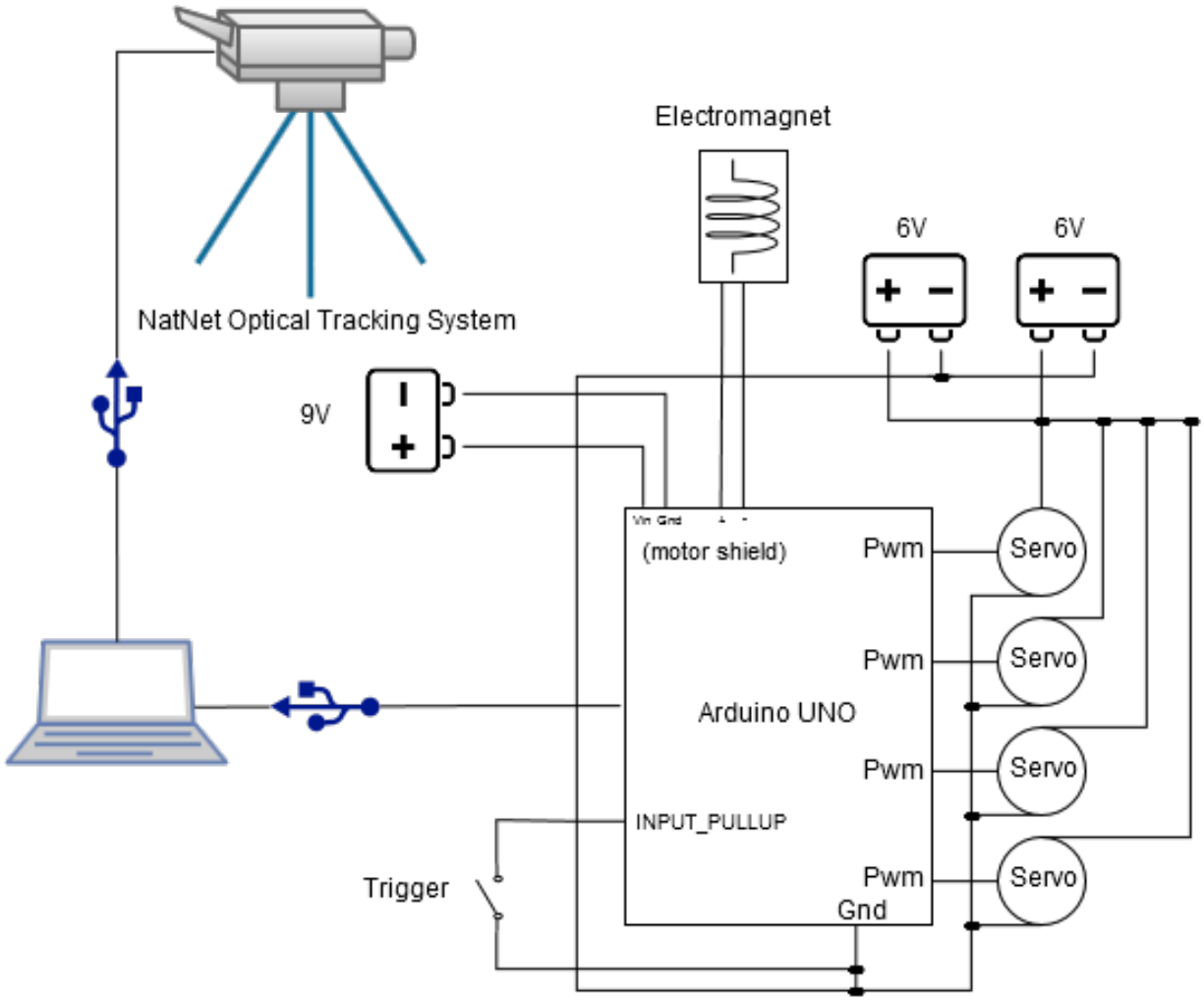


Figure 3.11: Schematic showing how the controlling equipment interfaces with the pc and the robot components.

Of course, the optical tracking system also interfaces with the controlling PC in order to receive the necessary tracking information. This is shown in the schematic for completeness. To send commands and receive information from the Arduino, a serial connection is set up in C# using the SerialPort class. Once a connection is established, instructions are sent in the form of a buffer of bytes, which are interpreted and responded to by the Arduino sketch. These commands include setting the position of a servomotor and activating the electromagnet; an example of this process, with the program sending instructions and the Arduino responding accordingly, is shown in Appendix H.1 and H.2.

All of the components were then assembled in place using an Arduino Proto-Shield (Appendix E.3), to solder the battery cables to, and a mini breadboard. This was set up in a manner that allowed the battery packs and servomotors to be readily interchanged, whereas the control circuitry was fixed in place. The finalised construction of the new handheld robotic device is thus illustrated in Figure 3.12.

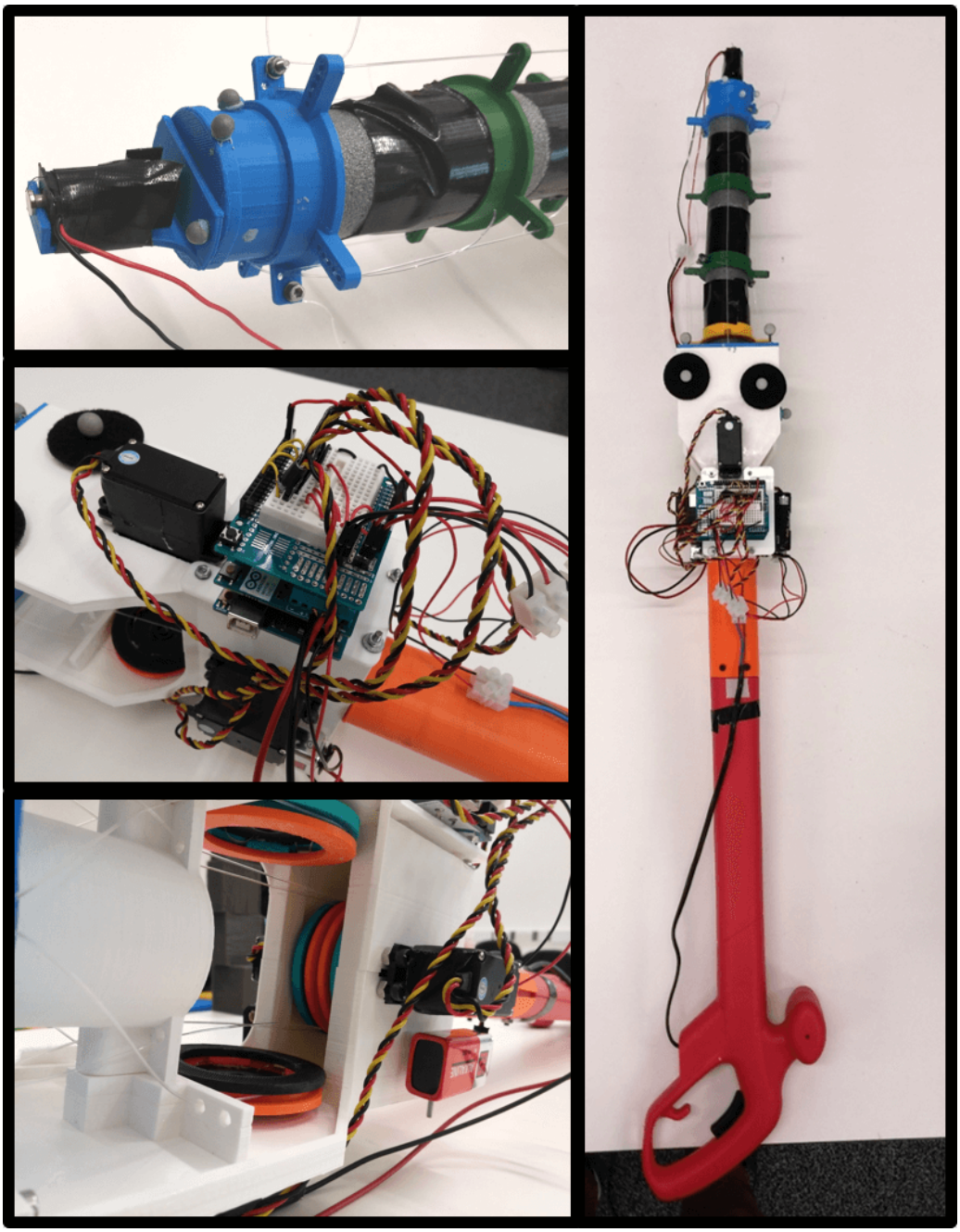


Figure 3.12: Right: Fully assembled device, Lower-Left: Servomotors and pulleys, Centre-Left: On-board controller set-up, Upper-Left: Electromagnetic end effector.

4 Developed Program

4.1 Interfacing With NatNet Optical Tracking System

In order to determine where the robot is in three dimensional space, as well as the relative distances between itself and other points, it is necessary to track physical objects. This was achieved using a single optical tracking device, the OptiTrack V120 Duo [42], of which the technical specifications are provided in Appendix F. The Duo features two fixed-distance cameras in order to provide depth perception, and works as a stand-alone device, unlike the syncing of several distinct cameras as used in [1] and [39]. The accompanying software library, the 'NatNet SDK', as well as the actual tracking software, 'OptiTrack Motive', is the same, regardless of the configuration. This library was provided to run in a .NET framework environment, hence the decision to use C# as the primary development language.

The device recognises the supplied retro-reflective markers, providing six degrees of freedom tracking. It is important to note that the coordinate system used by the software (the global axes) originates from the positioning of the Duo tracking device itself, as illustrated in Figure 4.1. Single markers can be identified in terms of their X, Y and Z positions, and can be grouped together to form 'rigid bodies'. These bodies, requiring a minimum of three markers, have a designated 'pivot point' that can be used to refer to the position of the body as a whole, and provide a reference about which rotation can be expressed. For reliable tracking, it is essential that the marker configuration be asymmetrical and distinctly unique, so that the program can easily distinguish it from others, and ascertain its true orientation. The rotation data for a body is inherently provided as a quaternion, which can be converted using built-in library functions to give Euler angles in the forms of roll, pitch and yaw (rotations about the global X, Y and Z axes respectively).

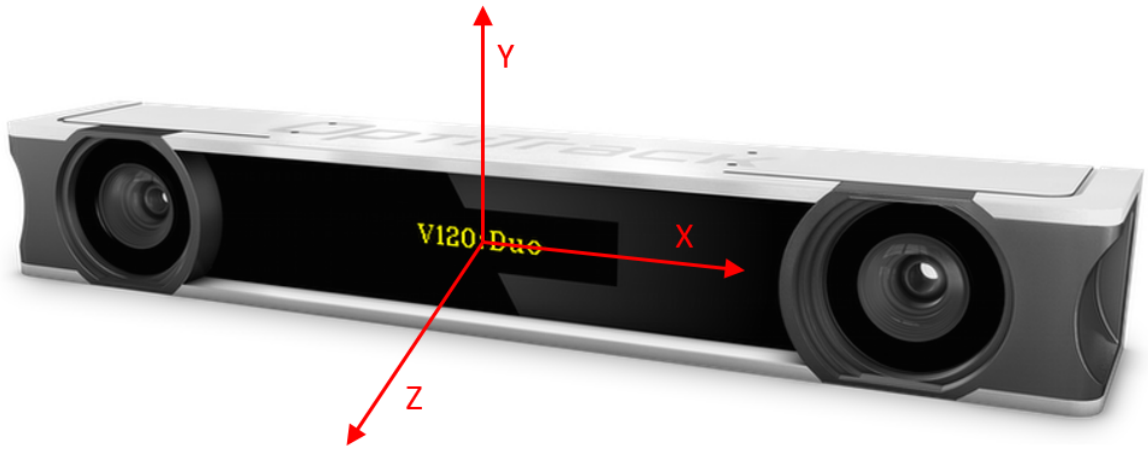


Figure 4.1: Axes as used by the OptiTrack Motive software, these global axes originate from the centre of the device, image adapted from [42]

The rigid bodies must be configured within the Motive program, selecting which markers comprise them and labelling them appropriately. Hence, a bodies were identified for the base and tip of the robot as a bare minimum. With the Motive program running in the background, the NatNet library can be used to extract tracking information from the software. This is done by identifying the program as a server application, and connecting to it via the local IP address. Once the connection is established and the fundamental NatNet client object is set up, the event based data handling is initiated. This refers to the server application (Motive) triggering an event with a corresponding callback every time a new frame of motion capture data is ready, which is then passed into the callback to be processed as desired. This frame of data contains all of the tracking information that Motive is configured to capture. For example, a list is provided that contains all the rigid body objects in the frame, each comprising the name, positional values, rotation (quaternion) and a contained list of the constituent marker objects.

Therefore, rigid body objects can be set up to represent the robot base and tip which, every time a new frame is sent from the program via the callback function, can be updated so they represent the most recent tracking information. Similarly, other variables pertaining to these bodies, such as the relative distance between tip and base, can be set up and updated as part of this process once the rigid body objects themselves have been refreshed. As a more practical alternative, a separate thread can be employed so that the user-created rigid body objects, or other relevant variables, can be updated as and when required by fetching information from the central 'Current Frame Of Data' object at any point. Of course, when doing so, it is important to ensure thread safety via a read-lock, so that the frame can be read, and data extracted, without the event based thread updating it mid-read.

Once the required tracking data has been extracted, it can then be manipulated to create target points relative to the base depending on the situation. These can then be used as inputs to the regression function, covered in the following section, in order to solve for the required servo angles. The tracking data at may also be written to an XML file, for later use in experimental or calibration procedures.

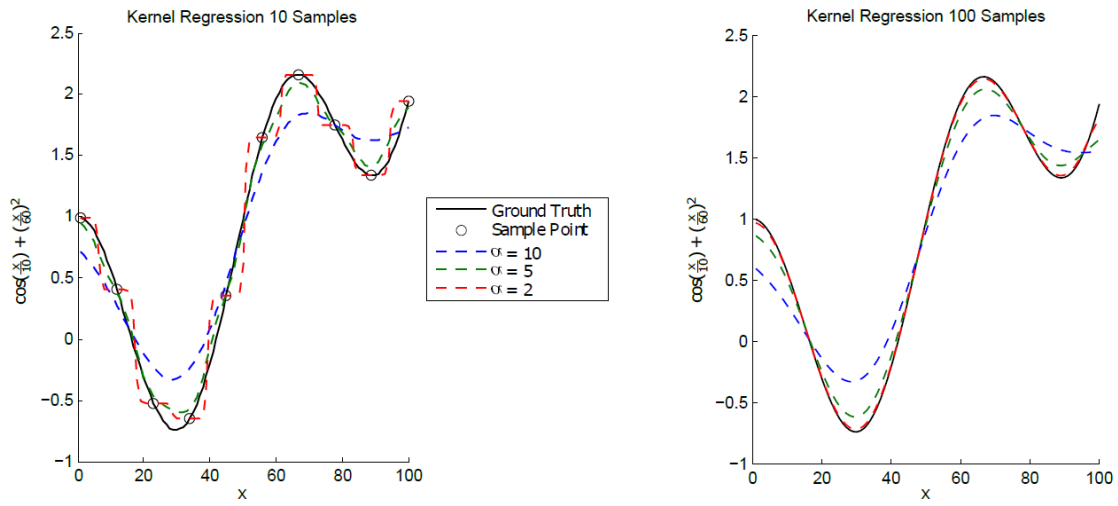
4.2 Inverse Kinematics via a Kernel Regression Method

Due to the nature of the design of the robot, particularly the flexible arm in which motion from one of the core arm linkages would affect the other, solving for the kinematics is non-trivial. The arm itself exhibits nonlinear characteristics when flexing, rendering established forms of kinematic modelling unsuitable. A nonlinear kinematics solution was provided for a single segment of a cable driven continuous backbone in [43], demonstrating the complexity involved when considering just one section. Thus, an alternative approach was taken, similar to that in [1], which is logical due to the parallels between the two designs; the arms are of different size and material but are both nonlinear in nature, and composed of multiple sections. The solution, as covered in [1], and again in more detail in [44], is to use a 'kernel regression method'.

This specific technique is known as the Nadayara-Watson kernel regression [45]. It operates on the principle of using known data points, interpolating between them to yield a solution from an input that is not necessarily covered by the existing data. The function itself is given below in (2). As with the Mark I design, it is safe to interpolate between the motor angles as the arm cannot self intersect.

$$\bar{y}(\bar{x}) = \frac{\sum_{i=1}^{n} y_i K(\frac{x-x_i}{\alpha})}{\sum_{i=1}^{n} K(\frac{x-x_i}{\alpha})} \tag{2}$$

Where K represents a kernel function, α the bandwidth, x the input and y_i and x_i the values of the stored data for the output and input values respectively. The bandwidth, as expected, will have a large effect on the output of the function. This is best represented by a plot of the output of the function for a range of inputs at various bandwidths. Of course, due to the number of dimensions in this case (three for the positional vector and four for the motor angles), it is difficult to give a visual representation, hence a 2D example is given in Figure 4.2 for the non-linear 2D function $cos(\frac{x}{10}) + (\frac{x}{60})^2$.



(a) *Kernel regression with 10 samples* **(b)** *Kernel regression with 100 samples*

Figure 4.2: Example of kernel regression being applied to a 2D function, image adapted from [44]. Lower bandwidth regression experiences a more accurate and smoother plot with more samples, whilst higher bandwidth remains smooth, but further away from the ground truth

As shown in the figure, for a smoother plot, covering the areas not contained in the stored points, at the expense of accuracy compared to the ground truth values, a higher bandwidth would be expected. However, a lower bandwidth would result in a rougher, more sporadic plot, with the effect of coming closer to the stored points at corresponding inputs. As shown by the two plots, adding more samples acts to mitigate the effect of this latter case, resulting in a smoother plot whilst improving the accuracy in relation to the actual values. An optimal bandwidth would of course produce a minimal error for comparison between ground truth values for the points the sampling data does not necessarily contain. High degrees of non-linearity are typically characterised by a comparatively lower optimum bandwidth.

Given the way this regression works, all of the recorded points will have an effect on the outcome. However, it is desirable for the recorded points close to the input value have a greater effect on the result than those further away, meaning smaller values of $x - x_i$ will contribute less to the $\bar{y}$ value. Therefore, to cater to this requirement, the chosen function, K, was a Gaussian kernel function, as given in (3), due to its bell-curve shape and how input values close to zero difference have a larger effect on the output.

$$K(u) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}u^2} \tag{3}$$

With the kernel regression function, the corresponding output values could be found for various different types of input values. The function itself does not depend on the number of dimensions, and hence the y and x values can be of any vector size, providing they are consistent, and that each input type value is paired with a corresponding output within the recorded data. This is because the value used as an argument for the kernel function depends on an absolute difference between the two vectors, which will be a single number regardless of their size, giving an overall output vector of the appropriate number of dimensions. The kernel regression function was then developed, as given in Appendix H.3, and for the scenario of the robot, and the data that it is concerned with, the corresponding output types for certain inputs is listed in Table 1.

Table 1: List of corresponding output vector types for a given input vector

Input vector	Output vector
motor angles $\bar{m}$	relative tip position $\bar{p}$
relative tip position $\bar{p}$	motor angles $\bar{m}$
relative tip orientation $\hat{d}$	motor angles $\bar{m}$
$\bar{p}$ and $\hat{d}$	motor angles $\bar{m}$

This list of input and output types is, as expected, similar to that given in [1], due to the comparable nature of the method. However, in this instance, only the tip position is returned when given the motor angles, due to less emphasis on physical precision. Once the regression method itself was formulated and tested with some sample data, it was necessary to acquire real calibration data, facilitating its use in practice, and to fine tune the bandwidth appropriately.

4.2.1 Calibration Procedure

The procedure for obtaining the calibration data involves sending a range of different motor angle configurations to the servomotors, and recording the resulting tracking information (relative tip-to-base position and orientation). Each data point then comprised all of these values, and was written to disk for future re-use within the regression function. When recording the data, it was imperative that the robot base remained stationary, so each data point is consistent with the same relative base position and orientation. As mentioned in Section 4.1, the base is said to have zero orientation about any of the global axes if it is in the orientation that it was when it was first identified in the Motive software (as a rigid body). Hence, the base was secured in a vice and then initialised in Motive (along with the tip), before commencing the procedure. This set-up is illustrated in Figure 4.3. The physical orientation of the base during calibration is of little importance (as long as all points are always visible to the tracking device), as the targets will be relative to the base and converted to its local axes before input to the function, as covered in more detail in Section 4.3.1.

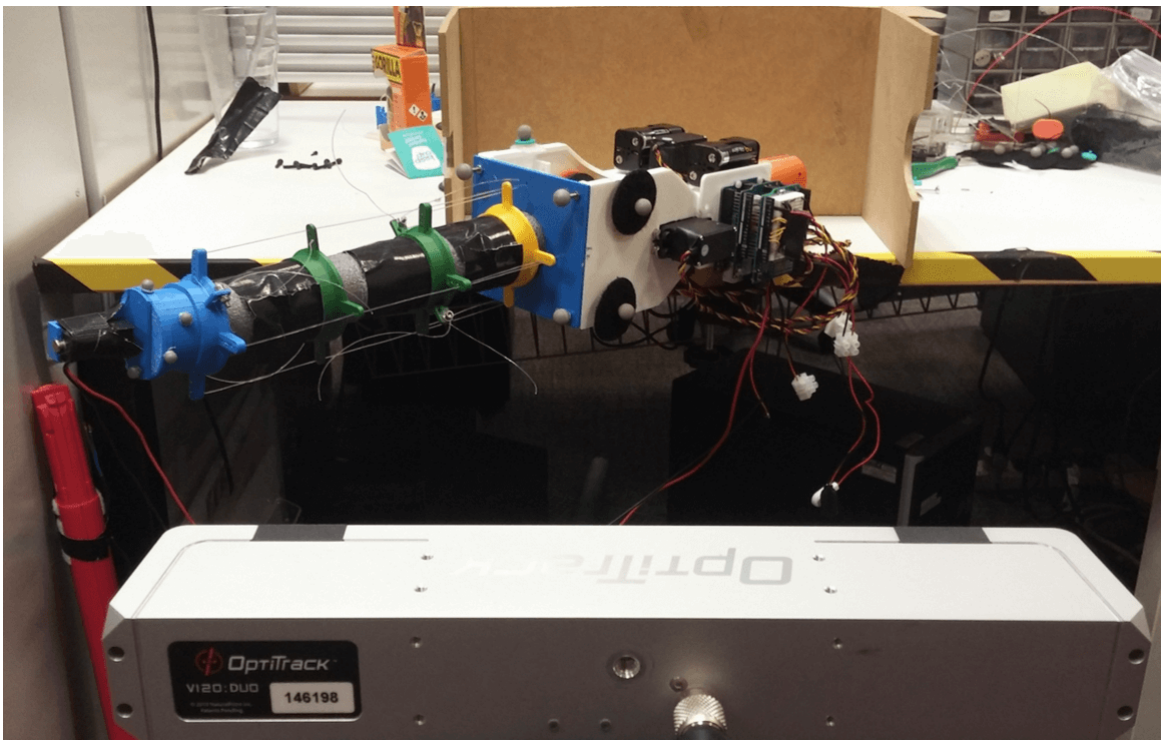


Figure 4.3: Set-up for the calibration procedure, clamping the device at an approximately 45° angle to the global axes of the OptiTrack Duo

The calibration sampling is then performed at a chosen resolution (number of degrees interval) by stepping through the full chosen range of motion for one servo, for each single step of a 'parent' servo. Thus if each servo (apart from the last) has a 'child' servo, the motions are performed recursively, with the servo stepping once, then its child covering its full range, with its grandchild covering its full range for each single step of its own parent, and so forth. The function is written such that after each step of a motor, its child motor then covers the range of motion in a direction opposite to its parent. This is to reduce the frequency of which a motor has to move all the way back to the start of its range before stepping again, providing smoother motion in general and reducing calibration time.

The state-space of the arm was initially sampled at a chosen resolution of five points over a total range of 60°per servo, giving a total sample number of $5^4 = 625$ calibration points. This gave a suitable range of motion, with an improvement of 20°over the device in [1]. Higher ranges were initially used, although occasional deformation of the foam arm meant that the cables needed to be re-tensioned more frequently, which was a time consuming process. A one second pause was issued after each calibration step to ensure the tip was steady before the position was logged. Hence this calibration took just over 10 minutes. Of course, higher accuracy could be obtained through more samples, with an increase in time, but this was kept short due to the fact that it needed to be repeated upon minor design changes to maintain reliability. Also, a meaningful increase in accuracy would require a large number of extra sample points, quickly increasing the required time to well over an hour, which introduces other problems such as battery life and gradual deformation.

4.2.2 Calibration Results

In order to find an appropriate bandwidth for the regression function, it was first necessary to obtain calibration data through sampling. Once enough data is available to perform the regression, the function can be tested against other sampled data, not covered in the initial calibration data, referred to as 'test data'. By comparing the output of the function for different bandwidth values against the ground truth values of the test data, an absolute positional error can be determined. The surface covered by this calibration procedure is illustrated in the Figure 4.4, depicting a portion of a sphere.

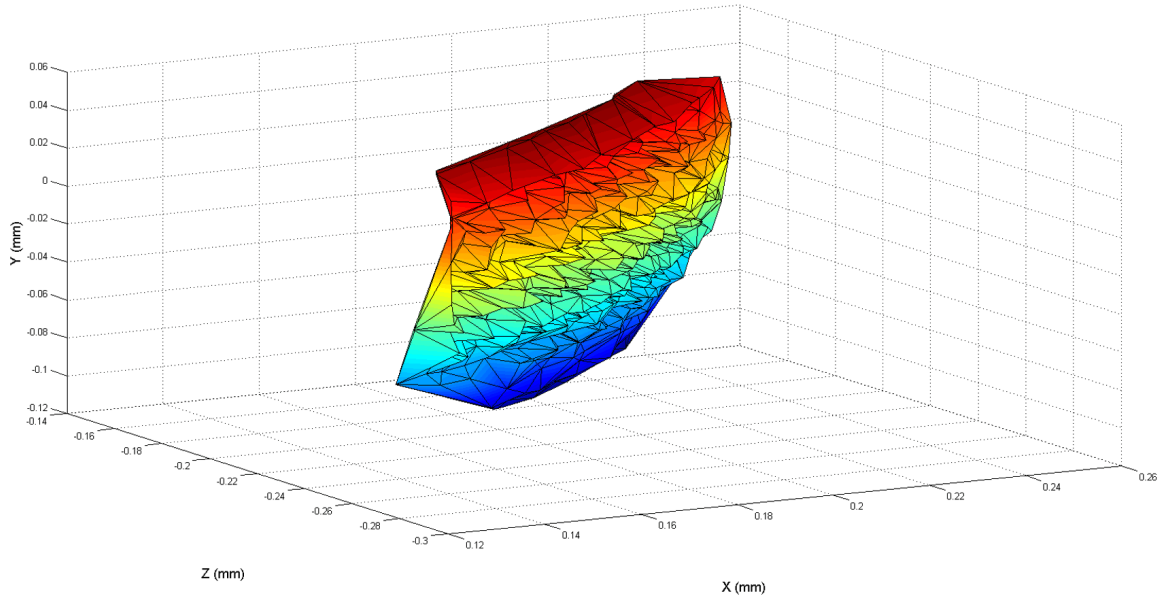


Figure 4.4: Surface plot for the 3D area covered by the calibration points, note the forward and backward spikes for when the arm took on an 'S' shape, pulling the tip inwards somewhat.

The number of test data points obtained was considerably smaller than the calibration data, with the primary aim of collecting data for motor angle vectors that were not covered in the initial sampling. Hence $3^4 = 81$ samples were obtained, with starting position and step selected to deliberately avoid any points sampled in the calibration data. For the comparison between the function output and the test data, the motor angles were to be the input, thus comparing the output positional values with the recorded ground truth positions for the same motor angles. This 'forward' approach was used for two reasons. The first is because the errors could be compared in terms of millimetres in space, which is more meaningful than servo degrees. The other reason is due to the fact that a given 3D position could potentially be achieved (or very close to) by more than one configuration of servo angles, whereas there is only one matching position for a given angle configuration. The 3D points for the calibration and the test points are illustrated in Figure 4.5.

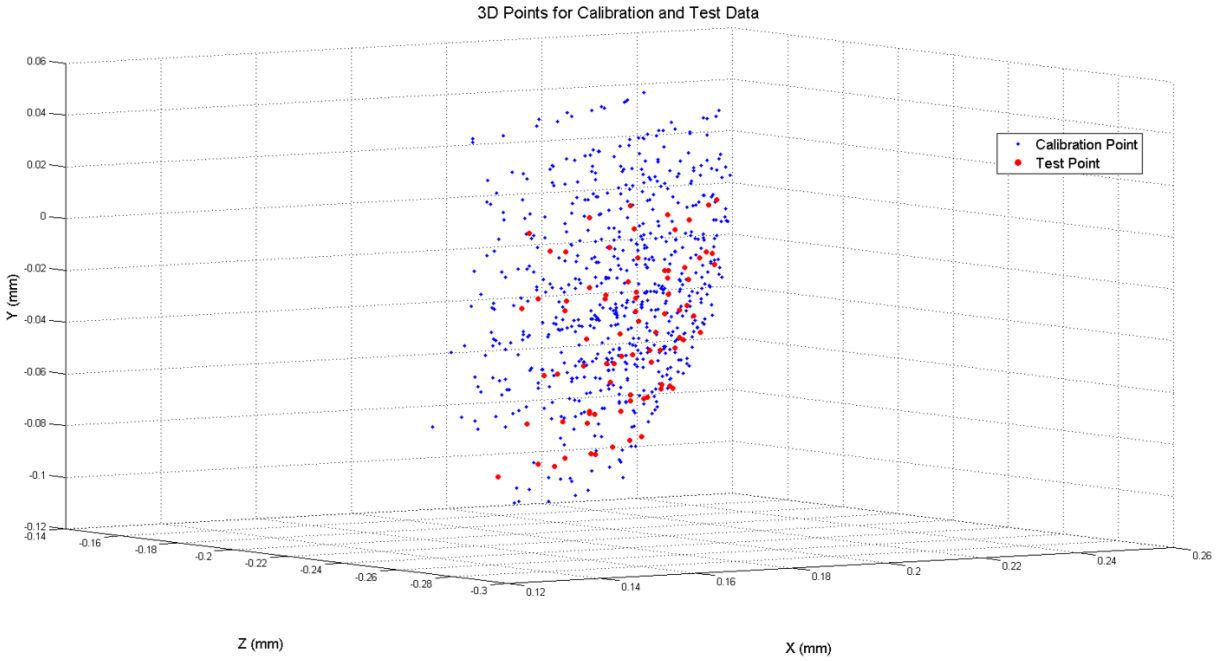


Figure 4.5: Calibration and test points represented in 3D space, note that although some of the points in 3D space may intersect, the underlying servo angles will all be different.

Due to the nature of the kernel function, large values for the input difference (the $x-x_i$ term) can cause issues. The input to the kernel function becomes larger when divided by the bandwidth (typically less than one), meaning when this value is squared, negated and made an exponent, the result is a very small number. This becomes particularly problematic when the value is so small that the number of bits required to represent it exceed those available in the 'double' data type, giving a result of zero out of the Gaussian kernel function. This then can then cause the regression itself to fail as the denominator becomes zero, giving NaN values as output as opposed to anything meaningful. Due to the differences in degrees typically observed between the servo angles of the input and the sample points, this issue was apparent. As such, the value of servo angles needed to be scaled so that the difference between any values is guaranteed to be in the range of $-1 < x - x_i < 1$. This is known as 'feature scaling', or min-max normalization [46], and is shown in (4), where x' is the new value after the original value, x, was scaled. This was performed for all servo angles as the regression function was carried out, and converted back after output to return corresponding results in degrees.

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{4}$$

The regression function was thus used for each of the 81 angle configurations covered by the test data, giving a mean positional error (in mm) between the output and the actual values from the test data. This was then repeated for 400 bandwidth values at an increment of 0.002 and plotted to determine the most suitable value to minimise error, as illustrated in Figure 4.6.

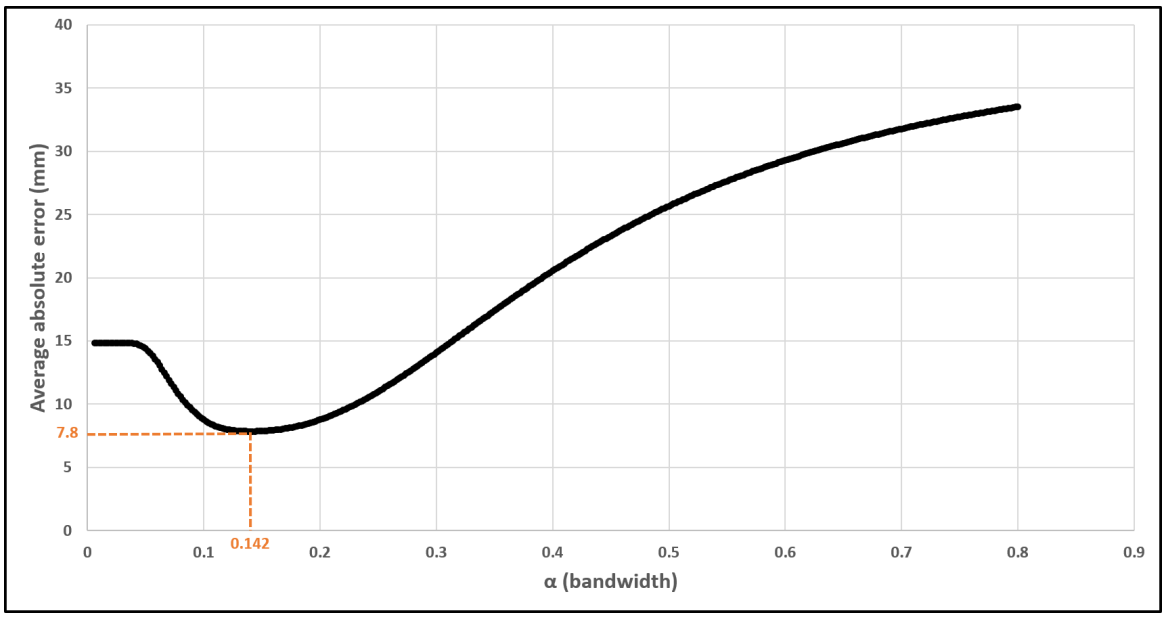


Figure 4.6: Mean positional error at various bandwidths for the output of the regression function compared with the corresponding real positional data

As shown in the plot, the optimal bandwidth (at the trough) occurs at a value of around 0.142, which yields an average error of 7.8mm. This bandwidth value can be considered relatively low, implying a high degree of non-linearity. Of course, this is expected for such a flexible structure, especially with four determining dimensions (the motor angles). The average error for this optimum value is not indicative of a high precision device. However, the physical capabilities of the robot are of much less concern in this project, needing only to be sufficient so that the device can cooperate with the user in a manner that can lend mental assistance. As such, this value was deemed acceptable for the intended purpose of the device.

Of course, if desired, actions could be taken to minimise this error, improving the physical accuracy of the robot in reaching a target point. The calibration procedure may be repeated at a higher sampling rate, such as 11 motor angle iterations per recursive call (total of 11^4 samples). In this instance, the added precision afforded by more calibration points would be of small benefit to this application when compared with the increased computational time required to carry out the regression function, causing added demand on the program and potentially affecting the responsiveness of the device. This effect could be mitigated by the introduction of a KD tree to search only for the closest calibration points, and only using such values in the regression function, as covered in [44]. However, for the scope of this project, and the required function of this device, such actions would be excessive.

The values found in this calibration procedure were then stored, and the optimal bandwidth figure was to be used as a constant in the program for performing the kernel regression function. The estimated error was also considered in experimental choices, knowing that any target regions that need to be reached reliably must be conservatively larger than this error figure in all three dimensions either side of the absolute centre.

4.3 Algorithm for Responding to Three-Dimensional State Targets

Once the robot has been calibrated in such a way that there exists a suitable range of data points in which to perform the regression function, along with the selection of an appropriate bandwidth, the device is then in a position to respond to 3D targets. This process is fundamentally built on inputting a positional vector into the regression function in order to ascertain the required motor angles. This input vector is with regard to the required tip position relative to the base, in accordance with the corresponding calibration data. The resulting motor angle vector is then written to the servomotors via the Arduino in a controlled manner.

However, in order for this approach of obtaining servo angles for a given position to be useful in practice, it is first essential to determine the appropriate input to the regression function. This can depend on a variety of factors relating to the absolute 3D target point and its relation to the base of the robot, such as the current orientation of the base and the relative proximity of the target.

4.3.1 Acquiring Target Position Relative to Orientation of Base

The robot was fixed in place while the optical tracking software is configured to recognise the rigid bodies formed by the retro-reflective markers associated with the base and the tip, as covered in Section 4.1. At this point, the base is first identified and is said to have zero orientation about any axes. The calibration procedure was then undertaken while the robot was fixed in this position, maintaining zero orientation. Thus, the stored relative tip-to-base calibration data for different servo angle vectors is only valid for the robot in a zero orientation state compared to the starting position. Hence, in practice, when the base has changed orientation about any of the three axes of the optical tracking device, a direct input for a relative tip-to-base position will yield incorrect motor angles to achieve this position. Therefore, the target position must be adjusted in order to account for the varying orientation of the base.

The orientation of the base itself is initially stored as a quaternion within the rigid body object from the NatNet library [47]. This is then converted to Euler angles using a built-in method to represent the orientation as three angles about the tracking device's X, Y and Z axes respectively, of which the order of rotation is critical. It is thus necessary convert a real target point to an equivalent point relative to the original orientation of the base (zero orientation), so that the regression function and stored calibration data can be used in accordance to yield servo angles that give the correct position in real space. This is accomplished using a rotation matrix that accounts for the rotation ordering of the base. Given that the base orientation is represented by rotation about the X, then Y, then Z axes, the converted point will be attained by taking the actual target point relative to the base and reversing these rotation operations; that is, the negatives of the base rotation values about the Z, then Y, then X axes. The rotation matrix in (5) is thus used to account for this rotation order about the axes.

$$R_x R_y R_z = \begin{bmatrix} \cos\beta\cos\gamma & -\cos\beta\sin\gamma & \sin\beta \\ \cos\alpha\sin\gamma + \sin\alpha\sin\beta\cos\gamma & \cos\alpha\cos\gamma - \sin\alpha\sin\beta\sin\gamma & -\sin\alpha\cos\beta \\ \sin\alpha\sin\gamma - \cos\alpha\sin\beta\cos\gamma & \sin\alpha\cos\gamma + \cos\alpha\sin\beta\sin\gamma & \cos\alpha\cos\beta \end{bmatrix} \tag{5}$$

This equation gives the rotation matrix for rotation about Z, then Y, then X axes. The symbols α, β and γ represent rotations about the X, Y and Z axes respectively, adapted from [48]. A simplified example of solving from a target point to get the appropriate relative positional vector as an input to the regression function is illustrated in Figure 4.7. Note that the target position in the image is not to scale.

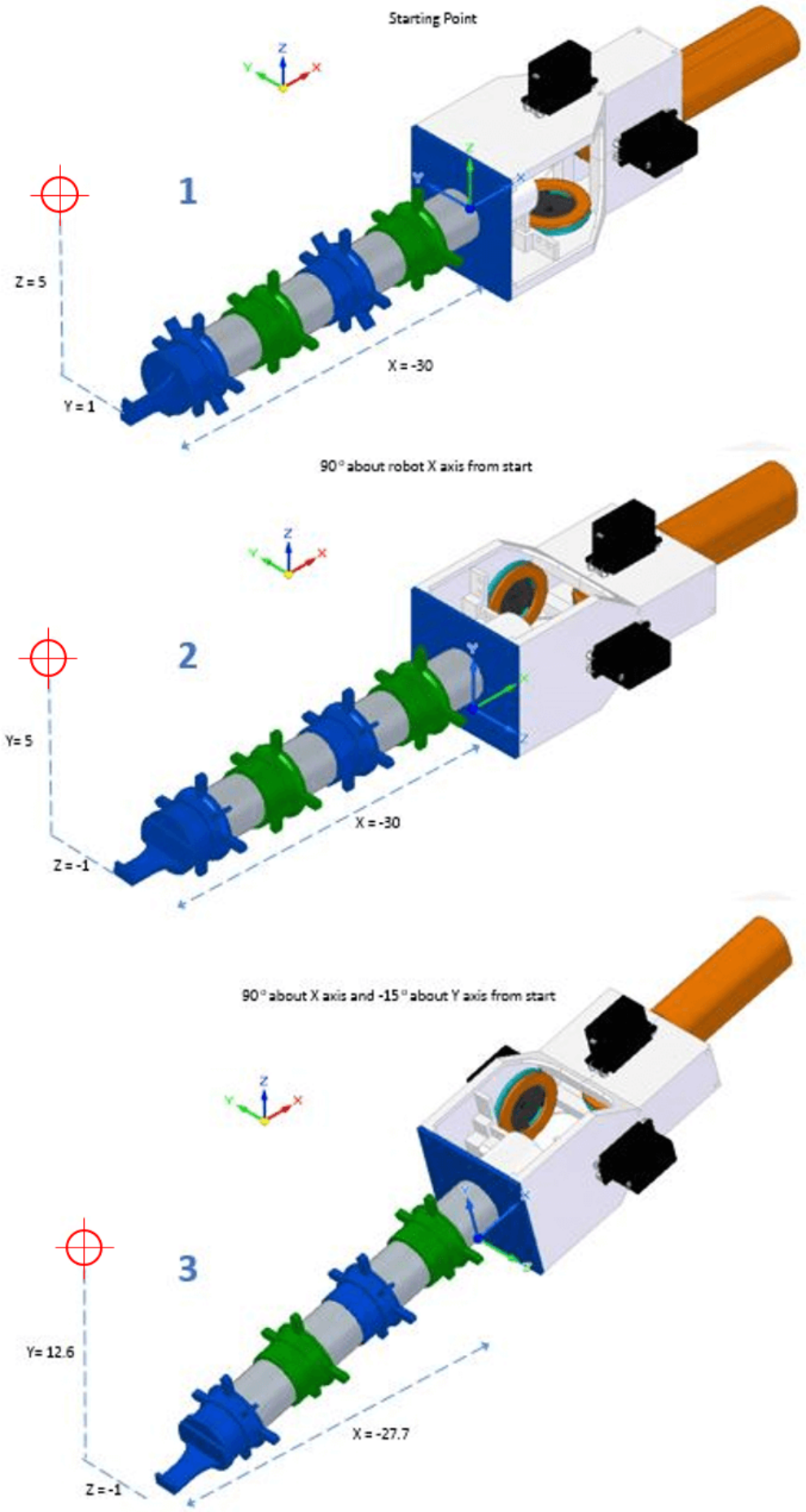


Figure 4.7: Example showing how the position and orientation of a point changes with base rotation

1. The robot is in its original starting orientation, and can be said to have zero rotation about any of the axes. There is no relative difference between the axes of the robot (at the base of the arm) and the global axes (shown in the top left of each image). The target point has an actual relative distance to the robot base of -30, 1 and 5 units respectively for the X, Y and Z dimensions. As there is no relative axes difference, the relative distance in relation to the robot base axes is identical in all three dimensions. Therefore in this situation, the actual relative positional vector could be inserted into the regression without adjustment.
2. In the second stage, the robot has been rotated 90°about the global X axis. The local axis of the robot base is therefore at a relative orientation difference of the same amount. The physical distance of the target point to the base is unchanged, but the target point relative to the local axes of the base is different, with the local Y and Z distances being swapped around, and the latter being negated.
3. The final image shows the outcome if the relative rotation of the base was a 90°rotation about the global X axis (as in the previous image) followed by a -15°rotation about the global Y axis. The physical distance of the target point to the base is again unchanged in the global axis as the base itself has not translated in any direction. Again, the relative dimensional distances in terms of the local axes of the base are altered further, with relative values now of -27.7, 12.6 and -1 respectively for the X, Y and Z dimensions. This is the vector that must be inserted into the regression function, so that motor angles can be attained that result in a 3D point which corresponds to the absolute target. This conversion of the target point into the local axes of the base is performed via the aforementioned steps, with the negation of base angles and the rotation matrix in 5.

4.3.2 Gesturing Algorithm for Distant Targets

In order to be cooperative and interactive in practice, a handheld robot must be able to effectively indicate positional preferences to the user. This may be with regard to where it must be located itself in order to carry out a given robotic operation, or even just the next point the user must consider for task progression. This process is referred to as the robot 'gesturing' to the user towards important 3D locations. In this scenario, the selected method for gesturing is comparable to typical 'pointing' operations, such as by finger, which is beneficial in terms of simplicity of use. Pointing is intuitive to the user as humans do this naturally to direct the attention of others to areas of interest, and is considered inherently linked to speech and communication [49]. Therefore confusion should be avoided in the user knowing where the robot is indicating to.

In order to gesture effectively to the user, the 3D target points must be categorised as achievable or not from the outset before deciding the next step. Initially, a point can be said to be immediately unachievable if the relative distance between itself and the base of the robot is greater than (or a certain amount less than) the typical tip-to-base distance in a straight-arm configuration. This tip-to-base distance can be used as the radius of a sphere which governs whether the points can be immediately disregarded as unattainable, requiring gesturing, or whether further checks are required to assess whether it is appropriate to attempt to manoeuvre to this point directly. This concept is illustrated in Figure 4.8.

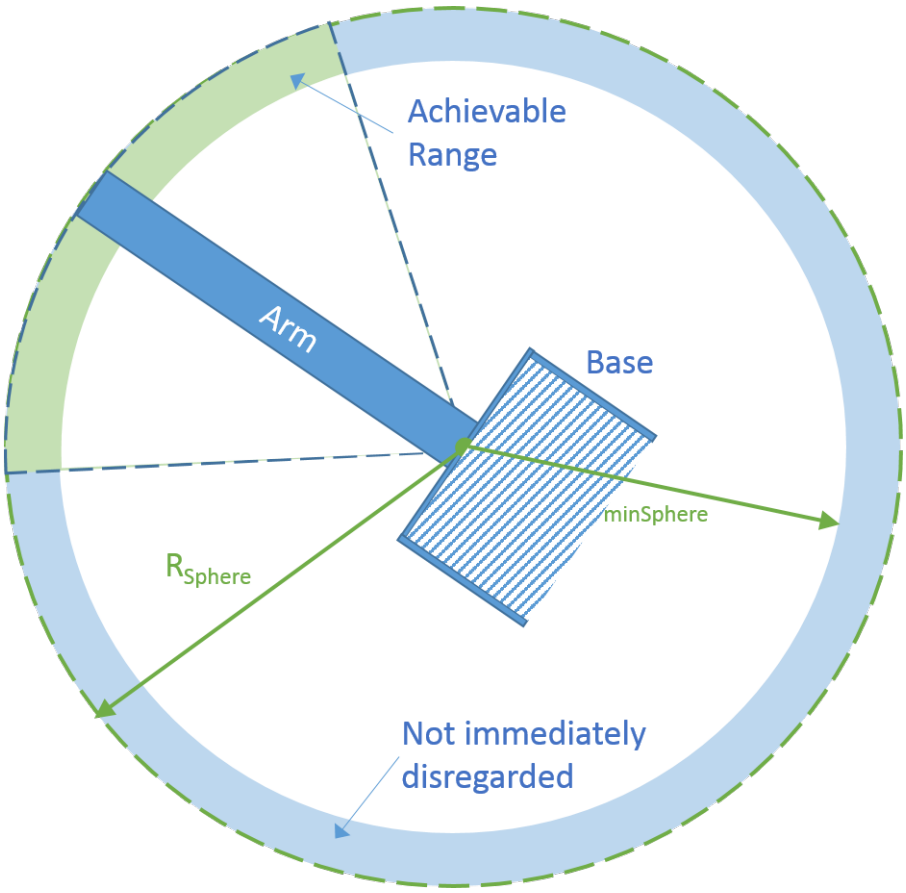


Figure 4.8: Concept of tip-to-base sphere marking immediately disregarded zone, and initial achievable range zone that requires further checking

If the point lies at an absolute distance that falls within the range between the radius of the sphere and another minimum radius, $R_{minSphere}$, the point is not immediately categorised as unachievable. This minimum radius is determined by the smallest absolute tip-to-base difference from the stored calibration data (via 3D Pythagoras). However, if it lies outside this range, a new target is required that is equivalent to shifting the original point along the vector between itself and the base until it falls within this range.

The current target then lies somewhere in this intermediate range, and a further check is carried out to determine whether the coordinates of the target point relative to the base fall outside any of the recorded minimum and maximum values for the X, Y and Z dimensions. If not, then the target may be directly achievable, and the positional vector is inserted straight into the regression function to check if motor angles can be yielded. However in the event of this still not generating a valid solution, or if the point does fall outside any of the min/max ranges, a compromise is required. This entails selecting a new target point from the stored calibration data that is geometrically closest to the target point in question. Thus, if not able to match the point, a new point should be determined to exhibit a suitable gesturing motion. The typical process for responding to a target is illustrated in Algorithm 1 and a typical example is provided in Figure 4.9 for a far-off point.

Algorithm 1 General algorithm for getting from 3D target position to solution in motor angles

while running experiment **do**
  get new frame data and separate into rigid bodies
  calculate target relative to base
  convert target orientation to local axes of the base
  if $relativeDistance > maxSphereRadius$ **or** $relativeDistance < minSphereRadius$ **then**
    shift along direct vector until on sphere
  end if
  if target falls within recorded max and min values for recorded X, Y, and Z positions **then**
    attempt direct solution from kernel regression function
  end if
  if 3D point outside recorded max or min values **or** direct solution failed (NaN values) **then**
    find closest geometrical point from calibration data
    use corresponding motor angles at this point
  end if
  set the motor angles to the solution via Arduino
end while

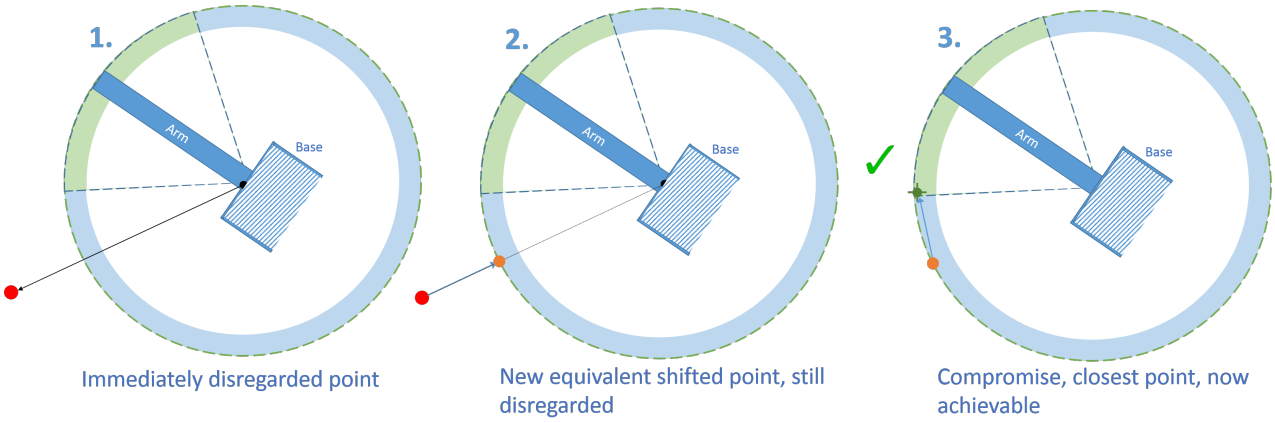


Figure 4.9: Typical steps in obtaining a compromised gesture solution from an initially unachievable point

1. The given target point is immediately disregarded as it lies at an absolute distance above the radius of the sphere.
2. A new target point is formed that is the equivalent of shifting the original point along the line from itself to the base. This is done by determining the required factor to shift the target to lie on the sphere. The factor itself is $ShiftFactor = R_{Sphere}/absolute\ distance$, and each coordinate of the target point is multiplied by this to give a new positional vector on the sphere.
3. The point is still unachievable as it lies beyond the max/min range of some of the dimensions from the calibration data. The compromise is therefore to find the point in the calibration data that has the closest absolute proximity to this point, and use that either as an input to the function or simply using the stored motor angles at this point (quicker).

With these steps of applying a rotation transformation, ascertaining the correct input and determining the most suitable target to aim for relative to the base, there now exists a general algorithm. This can receive any given target position, expressed relative to the base, and generate a solution in the form of angles to be applied to the servomotors, manoeuvring the arm to the required position, as outlined in Regardless of the task or application at hand, this algorithm will be central, receiving inputs based on the situation and yielding solutions to be applied. This fundamental step sequence is illustrated in the form of a flow chart in Figure 4.10.

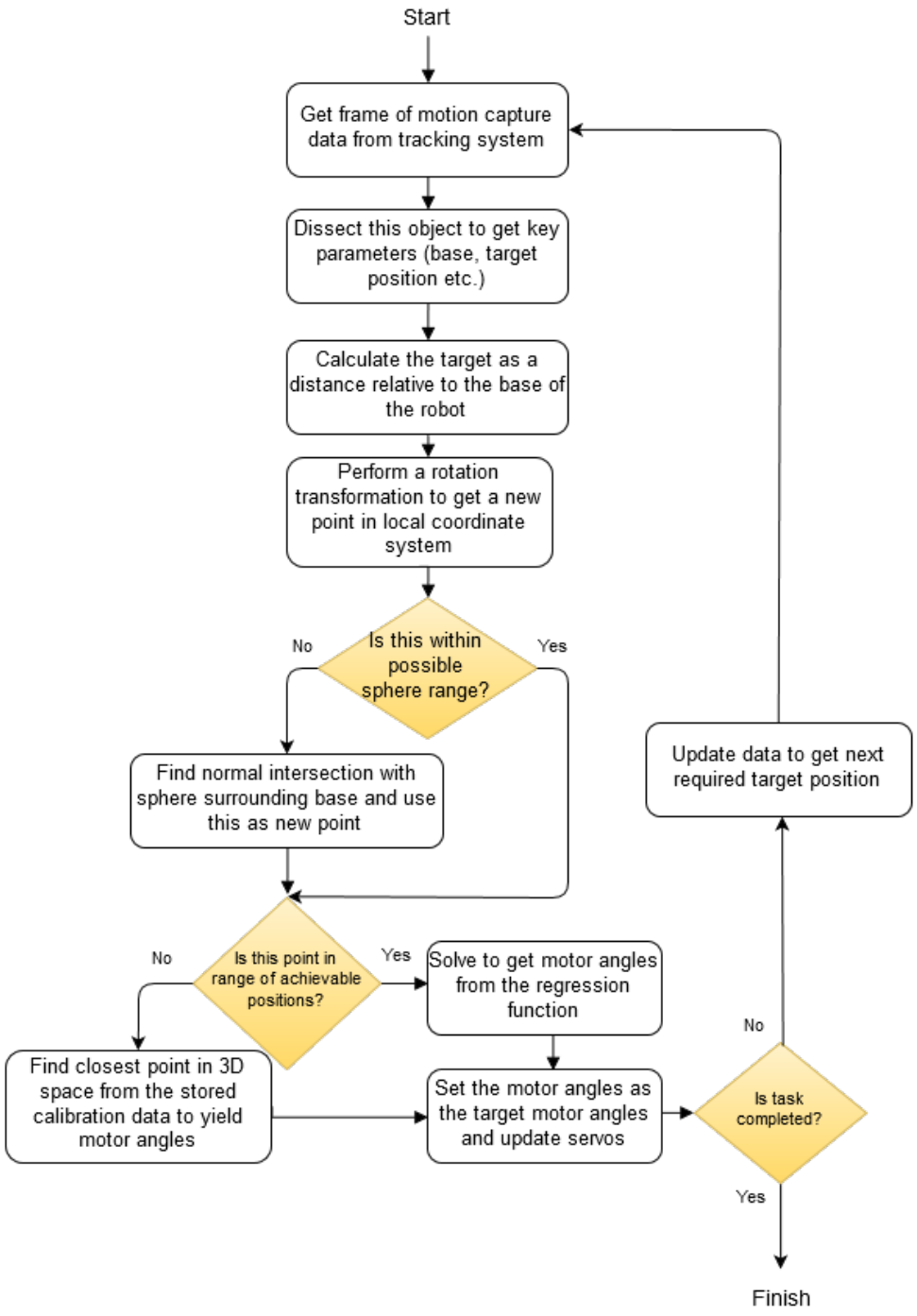


Figure 4.10: General algorithm procedure for taking a 3D target point relative to the base and producing the most suitable solution in the form of angles to be applied to the servomotors

4.3.3 Running with a Live Target in Real-Time

In order to be viable for handheld robotic applications, the device must be able to run with a live target in real-time, always attempting to move or gesture to the required position. Without this feature, the cooperation between the user and handheld robot would suffer, with the device appearing unresponsive, undermining the usefulness of the robot altogether. As such, it was imperative to set up the control of the device so that it could receive a live input, and respond accordingly to changes in its own position and orientation. This entails taking a fixed vector as a constant input, converting it to a position that is relative to both the position and orientation of the base, obtaining the solution angles and moving applying this target configuration to the servos. A simplified representation of this process in real-time is illustrated in Figure 4.11.

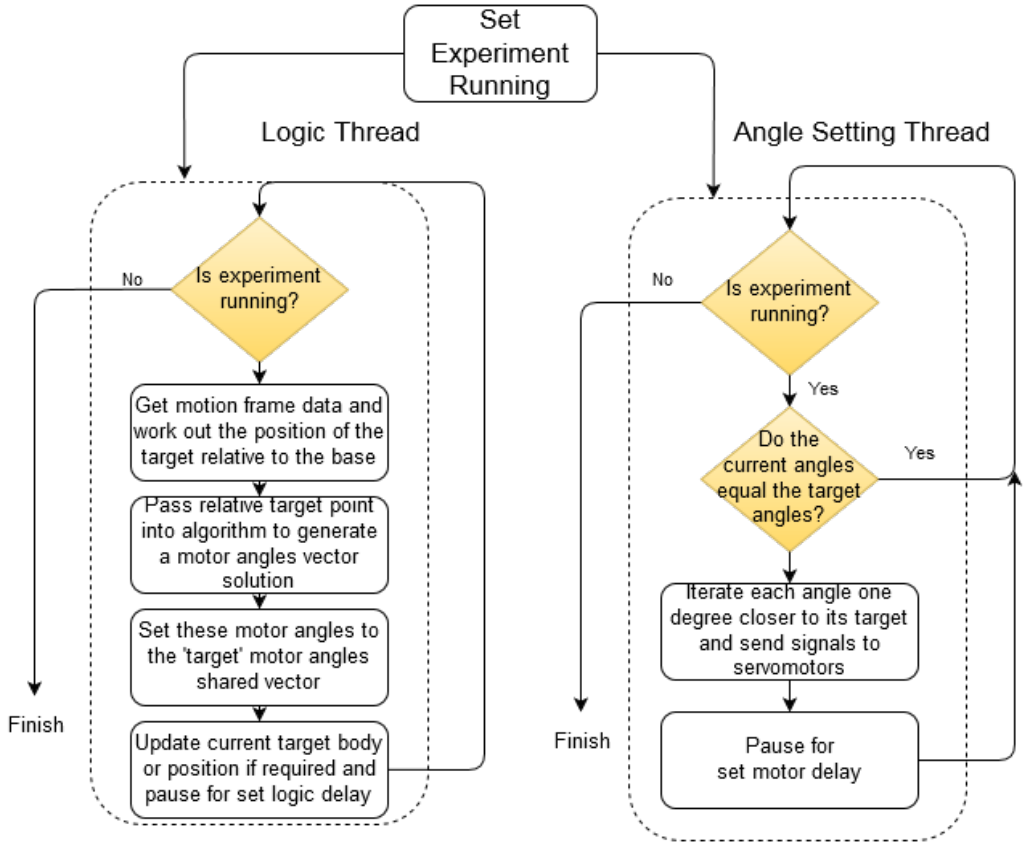


Figure 4.11: Simplified processes taking place in the two different threads, showing the servos are updated after a set interval regardless of the logic

The flow diagram shows how the control system splits this process into two threads. One thread is responsible for the logic of converting the target position and obtaining a servo configuration solution, as covered in the algorithm in the previous section. The other is solely responsible for taking the current solution of the servo configuration and iterating the servos towards this target (if they need to be changed). Hence these two threads share access to the servo configuration vector in a thread-safe manner, with the former updating it as necessary when a new solution is found, and the latter using it to determine how to iterate the servos towards the target. A typical situation where this multi-threaded technique is employed in code is given in Appendix H.4 for the user study code.

There is a ∼6 millisecond delay after iterating each servo by one degree towards its target to give a smooth motion of the arm, anything less than this was found to cause a 'jerked' motion that had the potential to pull a cable loose from its pulley groove. Moreover, by iterating towards the servo configuration one degree per loop with a small delay, the effects of overshoot will be greatly reduced. By rapidly advancing towards the target, the momentum of the heavier end effector (containing the magnet) means it is more likely to carry on past its target point. Also, there is a delay at the end of the logic operation to reduce computational demand and to try and minimise rapid perturbations in the target configuration from small fluctuations in the base position and orientation etcetera. This value can be varied accordingly to alter the responsiveness of the robot at the expense of avoiding these traits completely.

This method of splitting the two tasks into separate threads, with one of them iterating the values as opposed to setting them directly is highly advantageous. As soon as the relative target is changed in real-time, requiring a different servo solution, the servos are instantaneously iterated towards this new solution. Thus, the arm does not continue moving towards an old position when there is a more recent target; which may also be construed as minimising a form of overshoot where the tip moves past the current target position due to referring to a past point. This contributes towards the device's responsiveness in an ever-changing requirement, which is essential for the practicality of a cooperative handheld robot.

4.4 Program Design

The developed program for using the robot was built around the Motive software and the NatNet library. An example was provided in the NatNet SDK that simply establishes a connection between the tracking program and displays the tracked data in a live spreadsheet. This example took the form of Windows Form Application, due to the usefulness of a Graphical User Interface (GUI) with buttons and options to start/stop tracking etcetera. Hence the developed application was of the same type, re-using the necessary functions for setting up the connection with the Motive program and extracting the tracking data. Features were then added as necessary for the specific purpose of tracking and controlling the robot depending on the environment. This entailed creating new classes concerned with interfacing with the device, as well as the logic involved with the calibration, target selection and solution generation processes.

Connection with the optical tracking system is not possible unless the Motive program is running simultaneously, and is configured to track the necessary rigid bodies. Hence, before running the new program, Motive must be started and a 'project' loaded or set up. This Motive project contains the information necessary to track certain bodies, such as the exact marker configurations corresponding to specific rigid bodies, which must be labelled correctly. Certain features of the developed program require different tracking data; for example the calibration procedure only requires that the base and tip be tracked, whereas 'live body following' is not possible without tracking an extra rigid body. Various project files were therefore created for the different conditions. The interface of the Motive program, whilst tracking the base and tip of the robot, is illustrated in Figure 4.12.

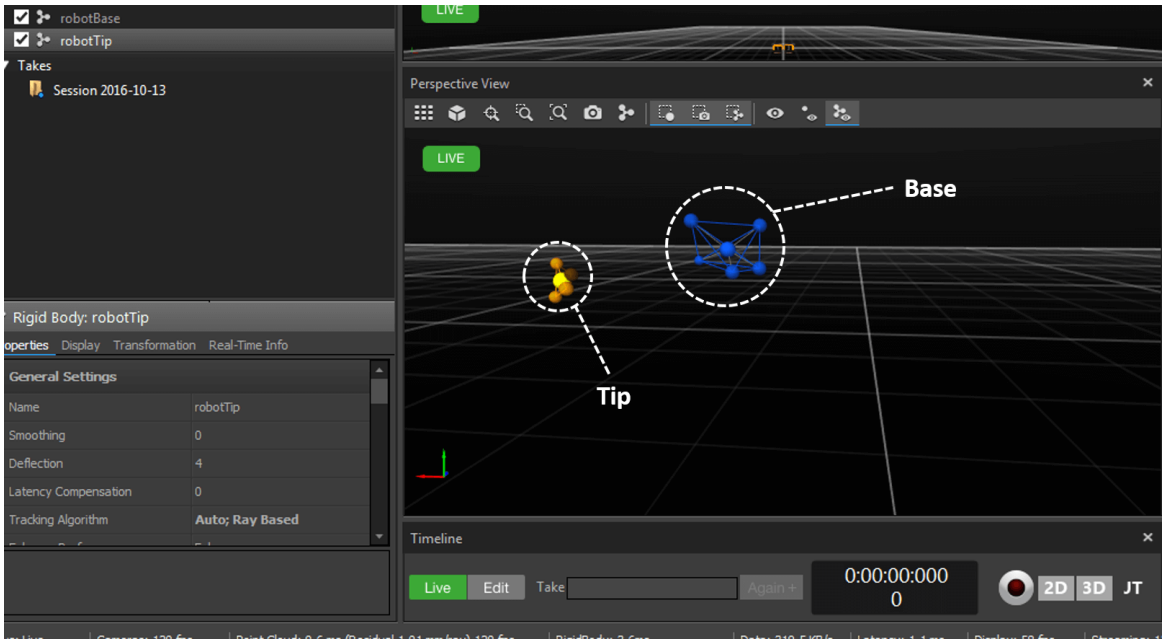


Figure 4.12: User interface of the Motive software, set up to track rigid bodies for the robot base and tip.

Small tests were also added to the program so that the motion of the robot, or solution generations from the regression function for instance, could be readily tested without having to fully connect to the external optical tracking system etcetera. Such tests would employ dummy variables where necessary, to quickly discern whether functions were performing as intended. The general process of adding a new features to the program would entail adding buttons to the GUI, and attaching methods to the callbacks associated with them. It is often necessary to have this functionality run in a separate thread so as not to hinder the updating of data or the user interface itself, rendering it unresponsive while the operation is carried out. Hence it was essential to ensure thread safety, using a certain lock depending on which shared object is being accessed, as well as delegating any user changes to the user interface to the thread responsible for updating it. This simple user interface of the developed program is illustrated in Figure 4.13.

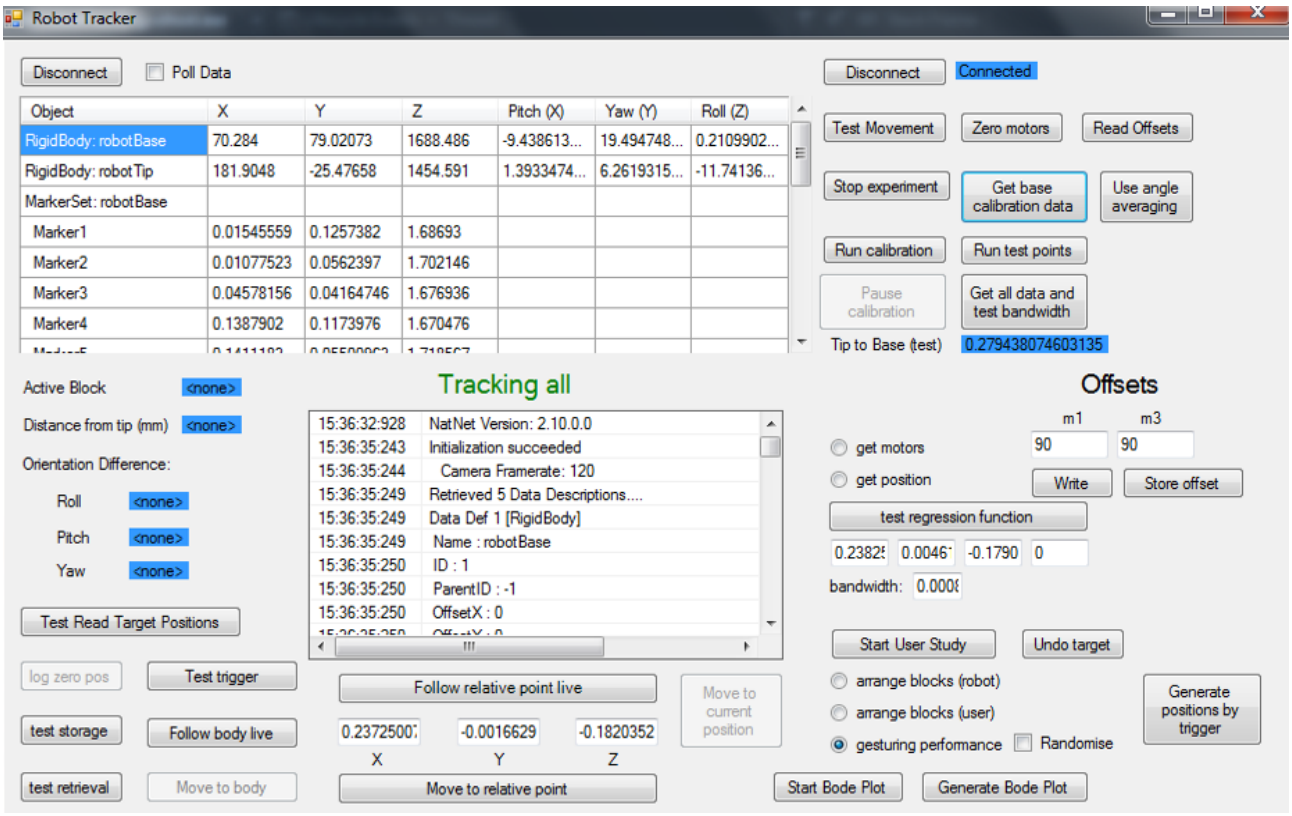


Figure 4.13: User interface of the program for controlling the current status and operations of the robot, as well as for connecting to the tracking system and reading key tracking data.

The upper left and right corners provide buttons for connecting to the Motive optical tracking software and the Arduino for robot control respectively. As the typical procedures regarding controlling the robot depend on analysing tracking data and sending instructions accordingly, both connections must be established before any practical operations can be performed. Hence full connection is required to start an 'experiment', which entails creating an Experiment object based on the information provided from both. However, as alluded to earlier, a 'dummy experiment' will be created without full connection, so that some basic functionality will be provided in order to carry out simple motion and logic tests without having to set up and connect to external infrastructure.

A number of options are provided for both testing and essential processes. The calibration procedure is initiated, stopped and paused (in case of battery failure) for both the main calibration data as well as the test sampling, using the corresponding buttons. Certain options were essential, such as reading stored calibration data into memory, without which the regression function could not be performed. The process mentioned in Section 4.2.1 for discovering the most appropriate bandwidth value can also be carried out. There is a live spreadsheet section in order to display the current tracking data being recorded, such as the position and orientation of the various rigid bodies and associated markers. Similarly, labels have been added to display values that have been calculated from this tracking data, such as the absolute distance from tip to base.

Generally, buttons were integrated in order to provide simple control the various features of the program. The callbacks attached to them were responsible for altering the GUI appropriately, such as activating other dependent buttons, and initiating the work for the corresponding action. This mostly entailed calling a wrapper function that would set up a new thread, if required, and call the methods in which the bulk of the computational effort is carried out, passing in any necessary parameters such as user entered information from the GUI. The key features of the program are listed in Table 2.

Table 2: List of key features of the program and their corresponding descriptions.

Feature	**Description**
Motion Test	Simply undergoes a brief procedure of setting each servo to a value either side of its neutral position to ensure they are all connected and working.
Servo Offsets	Allows offsets to be written to the two servos responsible for vertical motion, to compensate any 'leaning' when placed horizontally. Values can be entered and applied, added onto each instruction to the corresponding motor. Can also be stored to file and read in on start-up.
Calibration and Test Points	Runs the procedure associated with recursively iterating the servo angles for each procedure, and saves the tracked data to file accordingly. The calibration can be paused or stopped at any time, if any mechanical parts require attention. It pauses automatically if it doesn't detect recent motion, implying the batteries need replacing. Of course, the calibration data must be read in from file before any use of the regression function, so this is also facilitated.
Regression Test	Allows the user to enter a set of inputs and test the regression function to ascertain either the servo angle solution, or the positional solution, for a specific bandwidth. This is especially useful for debugging, and can reveal issues with the stored calibration data.
Test trigger	While running, this function simply prints whether the trigger is held down and when it is released, activating the magnet if it is connected. This was used to eliminate noise from the switch, 'debouncing' the trigger signal as appropriate so that it would not falsely assume it had been released.
Move to relative point	Allows user to input relative coordinates of the target to the base and have the tip move to this target. Can also follow this point live, so if only the base orientation is changed the tip should remain in the same physical position.
Move to current position	Takes the actual relative coordinates of the tip and uses this as an input to the solution function. The amount the tip then moves corresponds to the error at that point between the solution and reality. Ideal condition is zero motion.
Follow body live	Allows live following of another tracked body named "bodyToFollow". The current relative position of this body is repeatedly used as input to solution function. The same method is employed for recording the data for the response test plots, only with the current time value and positional data written to file. Can also just move once towards to the target for small tests.
Generate target positions	Initiates procedure whereby the arm remains straight and the user clicks the trigger to log the absolute position of the tip at the time. When stopped, all of the logged positions are then written to file, with the file name depending on which radio button is checked.
Perform user study	Allows a specific variant of the user study experiments to be performed, depending which radio button is checked. Stored positional data for the specific experiment is then loaded from file and used as a sequence of inputs to the robot. Pressing the trigger advances the current target position to the next in line. Also, an undo trigger button was supplied, to revert back to a previous position in the order if the trigger was pressed or released accidentally.

The core classes of the program are illustrated in Figure 4.14. Note that only the essential fields and methods that dictate how the program operates are shown, with the boilerplate NatNet methods, the smaller sub-methods and the less critical methods for testing and set-up all excluded for simplicity.

1. RobotTracker - The starting class that was built upon from the original NatNet code for interfacing with a tracking system via a simple Windows Form Application. This class is responsible for establishing the connection with the tracking system, as well as dealing with the corresponding data it receives. On top of this, it contains objects for the other main classes and methods that instantiate and utilise them. This is mostly achieved through callback methods associated with the user interface buttons, giving control to the user.
2. Experiment - The class that extracts the key information from the tracking data into specific variables, and performs the logic required for ascertaining the values that need to be passed into the regression function, and performing the regression itself. This entails all of the steps outlined in Section 4.3 regarding converting the target orientation and finding a sphere intersection etcetera. The calibration procedure, and the loading of the calibration data, is also performed in this class, along with other test processes.
3. DataPoint - An object to store the motor angle vector and its corresponding relative tip-to-base position and orientation values as recorded by the tracking system. Many of these objects will make up the calibration data, and the values contained within them will be used when the regression function is performed, depending on the type of input/output. Serializable as with CalibrationData.
4. CalibrationData - A collection of DataPoint objects, that can be accessed as if it were an array with an index, adding and removing points as necessary. Of course, this will be serializable as it will need to be written to and read from disk.
5. RobotControl - This class is responsible for physically controlling the robot via its connection to the Arduino which, in turn, controls the servomotors. A connection is established with the Arduino and values are sent via serial connection, which are subsequently interpreted by the Arduino script and servo signals are applied accordingly. As mentioned in Section 4.3.3, the values of the current motor angles are compared with their targets, and iterated one degree closer if required before being written to the servomotors.
6. UserStudy - This class is used for carrying out a user study with participants. It is designed to extract pre-recorded 3D positions value from file and load them into an array for the robot to access in order. It is constantly updated and informed of the trigger status and the base position to determine the relative target position and whether to move on to the next in the sequence(if trigger pressed).

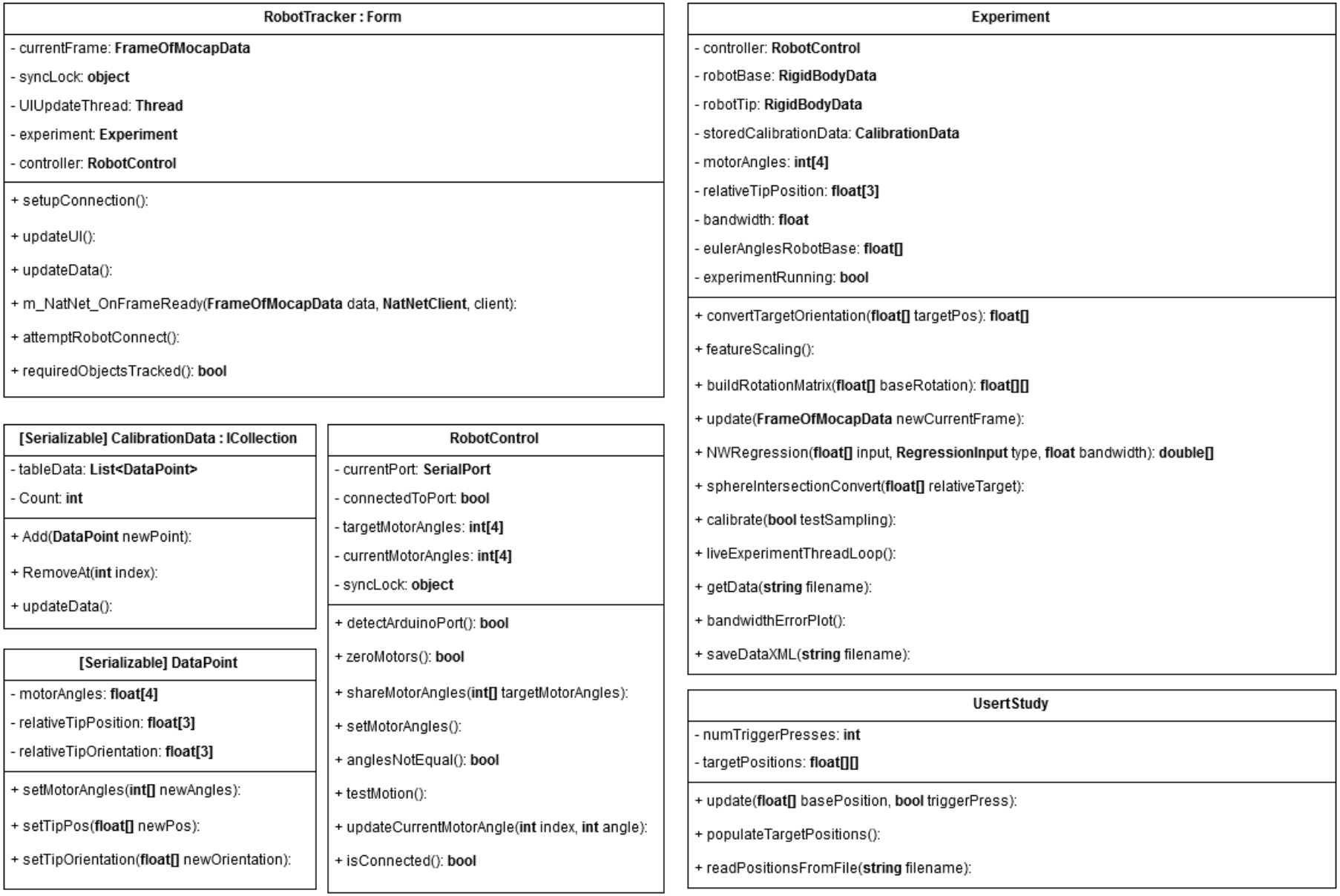


Figure 4.14: Core classes and their associated fields and methods that are most central to how the program runs.

5 Experimental Procedure

In order to evaluate the new handheld robotic device, it was important to identify the key characteristics that need to be identified so that the aims of the project can be achieved. This project is primarily focused on the design of a new device, and importantly, how it can utilise pre-acquired task knowledge to effectively cooperate with the user. Specifically, it is the alleviation of cognitive effort that is of concern, as this demonstrates the notion that unskilled or untrained operators in a certain task could achieve the objectives regardless, through assistance from the robot. Therefore it is essential to highlight this ability, and as such, a user study was to be carried out. The aim of this was to compare the completion of a mentally demanding task without any assistance from the robot, to a physically similar task where the cognitive burden is reduced through assistance from the device, gesturing the required actions to the user.

The physical performance of the robot is not of primary concern in this project, based on the tasks considered for evaluation being associated more with cooperation than precision motion. However, certain physical aspects, that would directly influence how effectively the robot can gesture (or cooperate) with the user, still need to be investigated. As such, it was necessary to determine whether the response characteristics of the robot were sufficient enough to actually carry out a user study test. For instance an excessive lag at typically expected input frequencies could seriously hinder the cooperation with the user and, in turn, the task performance. These traits could be dependent on the mechanical design of the robot, or the underlying algorithm and data that governs how the robot responds. Therefore, it was important to quantify the control performance of the robot in terms of step and frequency response. Similarly, it is also necessary to test how effectively and accurately the robot can gesture to specific target points, which is fundamental to its intended operation. Therefore an experiment was set up to ascertain if target positions could be reliably conveyed to the user.

5.1 Response Performance

In order to measure the response of the robot to various target inputs, it was necessary to constrain the robot and assign a live target that could be moved as necessary. As such, the robot was clamped in place using a vice, similar to the calibration procedure in Section 4.2.1. The OptiTrack Duo was then set up to have a clear view of the both the base and tip of the robot. Then, in the motive software, a new rigid body was defined using a small 'wand' made up of three-markers, as illustrated in Figure 5.1. The pivot point of this rigid body was then set as one of the single markers, giving an actual visual indicator in the real world of the 3D point that the robot will attempt to follow.

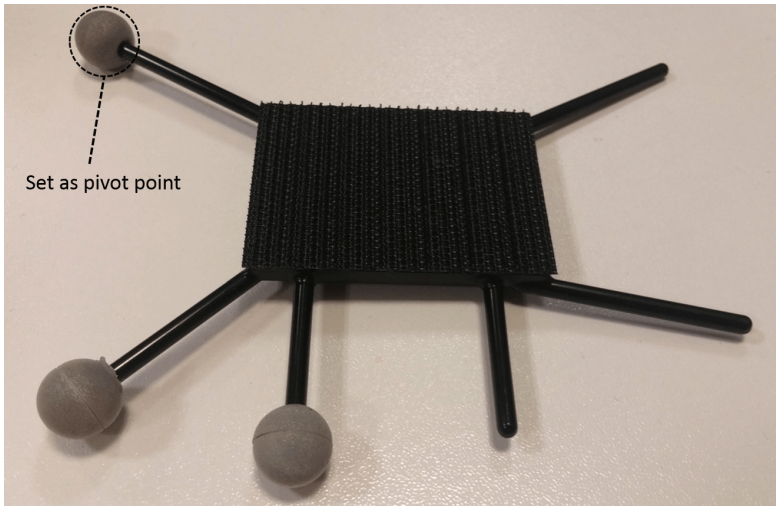


Figure 5.1: Wand used for live object tracking during response tests, showing which marker was used as the point to follow.

The absolute position of this wand, as ascertained by the optical tracking and assigned to the field of a rigid body object in the program, could then be extracted with each frame and used as the input to the general algorithm, converting it to a relative target position in the local axes of the robot, facilitating the solution of the servo angles. Of course, the multi-threaded technique for ascertaining the new solution while simultaneously iterating the servo angles towards their current target, as covered in Section 4.3.3, was also employed so that the wand could be used as a live target for the arm to follow.

Once in place, the wand was held at a point deliberately outside the range of possible solutions from the robot, meaning that it had to use the closest calibration point to gesture in that direction. This point was intended to be repeatable; even if there are small differences in the position of the wand, as long as it is consistently outside the range of recorded values in the same general direction each time then the robot will gesture at the same point. In this case, the starting position of the wand was up and to the right (if looking at the robot head-on down the axis of the arm). The wand could then be moved vertically down to another point outside the range of solutions (to the right and down) causing the program to find another set of solutions during its downward motion. As long as the wand is always kept outside the possible solutions in the X direction, and in front of the device, it should follow a vertical path downwards to the new max range gesturing position in the Y direction, while remaining at its maximum X position the whole time. Again, this makes the motion more repeatable, in that as long as the wand follows roughly the same path each time, the arm should follow the same exact path during the 'following' process. This experimental set up as well as the motions of the target and the arm is illustrated in Figure 5.2.

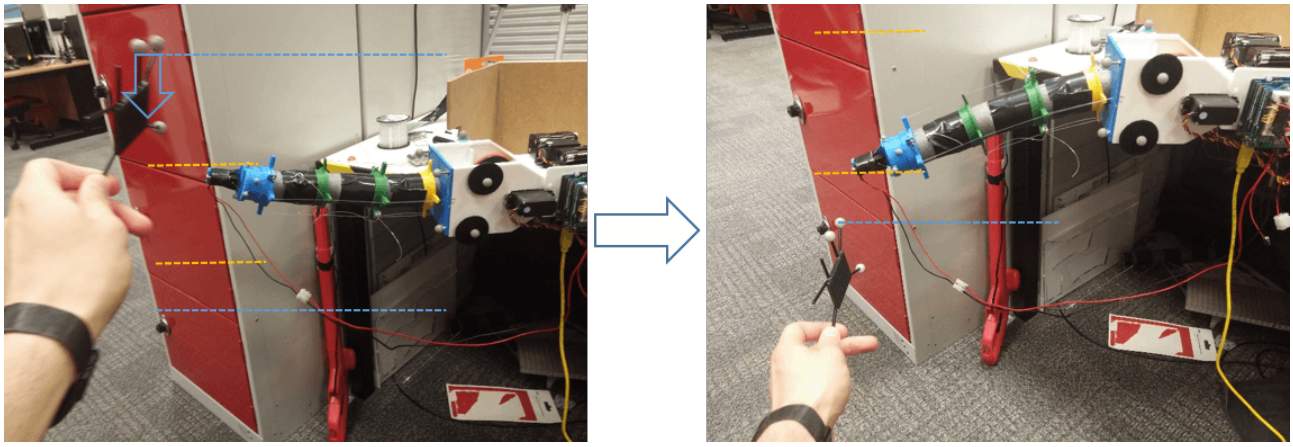


Figure 5.2: Tip of robot follows motion of wand, gesturing at the extreme positions

Once this was set up and the arm could follow the motion of the target wand, gesturing when it was out of range, the subsequent step was to record the response of the arm. This procedure necessitates the logging of the positions of the target and tip at a range of time steps. Therefore, when commencing a response plot by clicking the appropriate button on the GUI, a timer would start and the robot would begin following the target wand. At each interval of around 50ms, the positions of the wand and tip were logged. Finally, once the response plot was stopped, the recorded data was written to a csv file for post-experimental analysis.

This process was carried out for two types of response, where the Y positions of the target and tip were the values to be plotted as they were the key variables for the input and response motion. Firstly, a step response was performed, where the target was held still in the start position then moved rapidly down to the next point, pausing until the arm had completed its response, before repeating the motion in reverse. Then, the positions were recorded while the target wand was moved up and down at a roughly constant frequency. This was then repeated at higher frequencies, until anything greater would be difficult to perform by hand while maintaining a constant rate. In total, eight different frequencies were used to generate a basic assessment of the frequency response.

5.2 Effectiveness and Accuracy of Gesturing

In order to assess how effectively the robot could gesture to a target, a small experiment was to be carried out. This had to provide a means of ascertaining whether the device could accurately convey its desired location to the user, when other potential targets were nearby. As such, the user should be able to readily differentiate between the correct target and its neighbours, which would be an imperative aspect of alleviating mental workload; confusion as to the gestured target point could prove more mentally taxing. The experiment set up for this procedure was based around moving to physical target bodies, which were selected as lightweight foam blocks. The blocks were 7cm in width and height, with a depth ranging from 3.5 to 7cm. The depth is of less importance as they were to be arranged in such a manner that the robot gestures to the centre of the front face, giving a target surface area of $49cm^2$, which is thought to be a typical figure for manoeuvring to physical objects in applications that don't operate in a high precision environment, such as identifying the correct building blocks in a construction scenario.

The blocks were to be arranged in a four by four stack, giving a total of 16 blocks in a wall that was 28cm by 28cm. Then, using the technique mentioned in Section 4.4, the 3D target points that corresponded to the tip of the robot being at the centre of the front face, were to be obtained and logged for each block. In order to minimise any errors, uncertainties, or potential accuracy limitations of the OptiTrack Duo, the best position to place the camera would correspond closely to the same relative location it was at to the device during the calibration procedure. This took place with the robot angled at about 45°to the front of the Duo, so that it maintained a clear view of the markers on the tip and the base for all calibration points. Therefore, so that the response of the robot does not hugely depend on how accurately the Duo can provide the relative orientation of the base to its zero position, it was to be placed at a similar relative orientation in practice. The physical set-up for this experiment is thus illustrated in Figure 5.3.

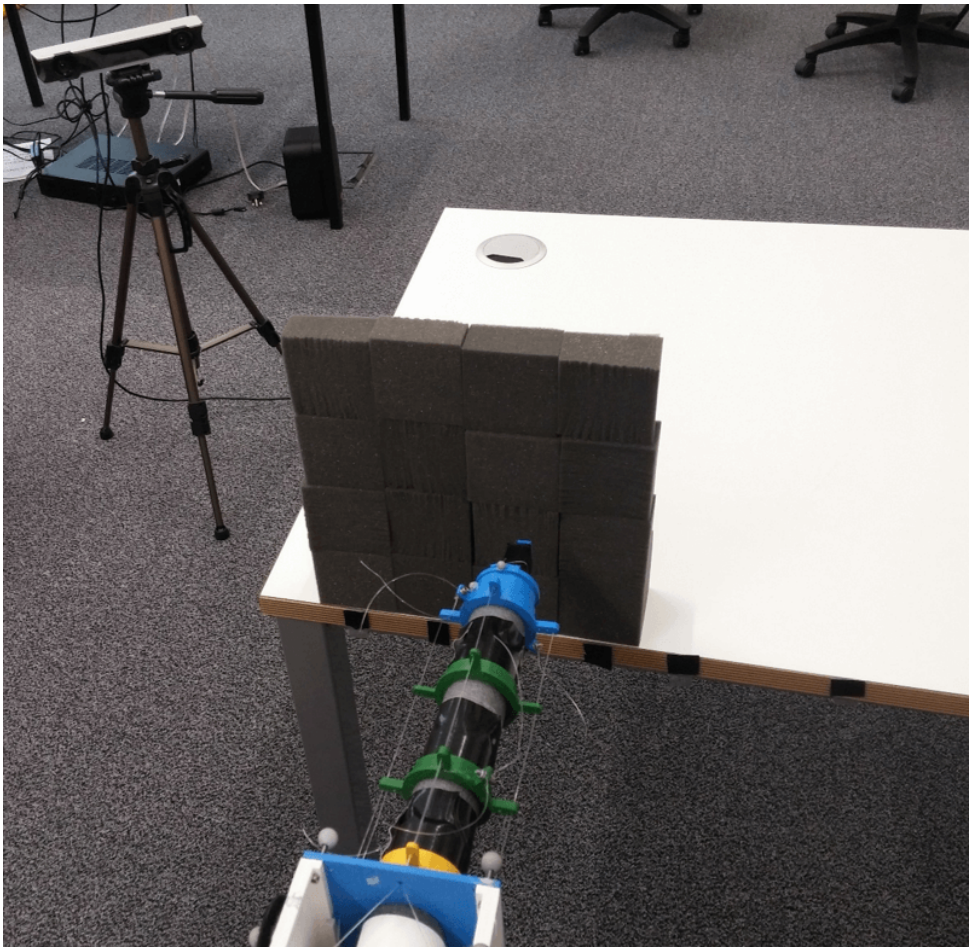


Figure 5.3: Set-up for the gesturing effectiveness experiment

The procedure for generating the target positions entails tracking the tip, positioning it in the desired location, and pressing the trigger to log the current coordinates, which can later be written to disk. However, this can have issues when considering the optimal position of the tracking device. The problem then arises from the Duo's visibility of the tip during this positional logging procedure. With the Duo positioned optimally for accuracy, it cannot maintain line-of-sight visual contact with the tip once it passes behind the wall of blocks. As such, the 3D locations had to be logged without having every block in place. A compromise then, was to assemble only the vertical stack on the furthest side. When the tip was lined up with these blocks, there are no other blocks obscuring the view from the camera, so they could be logged accordingly. Then, by placing markers on the surface to indicate where the central line of each column will be, this end column could be shifted over by the width of one block, representing the next closest column, and the positions logged. The process is repeated until all 16 positions have been logged, before placing the actual blocks in their correct positions. This procedure is illustrated in Figure 5.4.

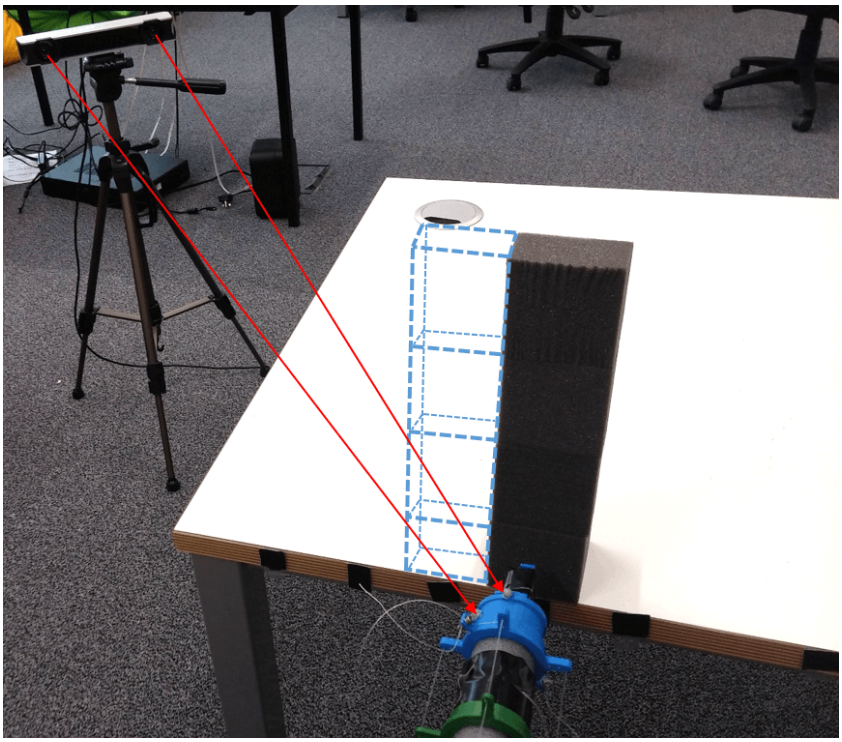


Figure 5.4: Technique used for logging the target positions of the blocks, the dashed lines shows where the column will be moved to next, with the OptiTrack maintaining visual contact

Of course, the issue of tracking the tip would then reoccur once the wall is fully in place. However, to move the tip towards the target position, the algorithm only needs to know this target point relative to the base, which it will then use in conjunction with the calibration data to obtain a servo angle solution. When applied, this servo configuration should then result in the tip moving to the target position, or gesturing in its direction. Therefore, in order to actually respond to target inputs, the program does not actually need to know the position of its tip. As such, once the target points were logged and the experiment was to be carried out, the Motive software was re-configured to only track the base of the robot, which, when positioned so the tip can move towards targets, will be far enough from the wall so as to remain visible at all times to the tracking device.

The logged positions were arranged in such a manner that the bottom right of the wall (as viewed by the user) would correspond to the first position, with the next position one to the left and so on. Upon reaching the end of a row, the subsequent position in the order would be the block at the right hand edge of the next row up. This order was referred to by zero index, with the bottom right and top left blocks assigned to 0 and 15 respectively, so that the positions can be loaded from file into an array of 3D vectors, and the notation for the physical order will match the indices of this array of coordinates.

Then, when performing the experiment, the stored block positions are loaded into the array, and a new array of five integers is formed by selecting random values from 0 to 15, representing a random order of positional indices of blocks. A new array of five 3D absolute target positions is then created using this integer array this, by accessing the stored positions at the corresponding indices within the stored positional array loaded to memory. An example of this random sequencing is illustrated in Figure 5.5. The operator has no indication as to what this new sequence of five targets is, other than the arm gesturing to the next target. The user could then move the arm so that the tip reaches its desired point, and state which block index position they think is the current target. Clicking the trigger then serves to move on to the next target position in the random order, and the process is repeated until all five targets have been cycled through. Once the experiment is stopped via the GUI, only then is the random order of position indices revealed, so that the order suspected by the user can be compared with the actual order, revealing how reliably the positions were conveyed by the robot gesturing.

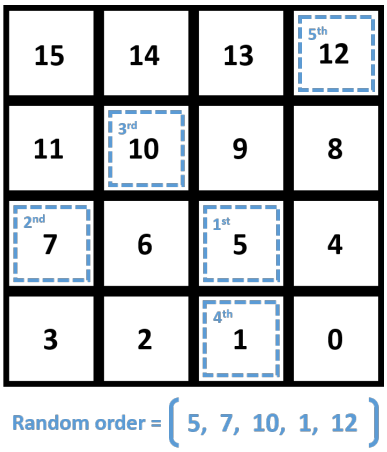


Figure 5.5: Positional order of the wall of the blocks as observed from the front.

5.3 Alleviation of Cognitive Workload in Users

Given the aims of the project, it was imperative to investigate whether the device could actually perform its intended function and further highlight the potential for cooperation between humans and handheld robotics. Specifically, its ability to draw on task knowledge to interact with the user to reduce perceived mental workload must be evaluated. The most appropriate way to carry out this evaluation was a live user study, in which participants use the robot in practice, and specify the extent to which it mentally assisted them. As such, an experiment was designed in which the users would perform two tasks using the robot to place coloured blocks in a specific order into a box. This was accomplished by attaching metal strips to seven of the foam blocks from the gesturing effectiveness experiment, and setting the program so that the electromagnet was activated and the arm held straight when the trigger was held down. Coloured stickers were attached to the front faces to give seven uniquely coloured blocks: red, orange, yellow, green, blue, white and purple, which were then placed in a straight horizontal line at the front of the table. As the physical accuracy was not under investigation here, a conservative spacing between blocks was selected at 7cm. The arrangement of blocks for the user study is illustrated in Figure 5.6

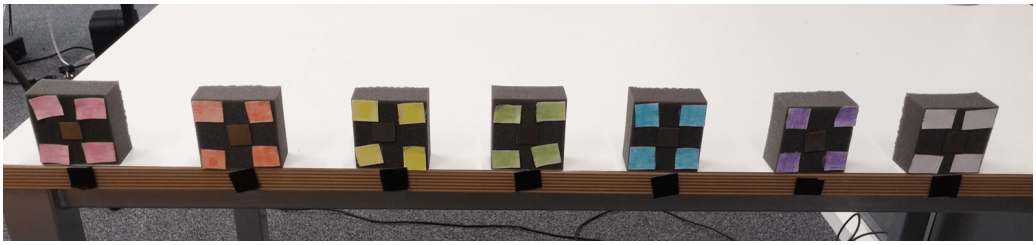


Figure 5.6: Arrangement of the coloured blocks, with metal strips, employed for the user study.

One of the tasks entailed dispensing the blocks in a sequence that required conscious cognitive effort in order to follow correctly, where the arm of the robot remained straight for the duration of the task. The only response to the trigger in this case was to activate the magnet. The instruction for this sequence was to place the blocks in an order that was **alphabetically ascending for the *third* letter of the colour**. That is, A comes before B, which was made clear to the user when informing them of the sequence instruction. Only six of the blocks were to be dispensed, with the final one to be left on the table, serving as a dummy. Once the instructions were acknowledged by the user, a timer was started and the order of the blocks they dispensed was recorded, with the timer being stopped after the sixth colour entered the box. The correct order was thus:

1. ORANGE
2. RED
3. GREEN
4. WHITE
5. YELLOW
6. PURPLE

However, if an incorrect block was selected, it would only count as a single error. That is, after an incorrect block, the score would then be based on the correct ordering of the remaining blocks, whatever they may be. Before the task, it was important to ask the user if they were colour blind, or non-native speakers of English, and note if so. They were then asked to verbally state what colours were laid out. This task served to simulate a mentally taxing set of operations, whereby the user must apply cognitive effort in order to perform successfully. The nature of this task took inspiration from the fact that typical sorting tasks have been shown to result in cerebral activation of the frontal lobe in healthy persons [50], the area of the brain responsible for problem solving and memory. These mental functions are thought to be the most pragmatic to assist with in order to alleviate perceived workload, and potentially improve performance. Note that this task was designed to be generally achievable through a degree of mental effort, not so challenging as to cause difficulties in actually completing the task correctly. This is because the investigation is focused more on how the device can actually cooperate and provide mental assistance to improve performance and/or alleviate effort, rather than facilitating the completion of tasks that would be otherwise impossible to users.

For the assisted task, a pre-defined order of six blocks was created, as before, by moving the tip to the target positions and logging it with the trigger. Although the initial selection of this order was random, it was held constant as to be the same for each participant. Again, visibility issues were encountered when all of the blocks were laid out so markers were put in place and only one block was used to log the positions, moving it between the markers until all the positions were recorded. The base was again the only rigid body tracked, for the reasons mentioned in the gesturing effectiveness experiment. The users would then be unaware of the order, and upon commencing the task the timer would be started and the robot would gesture to the next target. When the trigger was held, the magnet would activate and the arm would straighten up for easier manoeuvring of the blocks. Once the trigger was released, the arm would then begin to gesture to the next target, until after the sixth target when the experiment would stop and the arm would remain straight. Of course, if the trigger was released accidentally, the arm would move on against the user's wish. Therefore a button was added to the GUI to effectively 'undo' the trigger press and move back to the previous target. The top-level process associated with the program flow for the user study is given Appendix H.4. An example of the physical procedure of the assisted task is illustrated in Figure 5.7.

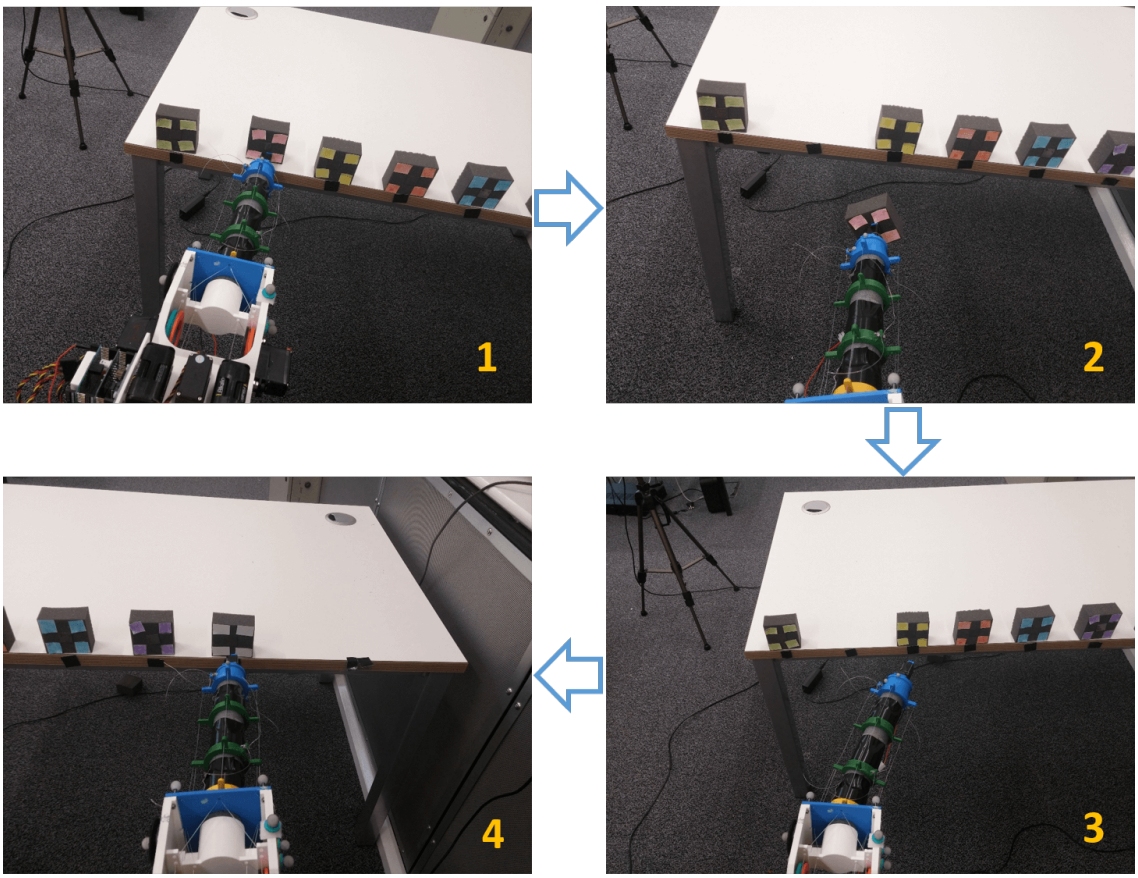


Figure 5.7: Typical example of the assisted procedure. Upper-Left: Robot has gestured the user to the position of the red block, Upper-Right: User activates trigger and dispenses of block, Lower-Right: Robot moves onto next block in sequence, gesturing to the right, Lower-Left: User follows gesturing until reaching white block, before repeating the process until completion

Before the assisted task commenced, a brief demonstration was given to the user, using a different and shorter pre-defined sequence, and they briefly practised using the robot while it gestured to a target. If, in the real task, an error was made and an incorrect block was taken, difficulties may arise, where the robot could later gesture to an empty space. In this event, a spare block would be placed in the empty space for picking up. These spare blocks were uncoloured as, due to the nature of this task, the colours are unimportant as it is the order of positions that is logged. So again, a single erroneous selection would only count as 1 mistake, providing the following order of selections were correct. The assisted task itself serves to represent the robot drawing upon task knowledge to cooperate effectively.

In this case, the task knowledge is just a pre-defined sequence of positions. However, this serves as a generic proxy for more specific tasks, in which, for example, the details of the task could be entered into a computer or captured via camera, to ascertain the necessary sequence of operations. Of course, the program could be set up here to receive user-entered colours and corresponding positions to determine a correct order, though this would be trivial and provide no real benefit to the investigation. The real focus is the robot gesturing a set of positions which are unknown to the user, whatever they may be, to reduce the cognitive effort of the user finding them out for themselves.

Upon completion of both tasks, a modified TLX form was then given to the participants to fill out, which is given in Appendix I. As mentioned in Section 2.4.4, this covered the key perceptions of task load; mental demand, temporal demand, perception of performance, total effort, frustration and physical demand. The latter is of little importance in such an investigation, with the two tasks expected to be identical in this regard, but was included for completeness nonetheless. This was filled out for both the mentally assisted task (gesturing) and the colour sequence (non-assisted) task. It was hypothesised that the completion time, relative performance and perceived workload would all be improved upon when comparing gesturing assistance with the mentally demanding sequencing task.

6 Results and Discussion

6.1 Response Performance

6.1.1 Frequency Response

After analysing the positional data for the results, the actual amplitude of the motions of the target and the tip of the arm could be ascertained. It was found that for the frequency response experiment, the amplitude of the target motion was mostly constant, as intended, at around 175mm (350mm peak to trough). As mentioned earlier, this amplitude was larger than the capped vertical range of motion of the arm, so that the arm moved from on extreme gesturing position to the other. The mid-point of the motion of both bodies was roughly at the same point in the Y direction, giving an equally distant target position on each side of the centre (Y = 0) of the robot. The maximum amplitude of the arm in terms of half the distance between the maximum and minimum Y positions from the calibration data (maximum cable tension) was recorded at 75mm. Hence an extra clearance distance of around 100mm would be present on each side of the arm from the target wand when stationary at the extreme values.

The target wand was found to oscillate at a range of eight frequencies from 0.4 to 2Hz. This was calculated by measuring the average time between peaks of the target motion, giving the wavelength, and using the standard equation:

$$frequency(Hz) = \frac{1}{wavelength(seconds)} \tag{6}$$

At each recording, a portion of the physical response graph would be identified where the frequency was mostly constant, as shown in the example in Figure 6.1, from which the amplitudes of the tip could then be calculated and averaged (the amplitude of the target was roughly constant). The magnitude of the tip amplitude in relation to the target amplitude was then converted into decibels using the average values at each frequency with the following standard equation, and the resulting bode plot for the amplitude response is shown in Figure 6.2.

$$magnitude(dB) = 20 \log \left(\frac{tipAmp}{targetAmp} \right) \tag{7}$$

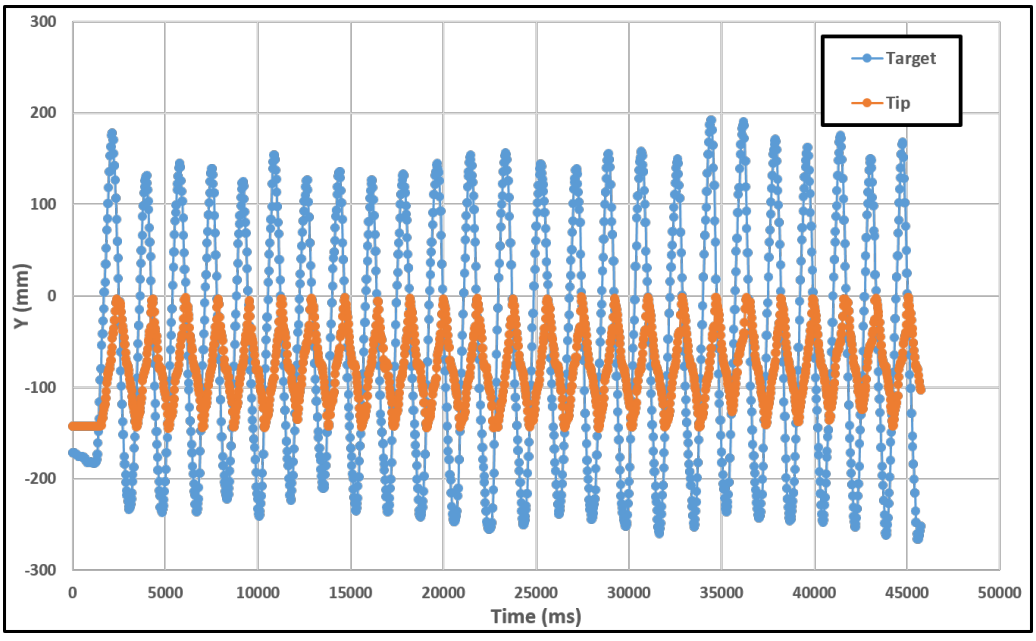


Figure 6.1: Example of the frequency response of the tip to the input of the target position (in Y direction).

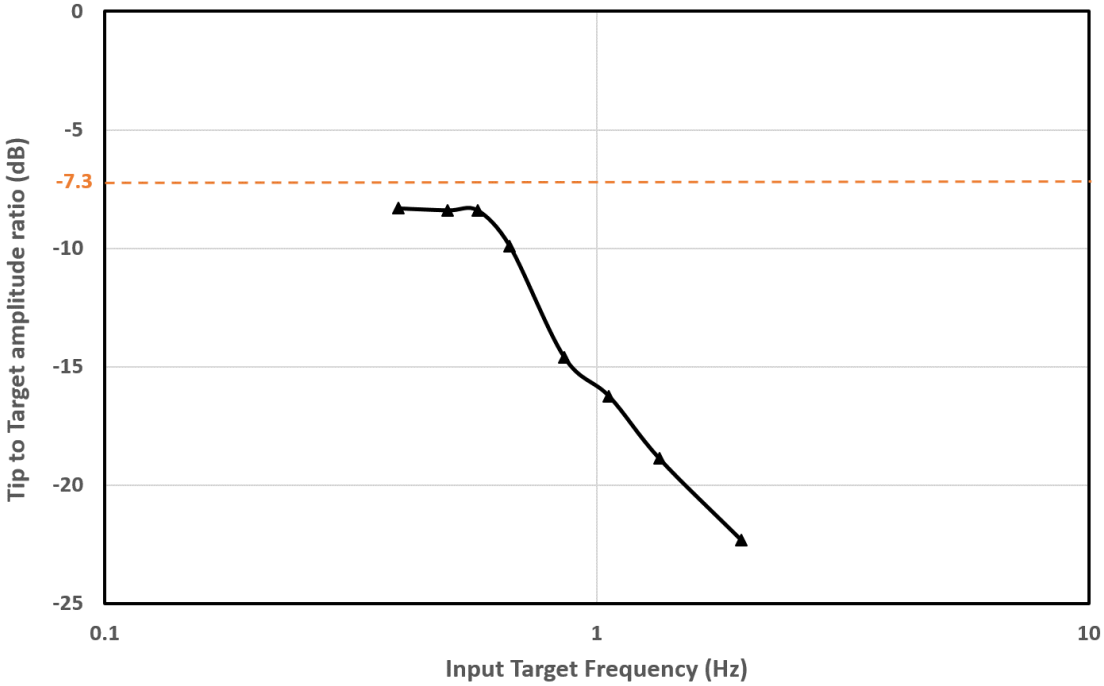


Figure 6.2: Amplitude response (in dB) of the tip to varying target frequency.

The amplitude response shown by the graph appears to be particularly small compared to the input amplitude. Typically, it would be expected that the amplitude (in dB) would be zero, or close to, with small frequencies, which is not the case here. However, it is important to note that due to the nature of the experiment, it is impossible for the amplitude of the output compared to the input to be 0dB. This is due to the fact that the physical amplitude that the target wand was oscillated at was greater than the maximum amplitude of the arm, at least with the imposed limitations. Therefore, the dashed line has been plotted on the response graph to indicate the theoretical maximum amplitude of the tip compared to the wand. The reason for this was to be certain that the arm is covering its full motion, by exceeding its range above and below so that it had to gesture to its extreme positions.

This maximum amplitude would be around -7.3dB, where the peak to trough distance of the tip is the same as the difference between the max and minimum recorded Y values from the calibration data. As seen in the response graph, this maximum is never reached, even when the frequency of the target oscillation is reduced to a point where the response appears to plateau. There are a number of possible reasons why this is the case. The first is that the frequency has not been reduced enough to observe maximum amplitude response. However, this is not the case as, at the lower frequencies, the arm motion was observed to be complete, coming to a full stop at the peak and trough before the target moved past it again. Another reason is that the maximum and minimum Y values from the calibration corresponded to when the tip was at different local X coordinates e.g. pointing to the left, not the right, as in this test. Again, this is unlikely due to the local and global Y axes being the same during the calibration procedure (base was clamped horizontally). Also, the issued servo angles would have been symmetrical about the arm axis, so the servos responsible for vertical motion would have been at the same values at some point for gesturing towards the left and right.

The most likely cause of this maximum amplitude not being met is due to a gradual loss of tension in the cables, highlighting one of the main weakness of cable-driven systems. The actual amplitude at the lower frequencies for the plateau was -8.3dB, being about 1dB shy of the maximum possible, which corresponded to an amplitude reduction of about 7mm. This value can thus contribute to inaccuracies in positioning, particularly at the extreme positioning. As this figure is not dissimilar from the error associated with the kernel regression testing, it is essential that the cables must be checked for tension before every user study test. This result was also the inspiration behind the development of the program feature that applied an offset to the arm so that it was straight when the servos were zeroed.

The actual value of the amplitude appears to drop off at around 0.7Hz reaching a value of -16dB (9dB below the max potential here) at around 1Hz. For this device and its intended applications, of mental assistance rather than physical precision, it is not expected to be used with a target that fluctuates between the maximum ranges either side at a rate above 1Hz. Therefore, as long as the robot maintains an easily perceptible amplitude at this frequency, which it does (around 56mm) then it is not a worry for the user study. The clear act of the arm moving in one direction is in itself an indication of the direction it will gesture to.

In this regard, the amplitude response is sufficient, at least for the intention of carrying out a user study. What is of more importance however, is the phase response, as an especially high lag for the arm will hinder the responsiveness of the device, and more importantly, contribute to user frustration. The phase shift of the tip in relation to the target could also be calculated for each constant frequency portion of a recording. This was done by averaging the time between the peaks of the target and tip, to obtain a time lag in seconds, and using the following equation:

$$Phase\ shift\ (degrees) = -1 \times lag(seconds) \times Frequency(Hz) \times 360(degrees) \tag{8}$$

This phase shift between the target and the tip of the arm was then plotted for the frequencies, as illustrated in Figure 6.3.

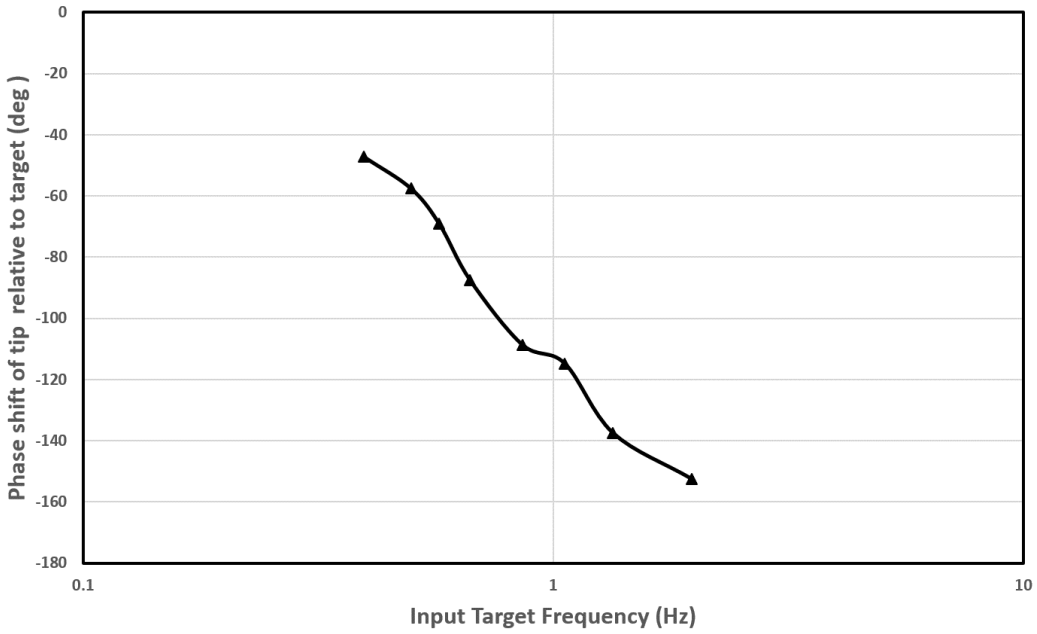


Figure 6.3: Phase shift response (in degrees) of the target motion to tip motion for varying target frequency.

As shown in the phase plot, there is a considerable lag for the tip to respond to the target when the frequency of the target oscillation exceeds 1Hz, with the value approaching 160 degrees around 2Hz. The naive view here would be to categorise the device with low performance based on these figures. However, it is important to consider the actual context that the device is intended to be used in. The device was designed with the intention of drawing upon task knowledge to alleviate the mental effort of its operator. It does this through the act of physically gesturing to the desired location. Therefore, in such a scenario, where the user requires the robot to gesture to the next target, to reduce mental workload, it is not as critical to have a rapid response when this target changes slightly. Situations where this physical target change may occur will typically be events where the user carries the device towards the target direction and slightly overshoots the target so the device must now gesture in the opposite direction. Such physical acts are unlikely to occur at frequencies higher than 1Hz, and will therefore not pose serious problems to usability.

The tasks such a robot would be expected to assist with, and what it was designed to prove, would not require complex mechanical motions in a rapidly changing environment, instead benefiting more from taking away the mental workload which would otherwise prove taxing. If the physical performance was imperative, the frequency response may in such case prove unsatisfactory, hindering the performance and benefits offered by the robots as a whole. For this project, it was decided to strike a balance between responsiveness and maintainability by setting the aforementioned motor delay (the delay after each individual servo is incremented by one degree before incrementing the next one) at 4 to 6ms. This was the highest delay found to still move the robot at appropriate subjectively acceptable speeds. This value was explored, being taken down to as low as 1ms where much greater speeds were observed, although reducing it beyond the selected amount was found to induce new problems.

At such small delays ($< 4ms$), the rapid motion of the servos was found to vibrate the servo horn, at the centre of the pulley, free from the servo shaft, requiring instant repair. In the worst-case scenario, the servo horn was found to shear loose from the glue connecting it to the pulley, which resulted in a lengthy downtime. These two issues came about from two flawed factors of the design; not screwing the horns to the rear servos due to accessibility issues, as well as using relatively weak glue for the pulleys on account of the rarity and potential need for re-use of the specific circular servo horns.

With more emphasis on the physical performance, changes could be applied to the design of the robot to allow access to screw the pulleys to the rear servos, as well as the delivery of replacement servo horns so that high-strength glue may be used on the pulley, addressing both of these problems. However, these relatively simple to implement changes would have added more time onto the design phase of the project, which was already strictly capped by the project time-frame. Although the robot could potentially endure the duration of this experiment at the minimal delay, the response was not investigated at such speeds as a fair representation could only be given if tested at the same speeds that will be experienced when actually using the robot in real task scenarios. An example scenario that is considered commonplace in such mentally assisted tasks is illustrated in Figure 6.4.

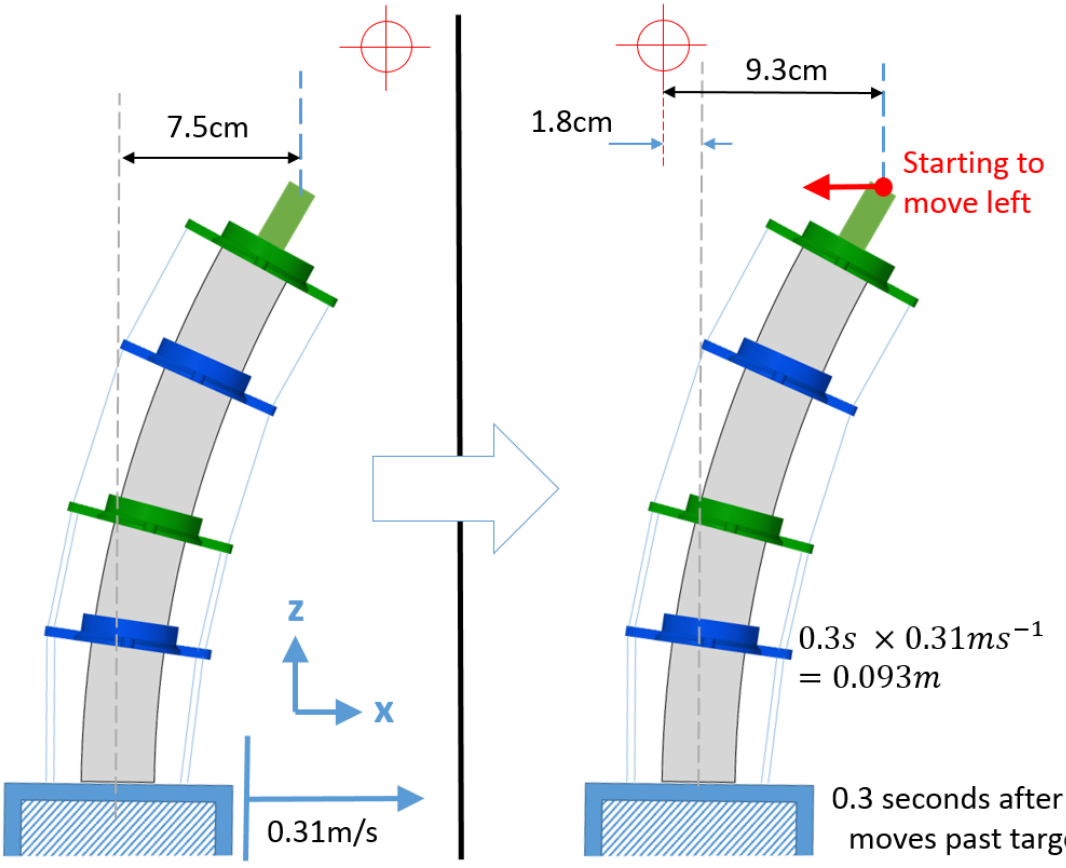


Figure 6.4: Example of typical conditions expected the robot is expected to be subjected to when mentally assisting with tasks, simplified to X translation only, showing potential base overshoot.

It is assumed that in typical applications for this robot, a changing target would not move at a rate from the extreme tip positions at a rate above 1Hz. This is comparable to saying that in half a second, the target position would move across the range of tip motion, which in such a case would be at a speed of 0.31m/s. The phase lag at 1Hz, as obtained from the raw data or Equation 8 and the graph, would be 0.3s (or 114°). In an environment where mental assistance is the primary concern and gestures are responded to, this target change relative to the base could come from moving the device past the target at a fixed orientation, rotating it so the target moves relative to the orientation of the base, or a combination of both.

For simplicity the diagram assumes a fixed orientation translation of the robot towards a gestured point out of range to the right. Eventually a point will be reached where the tip and the target are equal in terms of the X dimension. This in itself can indicate to the user that robot has just passed the target position in that direction, allowing them to intuitively ascertain the general target area, which may be sufficient to locate the target if there are few possibilities in that region, or they may have to wait a little longer for more precise gesturing. Either way, they should logically stop moving. In this case, the base would continue to move after the tip passes the target for 9.3cm, which is about 1.8cm more than the tip would be from the base when gesturing (7.5cm), so the base would roughly only overshoot by a maximum of 1.8cm from the point that is in line with the target, when the tip starts to move and the user halts. Considering this information as a typical case, where the robot would rarely move faster than this and would also usually be oriented towards the target direction, reducing the overshoot further, the robot appears suitably responsive. Hence this frequency response is deemed appropriate for a robot with the purpose of assisting with mainly mentally challenging tasks in a physical environment, thus suitable for the user study in this sense.

6.1.2 Step Response

Using the same set-up as for the frequency response test, a step response experiment was also carried out. The target wand was moved as fast as possible from one extreme position to the other, and the arm was allowed to fully move to the new position and settle before the subsequent step input. The positions of both bodies along with the current timer value were then logged as before. The resulting step response in terms of absolute position in the Y direction against time is illustrated in Figure 6.5.

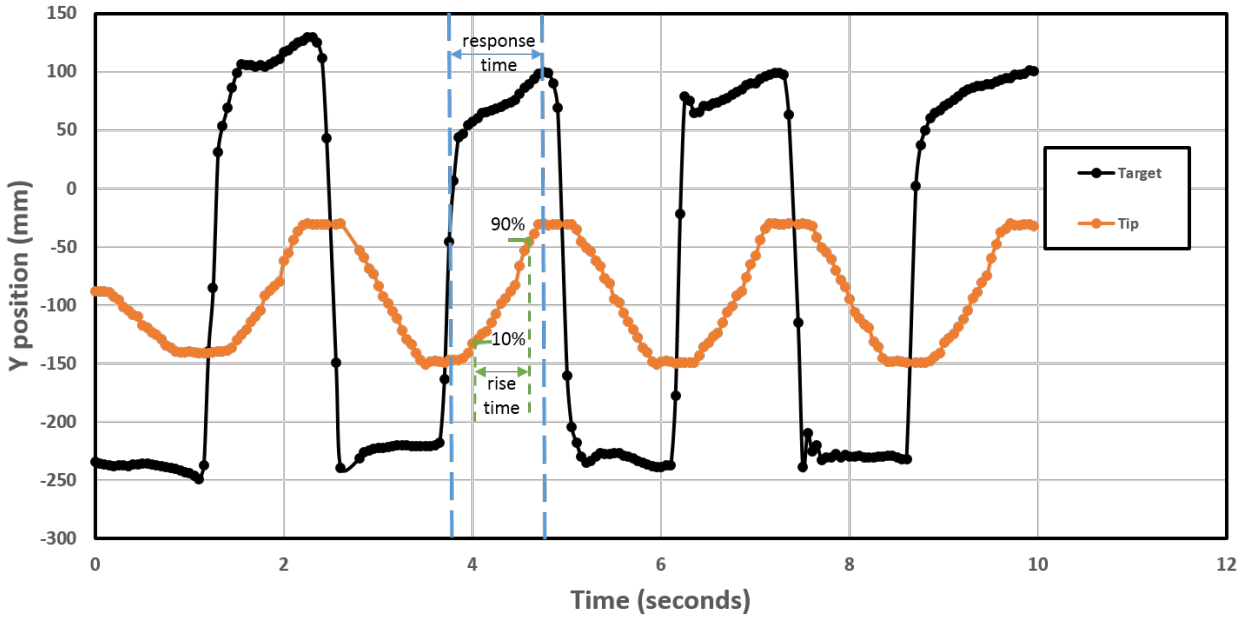


Figure 6.5: Physical response of the tip in the Y direction to a step input from the new target position.

Rather than receiving an input and immediately jumping to a servo configuration, the algorithm of setting the angles relies on checking the most recent solution, comparing this with the current values, and iterating the servos closer to the solution one degree at a time with a slight delay after each step. This process was covered in more detail in Section 4.3.3. As such, with a stationary robot, the servos themselves cannot technically overshoot. Once they reach the target angles (in degrees), the program will not iterate them any further. Of course, the tip itself could overshoot past its target based on the flexibility of the arm and the momentum of the end effector when the servos stop moving. However, when the motor delay was experimented with, the range of 4 to 6ms was found to provide smooth motion without any tip overshoot upon stopping.

Given this way in which the robot responds to target positions, eliminating servo overshoot, the most comparable level of damping that the robot is subjected to, in terms of typical control system terminology, would be 'over damped'. This implies that the servos could move at a greater speed than they do, while still reaching their target without any overshoot of themselves or the tip, which is a fair assumption. As such, it is common practice to quantify the rise time of the step response as the time taken for the response to move from 10% to 90% of its target motion. This is labelled in the step response graph. As such, this time was found for each step input that was recorded, and averaged to give a rise time value of 0.62 seconds. The total response time, that is, the time taken from the step input (one extreme to the other) until the arm moves to its new steady state position, was found to average at 1 second exactly.

If one considers the remit of such a device, with the user finishing a current operation, resulting in a new target position, the maximum time they will have to wait before the robot moves to the new gesturing position is only a second. Given the user could also intuitively see which direction it's 'starting' to gesture in, the robot can therefore be said to appear suitably responsive. Of course, a more reliable evaluation of the cooperation between the operator and device can only be given by results the robot being used in practice by real users.

6.2 Effectiveness and Accuracy of Gesturing

For the gesturing effectiveness experiment, the test was repeated with one user for a total of 14 times. The stated positions, for the randomly generated list of five indices, were recorded along with the actual values printed by the program at the end of the test. Therefore, a total of 70 random target positions were issued, with the results listed and illustrated in Table 3 and Figure 6.6 respectively.

Table 3: List of stated positions and corresponding real positions, showing percentage success.

Stated					**Real**					**Errors**
10	5	14	13	5	10	1	14	13	5	1
0	9	3	7	15	0	9	3	11	15	1
7	13	10	9	15	3	13	10	9	15	1
5	0	11	2	4	5	0	11	2	4	0
12	7	8	5	14	12	7	8	5	14	0
10	13	7	5	3	10	13	7	5	3	0
2	9	0	14	10	2	9	0	14	10	0
3	4	8	15	13	3	4	8	15	13	0
5	0	13	15	9	5	0	13	15	9	0
3	12	11	4	7	3	12	11	4	7	0
5	15	0	3	10	1	15	0	3	10	1
4	10	12	9	0	4	10	12	13	0	1
11	14	10	9	7	11	14	10	9	7	0
Total Errors			**5**							
Percentage Success			**92.9%**							

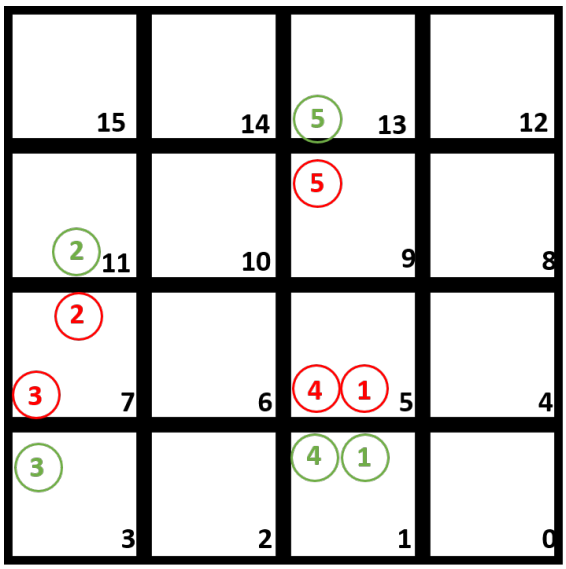


Figure 6.6: Recorded errors for the gesturing effectiveness experiment, shown by the matching numbers with green for the correct position and red for the stated position.

As shown in the error visualization, there were five incorrect statements out of the total 70, implying a total gesturing accuracy for the experiment of 92.9%. Although this is by no means indicative of low performance, the accuracy would ideally be 100% to prove that the robot can gesture effectively to the user its desired position, as a means of alleviating cognitive workload. However, there are some important factors to note when considering this figure. In this set-up, there was zero spacing between the blocks, which is the worst-case scenario for gesturing. In practice, the spacing cannot be reduced any further, and can only cause more gesturing difficulties if the block sizes themselves are made smaller. As mentioned, the physical precision for such a device is not under investigation here, and the blocks sizes are thought to be typical of non-precise applications. If greater accuracy was required, it could have been achieved through a more lengthy calibration and testing procedure, logging more physical points. Cable re-tensioning should also be performed during a lengthy calibration phase and before any meaningful experiment in order to maximise accuracy.

Even still, with these factors regarding the accuracy taken into account, the graphic in Figure 6.6 shows that all of the errors came about as an incorrect choice of the neighbouring vertical block. Therefore the horizontal location was correct in all cases. There are two potential reasons for this observation. The first is due to the viewing angle of the user to the device. If the device is held in the intended and natural manner with the handle of the strimmer base, then the operator would be looking vertically down on the axis of the arm from above, that is, down the Y axis according to the local orientation of the device. It is therefore much less obvious from this angle when the arm is gesturing vertically, than when it is horizontally. This could result in human uncertainty as to whether the tip is pointing straight at a block, or to one of its vertical neighbours.

The other reason the vertical gesturing could become less clear is down to the arm gradually 'sagging' when held in a horizontal position. At times, the constant force of gravity was found to cause a loss of cable tension on the upper side of the arm (responsible for resisting gravity). However, this effect was mitigated as much as possible by re-tensioning when sagging was observed, as well as through the introduction of an offset applied to the servos responsible for vertical motion. The latter was applied at the start of the experiment if a small offset could help straighten the arm out when the servos were in neutral positions, to reduce the frequency of the lengthy cable re-tensioning procedure.

Thus, given the fact that the user study was to be performed with greater block spacing and only on a horizontal plane (where there were zero errors), the accuracy was appropriate for the purpose of the robot. If a vertical element was to be applied to future user study tests involving gesturing, it is advised that there should be adequate spacing between blocks to make the gesturing clearer.

6.3 Alleviation of Cognitive Workload in Users

The user study to determine how effectively the device could alleviate cognitive effort was carried out with 10 participants. The time taken to complete both the gesturing assisted task and the instructed sequence task was recorded. The recorded times are illustrated in the form of a box and whisker plot in Figure 6.7.

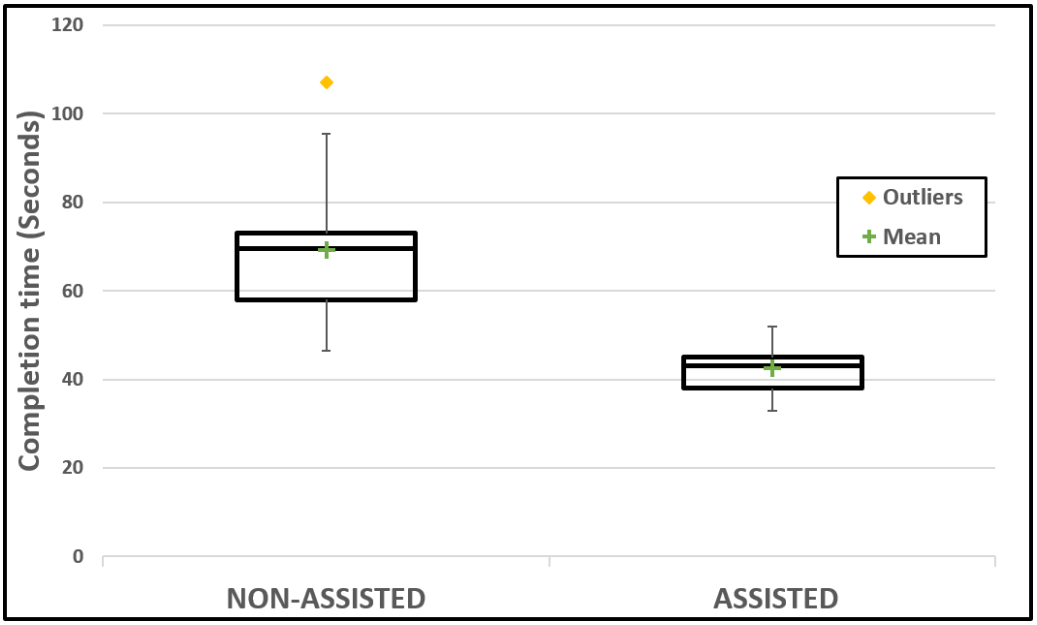


Figure 6.7: Box plots for the completion time of the non-assisted and assisted tasks (lower is better).

For the non-assisted sequencing task, the upper inner fence was defined by the upper hinge (75^{th}) percentile) plus the inter-quartile range multiplied by 1.5. Only one value was above this fence, and was thus classified as a suspected anomaly. This value corresponds to the only participant who identified as colour blind, solidifying its status as an anomaly. It can be seen from the plot that the height of the box is much shorter for the assisted task, indicating a more consistent completion time among users. The actual standard deviations of the assisted and non-assisted tasks were **5.93** and **15.73** respectively. This was to be expected as the assistance of the robot is designed to take away the mental aspect, and with it, the potential for differences in ability. Whereas this is not the case for the mental demand of the sequencing task.

Once a demonstration and brief practice was carried out with the robot, the operators should all have very similar competence in responding to the motion of the arm, which appears to be validated by small height of the box. The only differences observed in the way the operators used the robot was how some participants tended to respond to a gesture by moving the device at constant orientation, and others tended to rotate the device in the gestured direction and only then move it towards the target. However, this had no discernible differences in the completion time.

Conversely, greater differences among participants were expected in terms of attentiveness and mental response time and, as such, more variation in the completion time of the mental sequencing task was predicted. This was proved to be the case by the comparatively taller box plot. The greater range between times may have come about due to cognitive differences, or even because of varying levels of confidence. That is, some users may pause and wait longer until they are absolutely sure of the next block in the sequence, whereas others may take action as soon as they have an idea, which, in turn, may hinder task fulfilment. The completion time box plot shows that the box for the assisted task is considerably lower than for the non-assisted, with no overlap between the boxes, and only a slight overlap of the lower whisker of the non-assisted task with the box of the assisted task. Performing a 't-test' on these results yields a value for the completion times of **5.02**. The degrees of freedom in this case (number of samples in each set minus 2) is 18, and the risk level is assumed to be 0.05 (as per rule of thumb). The t-test result can then be looked up in a standard table of significance [51], where

it is found to be considerably greater than the required value of **2.10**. This implies a statistically significant difference between the two modes and the completion time, and that a strong improvement can be achieved through using the robot for assisting in a task that could otherwise be classified as mentally demanding.

The actual scores, ranked out of number of correctly ordered blocks for whatever remained, are illustrated in Figure 6.8 for each participant, with the appropriate box plot for these results illustrated in Figure 6.9.

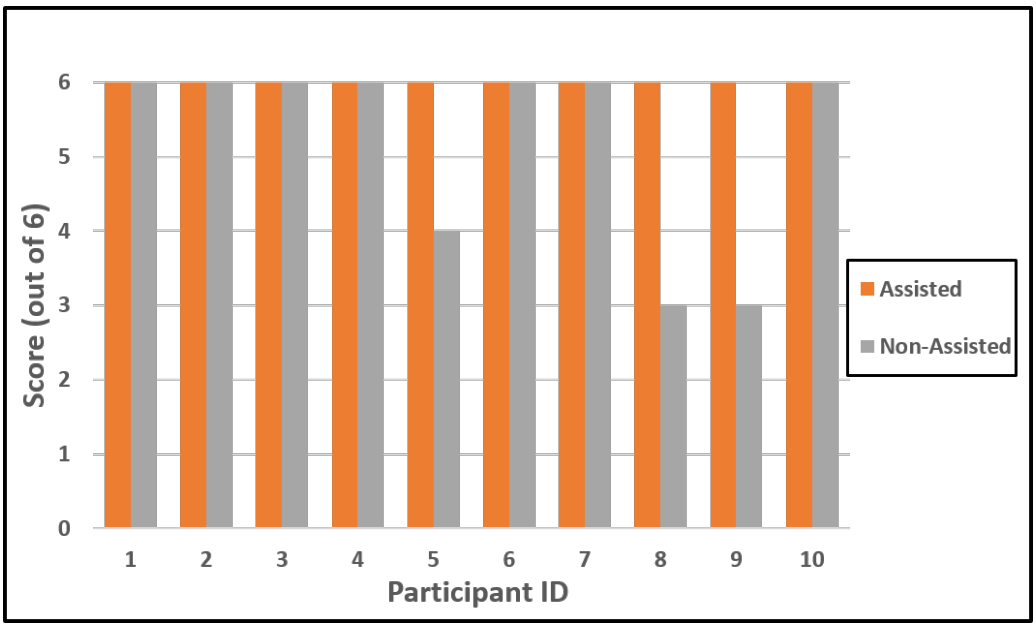


Figure 6.8: Scores for correctly ordered blocks for each participant, note that the scores were based on whatever blocks remained after each selection, so an incorrect choice early on would only reduce the score by one.

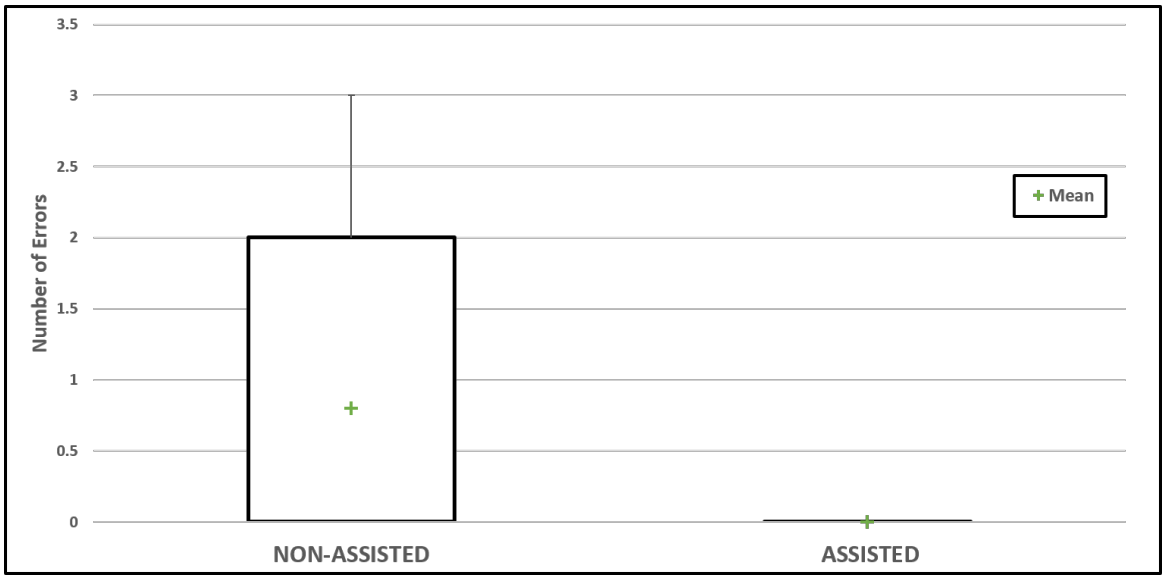


Figure 6.9: Box and whisker plot for the number of errors observed in the non-assisted sequencing task, note that as there were no errors for the assisted task, hence the box plot is a horizontal line.

It can be seen that all participants achieved the maximum score for the mentally assisted task, whereas three of them failed to achieve the maximum score for the non-assisted task, with corresponding scores of 4, 3 and 3. On the surface, this suggests that the actual successful completion of the task could benefit through operating the device. This could prove highly beneficial in practice, where costly mistakes, and the associated rectification time, could be vastly reduced. The t-test result for

comparing the number of errors in the two datasets was found to be **1.92**, and the required value for statistical significance, found the same way as before, was again **2.10**. In this case, the difference is considered not quite statistically different. This is not surprising, as the reduction of errors was not the sole focus of the study, and the non-assisted task was designed not to be so hard as to cause difficulty in actually achieving the objective, but to assess whether the robot can effectively assist in mental tasks. Of course, a much larger dataset would be required to determine any potential extent of error reduction, with increased mental task variety and duration being focused on in particular. Therefore further investigation may be warranted into specific tasks where the robot can actually prevent human mistakes through cognitive assistance, rather than just improve performance through reducing mental workload and completion time.

From a performance perspective, the results were as hypothesised, highlighting the potential usefulness of handheld robotics. The spread of completion times were much smaller when relying on mental assistance. This demonstrates the theory that handheld robots can mitigate for the variations in specific task training and skill level for operators, implying that they possess the potential for unskilled users to carry out operations previously reserved for those with considerable experience. Moreover, there appeared to be an overall improvement in completion time for the task through receiving mental assistance, which lends itself to improved performance, particularly in time-critical applications.

The other key parameter to investigate when evaluating how effectively the robot can cooperate with the user to alleviate mental workload, is the perceived effort of the participants. The performance gains are of little benefit if the user finds the robot difficult to operate, causing frustration and little incentive for re-use. This was obtained via the TLX form that they filled out upon completion of both the tasks, which asked them to rank each task on a scale of discrete points for the factors thought of as most indicative of subjective effort, as covered in [41]. The responses from the participants were then averaged for each category within each task, and are illustrated in Figure 6.10.

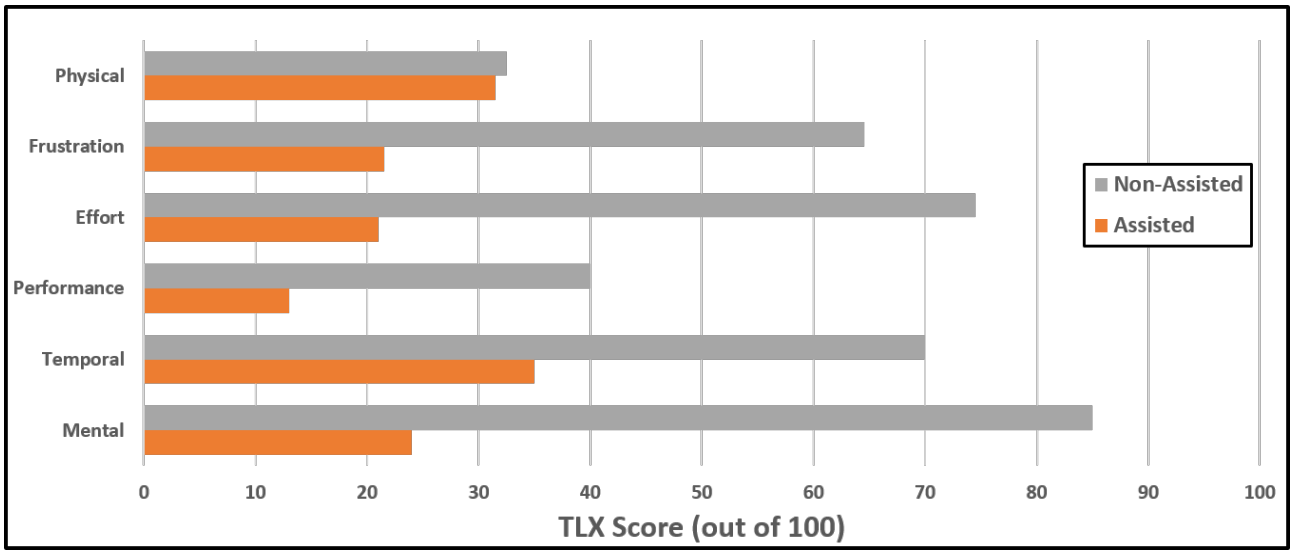


Figure 6.10: Average TLX scores for each category of perceived effort for both the assisted and non-assisted task, smaller is better.

As seen from the plot, the TLX scores are smaller for each category when the robot provides gesturing assistance, suggesting the all-round workload was subjectively smaller for the participants. The combined TLX scores were found from summing the individual scores (each out of 100) to give a total score out of 600. The t-test value was then found calculated for the ten samples as **6.71**, with the total t-test values for the evaluation listed in Table 4. This value, when compared with the same required value as before (**2.10**), proves that the reduction in total perceived workload is extremely statistically significant for receiving mental assistance from the device. In terms of physically operating the robot, the scores are virtually identical, as was expected due to the fact that the robot has to be used to move the blocks regardless, and the addition of mental assistance does not provide any reduction in total physical motion requirements. The only logical reason, in this scenario, why one would rank

the workload as smaller when mental assistance is provided, would be because the total time holding the robot is typically less due to the reduced completion time. The actual scores for these physical perceptions were still relatively low, with this being the smallest type of effort perception for the non-assisted mode.

Table 4: Resulting t-test values for the differences between the two tasks in each category

Task Difference	**t-test**	**Required**
Completion time	5.02	2.10
Total Errors	1.92	2.10
Combined TLX	6.71	2.10

The performance, where lower is considered better, implies that the users were satisfied with how they completed the task when gesturing was provided, being confident that the order they dispensed the blocks in was correct, even though a dummy block would still be left on the table. This was demonstrated by it having the lowest TLX score for the assisted task. It is still relatively low even for the non-assisted task, at a score of around 40%, which would fall on the left side of the scale closer to high performance than low. Of course, all the participants did achieve the correct order for the gestured task, hence their confidence in their performance. With the non-assisted task however, the users are likely to realize they have made a mistake as the task progresses and they notice a block remains with a lexicographically smaller third letter than of one of the previously dispensed blocks. For this reason, and for the comparatively longer time to complete the task on average, it is not surprising that the participants rated this with a higher score for no assistance.

The highest TLX score for the assisted task occurred for the temporal workload, which was a measure of how rushed the user felt in completing the task. This came out at about 35%. Perhaps the reason for this high score relative to the others for the assisted task was due to the nature of the instructions for the task. It was observed that when the users were practising with the gesturing of the device, they tended to spend a long time moving from one side of a correct block to the other, as the device alternated its varying position. This could be down to them wanting to be certain that the block was correct, or even out of intrigue in the device operation. Therefore, they were instructed to act as quickly as possible for both tasks, so as to not skew the results unnecessarily. Hence if the arm gestured to the right and only one block remained in that direction, they would immediately collect it, as opposed to narrowing in on the target. This instruction to operate quickly, without being rash and not having confidence in the target, may have contributed to the relatively higher score. Nevertheless, it is still about half of the corresponding score for the non-assisted task.

The three remaining parameters are all closely linked, particularly the mental effort and the overall effort required to achieve the level of performance. These are virtually synonymous, except that one may say it was mentally demanding to follow the instructions, while the overall effort to achiever their level of performance may have been smaller. For instance, they could struggle to follow the instructions, and after making a mistake may have just rushed towards the end of the task with less regard for performance given they already cannot achieve the maximum score. This could explain why overall, the users stated that it was less effort to obtain the performance they actually did for the non-assisted task, than the mental effort required to carry out the steps correctly. For the assisted task, these two parameters were similar at around 21% and 24% for overall and mental effort respectively. However, arguably the most important metric here is the difference in mental effort score for assisted and non-assisted task, as this was a primary focus for the project. This turned out to exhibit the highest difference amongst the parameters, at around 61%, and the value for the non-assisted task was the highest TLX score overall. The implications of this are: that the mental operation required for the non-assisted task was particularly taxing, and that a great deal of this mental effort is alleviated through use of the device.

The frustration experienced by the user was also relatively small, at around 21%, for when assistance was provided, which was a difference of 43% compared with when not. It is essential to the usefulness of the device that this frustration value, when cooperating with the robot, is low. This is because with high subjective frustration, it will be tiresome and unpleasant for a user to operate the device

over a long period of time. This figure then, is acceptably low. However, it is thought that physical improvements could be made to the design to further reduce both this figure and that of the physical effort perception values. One user remarked that the distribution of the weight of the robot meant that it must be held further down the strimmer base with one hand, resulting in a need to 'lean over' further. With more focus on the ergonomics, complex cable routing could be employed to allow the servos and the two 6 Volt battery packs to be shifted closer to the handle, moving the vast majority of the weight towards the user to mitigate this effect.

7 Conclusions

A project was undertaken concerned with contributing to the relatively small field of handheld robotics, with a particular focus on their ability to alleviate the cognitive workload of users in otherwise mentally demanding tasks. In order to evaluate this potential, a new device was designed and constructed, incorporating novel features over previous models. This process consisted of iteratively designing multiple parts in CAD, and prototyping them through 3D printing, until a suitable configuration was selected. The final design comprised a cable-driven elastic continuum arm, incorporating four servomotors in a housing, which was mounted to the base of a strimmer for handheld use. An electromagnet was implemented for the end effector, and was activated via the trigger of the strimmer, which served as an input signal from the user. A system was then designed in order to connect the robot to an external PC, allowing for controlled actuation of the servos via an Arduino. A program was developed in the background, with the purpose of interfacing with both the device and an external optical tracking system, controlling the robot accordingly.

Once the device was built, and the program at a stage to manipulate tracking data and control the robot, an algorithm was designed so that the robot could perform its required function of manoeuvring the tip towards 3D target positions. This was based around the use of a kernel regression function to interpolate between existing data points to achieve a servo angle solution for the required tip position. As such, a calibration procedure was carried out to obtain the relevant data, and ascertain the optimal bandwidth for the function. Other key stages of the algorithm involved converting the target point to the same orientation of the base for input to the function. If the target was unattainable, the robot was to gesture in its direction the best it could to cooperate its location to the user. The final program then allowed the robot to move towards a target in real-time, constantly updating the target based on data from the tracking system and responding through the actuation of the servos in a separate thread.

In order to evaluate the robots ability perform its intended purpose of cooperating with the user to relieve cognitive workload in mentally demanding tasks, it was decided to carry out a user study. However, before this could be accomplished, it was necessary to ascertain whether the robot could perform suitably in terms of physically moving to targets, otherwise user operation would not be possible. As such, two experiments were designed to assess its responsiveness and its ability to gesture effectively. The first entailed clamping the device and setting it to follow a target body in real-time. This target was then oscillated at different rates vertically in front of the device, and the positional information of the tip and target was logged at a set time interval, allowing the frequency and step response to be plotted. The resulting figures indicated that the robot was suitably responsive and could operate at speeds typically expected for the user study. The gesturing test comprised creating a wall out of the foam blocks to be used for the user study, and measuring how effectively the robot could gesture to a random order of blocks by comparing stated positions from the user with the actual order upon completion. This revealed a total accuracy of 92.9%, with all the errors occurring as a result of uncertainties in the vertical direction. As the blocks were to be placed horizontally for the user study, and at larger spacing, this gesturing was deemed suitably effective.

The user study entailed ten participants undertaking two tasks, using the robot to move coloured foam blocks into a box in a specific order. One of the tasks required that the user order the blocks according to an instructed sequence that required applied mental effort to complete. This was based on the alphabetical order of the third letter of the colour. For this non-assisted task, the arm of the robot remained straight, and the only action of the trigger was to activate the electromagnet. The other task incorporated assistance from the robot, where a pre-defined sequence of the blocks was unknown to the user, and the device gestured to the appropriate targets in the correct order. Upon completion of both tasks, the users filled out a Task Load Index (TLX) form to assess perceived workload. The completion time, number of task errors and the relevant TLX scores (individual and combined) could then be analysed to evaluate the performance improvements offered by the mental assistance of the robot.

The non-assisted task was designed to be achievable through conscious cognitive effort, therefore the key performance parameter was the completion time. This was found to be shorter for the mentally assisted time, and the reduction was proved to be statistically significant through a t-test value of **5.02**, which was greater than the required value for significance in each test of **2.10**. A box and whisker plot for the two tasks in terms of completion time also revealed a much smaller spread of data for the assisted task. This lends weight to the potential of the device to remove disparity between ability amongst users, allowing untrained or unskilled persons to perform sufficiently through use of a handheld robot. The larger spread for the non-assisted task demonstrates general differences in cognitive ability amongst participants. Whilst no errors were observed in the assisted task, a few were distributed amongst three users for the non-assisted task. However, a t-test showed this difference to be not quite statistically significant, falling short of the required value at **1.92** which was not surprising based on the aforementioned design of the task.

The TLX results indicated a smaller perceived workload for the assisted task in each of the six categories: mental, physical, frustration, performance, effort and temporal. The difference in combined TLX scores between the two tasks was shown to be extremely statistically significant for the t-test with a value of **6.71**, implying that subjectively, the effort required for such a task is considerably lower when receiving mental assistance from the device. As expected, there was only small differences between the two for factors that were inherently similar for both tasks, such as physical effort and perceived performance (in terms of errors). However, for the rest of the categories, larger differences were observed between the two tasks. In order for the device to have practical applications, it was imperative that the perceived frustration remained low, otherwise the long-term use would suffer. This appeared to be the case from the results, and was much smaller than for the non-assisted task, implying that the device can be operated to share mental effort and potentially improve performance without introducing new user-experience problems.

The most important metric for this specific study was the perceived mental effort, as this was what was hoped the device would alleviate. This exhibited the biggest difference in between the two tasks, indicating that the robot had been successful in cooperating with the user to lend cognitive assistance. This, combined with the improved completion time, helps prove the theory that the robot could improve all-round performance in mental tasks, and further outlines the potential for cooperative handheld robotics in general. With this, the aims and objectives of the project were accomplished, and further exploration has been contributed to the field, where the findings may be drawn upon in future developments.

8 Future Work and Improvements

There are a number of potential improvements for the work carried out in this project. Limitations in the actual design introduced difficulties in running the device for a long duration without required maintenance. In particular, this consisted of re-tensioning the cables, which generally took around 30-40 minutes. This however, is an inherent drawback to cable-driven designs, as eventual tension loss is avoidable. Although modifications were made to the linkages to reduce the frequency of re-tensioning, the act itself still required unscrewing the bolts, removing the cable from the interweaving holes, pulling it taut with needle-nose pliers before connecting it back up again. If a project of this nature was to be performed, it is strongly advisable to employ a re-tensioning method that can be done quickly, reducing the downtime of the device. Such designs were briefly explored, comprising a lead-screw to essentially 'push' the cable connection further away, requiring much less time in theory. If more serious maintenance was required, such as reconnecting a pulley to one of the rear servos, the downtime was severe. This was a direct result of the lack of accessibility to this inner region of the housing. With simple holes or gaps, a considerable amount of time could have been saved by reaching in to remove or reconnect components.

Although the highly flexible continuum structure of the design demonstrated its potential in terms of range of motion and resistance to accidental collisions or impacts, the material used in this case did have fundamental limitations. Over time, or when bent to extreme positions, a deformation from the starting position was observed, even when all force had been removed. As such, the arm structure itself had to be replaced three times throughout the project. It was found that by limiting the range of motion somewhat, and storing the robot in an upright position, the need for replacement could be substantially reduced. Of course, with more focus on the nature of such a structure, a more premium solution should be employed that has greater resistance to deformation, such as in [30]. Methods were considered to limit this vulnerability, such as internal ribs to provide more rigidity, or rods to help it move back to the starting position. However, these options defeat the purpose of an elastic continuum structure in the first place.

Regarding the user study, the amount of data obtained was deemed appropriate for the scope of this project, which focuses on the potential of handheld robotics. However, if a more thorough investigation was to be carried out into the benefits of mental assistance, much more data would be required. This would entail using a wider range of participants in terms of age and profession. This study was mostly performed by those working in computer science in some form or other. The actual nature of the non-assisted task here was to be achievable through mental effort. If more detail was to be obtained regarding alleviation of cognitive workload, it would be necessary to carry out a wide range of different mental tasks. These could include similar operations to this project, as well as situations where an untrained user is not expected to complete the task without assistance. Also, the combination of such cognitive tasks with more physically demanding aspects would highlight the all-round potential of handheld robotics. Of course, the field in general can still be considered in its infancy, therefore any further investigation into the nature of the field will prove valuable.

9 References

[1] A. Gregg-Smith and W.W. Mayol-Cuevas. The design and evaluation of a cooperative handheld robot. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1968–1975, May 2015.

[2] `http://handheldrobotics.org`.

[3] R. Chipalkatty. *Human-In-The-Loop Control For Cooperative Human-Robot Tasks*. PhD thesis, Georgia Institute of Technology, May 2012.

[4] S.J. Anderson, S.C. Peters, T.E. Pilutti, E.H. Tseng, and K. Iagnemma. Semi-autonomous avoidance of moving hazards for passenger vehicles. In *ASME 2010 Dynamic Systems and Control Conference*, pages 141–148. American Society of Mechanical Engineers, 2010.

[5] S. Haddadin, A. Albu-Schäffer, and G. Hirzinger. Safety evaluation of physical human-robot interaction via crash-testing.

[6] EN953. Safety of machinery - guards - general requirements for the design and construction of fixed and movable guards, 1997.

[7] C. Ott, O. Eiberger, W. Friedl, B. Bauml, U. Hillenbrand, C. Borst, A. Albu-Schaffer, B. Brunner, H. Hirschmuller, S. Kielhofer, R. Konietschke, M. Suppa, T. Wimbock, F. Zacharias, and G. Hirzinger. A humanoid two-arm system for dexterous manipulation. In *2006 6th IEEE-RAS International Conference on Humanoid Robots*, pages 276–283, Dec 2006.

[8] F.W. Heger and S. Singh. Sliding autonomy for complex coordinated multi-robot tasks: Analysis & experiments. 2006.

[9] G. Dorais, R.P. Bonasso, D. Kortenkamp, B. Pell, and D. Schreckenghost. Adjustable autonomy for human-centered autonomous systems.

[10] M. Tambe, P. Scerri, and D.V. Pynadath. Adjustable autonomy for the real world. *Journal of Artificial Intelligence Research*, 17(1):171–228, 2002.

[11] A. Poncela, C. Urdiales, E. J. Perez, and F. Sandoval. A new efficiency-weighted strategy for continuous human/robot cooperation in navigation. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 39(3):486–500, May 2009.

[12] C. Lenz, A. Sotzek, T. Röder, H. Radrich, A. Knoll, M. Huber, and S. Glasauer. Human workflow analysis using 3d occupancy grid hand tracking in a human-robot collaboration scenario. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pages 3375–3380. IEEE, 2011.

[13] M. Huber, A. Knoll, T. Brandt, and S. Glasauer. When to assist?-modelling human behaviour for hybrid assembly systems. In *Robotics (ISR), 2010 41st International Symposium on and 2010 6th German Conference on Robotics (ROBOTIK)*, pages 1–6. VDE, 2010.

[14] P. Lukowicz, J.A. Ward, H. Junker, M. Stäger, G. Tröster, A. Atrash, and T. Starner. Recognizing workshop activity using body worn microphones and accelerometers. In *Pervasive Computing*, pages 18–32. Springer, 2004.

[15] A. Elfes. Using occupancy grids for mobile robot perception and navigation. *Computer*, 22(6):46–57, 1989.

[16] M. Buss and M. Beetz. Cotesys—cognition for technical systems. *KI-Künstliche Intelligenz*, 24(4):323–327, 2010.

[17] J. Werfel, Y. Bar-Yam, D. Rus, and R. Nagpal. Distributed construction by mobile robots with enhanced building blocks. In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pages 2787–2794. IEEE, 2006.

[18] K. Petersen, R. Nagpal, and J. Werfel. Termes: An autonomous robotic system for three-dimensional collective construction. *Proc. Robotics: Science & Systems VII*, 2011.

[19] R.C. Arkin and R.R. Murphy. Autonomous navigation in a manufacturing environment. *IEEE Transactions on Robotics and Automation*, 6(4):445–454, Aug 1990.

[20] C.J. Payne and Guang-Zhong Yang. Hand-held medical robots. *Annals of Biomedical Engineering*, 42(8):1594–1605, 2014.

[21] N. Matsuhira, M. Jinno, T. Miyagawa, T. Sunaoshi, T. Hato, Y. Morikawa, T. Furukawa, S. Ozawa, M. Kitajima, and K. Nakazawa. Development of a functional model for a master–slave combined manipulator for laparoscopic surgery. *Advanced Robotics*, 17(6):523–539, 2003.

[22] C.J. Payne, H.J. Marcus, and G.Z. Yang. A smart haptic hand-held device for neurosurgical microdissection. *Annals of Biomedical Engineering*, 43(9):2185–2195, 2015.

[23] W.T. Latt, U.X. Tan, C.Y. Shee, and W.T. Ang. A compact hand-held active physiological tremor compensation instrument. In *2009 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, pages 711–716, July 2009.

[24] M. Fischer, M. Schraufstetter, C. Richter, F. Irlinger, and T. C. Lueth. Tremor compensation by use of a mechatronic cup holder. In *2010 4th International Conference on Pervasive Computing Technologies for Healthcare*, pages 1–8, March 2010.

[25] F. Echtler, F.Sturm, and K.Kindermann. The intelligent welding gun: Augmented reality for experimental vehicle construction. In *Virtual and Augmented Reality Applications in Manufacturing*, pages 333–360. Springer London, London, 2004.

[26] A. Gregg-Smith and W.W. Mayol-Cuevas. Inverse kinematics and design of a novel 6-dof hand-held robot arm. In *IEEE International Conference on Robotics and Automation (ICRA)*, ICRA Proceedings. IEEE, February 2016.

[27] I.S. Godage, D.T. Branson, E. Guglielmino, G.A. Medrano-Cerda, and D.G Caldwell. Shape function-based kinematics and dynamics for variable length continuum robotic arms. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 452–457. IEEE, 2011.

[28] Z. Zhang and S.H. Yang, G.and Yeo. Inverse kinematics of modular cable-driven snake-like robots with flexible backbones. In *2011 IEEE 5th International Conference on Robotics, Automation and Mechatronics (RAM)*, pages 41–46. IEEE, 2011.

[29] H.S. Yoon and B.J. Yi. A 4-dof flexible continuum robot using a spring backbone. In *2009 International Conference on Mechatronics and Automation*, pages 1249–1254, Aug 2009.

[30] J. Reinecke, B. Deutschmann, and D. Fehrenbach. A structurally flexible humanoid spine based on a tendon-driven elastic continuum. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4714–4721. IEEE, 2016.

[31] V. Anderson and R. Horn. Tensor arm manipulator design. *ASME Transactions*, 67-DE-57(2):1–12.

[32] S. Hirose. Biologically inspired robots, snake-like locomotors and manipulators. 1993.

[33] D.B. Camarillo, C.F. Milne, C.R. Carlson, M.R. Zinn, and J.K. Salisbury. Mechanics modeling of tendon-driven continuum manipulators. *IEEE Transactions on Robotics*, 24(6):1262–1273, 2008.

[34] R.J. Webster and B.A. Jones. Design and kinematic modeling of constant curvature continuum robots: A review. *The International Journal of Robotics Research*, 2010.

[35] P.M. Fitts. The information capacity of the human motor system in controlling the amplitude of movement. *Journal of experimental psychology*, 47(6):381, 1954.

[36] Y. Cha and R. Myung. Extended fitts' law for 3d pointing tasks using 3d target arrangements. *International Journal of Industrial Ergonomics*, 43(4):350 – 355, 2013.

[37] R.W. Soukoreff and I.S. MacKenzie. Towards a standard for pointing device evaluation, perspectives on 27 years of fitts' law research in hci. *International journal of human-computer studies*, 61(6):751–789, 2004.

[38] M. Akamatsu, I.S. Mackenzie, and T. Hasbroucq. A comparison of tactile, auditory, and visual feedback in a pointing task using a mouse-type device. *Ergonomics*, 38(4):816–827, 1995.

[39] A. Gregg-Smith and W.W. Mayol-Cuevas. Investigating spatial guidance for a cooperative handheld robot. In *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, February 2016.

[40] J.D. Gammell, S.S Srinivasa, and T.D. Barfoot. Informed rrt*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic. *arXiv preprint arXiv:1404.2334*, 2014.

[41] S.G. Hart and L.E. Staveland. Development of nasa-tlx (task load index): Results of empirical and theoretical research. *Advances in psychology*, 52:139–183, 1988.

[42] Natural Point. Optitrack v120:duo product page. Accessed: 2016-09-08.

[43] C. Li and C.D. Rahn. Design of continuous backbone, cable-driven robots. *Journal of Mechanical Design*, 124(2):265–271, 2002.

[44] A. Gregg-Smith. *Cooperative Handheld Robots*. PhD thesis, 2016. unpublished thesis.

[45] E. Nadaraya. On estimating regression. *Theory of Probability & Its Applications*, 9(1):141–142, 1964.

[46] I. Mohamad and D. Usman. Standardization and its effects on k-means clustering algorithm. *Res. J. Appl. Sci. Eng. Technol*, 6(17):3299–3303, 2013.

[47] Natural Point Optitrack. Natnet api user's guide version 2.10.0, May 2016.

[48] G. Murray. Rotation about an arbitraty axis in 3 dimensions. `http://inside.mines.edu/fs_home/gmurray/ArbitraryAxisRotation/`, June 2013. Accessed: 2016-09-08.

[49] G. Butterworth. Pointing is the royal road to language for babies. *Pointing: Where language, culture, and cognition meet*, pages 9–33, 2003.

[50] H. Volz, C. Gaser, F. Häger, R. Rzanny, H. Mentzel, I. Kreitschmann-Andermahr, W.A. Kaiser, and H. Sauer. Brain activation during cognitive stimulation with the wisconsin card sorting test — a functional {MRI} study on healthy volunteers and schizophrenics. *Psychiatry Research: Neuroimaging*, 75(3):145 – 157, 1997.

[51] San Jose State University. t-test table of significance. `http://www.sjsu.edu/faculty/gerstman/StatPrimer/t-table.pdf`, 2007. [Online; accessed Octoberber 9, 2016].

Appendices

A Servomotors

Figure A.1: Specifications for the Hi-Tec HS-7980TH Monster Torque Servomotors used on the robot.

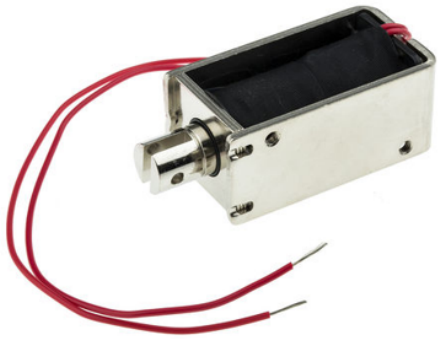
B Cable

Figure B.1: High strength fishing wire used as the cable.

C Electromagnet

Pull Action DC D-Frame Solenoid, 27mm stroke, 10W, 12 V dc

RS Stock No. **349-709**
Brand **BLP**
Mfr. Part No. **42-120-610-620**
Design change **Subject to design change**



Maximum Stroke	27mm
Action	Pull
Duty Cycle	100%
Magnetic Force	400g
Closed Power Continuous	10W
Supply Voltage	12 V dc
Length	32mm
Width	26.7mm
Depth	59.2mm
Weight	213g
Dimensions	32 x 26.7 x 59.2 mm

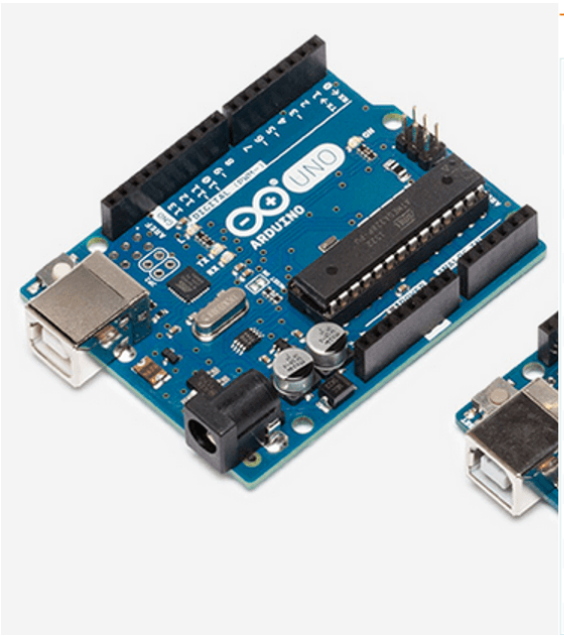
Figure C.1: Solenoid used for the electromagnetic end effector.

D Strimmer

Figure D.1: Wilko 250W grass strimmer used for the base (now discontinued).

E Arduino Board

E.1 Uno

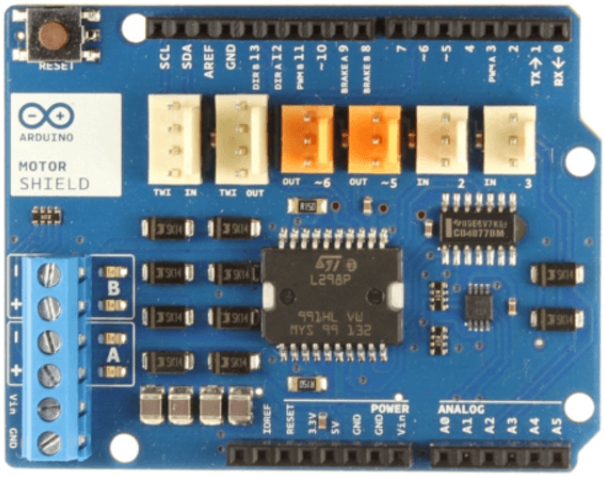


Technical specs

Microcontroller	ATmega328P
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
PWM Digital I/O Pins	6
Analog Input Pins	6
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328P) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz
LED_BUILTIN	13
Length	68.6 mm
Width	53.4 mm
Weight	25 g

Figure E.1: Specifications for the Arduino UNO, taken from https://www.arduino.cc/en/Main/ArduinoBoardUno

E.2 Motor Shield

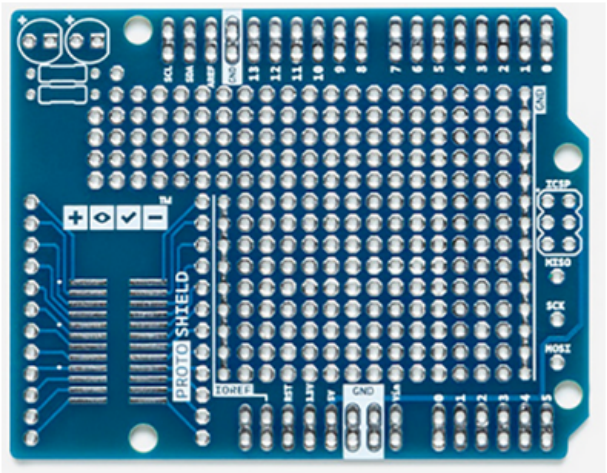


Summary

Operating Voltage	5V to 12V
Motor controller	L298P, Drives 2 DC motors or 1 stepper motor
Max current	2A per channel or 4A max (with external power supply)
Current sensing	1.65V/A
Free running stop and brake function	

Figure E.2: Specifications for the Arduino Motor Shield, taken from https://www.arduino.cc/en/Main/ArduinoMotorShieldR3

E.3 Proto Shield



The wide prototyping area has some extra features

- 1.0 Arduino Pinout
- 1 Reset Button
- 1 ICSP Connector
- 14 pin SMD footprint (50 mils pitch)
- 20 pin Through Hole footprint (100 mils pitch)

Figure E.3: Specifications for the Arduino Proto Shield, taken from https://www.arduino.cc/en/Main/ArduinoProtoShield

F Tracking Device

Camera Body

- Width: 11 inches (279.4 mm)
- Height: 1.6 inches (40.6 mm)
- Depth: 2 inches (50.8 mm)
- Weight: 1.3 pounds (0.6 kg)
- Mounting: 1/4"-20 tripod thread
- Display: 128 × 22 OLED

LED Rings

- No. of LEDs: 26 (× 2)
- Wavelength: 850 nm IR
- Strobe or Continuous Illumination
- Adjustable brightness

Lenses & Filters

- Standard M12 Lenses
  - Horizontal FOV: 47° [1]
  - Vertical FOV: 43°
  - Focal Length: 3.5 mm
  - F-number: 2.0
- Left and Right Cameras
  - 800 nm IR long pass filter

Input/Output & Power

- Data: USB 2.0
- Camera Sync: internal or external (via IO-X)
- 12V @ 3A

System Requirements

- Windows Vista/7/8/10
- 1GHz processor
- 1GB of RAM
- 50MB of available disk space
- USB 2.0 Hi-speed port

Image Sensors (×2)

- Imager Size: 4.5 mm × 2.88 mm
- Pixel Size: 6 µm × 6 µm
- Imager Resolution: 640 × 480 (VGA, windowed from 752 × 480)
- Frame Rate: 30, 60, 120 FPS
  - Frame decimation (transmit every Nth frame)
- Spatial decimation: 320 × 240, 160 × 120
- Latency: 8.33 ms
- Shutter Type: Global
- Shutter Speed:
  - Default: 1/1,000th of a sec. (1 ms)
  - Minimum: 1/50,000th of a sec. (20 µs)

Image Processing Types

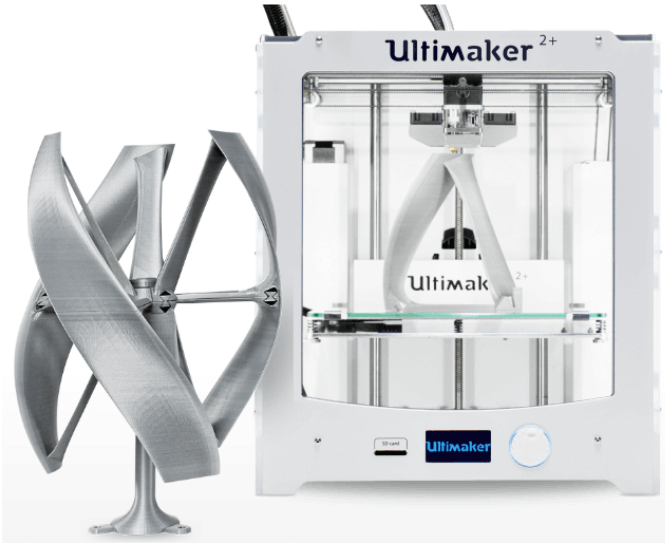
- Segment
- Precision Grayscale
- MJPEG Grayscale
- Raw Grayscale

In the Box

- 1 V120:Duo
- 1 Quick Start guide
- 1 license of Motive:Tracker
- 1 12V universal power supply (US/EU-compatible)
- 1 USB uplink cable (5 meters)
- 1 Hand rigid body

Figure F.1: Specifications for the OptiTrack Duo used for the project, taken from http://optitrack.com/products/v120-duo/specs.html

G 3D Printer



Layer Resolution	0.25 mm nozzle: 150 to 60 micron 0.40 mm nozzle: 200 to 20 micron 0.60 mm nozzle: 400 to 20 micron 0.80 mm nozzle: 600 to 20 micron
Build plate	50° to 100° C heated glass build plate
Build plate leveling	Assisted leveling process
Print technology	Fused Filament Fabrication (FFF)
Print head	Swappable nozzle
Build speed	0.25 nozzle: up to 8 mm^3/s 0.40 nozzle: up to 16 mm^3/s 0.60 nozzle: up to 23 mm^3/s 0.80 nozzle: up to 24 mm^3/s
Print head travel speed	30 to 300 mm/s
Feeder type	Geared feeder
XYZ accuracy	12.5, 12.5, 5 micron
Nozzle diameter	Included are 0.25, 0.4, 0.6 and 0.8 mm nozzles
Nozzle temperature	180° to 260° C
Nozzle heat up time	~ 1 minute
Build plate heat up time	< 4 minutes

Figure G.1: The 3D printer used for developing the bespoke parts was the Ultimaker 2+, taken from https://ultimaker.com/en/products/ultimaker-2-plus

H Code Snippets

H.1 Writing Motor Angles to Arduino

```
// Sends the appropriate signals to the Arduino to set the motor angles
public void setMotorAngles()
{
    // Already eliminated extreme angles in the Experiment class and no NaN values
        should have been added
    byte[] instructionBuffer = new byte[2];
    int oldAngle, newAngle, toWrite;
    while (anglesNotEqual())
    {
        for (int i = 0; i < 4; i++)
        {
            instructionBuffer[0] = Convert.ToByte(i + 1);
            oldAngle = currentMotorAngles[i];
            newAngle = targetMotorAngles[i];

            if (oldAngle < newAngle) oldAngle++;
            else if (oldAngle > newAngle) oldAngle--;
            else continue;

            toWrite = oldAngle;

            if (usingOffset && i == 0) toWrite += m1Offset;
            if (usingOffset && i == 2) toWrite += m3Offset;

            instructionBuffer[1] = Convert.ToByte(toWrite);

            lock(arduinoLock) // So other threads don't simultaneously write to
                Arduino and cause incorrect serial transmission
            {
                currentPort.Write(instructionBuffer, 0, 2);
            }

            Thread.Sleep(setMotorDelay);
            updateCurrentMotorAngles(i, oldAngle);
        }
    }
}
```

H.2 Receiving Instructions on the Arduino

```
void loop() {
...
  if (sentFromPC()) {

    int controlCode = Serial.read();
    int instruction = Serial.read();
    if(controlCode > 0 && controlCode < 5) {
      motorInstruction(controlCode, instruction);
    }
    else if(controlCode == 7) {
      triggerInstruction(instruction);
    }
  }

}

bool sentFromPC() {
  if(connectedToPC && Serial.available() == 2) {
    return true;
  }
  else {
    return false;
  }
}

void motorInstruction(int selectedServo, int angle) {
  switch(selectedServo) {
      case 1:
        currentServo = myservo1;
        break;
      case 2:
        currentServo = myservo2;
        break;
      case 3:
        currentServo = myservo3;
        break;
      case 4:
        currentServo = myservo4;
        break;
      default:
        break;
    }
    currentServo.write(angle);
}

void triggerInstruction(int instruction) {

    if(instruction == 7) {
      respondTriggerValue();
    }
    else if(instruction == 1) {
      digitalWrite(magnetControlPin, HIGH);
    }
    else if(instruction == 0) {
      digitalWrite(magnetControlPin, LOW);
    }
}
```

H.3 Regression Function

```
private double[] NWRegression(float[] inputVectorTarget, RegressionInput inputType,
    float alpha = startingAlpha)
{
    int numOutputDimensions = getNumOutputDimensions(inputType);
    double[] outputVector = new double[numOutputDimensions];
    double[] sumNumerator = new double[numOutputDimensions];
    double sumDenominator = 0;
    float[] newInputVector = getCopyVector(inputVectorTarget);

    if (inputType == RegressionInput.MOTORS)
    {
        convertMotors(newInputVector, true);
    }

    // loop through entire set of data
    for (int i = 0; i < storedCalibrationData.Count; i++)
    {
        float kernelInput = getKernelInput(i, inputType, alpha, newInputVector);
        float[] currentYValue = getcurrentYValue(i, inputType);
        for (int k = 0; k < currentYValue.Length; k++)
        {
            sumNumerator[k] += currentYValue[k] * kernelFunction(kernelInput);
        }
        sumDenominator += kernelFunction(kernelInput);
    }


    for (int k = 0; k < numOutputDimensions; k++)
    {
        outputVector[k] = sumNumerator[k] / sumDenominator;
    }

    if (inputType != RegressionInput.MOTORS)
    {
        convertOutputMotors(outputVector, false);
    }

    return outputVector;
}
```

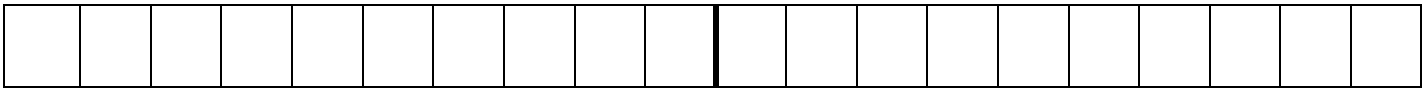
H.4 Logic and Angle Setting in Different Threads for User Study

```
// Sets the user study going, starts the thread to constantly update the motor
    angles
public void startStudy(UserStudyType type)
{
    activeStudy = new UserStudy(type, randomiseGesturing);
    experimentLive = true;
    new Task(liveExperimentThreadLoop).Start();
    runUserStudy();
    experimentLive = false;
    controller.zeroMotors();
    if(type == UserStudyType.GESTURING)
    {
        activeStudy.printRandomisedOrder();
    }
}

// Repeatedly sets the motor angles (delay is present within setMotorAngles
    function)
private void liveExperimentThreadLoop()
{
    while (experimentLive)
    {
        setMotorAngles();
    }
}

// Runs user study by informing object of base position and trigger, and fetching
    appropriate position
private void runUserStudy()
{
    bool triggerPress;
    float[] relativeTargetPosition;
    if (!activeStudy.isInitialised())
    {
        // Check that it has been initialise before attempting to run
        Console.WriteLine("unable to initialise user study");
        return;
    }
    relativeTargetPosition = activeStudy.getRelativeTargetPosition();
    runningStudy = true;
    while(runningStudy)
    {
        triggerPress = getTrigger();
        runningStudy = activeStudy.update(basePosition, triggerPress);

        if(triggerPress)
        {
            zeroMotorAngles();
            controller.activateMagnet(true);
        }
        else if (runningStudy)
        {
            controller.activateMagnet(false);
            getMotorAnglesForTargetPoint(relativeTargetPosition);
        }
        Thread.Sleep(newTargetDelay);
    }
}
```

Appendix I – TLX Questionnaire

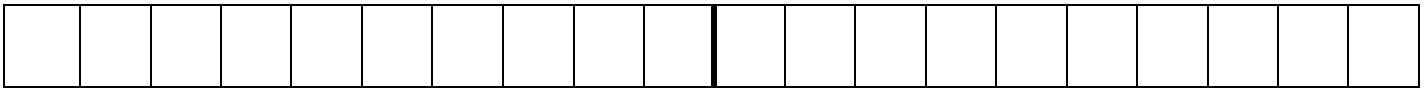
Participant ID number: Age:

1) **Mental Demand (no assistance)**: How much mental effort was required to follow the instructions for the non-assisted task?



Very low Very high

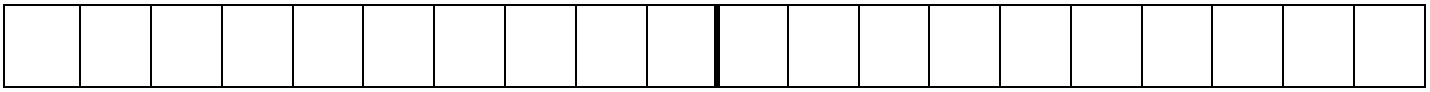
2) **Mental Demand (assistance):** How much mental effort was required to follow the instructions for the gesturing assisted task?



Very low Very high

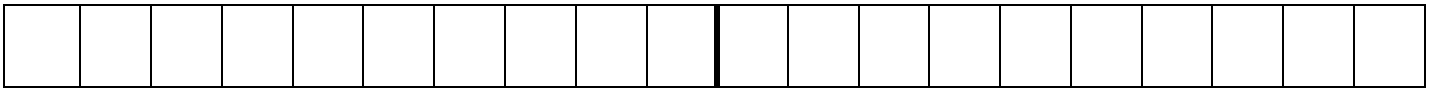
For the following questions please tick for the assisted task, and circle for the non-assisted task

3) **Temporal Demand**: How hurried or rushed did you feel during the task?



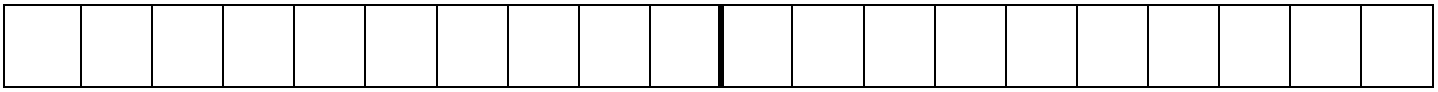
Very low Very high

4) **Performance**: How successful were you in accomplishing what you were asked to do?



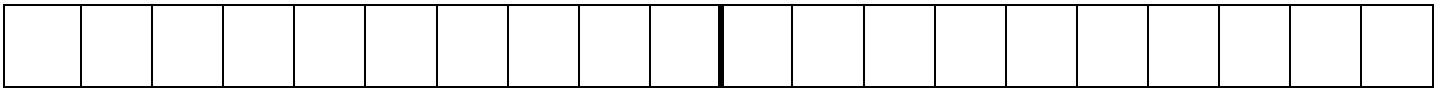
Perfect Failure

5) **Effort**: How hard did you have to work to accomplish your level of performance?



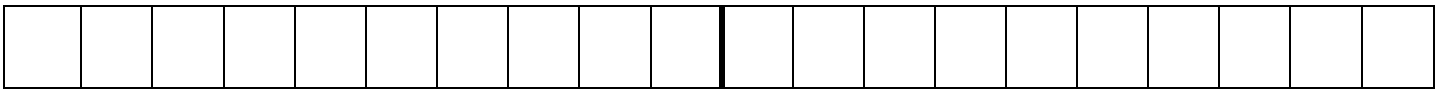
Very low Very high

6) **Frustration**: How frustrating was the task to complete?



Very low Very high

7) **Physical Demand**: How physically hard was it to manoeuvre the robot to pick up blocks?



Very low Very high