Declaration

I confirm that I have read and understood the Plymouth University regulations relating to Assessment Offenses and that I am aware of the possible penalties for any breach of these regulations. I confirm that this is my own independent work.

Candidate's Signature:

Sami Sequeira _____ Date: 11/10/2020

Supervisor's Signature:

Swen Gaudl _____ Date: XX/10/2020

Second Supervisor's Signature:

Chunxu Li _____ Date: XX/10/2020

Chunxu Li
Digitally signed by Chunxu Li
Date: 2020.10.12 09:16:43 +01'00'

Copyright & Legal Notice

University of Plymouth

Implementation of a Walking Sensor for Aiding Bipedal Locomotion

Sami Sequeira

Supervisor: Dr. Swen Gaudl

A thesis submitted to the University of Plymouth in partial fulfilment of the requirements for the degree of:

Master of Science, Robotics

September 2020

School of Computing, Electronics, and Mathematics

Faculty of Science and Engineering

Acknowledgements

I am immensely grateful for the influence and advice of Oliver Smith and my fellow Robotics colleagues during this project. Their contributions have been truly invaluable.

Table of Contents

Implementation of a Walking Sensor for Aiding Bipedal Locomotion

Sami Sequeira[1]

University of Plymouth, Plymouth, PL4 8AA, United Kingdom
sami.sequeira@postgrad.plymouth.ac.uk

Abstract. Humanoid robots' locomotive systems form one of the most important factors in their mechanical versatility. To truly work alongside a human, the ability to navigate human-centric environments is a great advantage. However, obstacles and semantic environmental knowledge that seems intuitive to humans can sometimes prove to be a significant challenge to humanoid robots.
This project will assess the state of the art of humanoid robotics, and in order to improve the locomotive ability of the Scott humanoid robot, will select a sensor-based strategy to be implemented. After selecting a sensor and implementing a strategy, its effectiveness will be assessed by contextually appropriate tests, and the results discussed and analysed. Insight gained from the project will be used to suggest improvements and further work to be carried out.

Keywords: Humanoid Robots, Bipedal Locomotion, Terrain Modelling, 2D LIDAR Sensor

1 Introduction

One of the most important parts of an autonomous robot is its locomotive system. Within this field, different applications vary in their requirements, and thereby produce different solutions - for example, search and rescue applications might require tracks, and self-driving automobiles might require wheels. However, in order for a robot to perform a variety of tasks in human environments, it becomes necessary for said robot to be able to navigate the same range of environments that humans often do. One of the most obvious methods to achieve this is by taking mechanical inspiration from humans themselves. Hence, there has been much interest in the development of legged robots (and bipedal, humanoid robots in particular).

Modern humanoid robots encompass all robots built with a basic structural similarity to the human body. However, this typically involves a reliance on servo motors and rigid connections between joints, rather than a more flexible, human-like movement control system. This trend can be seen throughout a wide range of humanoid robots over decades of the field's progression [1] [21] [30], and continues to be a cornerstone of humanoid robotic design. However, alternate

approaches and technologies have also seen considerable development, such as compliant actuators [38], different joint structures [15] and novel gaits [23].

Throughout their history, one of the ultimate metrics of humanoid robots has been their ability to compare to real human locomotive abilities. This would enable them to tackle many of the same tasks that humans do, opening up new avenues of robotic research and application. In pursuit of reaching a human-like level of performance, numerous measures have been devised, comparing results such as average speed and ability to deal with obstacles. One such measure is the RoboCup Humanoid League, a set of football matches played entirely by humanoid robots. Rather than measuring a single mechanical performance metric, these matches measure the collective performance of a variety of different robotic fields; a competitor robot might use artificial vision to locate a ball and teammates, advanced decision-making to decide where to move on the pitch, and kinematics in order to move using its legs.

The most relevant areas of such tests to this project are related to bipedal locomotion and terrain perception. The available types of terrain perception sensors must be evaluated in relation to the scope of the project, and their suitability for bipedal locomotion. In addition, consideration must be made of the mechanics of bipedal humanoid locomotion, as well as the underlying methods of control, when implementing the sensor mentioned in the title and the underlying code. This will determine the necessary processing and communication of the sensor's readings, such that they are able to benefit humanoid walking robots such as Scott.

Note that several sections of this project are pending publication as a research paper. Areas of major overlap include paragraphs from each major section of the project, though many subsections are omitted.

2 Literature Review

2.1 Motivation

Scott, the focal robot of this submission, is designed to perform well in tests similar to the RoboCup competition described earlier. It is a ROS-based, humanoid, bipedal robot with actuated arms and legs, and a wide variety of additional components that can be installed, such as exteroceptive sensors or specific end-effectors. It is a bespoke robot built by the University of Plymouth, and intended to participate in humanoid mechanical challenges (such as the aforementioned robotic football competitions). It now serves as an important stepping stone of a technical revival on that front. However, no specialised locomotive improvements have yet been implemented using said exteroceptive sensors, despite the greater depth of information available from sensing the terrain that Scott traverses. In addition, this could also enable Scott to make faster, lower-level decisions regarding gait and foot placement, mimicking reactive elements of human locomotion to a degree [43]. Sensors that are suitable for these geometric readings include LIDARs, cameras (alongside computer vision), ultrasonic sensors, and time-of-flight sensors, among others.

2.2 Preliminary Details & Analysis

Description: Scott has 18 degrees of freedom, spread out over two actuated arms (currently lacking end-effectors) and two actuated legs (including actuated ankles and feet). These arms and legs very closely mimic the structure of human limbs; shoulders and ankles have two degrees of freedom, elbows and knees have one, and hips have three. The angles of these degrees of freedom are mechanically similar to those of the associated human joints, broadly speaking. This mechanical similarity is the root of Scott's comparability with human walking.

Scott's torso connects each of these limbs and, in addition to being the mounting point of Scott's central microcontroller, also hosts a battery and a range of mounting points for other optional components. A good example of such optional components would be the sensory array used as Scott's 'head', which can be swapped depending on which sensor is required. It typically includes two further degrees of freedom, to actuate the sensory array's direction. Each component's degrees of freedom are controlled via a specified angular position, published by said torso-mounted microcontroller and received by a servo motor.

This range of equipment, however, has seen little improvement since Scott's original construction in 2015. Due to the bespoke nature of its construction, direct mechanical upgrades remain difficult to perform without completely rebuilding Scott, and thereby incurring an excessive project cost. As a result, the novel improvements to Scott that this project will focus on must be restricted to those that make significant use of Scott's preexisting chassis and components.

Analysis of Improvements: Potential improvements to Scott in this project have been restricted to those that are largely compatible with its preexisting chassis and components. In this vein, however, many separate approaches for improvement may be considered. Rather than directly mechanically improving Scott's locomotive ability, for example, improvements in programming can yield similar results. It is possible to employ these in conjunction with dynamic, flexible gaits or specialised components such as gyroscopes, for example. This enables humanoid bipedal robots to compensate for factors such as foot slippage, unperceived obstacles, and other perturbations (e.g. a teammate robot bumping into them during a match) [26] [20].

However, many such compensation strategies are reactive in nature; rather than decreasing the likelihood of an erroneous movement, they instead mitigate the impact of said error occurring. While very relevant to a complete movement programme, these solutions do not necessarily address minimisation of the likelihood of the initial error. Considering this as a potential avenue for design developments, therefore, may yield novel improvements to Scott's locomotion. It is also worth noting that many such reactive strategies rely on high-frequency motor position sensing and control; in order to sense and prevent a humanoid robot from falling, shorter delays in data transfer and control are advantageous. Scott's capability to match the required rates of data transmission may be in question, however, depending on the microcontroller and actuators used during its restoration. Insufficient processing power may slow down Scott's movements.

Reactive strategies that remain nonetheless appropriate for Scott include the approaches previously mentioned, among others. As described earlier, many such techniques revolve around estimating a foot position that minimises the danger of the robot falling. This is achieved with varying degrees of success using a range of methodologies; these may be either dependent on or independent of various sensory data and predictive algorithms [13] [26] [20] [14] [25]. However, they all make use of servomotors that are functionally similar to Scott's; they are controlled via feedback based on angular position. Depending on the necessary rates of data transmission, many of these may be considered for Scott.

More proactive solutions include those that reduce the frequency of the occurrence of motion errors. This include issues such as mis-perception of the walking surface, which can cause the mechanical considerations of walking to be significantly different than predicted internally by Scott. As a result, many of them are based on sensors and programming as opposed to direct mechanical improvements, as a more complete ability to sense, store and interpret data about the terrain being traversed would increase the probability of Scott's mechanical walking predictions remaining accurate [3] [4] [2].

Although this indicates that a wide variety of improvements are suitable for consideration in this project, consideration must be made of the ongoing restoration of Scott. Several of the example strategies cited that minimise the danger of the robot falling are built-in features of a generalised walking gait generator, and as a result, would require Scott to be fully restored to original functionality as a prerequisite to their implementation. Strategies such as these must therefore be relegated to implementation in future work, rather than during the current project. This is due to the ongoing nature of Scott's restoration. Furthermore, in light of the ongoing COVID-19 pandemic, it is necessary to consider improvements that can be implemented on a work-from-home basis. This presents an even more restrictive set of potential improvements, as the Scott robot is unavailable for any sort of testing outside of a laboratory setting.

In order to perform testing in a home environment, several options exist. A system can be developed and tested separately from the Scott robot entirely, to be potentially attached when the opportunity arises. Also, a simulation model of the Scott robot could be constructed, and work performed using this model to acquire experimental results. In the first case, the advantage of real-world results are retained, as well as the opportunity for the system to be physically implemented with Scott when normal work resumes. However, this requires components and tools to be acquired in a work-from-home environment. In the second case, the advantage of a more complete experimental testing setup is gained, but does not necessarily involve constructing the physical system for implementation on Scott. Furthermore, simulating results may lack some of the details of real-world results, and also requires high computational power to be available in a work-from-home environment.

2.3 Overview of Walking Strategies

The aforementioned variances in fall avoidance methodology are often found in technicalities of the walking gait. Walking gaits are typically derived using the traditional Linear Inverted Pendulum Model (LIPM). Determination of foot position in this model is often by methods such as ZMP (Zero Moment Point) preview control [44], which restricts the robot to walking on flat ground and involve a large amount of lateral centre of mass (COM) motion during walking. More recent developments, however, have seen LIPM walking gait derivations based on concepts such as the 3D capture point [22] and the divergent component of motion [12]. For the latter in particular, bespoke modification has broadened its capability, enabling it to cope successfully with uneven terrain [11].

A recently popularised alternative to this model is the Dual Spring-Loaded Inverted Pendulum (dual SLIP) model, which was originally proposed in 2D [19] but has since been extended to work in 3D. It has been studied less overall than the traditional LIPM, leading to fewer different gait implementations based on it being published. However, it shows generally superior modelling of human motion, producing effects very similar to those observed in real human walking. Among these are dual-peak ground reaction forces and vertical centre-of-mass oscillation. It also inherently supports a dual-support phase of walking, which directly contrasts with the LIPM, which requires separate theoretical considerations to be made for dual-support phase compatibility.

The dual SLIP model's superior similarity to real human movement has had notable effects in regards to the research based on it; when used to mechanically model the leg structure of a real human, it was able to accurately predict certain muscle usage timings [18]. Furthermore, its use in modelling humanoid walking was accurate enough to enable the control of a robot which was notably underactuated [32].

A wide range of walking gait approaches are therefore possible for Scott, assuming that the necessary hardware requirements are met. However, very few of them are able to be fully implemented and built upon in the timeframe of the current project, and as such must be relegated to a potential part of restoration efforts in Scott's future. In order for the contribution of this project to remain applicable to Scott throughout this future, it then becomes essential that this project's contribution to Scott must be compatible with all walking gaits examined. Currently, this seems to require compatibility with both the LIPM and dual SLIP models, depending on the gait chosen for implementation, but further common characteristics between the potential gait choices cannot be guaranteed. The vast majority of these gaits require Scott to continue using servos that are controlled via positional data. Considering this and the added difficulty of replacing Scott's chassis, it is likely that the current positional control system will remain in use with Scott for the foreseeable future. As such, the project can depend on similar positional data being available from Scott.

2.4 Approaches for Terrain Detection & Identification

Terrain detection is often performed using a variety of wide-area ranging sensors, which can vary from RGB-D cameras [2] to LIDAR sensors [33] [37], or a combination of both sensors in tandem. Rather than detecting a single point as a single laser rangefinding sensor might, these sensors broadly perceive the individual distances of each point in a swathe, forming a pattern. Cameras generally sense this within a square cone of vision, whereas LIDARs sense this in a set height above and below the sensor in a 360-degree radius. Robots making use of these techniques also often actuate the sensors, allowing them to perceive an even wider area of the terrain around the robot. In conjunction with a known reference frame relative to the ground, robots making use of these sensors can then relate the positions of these points to each other, allowing the construction of a map to be performed. It is also possible to probabilistically interpret the current sensory readings in order to determine the robot's likeliest current position in such a map; this is known as localisation. Performing both of these operations on the same sensory data in tandem is known as Simultaneous Localisation And Mapping (SLAM) [8].

Various methodologies can then be applied to transform the mapped terrain into a cost map for path planning algorithms and the like, enabling automatic navigation and similar applications. When SLAM is implemented without programming that attempts to identify the sensed surfaces, however, it is routine for objects to be mistaken for terrain by such sensing methods.

Terrain identification and object identification are distinct applications, but both involve usage of similar data. Broadly, these data can be classified into data that relates to the geometry of an object (such as its number of legs, or the arrangement of its largest surfaces), and data that relates to the texture of an object (such as lighting, colour, visual decals and decorations that have a negligible geometric impact on the object). Good examples of texture data include logos and icons (as in [10]), which are specific, consistent arrangements of features and colours that have little effect on geometric data. Good examples of geometric data include shapes and physical features of objects that are roughly consistent, such as desks having a large, flat, horizontal top face (as in [35]).

In order to identify objects, several approaches are used. Deep learning systems are able to train on various objects, enabling them to search camera images for said objects from a variety of angles with high degrees of success, depending on the particular objects involved [9]. Depending on the training data, this approach can be very robust to changes in lighting, orientation and distance of objects. Furthermore, depending on the depth of research conducted, they are capable of reaching very high accuracies for specific objects; a good example is bespoke facial recognition neural networks, which can currently exceed human performance in this area [40]. For applications depending more on textural data, one way to implement object identification is to use different forms of template matching [10]. This method may be based on processes such as edge matching (using Canny, for example), gradient matching, or greyscale matching, among others. Many feature-based methods exist that also involve geometric data in

the matching process - for example, using the Scale-Invariant Feature Transform (SIFT) [27] or pose clustering/geometric hashing.

All of the aforementioned strategies are technically compatible with Scott, conditional upon the use of the appropriate sensors. However, system sizing, timing and project cost provide limits to the available solutions; Scott's size is small relative to many robots in the literature. Using large sensor setups could therefore make Scott much more prone to overbalancing. Furthermore, since Scott's size limits battery capacity and microcontroller size (and thereby processing power), different sensors may require unsuitable amounts of energy or processing power to function efficiently. The requirement to process these images using computer vision systems increases the necessary computing power as a function of resolution. The implementation and potential training of more complex computer vision systems is another barrier to Scott's limited computing power, and also represents an arguably excessive time cost for a project of this scale. The fall avoidance applications described in previous sections also require Scott to process any sensed data within a fraction of a second, which provides more incentive to prioritise data processing speed over higher detail. Scott can therefore only depend on limited data from any sensors that will potentially be implemented, and may need to rely on the more less programmatically demanding forms of computer vision analysis.

2.5 Comparative Sensor Analysis

Cameras: In order to implement a camera for Scott, it is necessary to design a method of extraction of depth data. Among these solutions is the use of multiple cameras with vergence calculations [41], the implementation of both a camera and a separate depth sensor in tandem [37], and the use of specially designed RGB-D cameras (such as the Kinect) [2]. In the case of the Kinect in particular (as well as other commercial RGB-D cameras, such as the ASUS Xtion), this often revolves around the projection of a pattern in the IR spectrum (as in figure 1). These cameras enable depth to be calculated without disrupting any data present in the visible light spectrum.

Fig. 1: A photograph displaying the pattern of dots in the IR spectrum used by the Kinect to assess depth [16]

Given the previously mentioned restrictions on Scott's ability to process data, it becomes important to minimise the number of individual sensors used, otherwise image resolution must become even lower of a priority for Scott. However, each of these methods cited includes at least two sensors. No distinct advantages exist between each of the different camera-based methods in terms of the volume of data transmitted, except for the additional data load associated with images including an RGB channel and variations in frame rate. The availability of said RGB data may provide useful information for identification of terrain and objects, however. As a result, the availability of an RGB channel is advantageous overall.

There is also a distinction between commercially sold cameras, and manually constructing one from separate components. The Kinect, for example, offers a relatively low-cost, pre-assembled solution that is compatible with open-source tools to make use of it in a robotic context. This can minimise both financial and time costs for the project by reducing necessary design work and low-level coding. Constructing a sensor out of separate parts would enable more choice about the exact parameters of the sensors involved, but the development of a bespoke computer vision system with depth capability would increase the amount of time needed for the project. As such, a pre-built commercial RGB-D camera of appropriate scale for Scott is recommended if cameras are implemented. The presence of open-source code for said camera's implementation is also an advantage.

Ultrasonic Sensors: The measurement of distance data can be performed by directional ultrasonic sensors (as in [28], [31]), with the caveat that each individual sensor is only capable of sensing a single data point. This is performed by projecting a high-frequency sound wave towards a perceivable object, then recording the time taken for it to return, in order to calculate the distance travelled by the wave (and thereby the distance to the perceived object). In order to survey the terrain in front of Scott using such a sensor, it becomes necessary to use actuation or additional ultrasonic sensors, enabling multiple data points to be read in order to approximate the terrain Scott will traverse. This has the advantage of much lower data intensity than cameras or other sensors that return a high amount of data. By extension, this minimises the need for concern about Scott's limited computational power. These sensors might be implemented in several ways.

Adding actuation to an ultrasonic sensor may follow a similar design principle to a LIDAR [33]; incrementing the orientation of the sensor on a turntable. This is done in order to sense adjacent points of the terrain in an arc about the rotating sensor. However, a LIDAR is only able to make use of this operating principle due to its speed; a LIDAR reads data around it with enough speed to approximate an instantaneous measurement. An ultrasonic sensor, in contrast, has a more limited speed; sound waves travel much more slowly than the electromagnetic waves used by LIDAR sensors. Therefore, even with an optimised design, an ultrasonic sensor that attempts to operate on the same principle

might require a slower turning time to compensate. This would compromise the approximately instantaneous measurement of the data, however. Further to these points, the construction of such a sensor might significantly increase project costs and completion time.

Using an array of multiple ultrasonic sensors instead may alleviate this problem; they could be mounted directly to Scott's torso, such that they simultaneously measure the distance to several adjacent terrain points in a specified area in front of Scott. This solution has the advantage of being technically simple to implement physically, as well as requiring only a low amount of computational power to process the readings. However, small angular errors in the arrangement of each sensor might lead to significant error in the measured data. Moreover, unless sensors of a significantly reduced size compared to common industry entries are available, the number of sensors that could be mounted could not realistically be above 10. This greatly limits the resolution of the available data.

In either implementation, it is also notable that ultrasonic sensors remain unable to sense changes in colour or lighting. Therefore, the use of ultrasonic sensors would require an additional colour-compatible sensor, in order to read the same information available to approaches such as the camera-based methods considered earlier.

Laser Rangefinders: Laser rangefinders operate by firing a beam of electromagnetic waves (such as infra-red (IR) beams), and then use a sensor to detect the resultant spot on objects in the beam's path. This enables it to record the distance travelled by the beam, as shown in figure 2. A single laser rangefinder (also known as a time of flight sensor) is only capable of reading the distance to a data point directly ahead of it. This is a similar issue to the one that arose when considering a single ultrasonic sensor. Similarly, this limited sensing capability is unsuitable for the application required in this project, as a sensor must be able to provide enough data for a useful overview of the terrain that Scott will navigate.

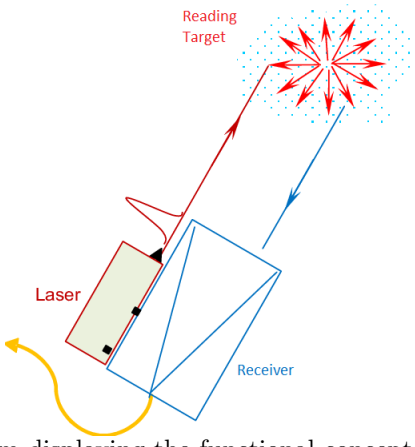


Fig. 2: A diagram displaying the functional concept of a LIDAR [24]

One way to overcome this limitation of a single sensor is to instead use a LIDAR, which enables approximately instantaneous measurement of distances all around the LIDAR while using a single laser. This has several advantages; the frequency of data being read is high [36], enabling Scott to interpret behaviour such as falling with minimal delay. This is helpful to facilitate faster reactions on Scott's part. LIDARs are commercially available as complete sensors with open-source code available, significantly cutting down on project time costs. Although they have financially high costs per unit, an exception can be made for this project, due to the availability of a Slamtec RPLIDAR A2M8 from a previous project. Restricting the data to a 2D plane also enables the project to progress in stages, developing a successful 2D process before attempting to implement it in 3D. Furthermore, at longer distances, LIDARs often exhibit improved accuracy compared to RGB-D cameras.

The disadvantages of using a LIDAR include the likelihood that the 360 degree field of vision will not be fully used, leading to an unnecessarily high amount of data being transmitted from the sensor. Furthermore, LIDARs require a significant amount of space and clearance from other moving parts, with a potentially increased risk of overbalancing depending on how it is mounted to Scott's chassis. Lastly, the Slamtec RPLIDAR provided from a previous project requires minor repairs to restore its functionality, potentially adding to the project's time costs. However, it is notable that a LIDAR is technically disallowed by the RoboCup rules, rendering its value more relevant to research outside of the competition if this sensor is used.

The repairable LIDAR provided is unable to sense RGB values of the surfaces it senses, but is able to return reflectance data instead. Although this data's capability is severely limited compared to RGB values, it is still able to be used to draw broad conclusions about the material being sensed, especially in conjunction with data such as the surface angle [17].

2.6 Literary Conclusions

A broad perspective of the potential improvements to robotic humanoid locomotion was taken, and their viability for Scott in this project was assessed. Although a wide variety of improvements were considered, Scott's bespoke construction, the scope of the project, and the ongoing COVID-19 pandemic restricted work choices. The most realistic options were either to simulate Scott, or developing a separate system for implementation with Scott after the completion of the project. However, both choices involve designing a similar form of improvement; the implementation of a novel sensor, aimed to improve Scott's locomotive ability.

This is aimed to mimic the mechanism by which humans are able to make reactive, almost instantaneous decisions about how to modify their gait when an obstacle is spotted in their path [43]. Hopping over small curbs, adjusting for factors such as grass collapsing under one's steps, and rotating one's gait to move up stairs are adjustments that humans generally perform as a matter of routine; enabling Scott to perform a similar feat could greatly improve its mobility.

In order to assess the most appropriate sensor that would remain compatible with other work being performed on Scott, prospective walking gait generation strategies were first examined. This revealed that specifying positional data was likely to remain the method by which Scott's gait was controlled for the foreseeable future, suggesting that any improvements should be designed with the aim of providing positional information on the terrain surveyed. It also reinforced that many of the most effective fall avoidance strategies are inherent in certain advanced walking gaits, rather than being able to be developed separately as part of this project.

The possible strategies for terrain and object detection were then summarised, and it was concluded that while many of them were technically capable of being used on Scott, Scott's size provides a significant limit in terms of the battery size, computational power and sensor size that it can physically support without becoming physically unstable. This should be considered in more detail in the design of the implemented sensor. Further analysis of several broad categories of sensors revealed that commercial RGB-D cameras and laser-based sensors such as LIDARs were the most applicable to Scott for this project.

It is therefore evident that out of the options available in the literature, the improvements applicable to Scott during this project should be based on either LIDAR technology or RGB-D cameras. These should be used to read data about the material of the terrain and its position relative to Scott, as it will provide Scott with additional information. This could enable it to traverse a range of terrains with a lower likelihood of errors such as slipping and falling, and thereby contribute greatly to Scott's ability to play robot football.

3 Design & Methodology

3.1 Test Setup Choice and Trialing

In order to aid in the ongoing restoration of Scott as a fully actuated humanoid robot, this project has the eventual design aim of enabling Scott to play robot football, ideally at a level that is competitive with recent entrants in said robot football matches. Therefore, as a result of Scott's construction being several years out of date, it becomes necessary to implement a method of improving Scott's performance in tests such as the RoboCup Humanoid League (alongside the independent work being performed to restore Scott's original mechanical functionality).

In order to decide which method would be used to carry out the project work of implementing a walking sensor for Scott (seen in figure 3), testing was performed with each method outlined in the previous section. The initial stages of work in each method were carried out, but only to the extent of basic simulation and building part of a sensing system in a home environment respectively.

Simulation: In order to test simulation, a virtual machine running Ubuntu 16.04 was installed onto a PC in the home working environment. This coding

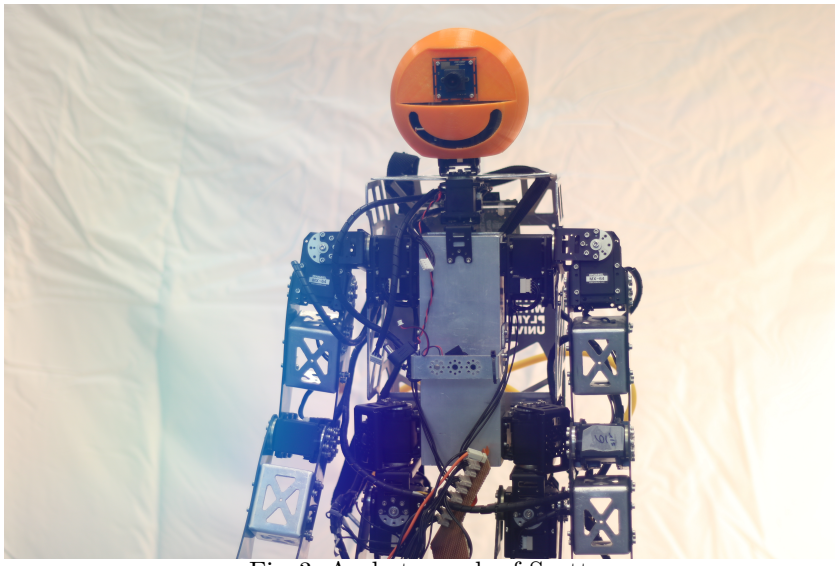


Fig. 3: A photograph of Scott

environment has the advantage of not only being simulation-capable via the use of ROS, but also of mimicking the coding environment that may be directly implemented on Scott. This involved using Ubuntu MATE on a Raspberry Pi, but was unable to be physically implemented due to the necessity of working from home.

To assess the simulation capability of the virtual machine, testing was performed by implementing a basic Kobuki simulation in Gazebo, with keyboard teleoperation control via a specialised ROS node [39]. This was deemed a valid test, as a simulation of Scott would require a model to be constructed and controlled using the same software. Unfortunately, the virtual environment was unable to stably support a simulation, leading to a low frame rate and excessive simulation times. This is likely due to the limited amount of accessible memory by the virtual operating system, and as such is unable to be made feasible within the timeframe of the project. This is especially alarming because of the simplicity of the Kobuki simulation; a simulation of Scott would involve multiple actuated components, whereas the Kobuki simulation involved an entirely rigid robot.

Furthermore, simulation of Scott would require much more complex nodes than the keyboard-operated differential drive of the Kobuki simulation; the internal software of Scott that enables it to walk would also need to be recreated. This would involve the simulation reading the angle of each of Scott's actuators, performing inverse kinematic calculations, and specifying new angles at a high enough frequency to avoid Scott falling. Furthermore, Scott currently lacks a gait generation program, which would be necessary in order to test the sensor system of this project with Scott's walking. This is self-evidently much more complex than the Kobuki simulation, requiring even more processing power; if the initial

simulation test failed due to a lack of processing power, it is very unlikely that a full simulation of Scott would be successful. The additional work to be done in replicating Scott's programming for inverse kinematics and gait generation is also a very significant disadvantage of this approach.

Physical Sensing System: To assess the feasibility of building a physical sensor system from home and without laboratory access, the initial task of replacing the motor of the repairable LIDAR mentioned previously was set. This LIDAR unit is a Slamtec RPLIDAR A2M8 [36], though its internal motor is faulty. By comparing rated motor speed and load, it was found that replacing this with an external DC hobby motor (exact model in Appendix A) powered from a phone charger should enable the LIDAR unit to rotate at approximately the same rate as the internal motor would. Rather than remove the internal motor, however, this additional DC motor will be attached to the LIDAR's mounting. Using other, similarly available hobby supplies, a basic belt drive can be built to actuate the LIDAR by treating the outer casing as a makeshift gear (as seen in Figure 4). This prevents the risk of any additional damage being done to the internals of the LIDAR unit, which is especially important in a work-from-home environment. Similar reasoning motivated the choice of a phone charger to power the DC motor; in the absence of laboratory power supply units, a phone charger (with a known voltage and current draw, and spliced with a variable resistor to adjust voltage) is able to fill a similar role in a safe manner.

Fig. 4: Hobby motor and makeshift belt drive to actuate the LIDAR

As is evident from Figure 4, assembly of the belt drive joining the external motor and the LIDAR sensor was successful. In order to create a mounting for testing, placeholder 3D-printed mountings were built to support this assembly. These mountings allow the DC motor to safely actuate the LIDAR using the belt drive system, and also keeps the LIDAR stable (and thereby the domain of its readings consistent), if necessary for software testing.

The DC motor used was rated at 3V and 1A, with an associated angular velocity of roughly 13000rpm. The phone charger is rated for 5V and up to 2A; this voltage was mitigated via adjustment of the previously mentioned variable resistor, allowing it to match the rated voltage of the DC motor.

Measurement of the radius of the LIDAR and the gear attached to the DC motor showed that the belt drive's gear ratio is approximately 3.5. This causes the angular velocity of the LIDAR to be decreased relative to the angular velocity of the DC motor by this ratio, and the torque exerted by the motor to be increased by a similar amount. The LIDAR's intended rotation speed is roughly 600rpm, but it can safely operate at a range of roughly 300-900rpm.

Taking this into consideration and assuming that the torque load exerted on the DC motor matched its rated load, the LIDAR would theoretically spin at roughly 3750rpm. As a result, it was necessary to use the variable resistor to reduce the voltage significantly before the first test, to minimise the risk of damage to the LIDAR.

Upon testing, it was evident that the voltage had been set too low using the variable resistor, as the LIDAR rotated at a visibly slow rate. Using the variable resistor to slightly increase the voltage of the DC motor produced a proportionate increase in the angular speed of the LIDAR. This suggested that the predicted torque load of the LIDAR was too low, so the variable resistor was slowly adjusted until the LIDAR reached a visible rotation rate of 3-4 Hz. Notably, this point was near the minimum resistance of the variable resistor. Due to a lack of instrumentation and tools in a home environment, exact measurement of the angular speed was not feasible.

After gradually testing a range of voltages, it was found that no additional resistance from the variable resistor was required, as the LIDAR's actuation torque requirement exceeded the rated torque of the DC motor, to the extent that the angular velocity of the DC motor was naturally reduced to within the operating range of the LIDAR. The variable resistor was therefore removed, to ensure a consistent angular speed of the LIDAR for later experiments.

In order to ensure that the LIDAR was fully functional, basic software provided by Slamtec for use with the RPLIDAR A2M8 was used to visualise its readings while it was actuated by the DC motor belt drive. This was performed using the same coding environment as the simulation trial was performed on (Ubuntu 16.04 using ROS), but the memory issues did not disrupt the LIDAR readings, despite the readings updating at a high frequency. The validity of the LIDAR's readings is evident from Figure 5, suggesting that no further repairs need to be made in order to progress the project using this testing setup.

Conclusions: From the trials performed, it was clear that simulation was a non-viable method of continuing the project on a work-from-home basis. However, the trial of developing a physical sensing system was successful, restoring the physical performance of the LIDAR. As a result, the physical sensing system is the most suitable choice of test rig for continuing the project.

In order to fully take advantage of the physical sensing system, however, it is important that it is designed and programmed with consideration for Scott. This involves the knowledge that Scott is most likely to use a gait based on positional control, as well as an awareness of Scott's limited system size. Minimising power consumption and computational power requirements of the system are

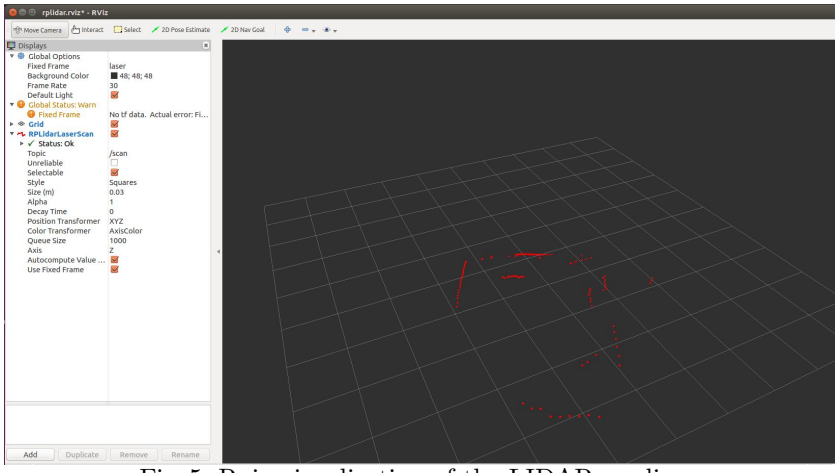


Fig. 5: Rviz visualisation of the LIDAR readings

therefore priorities, as well as interpreting the readings in a way that is useful to a positionally specified gait.

To ensure consistent performance, it is also important that an implementation of this system on Scott applies the same torque and speed. The motor information can be found in Appendix A, and the phone charger used to power it was rated at 5V and 2A output. If the same motor is used for the real implementation on Scott, then the power supply circuit should conform to similar values.

3.2 Code

Note that the complete package is available in the 'laser_reader' folder, found at 'https://github.com/Firnenenrif/scott'. The manufacturer's software is also included, in the 'rplidar_ros' package.

Extracting LIDAR Readings: The coding environment used to program the sensing system is the same as that described in the previous section. To recap, this is a virtual machine running Ubuntu 16.04. This is an approximation of the intended microcontroller for the sensing system, which is likely to be a Raspberry Pi (or similar microcontroller) running the Ubuntu MATE operating system.

The LIDAR communicates with this operating system via a USB adapter. The relevant port can be accessed from within Ubuntu by manipulating the relevant permissions for the virtual machine, using the command 'sudo chmod 666 /dev/ttyUSB0'. This method of connection is typically supported by the Raspberry Pi without the need for said permission manipulation, however; as a result, the code used for granting access permission will be left out of the completed program. This should ensure that the code for this project is compatible with immediate installation on a Raspberry Pi.

The basic ROS node used to access the readings of the LIDAR is provided on the manufacturer's website [36]. When the port permissions are granted and the LIDAR is attached via USB to the virtual machine, starting this node enables the LIDAR to publish its readings to the /scan topic, using the sensor message type LaserScan. Needless to say, the roscore node must also be active for this node to print the readings.

LIDAR Physical Position: Each LaserScan message published contains every point measured in a single 360-degree rotation, and a new one is sent every time a rotation is finished within a certain maximum amount of time. The data provided by the LIDAR fills the following fields of the LaserScan message: header, angle_min, angle_max, angle_increment, time_increment, scan_time, range_min, range_max, ranges, and intensities.

Although the aim of this project originally including assembling a mount to attach the physical system to the Scott robot, the strict usage of a work-from-home environment prevents such a mount from being constructed. Detailed measurements of Scott were also unavailable, so the design of a mount is also infeasible in this project. Regardless, it is necessary to determine the intended position of the sensor relative to Scott's chassis before processing its readings.

The position of the LIDAR must be suited to its intended function, which is to gather data that enables Scott to make better-informed decisions about where to place its next few steps. The relevant terrain is therefore the ground directly below and in front of Scott. This can be read in multiple ways, but given that this project is using a LIDAR that reads in a single plane, the most efficient method to accomplish this is to align the LIDAR's reading plane with Scott's sagittal plane. This ensures that data can be gathered about the different relevant parts of the terrain simultaneously, without any additional actuation, but limits the data gathered to a 2D cross-section of the terrain.

Based on the necessary data to be sensed, the intended position of the sensor is on the lower front of Scott's torso, where a visibly unused area exists. This enables the LIDAR to take readings from the terrain directly in front of and underneath Scott, which is the most relevant for locomotive information about Scott's current situation and next few steps. This also ensures that the distance between the LIDAR and the terrain is further than the LIDAR's minimum reading distance (15cm), due to Scott's leg length.

It is possible to gather 3D terrain information by adjusting the angle of the LIDAR's plane of readings between readings, and using the known actuation angle to localise the subsequent LIDAR readings relative to the earlier reading planes. For example, the LIDAR mounted as shown in the figure 6 only reads in Scott's sagittal plane (the YZ plane in the diagram). However, it can be actuated to gather 3D terrain information by oscillating about the Z axis; as described, the LIDAR would then be able to take multiple planes of terrain readings, gaining a 3D perspective on the region in front of Scott. This is conceptually similar to the actuated LIDAR arrangement seen on robots such as Boston Dynamics' LS3 [5].

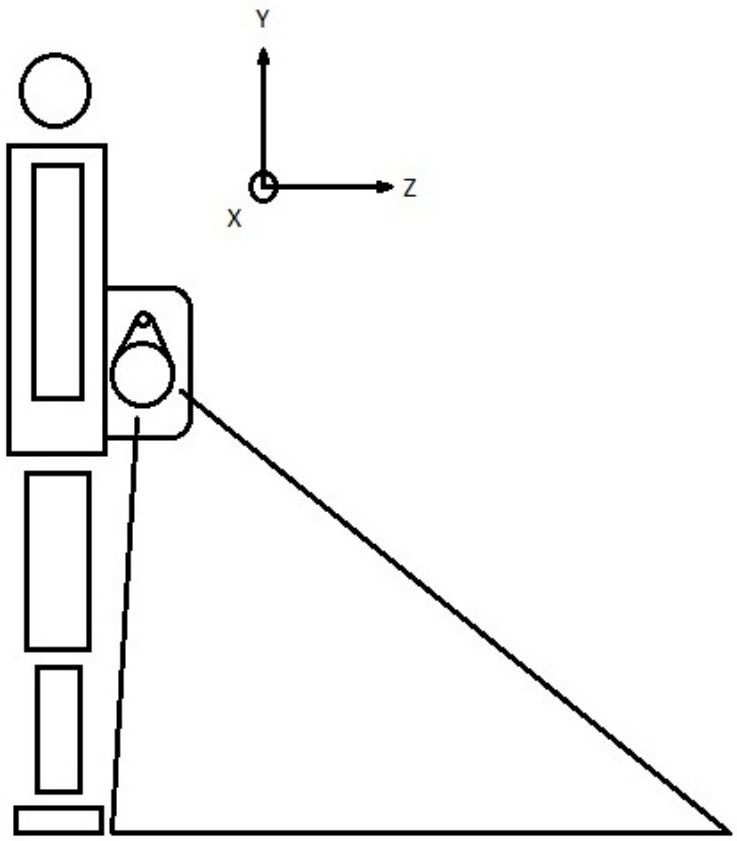


Fig. 6: Diagram of the proposed mounting point and orientation of the LIDAR relative to Scott's chassis

For similar reasons that it is infeasible to construct a mounting for the sensors in a work-from-home environment, however, the implementation of this actuation cannot be carried out during this project. This is because it would require additional mountings to be constructed, as well as a servo that operates at a carefully controlled actuation speed and a known angle, to ensure accurate placement of the different planes of sensed data.

Encoding LIDAR Readings to Matrix Form: In order to process the readings of the LIDAR such that visual analysis can be performed, it was necessary to ensure compatibility between the data and OpenCV computer vision algorithms. This was achieved by noting the similarity between the LIDAR's 2D data and a 2D image which the algorithms are capable of processing.

In order to convert the readings to an image with minimal processing power, it is important to note that a large proportion of the LIDAR's field of vision

provides no data about the terrain. Referring back to figure 6, it is visually obvious that there is no need for data in the 180-degree arc between the positive Y (or +Y) and negative Y (or -Y) axes (via the -Z axis), as this would involve the LIDAR recording points on the Scott robot itself. Similarly, there is no need to use data in the 90-degree arc of the +Y, +Z region, as this would involve sensing data too far away from Scott to be useful (i.e. approximately on the horizon) or sensing objects above Scott instead. As a result, the only necessary data lies in the 90-degree arc between the +Z and -Y axes. This quadrant of the LIDAR's field of view can be reduced to an image of a specified size and resolution, where the Z-axis forms the horizontal top of the image and the Y-axis forms the vertical edge of the image (from a frame of reference centred on the LIDAR)(figure 7).

In order to place each detected point within this image, a simple subscriber node was first created using ROS. This subscriber calls a callback function whenever a message is published to the /scan topic. This callback function, therefore, needs to contain all of the necessary code to process each image (or frame) of the continuously updated /scan topic, enabling it to apply it individually to each update.

The data read from the LIDAR is given in polar form; using the index and distance of the individual points in the 'ranges' array of each LaserScan message and the 'angle increment' information provided in each message, the angle and magnitude of each point's 2D polar vector can be calculated. As a result, converting them into a local X and Y form can be performed by using the following trigonometric relationships, where X and Y are the local horizontal and vertical axes respectively, and magnitude of the polar vector is given by R.

$$x = R * cos(\phi) \tag{1}$$

$$y = R * sin(\phi) \tag{2}$$

A code snippet found in the callback function is given here to illustrate the implementation of these equations (figure 8). Within the callback function, the 'imgmat' matrix was first defined as a 'blank' image matrix of zeros, with the same resolution as the intended final image. Then, the points (contained within the LaserScan message, 'msg') are individually filtered by whether their angle and magnitude fall within the specified bounds; angles must fall within the relevant 90-degree arc in Scott's sagittal plane, and ranges must be of a number that can be encoded into an image. In this case, this means that readings beyond the range of the LIDAR (recorded as 'inf') must be ignored. In short, this portion of the programming filters out points found to lie outside the dimensions of the image, preventing irrelevant coordinates from being calculated and wasting processing power.

Points that pass the filter (and are thereby confirmed as valid to appear on the image) are then encoded into the image's local X and Y coordinates, using the equations outlined earlier. They can then be filtered by any coordinates that exceed the image boundaries, to ensure that they are within the resolution of the image. If this is the case, then each point is individually set to 255 in the

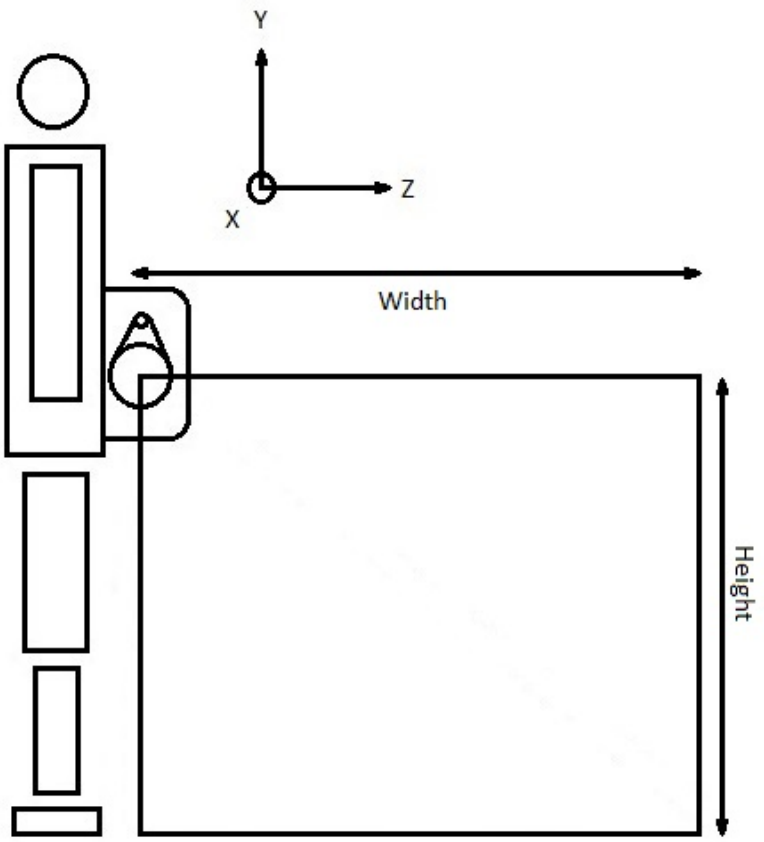


Fig. 7: Diagram of the proposed 2D image of the LIDAR readings, with image height and width labelled

```
imgmat = numpy.zeros((respixel,respixel)).astype(numpy.uint8)
for i in range(0, 359):
    if abs(msg.angle_increment*i) <= abs(max_angle) and abs(msg.angle_increment*i) >= abs(min_angle) and msg.ranges[i] != numpy.inf:
        #for every laserscan message, only proceed if data is between the specified angles and is a non-infinite distance

        #find each relevant point's local X and Y coord relative to the LIDAR. These are in the same axes as the official documentation
        #indicates for the A2 model
        xcoord = msg.ranges[i]*numpy.sin(msg.angle_increment*i)
        ycoord = msg.ranges[i]*numpy.cos(msg.angle_increment*i)
        if abs(xcoord) <= resmetre and abs(ycoord) <= resmetre:
            #only proceed if datum is within image resolution distance

            #Convert coordinates from metres to pixels based on image resolution
            xcoord = (xcoord*(respixel/resmetre)).astype(numpy.uint8)
            ycoord = (ycoord*(respixel/resmetre)).astype(numpy.uint8)

            #set the relevant pixel to 255
            imgmat[ycoord][xcoord] = 255
```

Fig. 8: Snippet of the main loop used to encode the polar LIDAR points into the image's local X and Y axis coordinates

matrix. This is because although black-and-white images typically use a scale of 0 to 1, the value 255 is necessary for use by the OpenCV computer vision package. In short, this part of the programming draws the filtered points onto the image matrix.

Note that there is an official method to encode the coordinates of the points into an image. This is performed via converting the message to the PointCloud2 message type and publishing CameraInfo from the LIDAR, as well as several other steps. However, this uses excessive amounts of processing power compared to the manual, filtered encoding described above.

Reading Terrain Geometry with OpenCV: So far, an image matrix has been created from the data, showing a 2D cross-section of the terrain. In order to process this data in more depth, contour and edge detection functionality can be employed. To do this, the image is first blurred using a 3x3 filter; in addition to minimising the presence of any noise-related readings in the image, this also helps fill in the visual gaps between the extremely sharp individual points detected by the LIDAR. Canny edge detection is then applied, as shown in the snippet below in figure 9. The threshold value is is left at 100. Effectively, this blurs the image points together, then detects the resultant edges.

```
#Process the image and set threshold
imgmatblur = cv.blur(imgmat, (3,3))
threshold = thresh

#Detect edges using Canny
canny_output = cv.Canny(imgmatblur, threshold, threshold * 2)

#Find contours
contours, hierarchy = cv.findContours(canny_output, cv.RETR_EXTERNAL, cv.CHAIN_APPROX_SIMPLE)[-2:]

#Draw contours
drawing = numpy.zeros((canny_output.shape[0], canny_output.shape[1], 3), dtype=numpy.uint8)
for i in range(len(contours)):
    colour = (255, 0, 0)
    cv.drawContours(drawing, contours, i, colour, 2, cv.LINE_8, hierarchy, 0)
```

Fig. 9: Snippet of the code used to detect edges and contours in the imgmat matrix

Contour detection is then applied using the Canny output, forming a basis for more complex object and shape recognition algorithms. In order to minimise processing time per image, the argument 'cv.CHAIN_APPROX_SIMPLE' is passed, greatly reducing the memory used, but also simplifying the results. The contour output is then drawn to a separate copy of the image using a for loop, as well as a specialised OpenCV contour drawing function. The result of this process can be seen below respectively (figures 10_11). Note that the LIDAR's centre of rotation is in the top left of each image in figures 10_11. In summmary, this groups the points and edges into established shapes.

Note that the background of the image is black, and the data points individually are represented by white dots. The contours are drawn over the points in red. The intermediate blurring step and Canny output are not pictured here. From the correct angle and zoom level, it is possible to view a similar portion of the LIDAR's field of view in traditional LaserScan viewing tools, such as Rviz (figure 5).

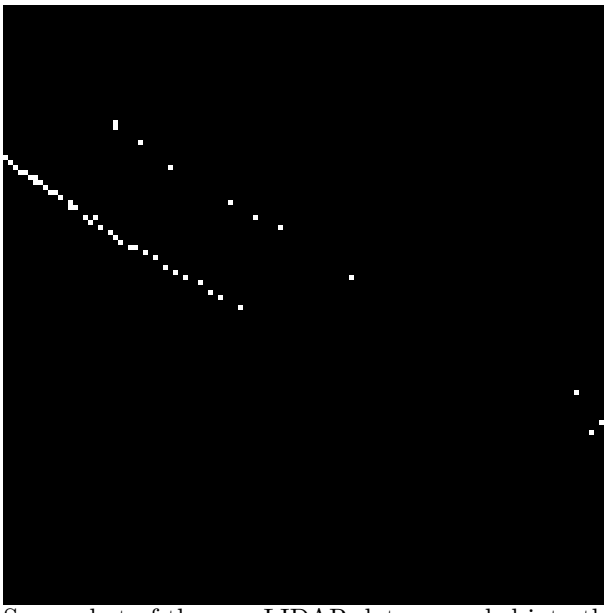


Fig. 10: Screenshot of the raw LIDAR data, encoded into the image

Fig. 11: Screenshot of the detected contours overlaid on the image

More advanced functions that rely on contours (e.g. programs for object categorisation) are not implemented in this report, as they are out of the scope of this project. However, the code is provided as a foundation for potential future projects based on Scott, or using similar sensing systems.

Geometric Terrain Estimation: Modelling the terrain perceived by Scott using the LIDAR is necessary for several reasons. Common models - such as a fitted polynomial equation - can be communicated much more efficiently than a matrix of all the detected terrain points. This could be performed by communicating an array of coefficients ordered by the power of the polynomial, as an example, needing only 3-5 variables per message. Passing the entire image matrix, however, would require 2 variables per coordinate, multiplied by the number of points shown on the image (barring any bespoke methods of compressing the data).

Furthermore, recording the occurrence of various geometric features (e.g. corners and straight lines) can enable Scott to perform useful guesses as to the nature of the terrain. For example, frequent right angles in the data might strongly suggest that Scott is standing in front of a set of stairs. Straight lines might indicate an artificial ramp or floor. Note that vertical corners (such as the edge of a doorframe) would not register, due to the vertical alignment of the LIDAR's plane.

To demonstrate this, a straight line is fitted to the points' coordinates. This can be seen applied to the results in figure 12, compared to the drawn line in figure 13. This is performed using simple mean squared error minimisation, but involves significant numbers of extra lines of code, largely due to the differences in structure between general-purpose numpy arrays and OpenCV's nested contour arrays.

The additional code required involves initialising an empty array (named 'lin' in subsequent snippets) before the loop that encodes X and Y coordinates. As the X and Y coordinates of the valid points are generated, they are appended in pairs to this array; this creates a nested array of arrays, where each entry in the array consists of a smaller array of two integers, representing the X and Y pixel coordinates of each point. Despite being nested, these arrays are not the type used by OpenCV. Effectively, this initialises an array of the correct structure to store point coordinates.

In order for the line fitting function to consider all the points, they must be sorted by their X coordinate value. This is achieved by sorting the nested array of coordinates according to the first integer in each nested array (the X coordinate of each point), which is assigned by naming the X and Y fields in the 'lin' array. However, the line fitting function also requires X and Y coordinates to be recorded in separate vectors; this was achieved using a for loop, which encoded each X and Y value in a separate array while maintaining the sorted order of points. In short, this orders the coordinates in the X axis, and places the points in separate arrays while preserving this order.

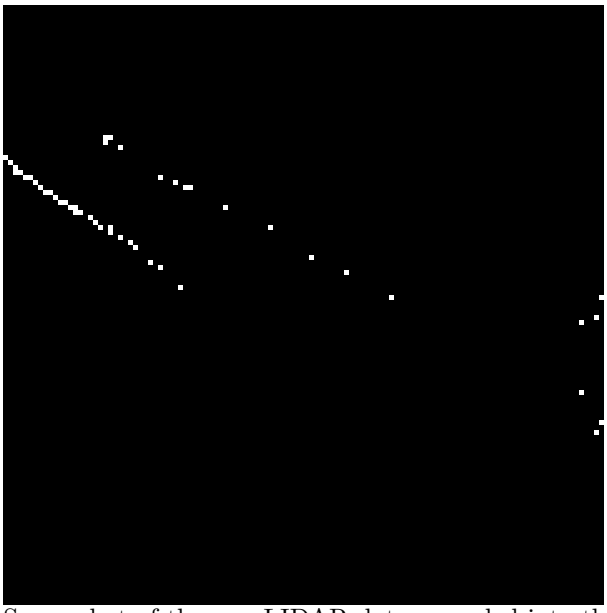


Fig. 12: Screenshot of the raw LIDAR data, encoded into the image

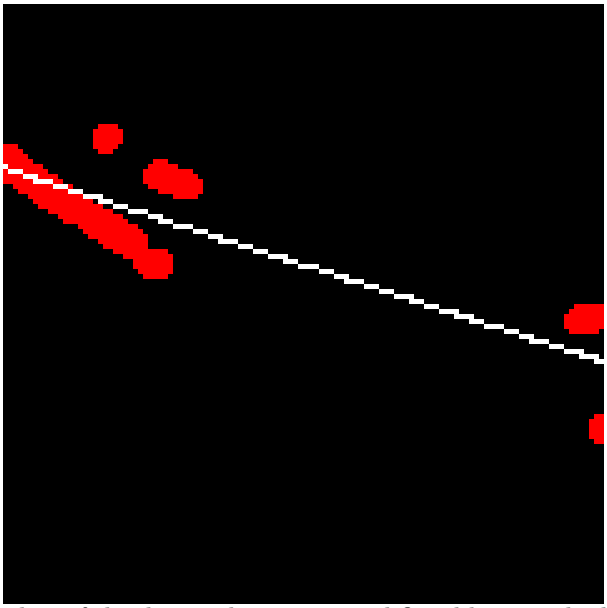


Fig. 13: Screenshot of the detected contours and fitted line overlaid on the image

After checking that there is a valid number of points to sort (for example, 0 points would returns an error in the fitting function, and is invalid as a result), the line fitting function is called. The order of the line's equation is specified as 1, to produce the equation of the straight line seen in figure 13. To summarise, the line is fitted to the ordered points, producing a tuned equation.

In order to ensure that the line drawn on the image is visible, it is important that it does not fill in pixels with gaps between them, as this may cause the line to appear unclear. The desired continuous line is achieved by ensuring that no gaps occur in the pixels shaded. By considering that the largest number of pixels a straight line can occupy on a 2D image is along the diagonal of the square image, this number of pixels (given as N in the equation) can be found as shown below, where W is image width in pixels:

$$N = W\sqrt{2} \tag{3}$$

The total width of the image is separated into this number of X-intervals and rounded to the nearest integer (stored in the 'approxx' array). Each X-interval is then evaluated using the line equation to find the relevant Y-intervals (also rounded, and stored in the 'approxy' array). The computed coordinates are then shaded in on the image matrix. The code for this can be seen in the snippet shown in figure 14. Effectively, this generates a suitably high number of coordinates to clearly illustrate the fitted line in the image.

```
#Sort the lin array with the lintype dtype to give the lintyped array, to ensure compatibility with curve fitting algorithm
lintype = [('xpix', int), ('ypix', int)]
##print('untyped: ' + str(lin))
lintyped = numpy.array(lin, dtype=lintype)
##print('unsorted: ' + str(lintyped))
linsorted = numpy.sort(lintyped, order='xpix')
##print('sorted: ' + str(linsorted))

#Assign the X and Y coords to separate vectors for the curve fitting algorithm
x = linsorted['xpix']
y = linsorted['ypix']
linx = []
liny = []
for i in range(len(x)):
    if y[i] < 120 and y[i] >= 0:
        linx.append(int(x[i]))
        liny.append(int(y[i]))
##print('linx: ' + str(linx))
##print('liny: ' + str(liny))

#Apply the curve fitting algorithm for each recorded point and generate coordinates
if linx == [] or len(linx) < 2:
    return
o = numpy.polyfit(linx, liny, 1)
p = numpy.poly1d(o)
approxx = numpy.linspace(0, int(respixel-1), int((numpy.sqrt(2))*(respixel-1)))
approxy = p(approxx)
approxx = numpy.round(approxx, 0)
approxy = numpy.round(approxy, 0)
##print('line x: ' + str(approxx))
##print('line y: ' + str(approxy))
```

Fig. 14: Snippet of the code used to sort the coordinates and fit a 1D line to them

Terrain Material Estimation: Other forms of modelling used include attempting to interpret the reflectance of the sensed materials. It is possible to identify unique material characteristics using the intensity of the returned beams from the LIDAR [29]. Given that the LIDAR is capable of only reading a 2D projection of a surface, however, the angle of the surface in a 3D environment is unknown. As a result, the simplifying assumption that the incident beam of the LIDAR is normal to the reflective surface at every point will be taken.

Although this assumption's inaccuracy can be accounted for during testing by placing materials at a 90-degree angle, it still has a significant effect at any other angle, and as a result is predicted to be very inaccurate when there is a mismatch between the assumed and real angle (i.e. the LIDAR is sensing an oblique surface). A fully realised implementation of this feature should aim to use semantic knowledge or other sensors to attempt to interpret the 3D terrain angle.

Another assumption is also made: due to the concept of polarised light planes becoming irrelevant when an incident beam is reflected from a normal surface, the s-plane and p-plane complexities of polarised light can be ignored. This is a result of the earler assumption that the incident beam strikes a normal surface.

From preliminary testing, the intensity data returned by the LIDAR is always in the form of a two-digit number between 0 and 100, returned individually for each point. Due to a lack of documentation on this data, therefore, it is presumed that this refers to a percentage measure of R, where 100% refers to the reflected beam having the same intensity as the incident beam of the LIDAR (e.g. when reflected from a mirror), and 0% refers to a negligible amount of reflected light being detected. As a result, this returned ratio will be referred to as the 'relative intensity' of each point.

The sensor used to record the presence of the reflected laser consists of a circular contact area, approximately 6mm in diameter. In order to calculate the measured power intensity, therefore, this is taken as the area of the reflected beam. The aperture of the laser emitter is circular, and approximately 2mm in diameter, which is taken as the dimensions of the incident beam. Using the given laser power of 3 milliwatts with the equation below and finding the area of the aperture, this gives an incident beam intensity of 238.73 watts per metre squared.

$$I = \frac{P}{A} \tag{4}$$

Relating this intensity to that of the reflected beam using the readings of relative intensity can also be performed, conditional upon a valid point being sensed by the LIDAR. Using this reflected intensity with the assumed angle of the object allows the classification of different materials similarly to that performed in [29]. However, this technique is unable to interpret data about the material beyond the specularity/diffusivity of its surface. A full reflective model (accounting for beam and surface angle, as well as beam divergence over distance, the characteristics of the laser emitter, etc) would be necessary in order to determine material parameters that could be compared to known data.

Kinematics: In order to specify the position of Scott's feet directly, inverse kinematics must be used to calculate the corresponding angles at the leg joints that contribute to the foot's position. In terms of functionality, inverse kinematics must be able to find the angles at each leg joint given that the desired position of the end-effector of said leg is known. In order to apply this to the robot, several conceptual simplifications can be applied.

Consider a single leg: other than the hip and ankle joints rotating it about the sagittal axis, the leg joints all move in the same plane. This plane is defined not only by its inclusion of the positions of both the hip and knee joints of the leg, but also by the position of the end effector. As a result, the leg can be thought of as moving within a single plane, which itself is rotated about the sagittal axis by the hip joint. The ankle joint in the sagittal axis can be dismissed as part of the end effector, as its role is to align the foot with the terrain, rather than alter the foot's position. Thereby, the 2D inverse kinematics of the coplanar joints can be combined with the known angle of the hip joint to produce the complete leg position in 3D.

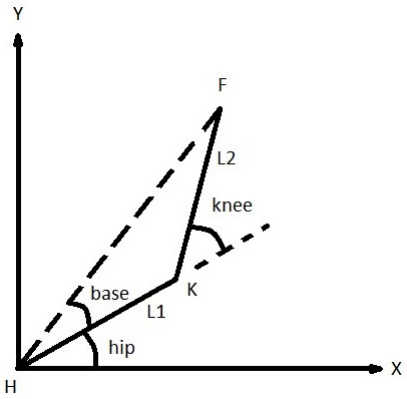


Fig. 15: Simplified diagram of a humanoid biped leg

Figure 15 shows a simplified diagram of the hexapod leg, projected onto the aforementioned plane. By producing a geometric solution for the angles in the diagram (at the hip and knee respectively), it is possible to calculate the inverse kinematics of the leg in 2D, similarly to the derivation carried out in [42]. The angle at the knee can be found using the following equation, where X and Y represent the desired X and Y coordinates of the end-effector in the coordinate frame shown:

$$\theta knee = cos^{-1}(\frac{X^2 + Y^2 - L1^2 - L2^2}{2L1L2}). \tag{5}$$

Note that since cosine is a symmetrical function about 0, there is no ambiguity about the magnitude of the answer produced by this equation. As a result, only one equation is necessary. In order to find the angle at the hip, it is necessary to find the base angle first. As shown on the diagram, this is the angle of a line passing straight through both the hip and the end effector of the robot.

Equations (6)(7) can be used, in order to produce a unique value for the base angle:

$$\theta base = sin^{-1}(\frac{L2sin(\theta knee)}{\sqrt{(X^2+Y^2)}}). \tag{6}$$

$$\theta base = cos^{-1}(\frac{L1+L2cos(\theta knee)}{\sqrt{(X^2+Y^2)}}). \tag{7}$$

These equations are dependent on the value of the angle at the knee, and as a result, that must be worked out as a prerequisite to calculating the angle at the hip. The angle at the hip can then be calculated by Equations (8)(9):

$$\theta hip = sin^{-1}(\frac{Y}{\sqrt{(X^2+Y^2)}}) - \alpha. \tag{8}$$

$$\theta hip = cos^{-1}(\frac{Y}{\sqrt{(X^2+X^2)}}) - \alpha. \tag{9}$$

Thereby, the angles at the hip and knee of the robot can be found from the coordinates of the end effector. In order to apply this to the robot in 3D, however, it is necessary to rotate this plane about the sagittal axis, according to the position of the leg's hip motor which rotates about said vertical axis. Taking this into account mathematically is a matter of using the intended sagittal plane coordinates of the end effector with the sagittal angle of the hip motor. Using the angle and the magnitude of the vector in the local X and Y axes as shown in the diagram above, the end effector position can be immediately expressed as a polar vector from the hip joint.

In order to calculate the sagittal ankle joint, subtracting the angle of the terrain in the frontal plane from the sagittal angle of the hip should provide the value to specify.

This enables all the relevant servo angles to be specified from the end effector position for Scott, or any structurally similar humanoid robot. Consequentially, these angles can be used to perform frame of reference transforms between the base and the LIDAR frame of reference, or vice versa.

If work environment restrictions had not eliminated access to the Scott robot, testing could have been performed in transforming the fitted line in the previous subsections into the frame of reference of Scott's torso. This would have enabled Scott to place its feet using the fitted line as a reference position, or as a saturation limit on the specified foot position.

3.3 Testing & Results

In order to evaluate the performance of this system, it is necessary to investigate the depth of data that can be determined from its readings. This consists of the LIDAR's readings of the 2D geometric structure of sensed terrain and objects, as well as the reflective capability of each sensed point. The intended purpose of

this section is to provide the evidence necessary to enable the assessment of the LIDAR's adequacy for Scott's sensing requirements.

In an ideal situation, Scott would be able to employ its sensing system to extract maximal data about the terrain surrounding it. In a visual sense, this would involve interpreting the environment's colour, texture (among other surface factors relating to the materials and construction methods used), and geometric position. In tandem, these data enable visual information to be interpreted for further processing. Such processing includes fields as broad as object recognition, material classification, mapping, and localisation (via the Monte Carlo method [34]) among others, potentially providing Scott with similar levels of environmental knowledge to humans. However, many of these techniques rely on the availability of the aforementioned data on colour, texture, etc. As such, basing testing on the availability of these data is a valid method to assess the completeness of Scott's sensing capability.

The broad areas of information about the terrain that can be extracted can be roughly arranged into the following key categories:

- The applicability of the readings to identifying different materials
- The applicability of the readings to assessing terrain geometry
- The ability to adjust gait generation programmes according to the processed sensor output

Identifying Terrain Materials: The identification of terrain materials is performed on a point-by-point basis, using the intensity data returned from the LIDAR readings for each point. In order to test this, the relevant 90-degree arc of the LIDAR was faced towards a surface made up of several common household materials. Due to the immobility of the experimental setup on a work-from-home basis, materials more relevant to Scott's intended function of a football robot were unavailable.

In order to ensure that the data returned was reliable, each material was tested under several different environmental lighting conditions. These consisted of three brightness layers: sensing carried out with minimal lighting, sensing carried out with passive, stochastic lighting, and sensing carried out with directly illuminated lighting.

Minimal lighting describes the lowest possible achievable amount of external light interference. By performing the tests at night and turning off other light sources in the testing room, it was possible to produce an environment that reasonably approximated being interference-free in a work-from-home scenario. Passive stochastic lighting describes the lighting that results from indirect sunlight during the day; although this cannot be precisely quantified without laboratory instrumentation, all results were taken during a single sampling session, to maximise lighting consistency between each test. Directly illuminated lighting involved illuminating the material directly with a lamp while it was sensed by the LIDAR, in addition to the stochastic lighting described earlier. For consistency in passive stochastic lighting, the directly illuminated set of results were recorded immediately after the passive stochastic results.The results

are given in the table below, where each value is the returned value of relative intensity provided by the LIDAR.

In order to ensure a fair test, the LIDAR was actuated at the same speed for each test. All results were recorded at approximately the same distance and surface angle from the LIDAR. Each material was held in front of the LIDAR long enough for the value reported at a single point on the sensed material to stabilise at a single value of relative intensity.

Material	Minimal	Stochastic	Illuminated
White Paint	47	47	47
Black Paint	0	0	47
Mirror	47	47	47
Plywood	47	47	47
Carpet	0	47	47

The results show a very strong tendency towards the same value of relative intensity at 47, regardless of the material or light conditions being sensed. This trend continues very consistently between all materials and light conditions, barring occasional anomalies. Notably, examination of the results in detail did not show that this issue was unique to the data point examined; similar results were observed for all points on the material for each test. Although the test conducted was repeated for several materials in order to verify these results, the same data were obtained.

Assessing Geometry: Sensing geometric features via individual points is the main intended function of the LIDAR, with the polar coordinates of each sensed point being returned and updated at a high frequency. Although examples were shown in the methodology section to illustrate the functionality of the code, the images used did not reflect Scott's intended environments.

In order to assess the functionality of the sensor with common indoor terrain geometries, several different types of terrain cross-sections were examined using the LIDAR. These have been split into three broad types: flat floors (figures 16_17), ramps (figures 18_19), and stairs (figures 20_21). Both the recorded results and the contour and line fitting results are shown for each case.

The point readings generally provide a visually clear and low-noise cross-section of each set of terrain, though there is occasional inconsistency in the resolution (or spacing) of the points. The contours largely match the points with visually evident accuracy, though there are points of separation, broadly coinciding with the areas of lower point resolution. Depending on how similar the arrangement of points are to a straight line, the accuracy of the approximation of the straight line varies; for example, the ramp is more accurately reflected by the fitted line than the stairs. In the cases of the stair and ramp data, when the line reaches the top of the image, a line of similar orientation appears in the same place, but near the bottom of the image.

Fig. 16: Screenshot of the flat floor raw LIDAR data, encoded into the image

Fig. 17: Screenshot of the detected flat floor contours and fitted line overlaid on the image

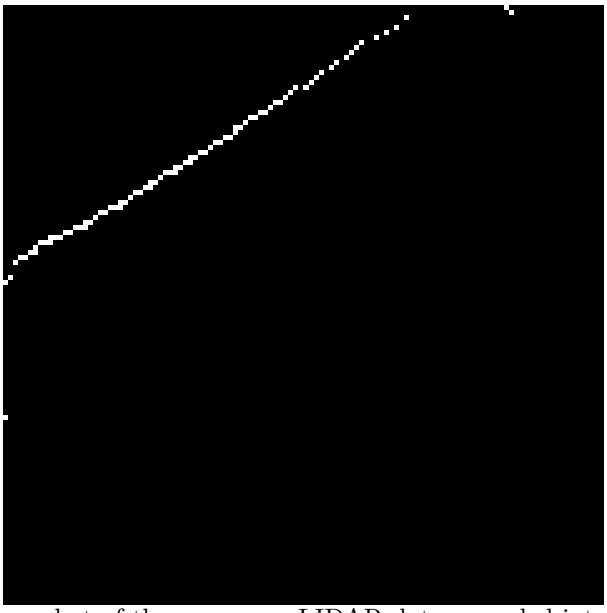


Fig. 18: Screenshot of the ramp raw LIDAR data, encoded into the image

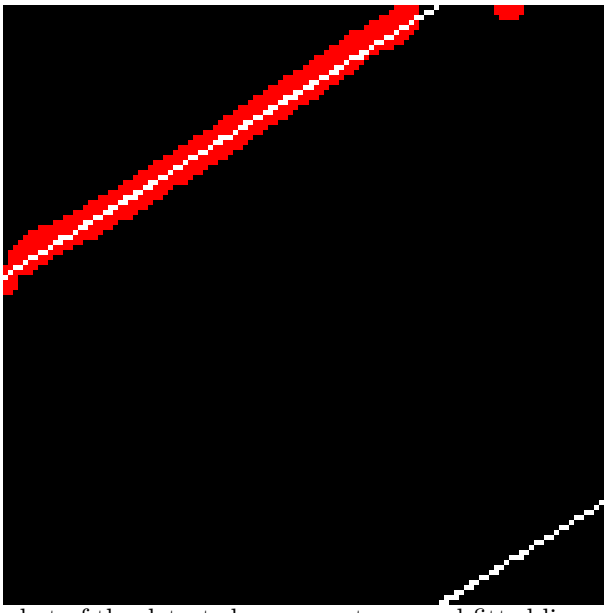


Fig. 19: Screenshot of the detected ramp contours and fitted line overlaid on the image

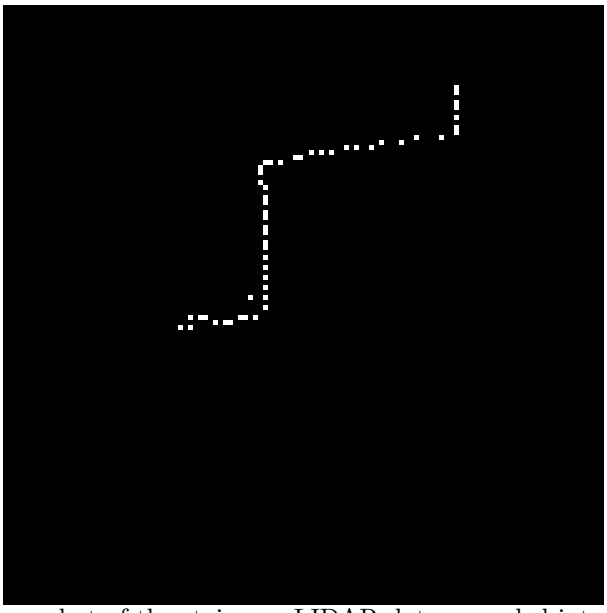


Fig. 20: Screenshot of the stair raw LIDAR data, encoded into the image

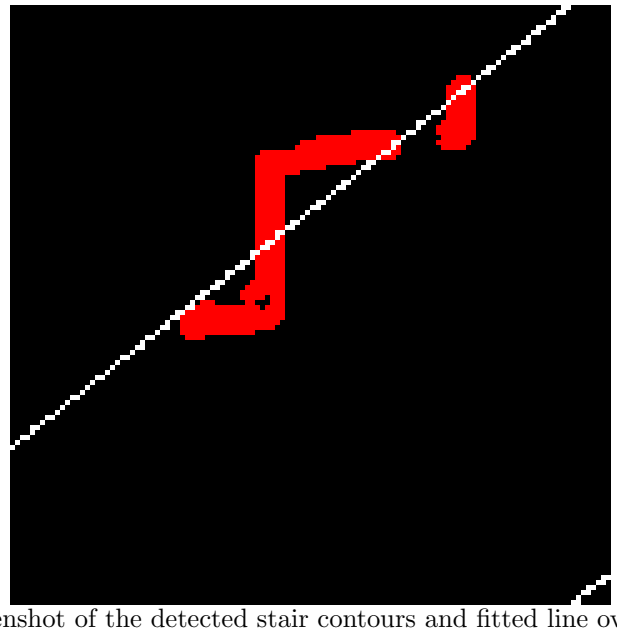


Fig. 21: Screenshot of the detected stair contours and fitted line overlaid on the image

Gait Adjustment: Originally, this section's results were intended to be demonstrated on the Scott robot, with a simple walking gait implemented during the separate restoration project. However, the COVID-19 pandemic has prevented both access to the Scott robot, and the computing power necessary to simulate Scott is unavailable. As a result, the ability for the implemented LIDAR to sense and perform characterisation upon the terrain is tested directly instead.

The expected method to pass the terrain data to the Scott robot is by publishing the fitted line's coefficient to a topic accessed by Scott's other nodes. Given that this line is successfully fitted to the variety of terrains tested (figures 17, 19, and 21), it is therefore possible to successfully publish these data to the relevant gait planning nodes, conditional upon the restoration of Scott's original functionality.

The equation of this line can be attached to the reference frame of Scott's torso in either a simulation, or within Scott's internal programming, though this could not be performed during this project as it is conditional upon Scott's restoration. Attaching this reference frame can be performed using a simple translation, since in its intended implementation, the LIDAR's frame of reference remains fixed relative to Scott's torso. In conjunction with the kinematics that should be present upon Scott's restoration, this enables the coordinates of the LIDAR points (and thereby, the fitted line) to be overlaid on the real position of the terrain.

The fitted line overlaid on the projected position of the real terrain can then be used as an approximation of said terrain, and thereby form the basis of various gait adjustment strategies. This could range from strategies as simple as applying the equation of the line as a saturation limit on Scott's planned foot positions in the sagittal plane, to replacing the fitted line with complex object recognition algorithms that can provide Scott with context about the terrain being traversed. For example, recognising a set of stairs or a flat platform could enable Scott to make more accurate choices regarding identifying the terrain material. However, selecting the details of this strategy fall beyond the scope of this project; proving that the terrain can be modelled for gait adjustment purposes is sufficient.

In order to successfully use the LIDAR to adjust the gait, therefore, this program only needs to supply enough LIDAR data points to support a line being fitted to the points, as a basic form of modelling. In each of the figures shown, this line fitting process is successful. However, when the fitted line reaches the top of the image, a line of similar orientation appears in the same place, but near the bottom of the image.

The visually high amount of inaccuracy in certain cross-sections compared to the fitted line is also notable. To illustrate this, the sum of the mean squared fit errors (also known as the residuals) between the fitted line and the LIDAR points were returned as approximately 36.43 and 61.49 for figures 17 and 19 respectively, while the mean squared fit error value for the 21 was returned as 5205.97.

4 Discussion and Analysis

Terrain Materials The results of the terrain identification via relative intensity remain consistent at the same value, despite changes in material and background lighting. This trend has little basis in the underlying theory of reflection, given the markedly obvious differences in reflective properties between mirrors and carpets, for example. This suggests an error or defect in the LIDAR's implementation as the cause of these unexpectedly consistent results.

Conducting further research into this issue revealed that the driver software of the LIDAR did not return genuine intensity readings for the points read; instead, the values 47 and 0 were directly coded by manufacturers into the device's software. The LIDAR's documentation does not include any options for enabling intensity data to be returned. This explains the lack of variation in the values seen in the readings.

The occasional 0s appear to be related to the ability of the LIDAR to sense points; when a point's results are returned as 0, this often coincides with points being read at a range of 'inf'. Effectively, if the LIDAR cannot detect a point, an intensity value of 0 is returned. This suggests that a combination of the lighting level and the material can affect whether or not the LIDAR is able to sense a point. This interpretation is supported by the literature; as outlined in [29], the material properties of an object's surface can affect the point detection range of the LIDAR.

The data shown also suggests that background lighting conditions affect the ability of the LIDAR to detect points, as well. From the data, lower levels of background lighting appears to reduce the likelihood of the LIDAR detecting a point. The arrangement of 0s relative to each material seems to suggest that darker materials are less frequently detected by the LIDAR. This seems to conflict with the results obtained in the literature, however [6]; it is therefore likely that this trend of lower reading reliablity when sensing darker materials is an experimental error. Alternately, the hypothetical criteria for the manufacturer's software's provision of either 47 or 0 may be incorrect; the LIDAR may have sensed the points in both cases, and returned 0 regardless.

Geometry The points sensed by the LIDAR are generally geometrically accurate to the tested objects. They consistently outline objects with a range of profiles and materials with visual accuracy, as can be seen from the points and contours for figures 17, 19 and 21. Due to the nature of the LIDAR's readings, little to no noise is visually present in the readings, minimising the current need for filtering.

However, the images show reduced point density for surfaces that are further away from the LIDAR, as well as surfaces at a more oblique angle. This is particularly problematic for cases such as the stairs, where the canny edge recognition (and thereby, the contours) are unable to recognise the stairs as a single profile due to the point density being too low on certain parts of the stairs' surface.

This is likely a result of the LIDAR's approach to taking readings; each point is read at an equal angular spacing, giving 360 individual points per LaserScan

returned by the LIDAR. When the range of a detected point from the LIDAR is low, these points appear close together (as in figure 18, at the portion of the ramp nearest the LIDAR in the top-left of the image). However, as the range of the points increases, this angular difference causes the distance between each point to increase. This can be observed by looking at the points in figure 18 that are closer to the edge of the image. As the surface gets further away from the LIDAR, it can be observed that the separation of points along the surface also increases.

This increased separation also occurs when a surface is at an oblique angle relative to the beam between the LIDAR and the points being sensed. A small angular rotation might read a point that has a much farther range than the previously read point, due to the slope of a surface relative to the LIDAR beam. This is most obvious in figure 20, where the long horizontal surface of the stair is read with a low resolution due to its oblique angle relative to the LIDAR beam. This decrease in resolution is likely to be the main cause of the aforementioned disruption of the canny edge detection.

Solutions to this problem may lie in changes to the image processing techniques used. The image is currently blurred with a 3x3 matrix before being processed using canny edge detection; increasing the size of this blur matrix and lowering the detection threshold for canny edge detection may allow points to be registered as a single edge, even with high separation. This may also be accomplished by modifying the parameters of the contour function. However, this technique would also greatly increase the effect of noisy readings on the results. As a result, it may also necessitate the application of additional filters to the results in future.

Gait The fitted line is successfully overlaid onto all of the different terrain geometries that were tested with the LIDAR. Although the line is fitted, the accuracy of said fit varies strongly with how similar the fitted line is to the detected terrain profile. This is evidenced by the residuals for the floor and ramp (which are similar in profile to the fitted line) being 36.43 and 61.49, whereas the stairs (which have a profile very distinct from that of the fitted line) had a residual of 5205.97. These correspond to the inaccuracy of the fit. Therefore, it is clear that the fitted line provides a better fit on terrain of a similar profile, making it very inflexible to terrain corresponding to any differently shaped profiles.

Profile	Residual Error
Floor	36.43
Ramp	61.49
Stairs	5205.97

For terrain of a different profile to the line, the high level of inaccuracy makes taking the line as an approximation of the sensed terrain a dangerous strategy. This is because the terrain could be much higher or lower than the fitted line; if Scott's foot planning node aimed for the fitted line, it might reach the terrain a significant distance away from the line, disrupting the planned gait. This may

even cause Scott to tip over, if the planned leg extension is carried out regardless of the actual terrain.

This strategy of approximating terrain using a fitted line is also very vulnerable to noise; a small collection of points from an unexpected object is capable of very strongly influencing the fit of the polynomial, and thereby Scott's perception of the terrain. However, it is also a relatively simple process to perform, and it greatly minimises the size of the messages that need to be passed to Scott to approximate the terrain. Transformations of a fitted polynomial are also computationally cheaper than transforming each result point individually.

It is suggested that rather than fitting a polynomial, the use of more flexible algorithms is implemented to model the terrain. As a good example, the Ramer–Douglas–Peucker algorithm [7] is usable in OpenCV, and is able to flexibly approximate a shape that has been processed using the OpenCV contours functionality. Flexible algorithms such as these have the notable advantage of being able to model shapes with much higher accuracy than the fitted line, especially when the terrain's profile differs strongly from the fitted line's profile. Approximations based on several simpler shapes rather than the precise points of the sensed terrain also retain the advantage of being able to be transformed into different reference frames with minimal computational load.

In particular, the Ramer–Douglas–Peucker algorithm's method of structuring the approximated shape involves several straight lines, as seen in the example below 22. This could enable further strategies for approximating the surface angle of the terrain at various points, and assessing the artificiality of terrain based on how many straight-line segments are used, for example. In addition to being a more accurate model, this could provide much more information than the current fitted line model does about the terrain Scott is traversing. However, it may also involve a much higher computational load than line fitting, potentially slowing down the effective rate of LIDAR data updates to Scott.

Fig. 22: Example of a shape (black) being processed by the Ramer-Douglas-Peucker algorithm (green) in OpenCV

Other options for flexibly modelling the terrain and extracting information include using SIFT (Scale Invariant Feature Transform) to identify features such as right angles and straight lines, and drawing them onto the image where they are located. Although these solutions are suggested, an exhaustive list is beyond the scope of this project.

4.1 Future Work

It is suggested that in light of the LIDAR's inability to sense intensity data, an alternative method of assessing the material being traversed by Scott is imple-

mented. Examples could include an RGB camera, with a known transform from the LIDAR frame of reference. Overlapping the two areas of perception would enable both geometric and material data to be gathered about the same points simultaneously. Alternately, an RGB-D camera may be able to provide both the geometric and material information necessary, without having to concatenate the readings of two different sensors [2].

Another way to obtain 3D geometric readings would be to actuate the LIDAR such that it also oscillates in the sagittal axis, as seen in Boston Dynamic's LS3 [5]. This could enable the LIDAR to read a wider range of the terrain that Scott is traversing, but potentially requires mapping to also be implemented to make full use of this range of movement.

If a new LIDAR or other method of reading intensity is implemented, it is recommended that a full physical model of the reflected beam is constructed. Fresnel equations using this model's data may be able to return estimates for quantities such as the reflective power coefficient of the material. This could lead to more advanced conclusions about what material Scott may be traversing.

It is suggested to develop a more advanced terrain profile modelling technique. This could use one of the methods mentioned earlier in the discussion (e.g. SIFT feature detection, or the Ramer-Douglas-Peucker algorithm) to provide a more detailed model, or a novel one. However, it must be kept in mind that the method should yield a higher accuracy over a range of terrain profiles compared to the fitted line strategy demonstrated in this project.

It is also suggested that once the separate Scott restoration is complete, that this project is fully integrated with Scott. This involves work that was originally planned to be part of this project, but became unavailable to perform due to the COVID-19 pandemic, such as the design of a mounting for the LIDAR, and testing the gait planning using the fitted line as a saturation limit for Scott's foot position in the sagittal plane.

5 Conclusions

This project successfully implemented a LIDAR to improve Scott's locomotive ability. After evaluating approaches in the literature, a LIDAR was selected for implementation, with the aim of enabling Scott to sense the terrain in front of it. This was selected after an extensive evaluation of different approaches in the literature, as its implementation would enable Scott to assess the terrain in front of it. This LIDAR was implemented with a custom ROS node, on a virtual machine that mimics the microcontroller environment that is to be implemented on Scott.

The LIDAR readings were filtered into an image representing the terrain-adjacent quadrant of the LIDAR's field of view, which was then processed into a 2D image matrix in Scott's sagittal plane via the custom ROS node. Basic artificial vision functions were then performed from the OpenCV library on these readings, as well as curve fitting. These processes both tested the robustness of the data read, and demonstrated the possibility of deeper analysis being per-

formed. Data pertaining to the reflectivity of the terrain sensed was also recorded and interpreted.

The artificial vision functions were successful, with edge detection and contour functionality successfully being applied to the image matrix. In addition, a line was fitted to simplistically model the terrain's geometry. Despite the elementary nature of the functions applied, the data were proved to be robust and accurate enough to make several conclusions about the terrain, pending further processing. However, the nature of these conclusions is entirely geometric, as the reflectivity data proved unable to add any further information. Although this data does not provide information regarding the material of the terrain being traversed, the implementation of the sensor system designed here is more than sufficient to assess the geometry of the terrain, and enable Scott to adjust its gait as a result.

References

1. History of the Humanoids: P1-P2-P3 (1993 - 1997) (July 2008), http://world.honda.com/ASIMO/history/p1_p2_p3.html, available at https://web.archive.org/web/20080708035640/http://world.honda.com/ASIMO/history/p1_p2_p3.html,,(accessed Jun 01, 2020)
2. Belter, D., Łabecki, P., Fankhauser, P., Siegwart, R.: Rgb–d terrain perception and dense mapping for legged robots. International Journal of Applied Mathematics and Computer Science **26**(1), 81–97 (2016)
3. Belter, D., Łabecki, P., Skrzypczyński, P.: Adaptive motion planning for autonomous rough terrain traversal with a walking robot. Journal of Field Robotics **33**(3), 337–370 (2016)
4. Belter, D., Skrzypczyński, P.: Rough terrain mapping and classification for foothold selection in a walking robot. Journal of Field Robotics **28**(4), 497–528 (2011)
5. BostonDynamics: Legacy robots — boston dynamics (2012), https://www.bostondynamics.com/legacy
6. Dong, H., Barfoot, T.D.: Lighting-invariant visual odometry using lidar intensity imagery and pose interpolation. In: Field and Service Robotics. pp. 327–342. Springer (2014)
7. Douglas, D.H., Peucker, T.K.: Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. Cartographica: the international journal for geographic information and geovisualization **10**(2), 112–122 (1973)
8. Durrant-Whyte, H., Bailey, T.: Simultaneous localization and mapping: part i. IEEE robotics & automation magazine **13**(2), 99–110 (2006)
9. Eitel, A., Springenberg, J.T., Spinello, L., Riedmiller, M., Burgard, W.: Multimodal deep learning for robust rgb-d object recognition. In: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 681–687. IEEE (2015)
10. Ekvall, S., Jensfelt, P., Kragic, D.: Integrating active mobile robot object recognition and slam in natural environments. In: 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 5792–5797. IEEE (2006)
11. Englsberger, J., Koolen, T., Bertrand, S., Pratt, J., Ott, C., Albu-Schäffer, A.: Trajectory generation for continuous leg forces during double support and heel-to-toe shift based on divergent component of motion. In: 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 4022–4029. IEEE (2014)
12. Englsberger, J., Ott, C., Albu-Schäffer, A.: Three-dimensional bipedal walking control using divergent component of motion. In: 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 2600–2607. IEEE (2013)
13. Englsberger, J., Ott, C., Albu-Schäffer, A.: Three-dimensional bipedal walking control based on divergent component of motion. Ieee transactions on robotics **31**(2), 355–368 (2015)
14. Feng, S., Xinjilefu, X., Huang, W., Atkeson, C.G.: 3d walking based on online optimization. In: 2013 13th IEEE-RAS International Conference on Humanoid Robots (Humanoids). pp. 21–27. IEEE (2013)
15. Ficht, G., Farazi, H., Brandenburger, A., Rodriguez, D., Pavlichenko, D., Allgeuer, P., Hosseini, M., Behnke, S.: Nimbro-op2x: Adult-sized open-source 3d printed humanoid robot. In: 2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids). pp. 1–9. IEEE (2018)
16. Fisher, M.: Matt's webcorner - kinect sensor programming (2014), https://graphics.stanford.edu/~mdfisher/Kinect.html

17. Georgoulis, S., Vanweddingen, V., Proesmans, M., Van Gool, L.: Material classification under natural illumination using reflectance maps. In: 2017 IEEE Winter Conference on Applications of Computer Vision (WACV). pp. 244–253. IEEE (2017)
18. Geyer, H., Herr, H.: A muscle-reflex model that encodes principles of legged mechanics produces human walking dynamics and muscle activities. IEEE Transactions on neural systems and rehabilitation engineering **18**(3), 263–273 (2010)
19. Geyer, H., Seyfarth, A., Blickhan, R.: Compliant leg behaviour explains basic dynamics of walking and running. Proceedings of the Royal Society B: Biological Sciences **273**(1603), 2861–2867 (2006)
20. Joe, H.M., Oh, J.H.: Balance recovery through model predictive control based on capture point dynamics for biped walking robot. Robotics and Autonomous Systems **105**, 1–10 (2018)
21. Kaneko, K., Harada, K., Kanehiro, F., Miyamori, G., Akachi, K.: Humanoid robot hrp-3. In: 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 2471–2478. IEEE (2008)
22. Koolen, T., De Boer, T., Rebula, J., Goswami, A., Pratt, J.: Capturability-based analysis and control of legged locomotion, part 1: Theory and application to three simple gait models. The international journal of robotics research **31**(9), 1094–1113 (2012)
23. Kryczka, P., Hashimoto, K., Kondo, H., Omer, A., Lim, H.o., Takanishi, A.: Stretched knee walking with novel inverse kinematics for humanoid robots. In: 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 3221–3226. IEEE (2011)
24. Liu, B., Yang, Y., Shuo, J.: Review of advances in lidar detection and 3d imaging. Guangdian Gongcheng/Opto-Electronic Engineering **46**, 2019–2065 (07 2019). https://doi.org/10.12086/oee.2019.190167
25. Liu, Y., Wensing, P.M., Orin, D.E., Zheng, Y.F.: Trajectory generation for dynamic walking in a humanoid over uneven terrain using a 3d-actuated dual-slip model. In: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 374–380. IEEE (2015)
26. Liu, Y., Wensing, P.M., Schmiedeler, J.P., Orin, D.E.: Terrain-blind humanoid walking based on a 3-d actuated dual-slip model. IEEE Robotics and Automation Letters **1**(2), 1073–1080 (2016)
27. Lowe, D.G.: Object recognition from local scale-invariant features. In: Proceedings of the seventh IEEE international conference on computer vision. vol. 2, pp. 1150–1157. Ieee (1999)
28. Marques, T.P., Hamano, F.: Autonomous robot for mapping using ultrasonic sensors. In: 2017 IEEE Green Energy and Smart Systems Conference (IGESSC). pp. 1–6. IEEE (2017)
29. Muckenhuber, S., Holzer, H., Bockaj, Z.: Automotive lidar modelling approach based on material properties and lidar capabilities. Sensors **20**(11), 3309 (2020)
30. Pandey, A.K., Gelin, R.: A mass-produced sociable humanoid robot: Pepper: The first machine of its kind. IEEE Robotics & Automation Magazine **25**(3), 40–48 (2018)
31. Rao, A., Elara, M.R., Elangovan, K.: Constrained vph+: a local path planning algorithm for a bio-inspired crawling robot with customized ultrasonic scanning sensor. Robotics and biomimetics **3**(1), 1–13 (2016)
32. Rezazadeh, S., Hubicki, C., Jones, M., Peekema, A., Van Why, J., Abate, A., Hurst, J.: Spring-mass walking with atrias in 3d: Robust gait control spanning

zero to 4.3 kph on a heavily underactuated bipedal robot. In: ASME 2015 dynamic systems and control conference. American Society of Mechanical Engineers Digital Collection (2015)
33. Ring, J.: The laser in astronomy. New Scientist p. 672–673 (Jun 1963)
34. ROS.org: amcl - ros wiki (2020), https://wiki.ros.org/amcl
35. Ruiz-Sarmiento, J.R., Galindo, C., González-Jiménez, J., et al.: Mobile robot object recognition through the synergy of probabilistic graphical models and semantic knowledge (2014)
36. Slamtec: Slamtec product documents download and technical support (2019), https://www.slamtec.com/en/Support
37. Sock, J., Kim, J., Min, J., Kwak, K.: Probabilistic traversability map generation using 3d-lidar and camera. In: 2016 IEEE International Conference on Robotics and Automation (ICRA). pp. 5631–5637. IEEE (2016)
38. Spröwitz, A., Tuleu, A., Vespignani, M., Ajallooeian, M., Badri, E., Ijspeert, A.J.: Towards dynamic trot gait locomotion: Design, control, and experiments with cheetah-cub, a compliant quadruped robot. The International Journal of Robotics Research **32**(8), 932–950 (2013)
39. Stonier, D.: kobuki - ros wiki (2016), https://wiki.ros.org/kobuki
40. Taigman, Y., Yang, M., Ranzato, M., Wolf, L.: Deepface: Closing the gap to human-level performance in face verification. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1701–1708 (2014)
41. Theimer, W.M., Mallot, H.A.: Phase-based binocular vergence control and depth reconstruction using active vision. Computer Vision and Image Understanding **60**(3), 343–358 (1994)
42. Turner, P.: Mathematics required for robotic motion (2017), https://web.archive.org/web/20170511015320/http://www.tribotix.info/Articles&Tutorials/MathsforRobotics/Mathematics%20required%20for%20Robotic%20Motion.pdf, available at https://web.archive.org,(accessed Jun 04, 2020)
43. Weerdesteyn, V., Nienhuis, B., Hampsink, B.M., Duysens, J.: Gait adjustments in response to an obstacle are faster than voluntary reactions. Human Movement Science **23**(3-4), 351–363 (2004)
44. Wieber, P.B.: Trajectory free linear model predictive control for stable walking in the presence of strong perturbations. In: 2006 6th IEEE-RAS International Conference on Humanoid Robots. pp. 137–142. IEEE (2006)

List of Figures

A Appendix

Motors

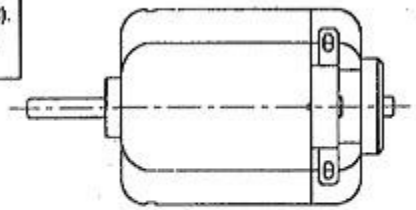
Order code	Manufacturer code	Description
37-0140	n/a	3V 13100 RPM DC MOTOR

Sales: 01206 751166 **Technical: 01206 835555** **Fax: 01206 7551188**
Sales@rapidelec.co.uk **Tech@rapidelec.co.uk** **www.rapidelectronics.co.uk**

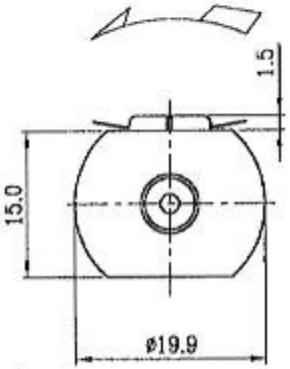
REMARKS:

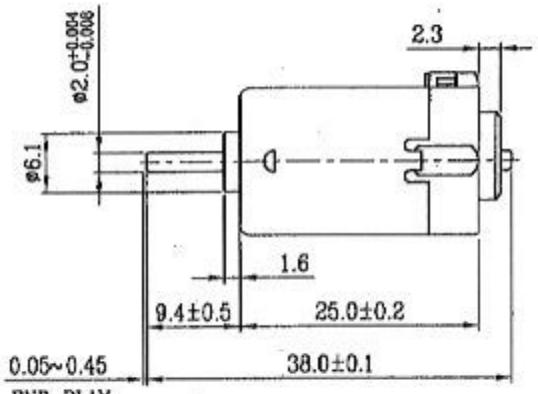
1) WHEN PRESS FITTING A GEAR,PULLEY OR ANYTHING ALIKE ONTO MOTOR OUTPUT SHAFT, DO PUSH THE OTHER SHAFT END VERTICALLY, DON'T PUSH THE REAR BRACKET
2) SOLDERING TERPERATURE AND TIME MUST BE 360°C(3SEC) OR 400°C(2SEC).
3) MEASURING THE DIMENSION OF THE SHAFT EXTENSION WHEN THE OTHER END OF THE SHAFT IS AGAINST THE REAR BRACKET END.

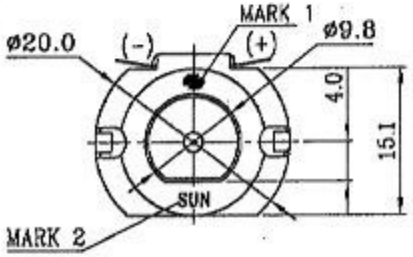
ISSUE NO.



DIRECTION OF ROTATION







2005-01-06

REV.	MAS	DES'D	APP'D	DATE

CONFIRMED & ACCEPTED BY CUSTOMER
Company, Chop & Signature
DATE:

MODEL NO.	S10-2270-38Z-02		
SSO#:	57882	REV	0

TOLERANCE EXCEPT AS NOTED		
1PL DECIMAL	2PL DECIMAL	ANGLE
±0.15	±0.1	±2°

DES'D 4/-05
CHK'D
APP'D 4/2/05
DO NOT SCALE DRAWING
CUSTOMER NO.

DNG-3A-3.0.

MOTOR OUTGOING SPECIFICATION

馬達出廠規格

Model No. :S10-2270-38Z-02 SSO#:57882

REV : 0

1. **Motor Type:** Small PMDC motor
馬達類型 微型直流馬達

2. **Standard Operating Conditions:**
標準工作條件

2-1	Rated Voltage(額定電壓):	3.0V DC CONSTANT between motor terminals(端子標壓3.0V).
2-2	Direction of Rotation(旋轉方向):	CCW when viewed from output shaft side(從輸出軸端看CCW).
2-3	Operation Temperature Range(工作溫度范圍):	Between 0℃ and 50℃(在0℃ ~50℃)
2-4	Operation Humidity Range(工作濕度范圍):	Between 5% and 95%(在5%~95%)

3. **Measuring Conditions:**
測試條件

3-1	Motor Position(馬達放置狀態):	Motor to be held with shaft horizontally(馬達測試時水平放置).
3-2	Power Supply(電源):	Regulated power supply which assures unquestionable measurement(確保測試沒問題的穩壓電源).
3-3	Environment Temperature and Humidity(環境溫度及濕度):	The test is made in principle at a temperature between 15℃ and 35℃ and at a relative humidity 75%(一般在15℃~35℃及相對濕度75%的環境下測試).

4. **Electrical Characteristics (at initial stage after 30 seconds run-in):**
電氣特性(起動30秒后的初始階段)

4-1	No Load Current(無負荷電流):	0.34A max.
4-2	No Load Speed(無負荷轉速):	16400±2460rpm.
4-3	No Load Starting Voltage(無負荷起動電壓):	0.6Vmax
4-4	Rated Load(額定負載):	8.0 g-cm.
4-5	Rated Load Current(額定負載電流):	1.07A max.
4-6	Rated Load Speed(額定負載轉速):	13100±1570rpm.
4-7	Fan Load Current(風葉負載電流):	1.28A max.
4-8	Fan Load Speed(風葉負載轉速):	12900±1550rpm.
4-9	Fan Load(風葉負載):	I-shaped fan load with size30x15.0mm(approximately equivalent to9.4g-cm). 30X15.0mm一字形風葉(大約等于9.4g-cm)
4-10	Motor Characteristic Curve(馬達性能曲線):	See Motor Performance Curves And Characteristics(見馬達特性曲線圖).

5. **Mechanical Characteristics:**
機械特性

5-1	Dimensions(外形尺寸):	As Per Drawing attached(所附圖面57882)
5-2	End Play(歲位):	0.05~0.45 mm
5-3	Runout of Shaft Extension(輸出端跳動):	0.05 mm max.

CONFIRMED & ACCEPTED BY CUSTOMER
客戶接受&確認
COMPANY, CHOP & SIGNATURE
蓋章&簽名
DATE : 2005-01-06

PREPARED BY: CHECKED BY: APPROVED BY: 2005/1/6

Part List

Model No. : S10-2270-38Z-02 SSO#:57882

REV. : 0

---

1. Plastic Cap

a)Manufacturer : Taita chemical co.,Ltd
b) Material : Acrylonitrile butadiene styrene.

2. Body Casing

a) Material : Electrolyti zinc plated steel sheet SECC.

3. Leaf Spring

a) Material : Phosphor bronze.

4. Lamination

a) Material : Laminated silicon steel.

5. Armature Magnet Wire

a) Manufacturer : Tai-I Electrical Wire & Cable Co., Ltd.
b) Material : Polyurethane coated copper wire, UEW.
c) UL File No. : E85640 (S)
OR
a) Manufacturer : Xin Long Magnet Wire Co.,Ltd.
b) Material : Polyurethane coated copper wire, UEW.
c) UL File No. : E171082 (S)

6. Magnet

a) Material : Isotropic ferrite magnet.

7. Commutator

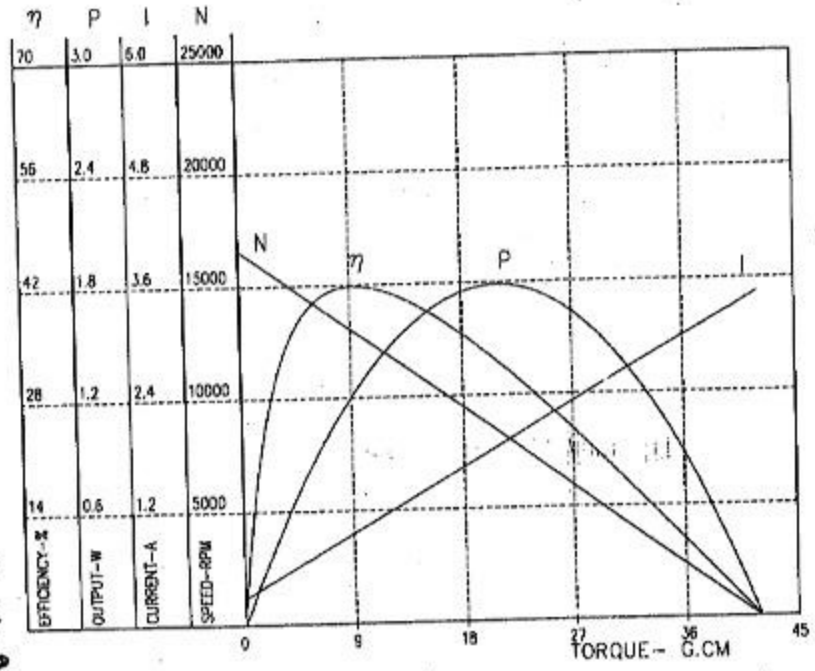
a) Material : Three copper bars insulated from shaft by glass reinforced 66 nylon.

2005-01-06

PREPARED BY: CHECKED BY: / APPROVED BY:

MOTOR PERFORMANCE CURVES AND CHARACTERISTICS.
馬達特性曲線

MODEL: *S10-2270-38Z-02*　　VOLTAGE: *3.0 V*　　SSO#: 57882



PERFORMANECE(性能)

AT NO LOAD(無負荷時)

SPEED(轉速)	=16600	RPM
CURRENT(電流)	=0.280	AMP

AT STALL EXTRAPOLATION(堵轉時)

TORQUE(力矩)	=42.0	G.CM
CURRENT(電流)	=3.450	AMP.

AT MAXIMUM EFFICIENCY(最大效率時)

EFFICIENCY(效率)	= 41.85	%
SPEED(轉速)	= 12919	RPM
TORQUE(力矩)	= 9.3	G.CM
CURRENT(電流)	=0.983	AMP.
OUTPUT(功率)	= 1.234	WATTS

AT MAXIMUM OUTPUT(最大輸出功率時)

SPEED(轉速)	= 8300	RPM
TORQUE(力矩)	=21.0	G.CM
CURRENT(電流)	=1.725	AMP.
OUTPUT(功率)	=1.788	WATTS

THE CURVES REPRESENT THE THEORETICAL PERFORMANCE OF THE SAMPLE ONLY.
特性曲線僅代表樣品的理論特性值

06-10-2005

PREPARED BY: [illegible]　REVIEWED BY: /　APPROVED BY: 4/2/05

FILENAME: UNNAMED.PMG