Andreas Torgrimsen Eggesvik

Robot for automated seaweed deployment and harvesting

Masteroppgave i Kybernetikk og robotikk
Veileder: Morten Omholt Alver, Torfinn Solvang
Juli 2019

Masteroppgave

NTNU
Norges teknisk-naturvitenskapelige universitet
Fakultet for informasjonsteknologi og elektroteknikk
Institutt for teknisk kybernetikk

NTNU
Kunnskap for en bedre verden

Andreas Torgrimsen Eggesvik

Robot for automated seaweed deployment and harvesting

Masteroppgave i Kybernetikk og robotikk
Veileder: Morten Omholt Alver, Torfinn Solvang
Juli 2019

Norges teknisk-naturvitenskapelige universitet
Fakultet for informasjonsteknologi og elektroteknikk
Institutt for teknisk kybernetikk

NTNU
Kunnskap for en bedre verden

Abstract

On a global scale, the total market for kelp is growing rapidly. In order to meet the demand and keep cost of production down, there is a need for technology development and automation in the kelp cultivation and harvest process.

A concept for a modular cultivation plant designed to enable a high degree of automation named SPOKe has previously been proposed. This concept consists of a ring-shaped plant and a manipulator capable of both attaching kelp seeds to the plant and harvesting the full grown kelp. The manipulator in this concept can be attached or detached to the plant, enabling a single manipulator to operate several plants.

The work of this report includes describing the desired behaviour of the manipulator. Based on this desired behaviour, a control system is developed. This control system creates a continuous desired path for the manipulator. In order for the controller to be capable of operating the manipulator, the propulsion and sensory systems of the manipulator was specified.

A physical prototype of the SPOKe concept was developed in order to test the controller and the viability of the propulsion and sensory systems, as well as the viability of the concept in general. In order to create a physical prototype capable of performing the desired tasks, several specific design decisions had to be made. This prototype includes a computer, sensors and motors and a touch screen. The control system can be initialised, started and stopped by the user though a touch screen.

The prototype proved to be able to physically move the tool effortlessly in all desired positions, validating the plausibility of the design. The trajectories created by the controller are all as desired. Due to time constraints, only one of the two positional sensors was tested. The controllers ability to control the motor, and make the manipulator follow the trajectories could therefore not be tested.

Sammendrag

Det globale markedet for sukkertare er raskt voksende. For å møte markedets behov og samtidig holde produksjonskostnader nede, er det nødvendig å utvikle teknologi som automatiserer prosessen med å så og høste denne taren.

Et design for taredyrkningsanlegg kalt SPOKe er tidligere blitt foreslått. Dette designet er laget med tanke på å være skalerbart og å automatisere dyrkeprosessen. Dyrkningsanlegget består av en ringformet struktur hvor man kan feste tau som eiker på et sykkelhjul. På dette tauet er det sådd sukkertare. En robotmanipulator har som oppgave å spenne opp tauet og høste tau med ferdiggrodd tare. Denne roboten er løs fra ringstrukturen og kan løsnes og festes etter behov. På denne måten kan én robot så og høste tare fra flere dyrkningsanlegg.

I denne rapporten beskrives den ønskede oppførselen til denne roboten. Deretter utvikles et kontrollsystem som lager en kontinuerlig bane som roboten skal følge for å holde ønsket oppførsel. For at kontrolleren skal være i stand til å kontrollere bevegelsen til roboten spesifiseres det hvilke sensorer og pådragsorgan roboten skal ha.

For å teste kontrollsystemet og funksjonaliteten til designet, utvikles det en fysisk prototype av SPOKe konseptet. For å kunne lage en fysisk prototype var det nødvendig å ta noen designvalg for robotens utforming. Prototypen som ble utviklet består av en ringstruktur og en robot som styres av en datamaskin, og har motorer, sensorer og en trykkskjerm. Kontrollsystemet kan initialiseres, startes og stoppes fra trykkskjermen.

Prototypen som ble utviklet var fysisk i stand til å bevege seg på alle ønskede måter. Banene laget av kontrolleren var alle som ønsket. Grunnet tidsbegrensning ble kun én av de to posisjonssensorene testet. Det var derfor heller ikke mulig å teste om kontrolleren var i stand til å sørge for at roboten fulgte den ønskede banen.

Preface

This thesis concludes my M.Sc degree in Cybernetics and Robotics at Department of Engineering Cybernetics at Norwegian University of Science and Technology (NTNU).

The thesis is based on previous work conducted by Emil Bale for the research organisation SINTEF, in introducing the SPOKe concept. The work done for SINTEF by Ingrid Wiik has also been valuable during the process of writing this report. This thesis is a continuation of the work done in my specialisation project. some sections from that report are recited here.

I would like to thank my supervisor Morten Omholt Alver both for the interest shown in my work, and for being supporting through the last year. I would also like to thank my co-supervisor Torfinn Solvang at SINTEF. The practical know-how and drive to make this project happen has been very helpful throughout the last year. Having two supervisors enthusiastic and engaged in the work I've done has been motivating and inspiring.

The creation of the digital model for the prototype as well as the physical implementation of it including mounting of the electronics has been done by Daniel Bogen working at the workshop of the Department of Engineering Cybernetics. When I five months ago described the concept of the prototype with limited specifics relating to implementation, I had never believed a prototype as well built as this would be the end result. With my limited experience of evaluating physical practicality of solutions, it has been very helpful be able to present ideas and get feedback on the feasibility of my proposals.

The cost of materials for the prototype was funded by SINTEF through the MACROSEA project and the labor cost for the prototype was funded by the Department of Engineering Cybernetics.

Working with electronics for the prototype has been a pleasure thanks to the student workshop “Omega Verksted”. The student workshop with its variety of tools, electrical components as well as curious and helpful students has been a good place to get to know. The H-bridge used to control the motors is developed at Omega Verksted by students.

Working on this master thesis has been easier thanks to my fellow students in the office, who have been engaged and committed to both fun and contestation. A special thanks goes to Eva Kristiansen, for being both a good friend and sparring partner.

Andreas Torgrimsen Eggesvik
Trondheim, June 2019

Chapter 1

Introduction

1.1 Motivation

This motivation is a direct citation from the project report [1].

The global market for kelp grows rapidly. In 2008, the global production was at 16.8 million tonnes, with a value of 7.4 million USD [2]. The global production of farmed aquatic plants reached 27.3 million tonnes in 2014 [3]. This development is expected to continue, and it is expected that the potential for micro-algae production in Norway can reach 20 million tonnes, for a value of 40 Billion NOK by 2050 [4].

One reason for the big market potential for kelp, is the need for new ways to produce sustainable food to feed the growing global population. The increase in world population will require an increase of 69% crop calories in 2050 compared to 2006 [5]. Kelp can be a food-source both directly and indirectly as feed source for both land and sea-based livestock.

Another reason for the potential in production of kelp is the transition in the energy sector from being heavily fossil-fuel dependant, to being based on renewable energy. This transition needs to be quick, in order to limit the global effects of climate change. The best case scenario by IPCC estimates that the proportion of energy sourcing from bio-fuel will increase from 3% today to 27% in 2050 [4]. Bio-fuel based on energy sourcing from algal production has potential to be part of this increase, and in opposition to many other bio-fuel based solution, it does not require using resources currently utilised for food production. The three prerequisites for having a large micro-algae production in Norway by 2050 is according to a SINTEF report [4] listed as

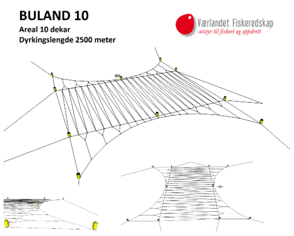
- The exploitation of macroalgae for applications requiring high production volumes, such as bioenergy and feedstuffs, must be maximised.

- Universal acceptance of a further expansion of the areas of coastal waters occupied by mobile facilities and the development of space-efficient cultivation technologies.

- Know-how and technology are developed which enable the full exploitation of harvested biomass (the bio-refinery concept).

Three examples of state of the art technology for kelp cultivation is briefly described in the report by Emil Scott Bale [6]. The three different designs from Seaweed Energy solutions, Seaweed AS and AT-Sea/SIOEN are illustrated in Figure 1.1. Common for all these technologies is labour extensive harvesting methods and low degree of automation.

(a) Seaweed Energy solutions production site [7].



(b) Seaweed AS cultivation system Buland 10 [8].

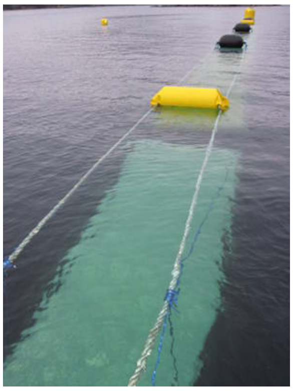

(c) Algaesheet from At-Sea/SIOEN [9].

Figure 1.1: Current Kelp cultivation technologies.

SINTEF is a Norwegian research organisation with projects in a large variety of research areas. One of these projects is named "MACROSEA". The goal of the MACROSEA project is "successful and predictable production of high quality biomass thereby making significant steps towards industrial macroalgae cultivation in Norway" [10]. As a part of this project, a concept for a macroalgae cultivation plant has been developet. The concept developed by Bale was named Standardized Production of Kelp, SPOKe. In the SPOKe concept, kelp is cultivated on a one-dimensional substrate (rope), which is attached to a structure made up of two rings illustrated in Figure 1.2. The goal of MACROSEA SPOKe was the possibility of automation and scalability. The carrier rope attachment points and tool for this concept were developed further in the report "Concept development of details for macroalgae cultivation" [11]. A mathematical model for the movement in one dimension of the manipulator submerged in water has been developed [1]. Considerations for propulsion and sensory system has also previously been made [1]. This report will focus on the automation aspect of the SPOKe concept, and implementation considerations related to automation.

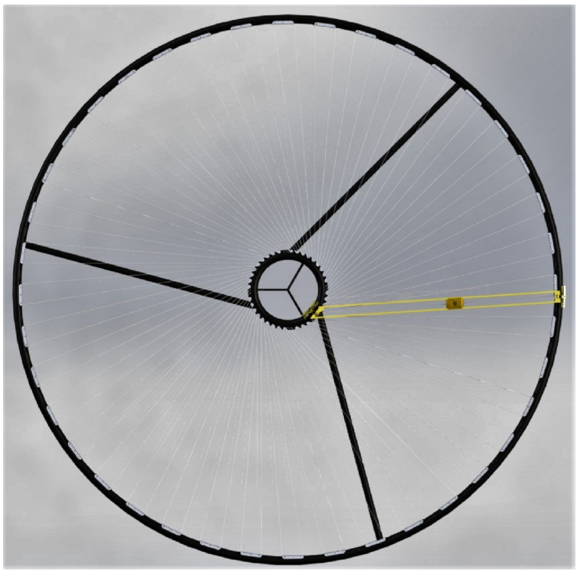


Figure 1.2: Spoke ring structure and manipulator [6]

1.2 Basic Theory

1.2.1 Workspace and operational space

All manipulators are designed to perform operations. In order to do this, the manipulator is equipped with a tool suitable for the operation performed. The pose of this tool can be described by its position and orientation. In three dimensions, this results in two three-dimensional vectors, one for position and one for orientation. This results in a pose vector $\mathbf{x}$, with $\mathbf{x} \in \mathcal{R}^{\mathbf{6}}$. This vector is named operational space [12].

Joints on a manipulator will have geometrical constraints limiting the space in which they can move. As a result of this, the tool can only operate in a subset of the operational space, namely the workspace. The workspace consists of all physically possible poses of the manipulator tool.

1.2.2 Joint variable, Joint space configuration space

Each joint in a robot manipulator has a variable configuration. The parameter used to describe the configuration of each joint is names a “joint variable”. For linear actuator joints, this variable describes the extension of the joint. For rotary joints, this variable describes the rotation of the joint. The set of all joint parameters are named the joint space. This causes the dimensions of the joint space to be equal the number of joints of the manipulator.

The configuration of a robot is a description of its geometry. The space in which the manipulator can operate is named configuration space. Assuming the individual links of the manipulator are rigid, the configuration space can be described by the joint variables. This property causes terms “joint space” and ”configuration space” to describe the same space.

1.2.3 Inverse kinematics

When describing a desired pose of the tool of a robot, it is intuitive to describe it in configuration space $\boldsymbol{x} \in \mathcal{R}^6$. In order to control the robot movement, positions in configuration space needs to be transformed to orientations of the actuators $\boldsymbol{q} \in n$ for n actuators (joint space). The transformation from configuration space to joint space is named “Forward kinematics”. The transformation from joint space to configuration space, is named “Inverse kinematics” [12].

Depending on robot configuration, solutions of inverse kinematics will differ and is not always unique. The solution can be non existing, unique, multiple or infinite. The process of estimating the pose of the manipulator based on joint variables is only straight forward when the solution of the inverse kinematics is unique.

1.2.4 Over actuated system

An over actuated system is a system containing more actuators than degrees of freedom. An object moving in space has six degrees of freedom, three for position (x, y, z) and three for orientation (ϕ, θ, ψ). For a system to arbitrarily control every degree of freedom separately, six actuators are needed, one for each degree of freedom. By using seven actuators to move in six-dimensions, the system is over actuated. An over actuated system can still operate in all six degrees of freedom, while it also achieves a set of new properties, due to redundancy in the design. Properties of over-actuated include possibility to control movement to minimise power consumption, wear and tear, singularity avoidance and avoiding forbidden sectors.

1.2.5 Velocity kinematics

In order to control a manipulator by describing velocities in configuration space, a relation between velocity in configuration space ($\dot{\boldsymbol{x}}$) and velocity in joint space ($\dot{\boldsymbol{q}}$) must be found. This is described by the velocity kinematics as:

$$\dot{\boldsymbol{x}} = \boldsymbol{J}\dot{\boldsymbol{q}} \tag{1.1}$$

and the inverse transformation is described by

$$\dot{\boldsymbol{q}} = \boldsymbol{J}^{-1}\dot{\boldsymbol{x}} \tag{1.2}$$

where J is called the ”Jacobian matrix”. If the robot is equally actuated, the jacobian matrix is a square matrix. If the robot however is over-actuated, the matrix J is a fat matrix. The Jacobian matrix is a function of the joint variables. The rank of the matrix is the maximal number of independent columns or rows. For a manipulator to be able to move the tool in an arbitrary direction, having six joints is in it self not necessarily sufficient, the rank of the Jacobian matrix must also at all times be six. Configurations of the manipulator where the Jacobian matrix does not have full rank are called ”singularities”. Among the reasons for these being interesting is that at singularities, bounded end-effector velocities can correspond to unbounded joint velocities [12].

1.2.6 Control allocation

This is a topic for over-actuated systems, which we don’t have. We could however possibly ended up with over-actuated system, and knowing its complications helped argue for not getting over actuated.

1.2.7 Path and Trajectory planning

When controlling a manipulator, an initial and final configuration is defined. As there might be obstacles between the two configurations, a path must be found, avoiding the obstacles. A path is defined as a continuous map $\tau : [0, 1] \rightarrow \mathcal{Q}$, with $\tau(0) = q_{init}$ and $\tau(1) = q_{final}$ [12]. A trajectory is a function of time $q(t)$, such that $q(t_0) = q_{init}$ and $q(t_f) = q_{final}$. Here, $t_f - t_0$ is the time it takes to move from the initial to final configuration. By considering the parameter of τ as a variable of time, a path is a special case of a trajectory, with the time parameterized by the amount of time is takes to execute the movement.

The path is often calculated by an initial and a final configuration and constraints on movement. The desired initial and final configuration can be described by specific values to all derivatives of the joint variables. The constraints on movement can consist of positions which the manipulator should avoid, or limitations on velocities and accelerations along the trajectory. In a scenario with no obstacles and no constraints on movement, a trajectory can be calculated solely based on initial and final configuration, given the initial and final time. In order to describe a trajectory as a polynomial, there must be the same number of variables in the polynomial as constraints.

Point to point trajectory

Given two scalar points with constraints on velocity both at initial and final position as given in 1.3-1.6, the trajectory can be described by a polynomial with four coefficients.

$$q(t_0) = q_0 \tag{1.3}$$
$$\dot{q}(t_0) = v_0 \tag{1.4}$$
$$q(t_f) = q_f \tag{1.5}$$
$$\dot{q}(t_f) = v_f \tag{1.6}$$

By describing the trajectory as a third degree polynomial, the trajectory gets the form of equation 1.7.

$$q(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 \tag{1.7}$$

Combining this equation with the constraints, we get four equations with four unknowns:

$$q_0 = a_0 + a_1 t_0 + a_2 t_0^2 + a_3 t_0^3 \tag{1.8}$$
$$v_0 = a_1 + 2a_2 t_0 + 3a_3 t_0^2 \tag{1.9}$$
$$q_f = a_0 + a_1 t_f + a_2 t_f^2 + a_3 t_f^3 \tag{1.10}$$
$$v_f = a_1 + 2a_2 t_f + 3a_3 t_f^2 \tag{1.11}$$

These equations can be written in matrix form:

$$\begin{bmatrix} 1 & t_0 & t_0^2 & t_0^3 \\ 0 & 1 & 2t_0 & 3t_0^2 \\ 1 & t_f & t_f^2 & t_f^3 \\ 0 & 1 & 2t_f & 3t_f^2 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} q_0 \\ v_0 \\ q_f \\ v_f \end{bmatrix} \tag{1.12}$$

$$\mathbf{Ma} = \mathbf{b} \tag{1.13}$$

From equation 1.13 equation, the parameters of the trajectory, **a** can be found by $\mathbf{a} = \mathbf{M^{-1}b}$. This approach can be used to create trajectories of higher order by applying more restrictions on initial and final acceleration.

A point to point trajectory described by a polynomial of degree five, calculated by initial and final constraints for position, velocity and acceleration, is illustrated in figure 1.3.

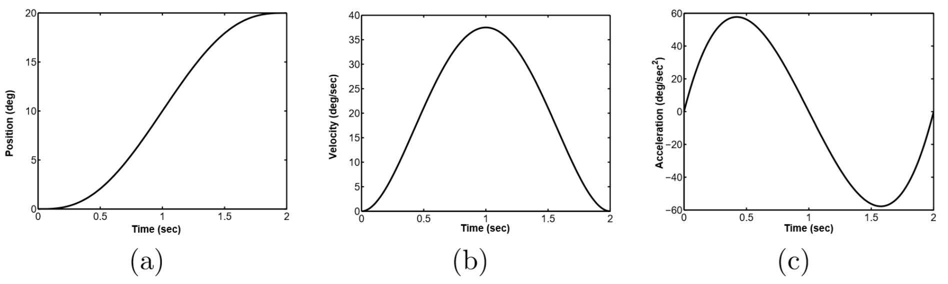


Figure 1.3: Position, velocity and acceleration plot of a trajectory of a polynomial of degree five [12].

Linear segments with parabolic blends For some operations, having constraints on the start and end positions alone is not satisfactory. In situations where it is desired to have a constant velocity along the path, a trajectory can be calculated using *Linear segments with parabolic blends* (LSPB) [12]. This trajectory planning can be considered a point to point trajectory with an additional two points between the initial and final position. This results in a trajectory consisting of three components, first a parabolic trajectory moving from initial configuration to constant velocity, then a linear trajectory and finally a parabolic trajectory moving from the constant velocity to the final configuration. The parabolic blends ensure both the end-conditions and the constant velocity constraints are met.

A plot of a LSPB trajectory is illustrated in figure 1.4. This trajectory is computed with the constrained desired constant velocity at 60 deg/sec and both initial and final velocity equal zero. The initial position is zero, and the final position is 40 deg. The parabolic blends are modelled by a polynomial of degree two, as the point to point trajectories for the parabolic blends have a total of three constraints.

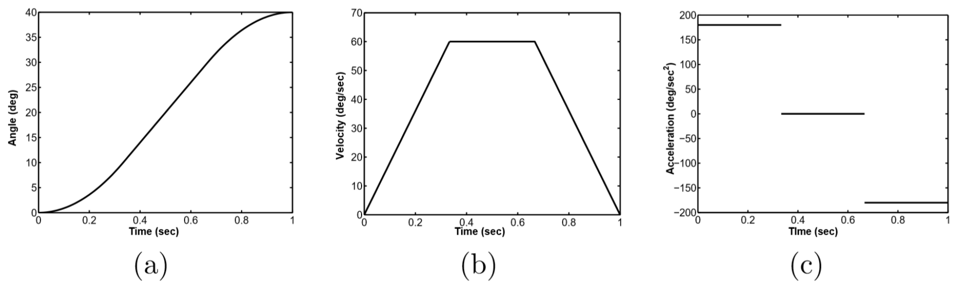


Figure 1.4: Position, velocity and acceleration plot of a linear segment with parabolic blends trajectory [12].

1.2.8 Rotary Encoder

A rotary encoder is a sensor that converts angular position to a digital or analogue output signal used to estimate the angle of a shaft. Rotary encoders are generally divided in two categories: absolute encoders and incremental encoders. Absolute encoders output a signal corresponding to the current shaft angle relative to absolute zero angle. Reading absolute angle is practical as it does not require the user to keep track of every signal produced by the encoder. This allows the user to read the value from the encoder only then it is needed, while keeping track of the number of rotations in scenarios where the shaft rotates more than 360 deg. This type of encoding does however require more space for the encoder, restricting the resolution of the sensor [13]. An incremental encoder sends out a binary signal on the out-port, acting like a square wave as the shaft rotates. This signal gives no information on the orientation of the shaft relative to an absolute zero position like an absolute encoder does. The change in signal from the incremental encoder however represents an incremental change in position. By counting all signal changes from a known position and knowing the number of cycles per revolution for the encoder, the orientation of the shaft can be known exact. As the signals relate to a change in position, the outputs from the encoder must be continuously counted in order keep the estimated shaft angle accurate, not drifting from the true value. As incremental encoders are able to send data at high rate and with low latency, these are commonly used for high speed real-time applications. The signals from an incremental encoder are represented using quadrature encoding.

Quadrature encoding

By counting the rising or falling edges of the square wave produced by the encoder when it rotates, the position change of the encoder can be estimated. By only having one output however, it is impossible to get information about the direction of rotation. In order to know this, incremental encoders are equipped with two sensors and has two outputs named A and B. The square wave from these sensors when rotating the shaft are shifted with a 90 deg offset. The relation between these two pulses gives directional information [13]. Depending on the state of pulse B when pulse A is rising, the direction of rotation can be determined: If pulse train A leads B, the shaft rotates clockwise. Similarly, the signal on output B leads signal A, the shaft rotates counter clockwise. By counting only the rising edges of output A, it is possible to detect as many positions per rotation as there are cunts per turn on the sensor. By also counting the falling edges of output A, it is possible to detect two times as many positions as there are counts per turn on the sensor. As these two pulse trains are 90 degrees out of face, rising and falling edges of both signal A and B can be used to determine directional change in position. This enables the possibility to count four times the number of positions for each rotation as there are counts on the encoder [14]. The different types of data evaluations are illustrated in Figure 1.5.

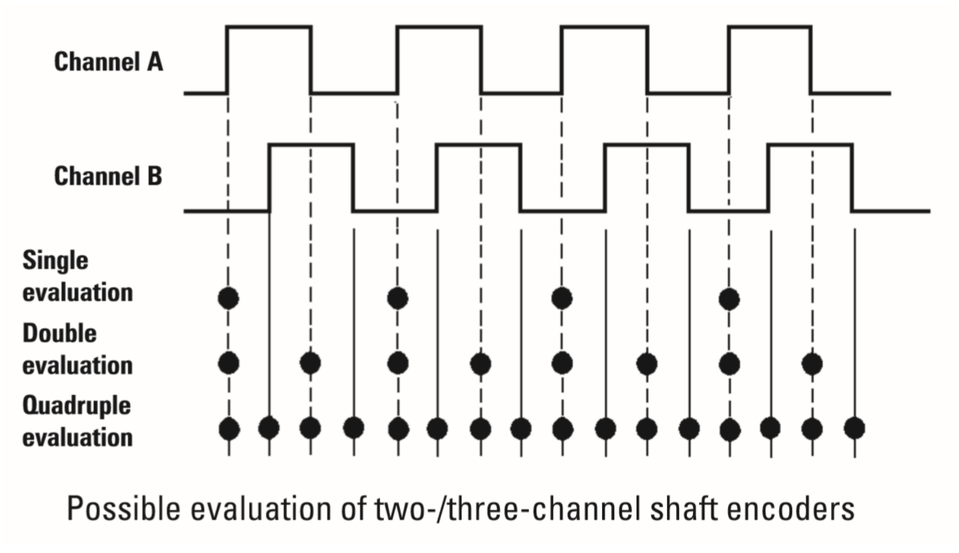


Figure 1.5: Quadrature encoder evaluation for counter clockwise rotation [15]

As incremental encoders depend on counting rising and falling edges of the signals, and only gives information on position change, it is prone to errors. If the shaft speed is too fast for the decoder to read, some increments can be missed, causing the estimated shaft angle to drift from the true shaft angle. In order to solve this problem, many incremental encoders are equipped with a third channel sending a signal when the shaft is at a specific position, sending one signal for each rotation. This signal is named the “index” signal, often referred to as Z. The index signal makes it possible to know when the shaft has reached a full rotation and to compensate for the accumulated error.

1.3 Problem description

Existing process and methods for cultivation of macro algae are industrially under-developed and labour intensive. As the global demand for macro algae increase, there is a need for technology development and automation of both the cultivation and harvest process.

The research organisation SINTEF has through its project MACROSEA presented a modular concept for cultivation of macro algae, named Standardised Production of Kelp (SPOKe). The concept includes a manipulator operating the cultivation substrate which is attached to a ring structure moored to the sea floor. The SPOKe concept requires further development both in form of theoretical evaluation of potential and limitations, as well as practical implementation and testing.

Building on the work done relating to the SPOKe concept, the assignment is structured to the following tasks:

- Investigate different specific design implementations required to form a prototype based on the SPOKe concept. This involves solutions to include the attachment points for the carrier rope to the work space of the manipulator. Design of the attachment points and mechanism to tighten the rope as well as propulsion and sensory system enabling control of the manipulator should also be investigated.
- Describe and propose solutions for the control problem of deploying rope with kelp seeds on the cultivation plant.
- Develop a prototype based on the SPOKe concept for testing the implementation of the control system.

1.4 Outline

The work in this project involves understanding the already proposed Macrosea SPOKe concept, make changes to the concept for it to be possible to implement, developing a physical prototype and finally controlling the prototype to do simple movement.

Chapter 2: Current SPOKe concept, describes the original SPOKe concept. This includes motivation, unique features and geometry. A set of joint variables are proposed for further development. The chapter also briefly describes some limitations to the concept.

In chapter 3: Extending SPOKe design, several design changes to the original concept are proposed. The new proposals include mechanisms for securing the rope, connecting it to the ring structure, tightening the rope as well as proposing a propulsion and sensory system for movement. The prototype is based on these choices.

Chapter 4 describes the mechanical and electrical components of the prototype.

In chapter “5 Control of robot manipulator”, the control problem is introduced. The state machine describing system operations is described as well as the implementation of the controller based on the state machine. Calculation of trajectories based on point references is also described in this chapter.

The experiments used to test the functionality of the prototype are described in chapter “6 Experiments”. The results from these experiments are presented in chapter “7 Results”.

Finally, the results are discussed in the final chapter, “8 Discussion”. Here the relevance of the prototype and tests are also discussed. The chapter ends with recommendations for future work for further development of the SPOKe concept.

Chapter 2

Current SPOKe concept

In the report "*Development of area efficient and standardized structures for large-scale macroalgae cultivation*", Emil Scott Bale introduced a concept for large scale production of kelp [6]. The goal of the MACROSEA project is "successful and predictable production of high quality biomass thereby making significant steps towards industrial macroalgae cultivation in Norway" [10]. The concept developed by Bale was named Standardized Production of Kelp, SPOKe. In the SPOKe concept, kelp is cultivated on a one-dimensional substrate, which is attached to a structure made up of two rings, as illustrated in Figure 1.2. When the carrier rope is attached to the ring structure, it looks like spokes on a wheel. This concept is designed to make it possible to automate the process of deploying the carrier rope, harvest the kelp and detach the carrier rope after or during harvest. The SPOKe concept is proposed to have a mooring-system inspired by the fish cultivation industry. This allows several ring structures to be attached to the same anchor-points on the seabed and makes the plant modular.

The deployment and detachment of the carrier rope, as well as harvesting of the kelp, is proposed done using a manipulator. In order to do this, the manipulator must be able to attach to the ring structure and move a tool to any position between the two rings. The manipulator will have a propulsion system allowing the rail to move angular along the ring structure, and a separate propulsion system allowing the tool to move along the rail

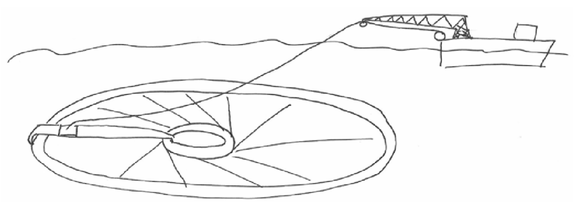


Figure 2.1: Surface vessel feeding rope to manipulator [11].

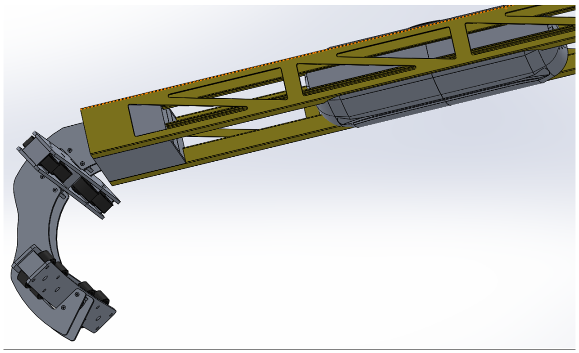


Figure 2.2: Attachment mechanism for SPOKe manipulator

of the manipulator. The carrier rope as well as power, is provided to the manipulator by a surface vessel as illustrated in figure 2.1. This then goes through the tool of the manipulator, which is capable of connecting the rope to connection points on the ring structure. By moving the tool of this manipulator between the connection points on the ring structure, this manipulator will be able to perform its proposed tasks. The SPOKe concept achieves its potential for scalability by allowing the manipulator to be independent of the ring structure. This way, the expensive manipulator required to perform the proposed tasks can be used to operate several ring structures. A small number of manipulators can then be used to operate a large number of kelp cultivation plants. The manipulators can be moved from one ring structure to the next on a boat.

The attachment mechanism keeping the manipulator attached to the ring structure in this design, is a gripping mechanism on each side of the rail of the manipulator. This attachment mechanism is illustration in figure 2.2. The grip is connected to the rail in one single joint, which can be used to change the configuration of the grip to either be open or closed. When the manipulator is moved from the boat down the ring structure, the grip is open. Once the manipulator hits the ring structure, the grip closes, ensuring the manipulator is locked in place.

2.1 Geometry

In this report, many components of the SPOKe design will be referenced. *The ring structure* refers to the structure consisting of the outer ring, inner ring, and the support beams connecting the two. The moving beam containing actuators for moving in both radial and tangential direction, is referred to as *the manipulator*. The part of the manipulator moving in angular direction, is called *the rail*. The part of the manipulator moving linearly along the rail is called *the gantry robot*. The mechanism on the gantry robot for feeding rope to the ring structure is called *the tool*. The attachment-points for the carrier rope on the ring structure, are referred to as cleats. These cleats are positioned both on the inner and the outer ring.

The manipulator moves along the ring structure between the cleats in order to attach the carrier rope to them. As the movement about the ring is defined by these cleats, both position must be known exact. The extension of the cleats can be described both by length and by angle about the centre of the ring structure, as illustrated in figure 2.3a. The relation between the arc of a circle and the length of that arc, is given by equation 2.1. The radius of the inner ring in this illustration is r_3 and the radius of the outer ring is r_4. As the length of the clams are short relative to the radius of the circle, their arc length is assumed equal to their length.

$$\alpha = \frac{l}{r} \tag{2.1}$$

(2.2)

The arc between the clams on one ring must at least be as large as the arc of the clams on the other ring for there to be an equal amount of clams on both rings. This is illustrated in figure 2.3b and 2.3c. In this illustration, the circles on each side of the clams illustrates the dimension of the tool. The clams are positioned in a way that allows rope to be attached between them to be parallel to the rail of the manipulator. When the manipulator tool is next to a clam on one of the rings, it will be positioned next to a clam on the other ring by only moving along the gantry robot. In order for this to be possible, the spacing between the clams on the inner ring is equal to the angle of a clam on the outer ring, in addition to the size of two tool diameters, as illustrated in figure 2.3b and 2.3c. The dimensions for the clams are summarised in table 2.1.

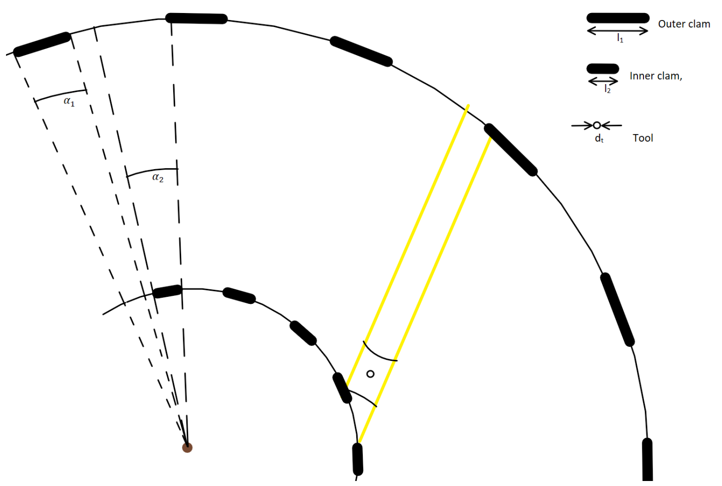
	length	arc angle [rad]	spacing length	spacing [rad]
Inner clam	l_2	$\alpha_2 = \frac{l_2}{r_3}$	$2d_t + \alpha_2 r_3$	$2\frac{d_t}{r_3} + \alpha_2$
Outer clam	l_1	$\alpha_1 = \frac{l_1}{r_4}$	$2d_t + \alpha_1 r_4$	$2\frac{d_t}{r_4} + \alpha_1$

Table 2.1: Clam dimensions

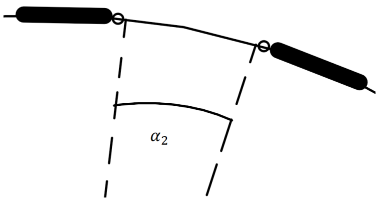
When describing possible tool positions relative to the ring structure, a coordinate frame is needed. A Cartesian coordinate system attached to the SPOKe system with origin located in the centre of the ring structure is illustrated in Figure 2.4 by the x and y axis. The z direction follows the right hand rule. As the SPOKe concept is circular, it is more intuitive to describe positions using polar coordinates. A position in polar coordinates is given by a distance r relative to origin of the cartesian coordinate system and an angle θ relative to the x-axis.

The ring structure is anchored to the seabed using a mooring system, with some slack. When subjected to waves and changing current, the orientation and position of the ring structure will vary. This work will focus on the movement of the manipulator relative to the ring structure, and the position and orientation of the ring structure is not considered.

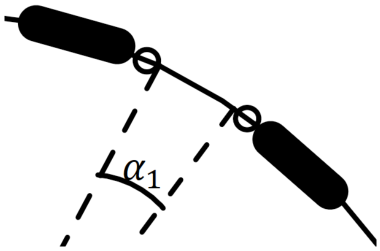
In order to describe the geometry of the SPOKe module, several parameters are named and described. The angle θ_1 is defined as the angle between the x-axis and the line inter-



(a) Cleat dimension relation.



(b) Spacing between outer clams.



(c) Spacing between inner clams.

Figure 2.3: Geometry of clams

secting both the origin of the cartesian coordinate system and the tool. The length r_1 is the distance from origin to the tool of the manipulator. The coordinate pair θ_1 and r_1 is therefore the polar representation of the position of the tool as is illustrated in Figure 2.4. The length r_2 describes the distance from the anchor point of the rail on the inner ring to the tool.

The angle θ_4 is the angle between the x-axis and the line intersecting both the SPOKe origin and the anchor-point of the rail on the outer circle, as illustrated in Figure 2.5. The distance from origin to the outer circle is defined as r_4, and the total length of the rail is named l_{total}.

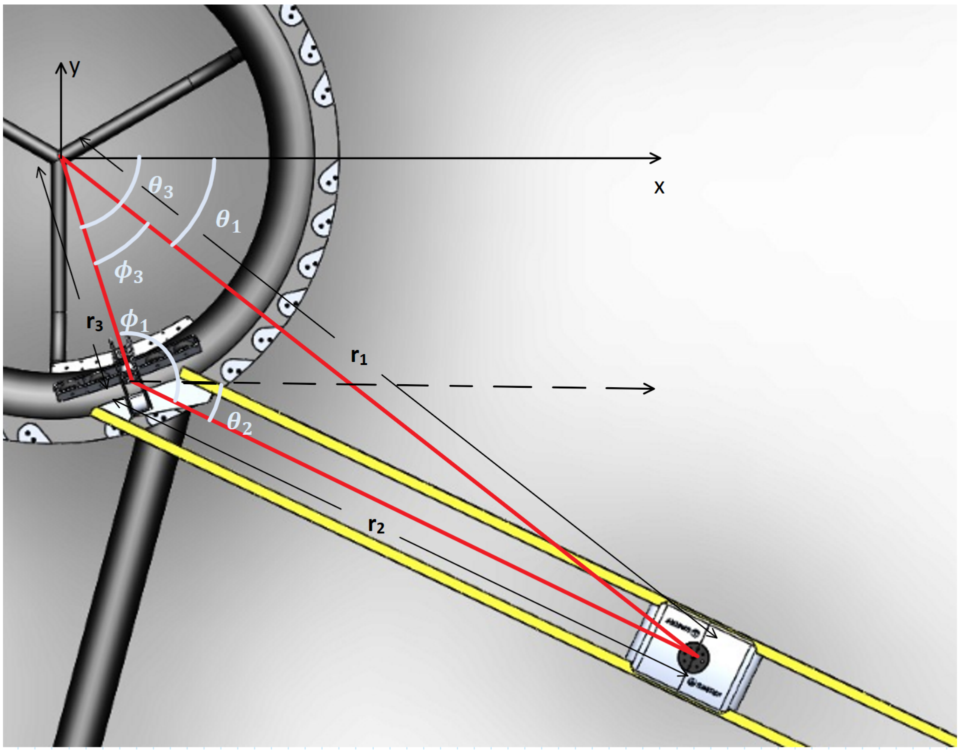


Figure 2.4: The illustration shows how the angles $\theta_1, \theta_2, \theta_3, \phi_3$ and lengths r_1, r_2, r_3 relate to the robot-configuration. The SPOKe-frame is illustrated in x-y.

2.2 Modelling, Workspace and joint variable of the SPOKe system

As the manipulator only moves in two dimensions, the x-y plane, the orientation only varies about the z-axis. The purpose of the tool is to be the link between the rope connected to the boat and the rope connected to the ring structure. Depending on the tool design, it is possible for the tool to fulfil its task independently of its orientation. Due to these properties of the manipulator, it is possible to model the pose in only two dimensions. The choice of joint variables for the manipulator will affect the complexity of control. Three different representations are considered, namely $[\theta_1, r_1]^T$, $[\theta_2, r_2]^T$ and $[\theta_4, r_2]^T$.

The polar representation using θ_1 and r_1 as joint variables gives an intuitive representation of the position of the tool-frame. The task of the manipulator is to attach ropes as angled spokes as illustrated in Figure 2.6. When the gantry robot moves along the rail while the angle of the rail is constant, both θ_1 and r_1 will vary. This property will result in the variables θ_1 and r_1 oscillating when the manipulator attaches the rope to the ring

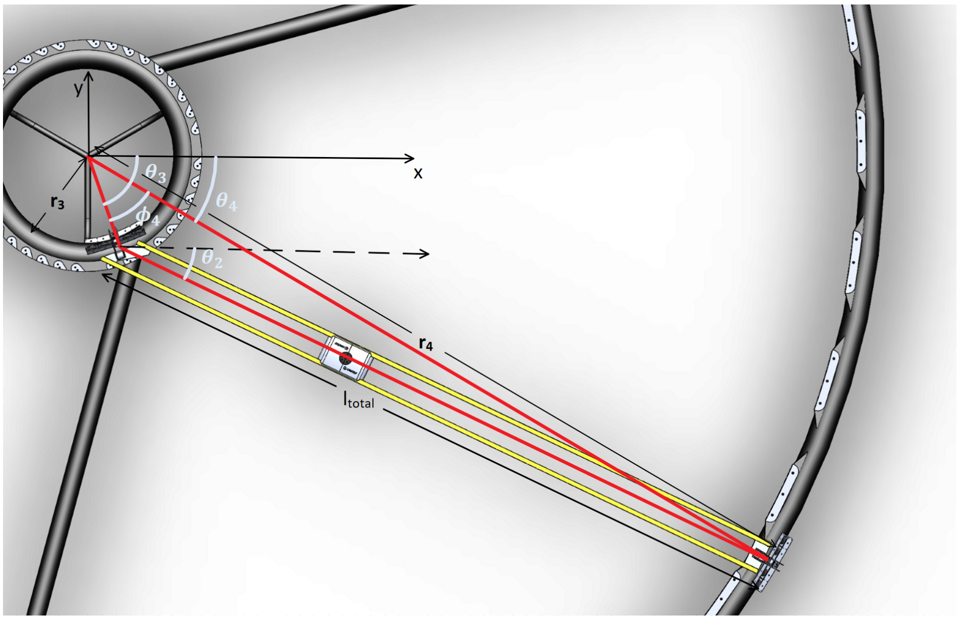


Figure 2.5: The illustration shows how the angles $\theta_2, \theta_3, \theta_4, \phi_4$ and lengths r_3, r_4, l_{total} relate to the robot-configuration. The SPOKe-frame is illustrated in x-y.

structure. This representation is therefore not optimal for controlling the main tasks of the manipulator.

By using the parameters θ_2 and r_2 for representing the movement under attachment of the rope, the angle θ_2 is strictly increasing, while only r_2 is oscillating. In the scenario where angle of the rail is constant while the tool moves along the rail, only the value of r_2 changes. In reality, the angular movement of the rail is controlled by a motor placed by the outer ring. It can therefore be considered more intuitive to represent the manipulator movement by using the angle θ_4 instead of the angle θ_2. Based on these considerations, the workspace is described by the vector of joint variables $\mathbf{x} = [\theta_4, r_2]^T$. These two variables will form the basis of a coordinate frame which will be named the *SPOKe coordinate frame*.

2.3 Coordinate transformations

This section will describe equations for transformation between different variables describing the pose of the SPOKe manipulator. As described in the previous section, the rail is oriented with an angle offset relative to the radial direction of the ring structure. Transformation between different variables therefore require the introduction of geometric properties of the SPOKe manipulator. The angle ϕ_4 as illustrated in Figure 2.5 is a

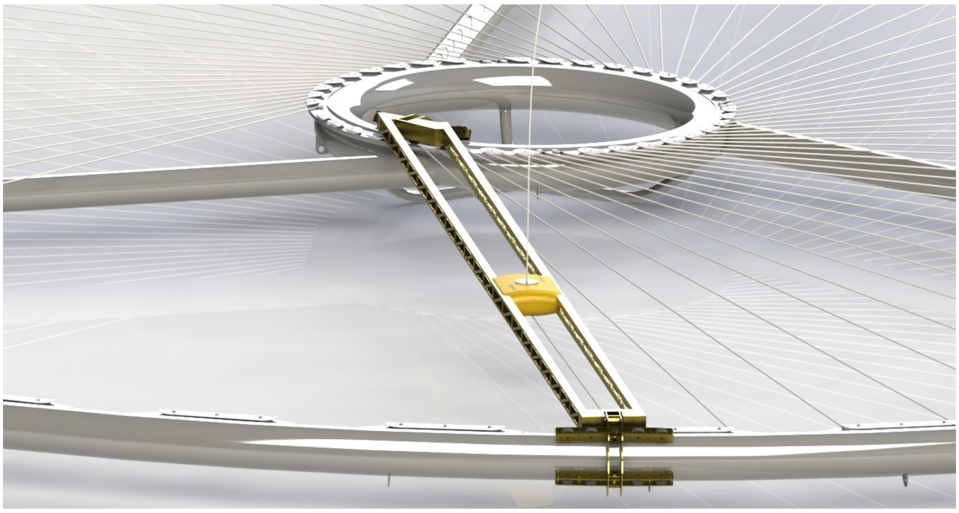


Figure 2.6: Spokes are slightly angled

constant angle describing the geometry of the manipulator. Angles θ_3 and θ_2 shown in the same illustration has constant offset values relative to θ_4. In figure 2.4, ϕ_3 is illustrated as an angle varying as the tool moves along the rail, while ϕ_1 is a constant angle given by the manipulator dimensions. Relations of several of the geometric parameters of the SPOKe manipulator are given in the equations 2.3 - 2.9.

$$\phi_4 = cos^{-1}(\frac{r_3^2 + r_4^2 - l_{total}^2}{2r_3r_4}) \tag{2.3}$$

$$\phi_1 = cos^{-1}(\frac{l_{total}^2 + r_3^2 - r_4^2}{2l_{total}r_3}) \tag{2.4}$$

$$\theta_2 = \theta_4 - sin^{-1}(sin(\phi_4)\frac{l_{total} - r_4}{l_{total}}) \tag{2.5}$$

$$\theta_3 = \theta_4 - \phi_4 \tag{2.6}$$

$$r_1 = \sqrt{r_2^2 + r_3^2 - 2r_2r_3cos(\phi_1)} \tag{2.7}$$

$$\phi_3 = cos^{-1}(\frac{r_1^2 + r_3^2 - r_2^2}{2r_1r_3}) \tag{2.8}$$

$$\theta_1 = \theta_4 + \phi_3 - \phi_4 \tag{2.9}$$

The values for the angle θ_4 and the length r_2 can be estimated based on values given by sensors on the joints. When controlling the manipulator to move the tool to a specific position on the ring structure, it is intuitive to give the instruction in polar coordinates θ_1, r_1. The transformation from polar coordinates to joint variables are given in the equations

2.10 and 2.11. Equation 2.11 shows that $r_2 > 0$ for $r_1 > r_3$.

$$\theta_4 = \theta_1 + \phi_4 - \phi_3 \tag{2.10}$$

$$r_2 = r_3 cos(\phi_1) + \sqrt{r_3^2 cos^2(\phi_1) - r_3^2 + r_1^2} \tag{2.11}$$

The inverse kinematics is given by equations 2.7 and 2.9. As the length r_1 is defined as a positive value, the inverse kinematics never has multiple solutions. The solutions for the inverse kinematics has unique solutions for $r_2^2 + r_3^2 > 2r_2r_3cos(\phi_1)$.

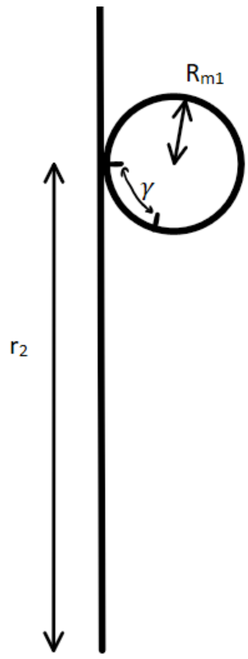
On the prototype, the manipulator is attached to the ring structure with a set of wheels. Assuming there is no slip between the wheels and the ring structure, their position can be represented by their orientation. Assuming that the movement of the gantry robot along the rail also involves a wheel with no slip, the position along the rail can be represented by the orientation of this wheel. Estimating position based on integrating the values from motion sensors such as as incremental encoders, is named odometry.

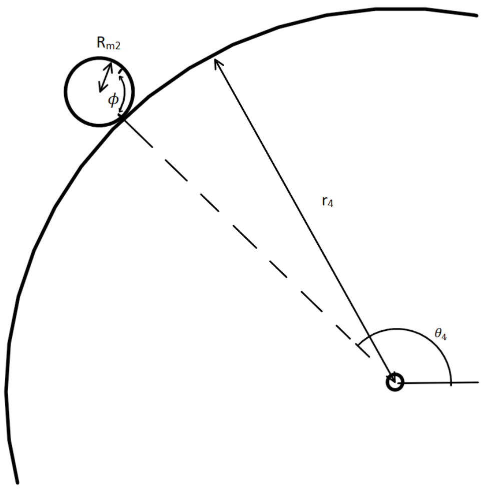
The relation between the orientation of the wheel connected to joint variable r_2 and the actual length r_2 is illustrated in Figure 2.7a. From this illustration, it follows that the relation between motor angle and joint variable r_2 is given by Equation 2.12. In this equation, R_{m1} is the radius of the wheel and $\delta\gamma$ is the change in angle of the wheel given in radians.

$$\delta r_2 = R_{m1}\delta\gamma \tag{2.12}$$

Similarly to this transformation, the transformation for the second joint variable, θ_4 is given by Equation 2.13. In this equation $\delta\phi$ represents the change in angle of the wheel.

$$\delta\theta_4 = \frac{R_{m2}}{r_4}\delta\phi \tag{2.13}$$





(a) Relation between motor angle r_2. **(b)** Relation between motor angle ϕ and joint variable θ_4.

Figure 2.7: Relation between motor angles and joint variables r_2 and θ_4.

2.4 Design limitations

When introducing the SPOKe concept, a three dimensional model was created in order to explore the immediate possibilities and limitations of the SPOKe design. This model also creates the basis for further design development. This model does not take in to consideration actuator design, choice of sensors, exact movement limitation, material choice or structural integrity. This allows the concept to illustrate requirements for the SPOKe concept without applying limitations to future considerations needed to be taken. Before creating a physical prototype based on this model, there are five important aspects of the design that need further development:

- In the existing model of the SPOKe manipulator, the tool does not have the reach ability to move the rope about the cleats. The manipulator must have a workspace including the clams both on the inner and outer ring in order to attach the rope to the ring structure.
- The design of the cleats must be chosen. The cleats should be designed for easy attachment of the rope. Further, the rope should not detach from the ring structure, even if the rope breaks at some point.

- In the existing model, there is not defined what mechanism should be used to tighten the rope between the cleats. There are several possible solutions that must be considered.

- Propulsion systems making the manipulator capable of moving both joints should be defined. In this, the unique challenges resulting from the ring structure being submerged in sea water should be considered.

- A proposal must be made of sensors capable of measuring the pose of the manipulator accurately enough to ensure precise movement. What sensors relevant for the manipulator will depend on what propulsion system is chosen.

These considerations will be further explored in the next chapter.

Chapter 3

Extending SPOKe design

This chapter will look in to different designs aimed at extending the original SPOKe model when it comes to the five points mentioned in section 2.4. These design changes are necessary for the manipulator to attach rope to the ring structure. The first three sections of this chapter each describe design proposals for fixing different task operations: 3.1, 3.2 and 3.3. Section 3.4 examines combinations of the previously described design proposals. Finally, a propulsive and sensory system is defined in section 3.5, and an arm is considered for stability purpose in section 3.6.

3.1 Securing rope

In order to make sure that the rope is attached securely to the ring structure and does not loosen, a design for securing the rope needs to be chosen. This securing of the rope has three purposes:

1. Ensure the rope does not fall off the ring structure.
2. Ensure that the rope does not slide on the ring structure.
3. Ensure that the rope does not loosen if the rope breaks at one point.

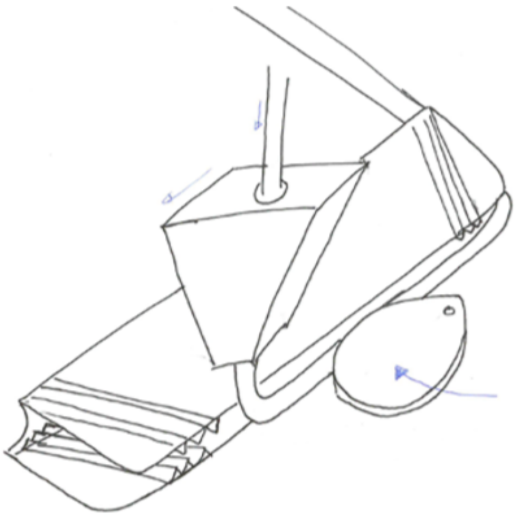
Two different designs are considered, and differs in what purposes they fulfil.

3.1.1 Clam cleat based connection

Clam cleats are u-shaped tracks with angled teeth inside, as shown in Figure 3.1b. When pulling the rope through the clam cleat in one direction, the rope moves through with little friction. When pulling the rope in the other direction, the teeth grip on to the rope, and their angle force the rope more tightly in to the clam cleat. In order to securely attach a rope using a normal clam cleat, the rope needs to be pulled tightly in one direction, creating a large rope tension. This tension enables the teeth to grip and lock the rope in place.

A lock like this only locks the rope in one direction.

A design for rope attachment based on clam cleats was introduced by Ingrid Wiik [11]. This design is illustrated in Figure 3.1a. In this design, the cleat has two clams with teeth tightening the rope in opposite directions. This allows the cleat to fixate the rope from moving in both directions. The downside with this design in the challenges it introduces when it comes to mounting. As the two clams are faced in opposite directions, the rope can not simply be tightened by pulling it in one direction as for a regular clam cleat. A mechanism involving a drop-formed manipulator was therefore introduced for attaching the rope to the locking mechanism. In order to avoid the rope falling off the ring structure, the locking mechanism is positioned horizontally. The tension of the rope will naturally keep it from falling out of the opening on the locking mechanism. If the rope breaks at one point, the rope will have slack and no tension to the clam. Without tension, the clam can not ensure the rope does not loosen in the open part of the locking mechanism. This locking mechanism fulfils design-purpose 1. and 2. described in section 3.1 but not 3.


(a) Locking mechanism using cleat with two clams [11]

(b) A standard clam cleat [16].

Figure 3.1: Clam cleat based locking mechanisms for rope.

3.1.2 Bolt design

A different design is the bolt design. This involves a bolt being inserted through the rope and a bollard with a notch, as illustrated in figure 3.2. When the rope has high tension over the bollard, the bolt will lock it in place and the notch will help to keep the orientation of the rope stable. If the rope breaks at one point, the bolt will keep it locked in place at the bollard, and keep it from unwinding. As the rope is locked in place using a bolt, the bollard does not need teeth for keeping the rope tight. The kelp seeds passing through this bollard is therefore handled more gently compared to kelp passing through a clam cleat. In order to get the desired tension of the rope, metres of the rope may have to move past the bollard, making the potential for damage large.

This design fulfils all three purposes of the securing mechanism. This does however require a manipulator on each side of the rail to insert the bolt under deployment and pull it out during harvest. Alternatively, the bolt could be inserted only on one of the rings, allowing some rope to unwind.

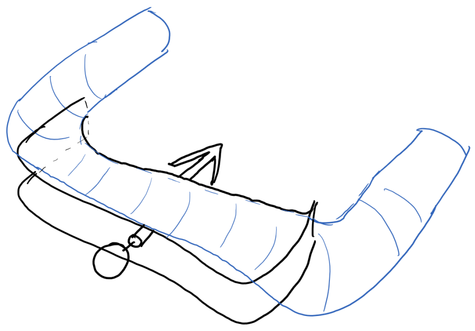


Figure 3.2: Locking mechanism using a bolt through rope.

3.2 Connecting rope

In order to include the locking mechanism for the rope in the work space of the manipulator, the current SPOKe design needs modification. The current design only allows the gantry-robot to move close to the ring, but not around it, as illustrated in Figure 3.3. This does not enable the tool to connect the rope to the ring structure. In this section, two alternative designs for this will be presented. The designs differ in number of actuators, complexity of attachment and complexity of inverse kinematic.

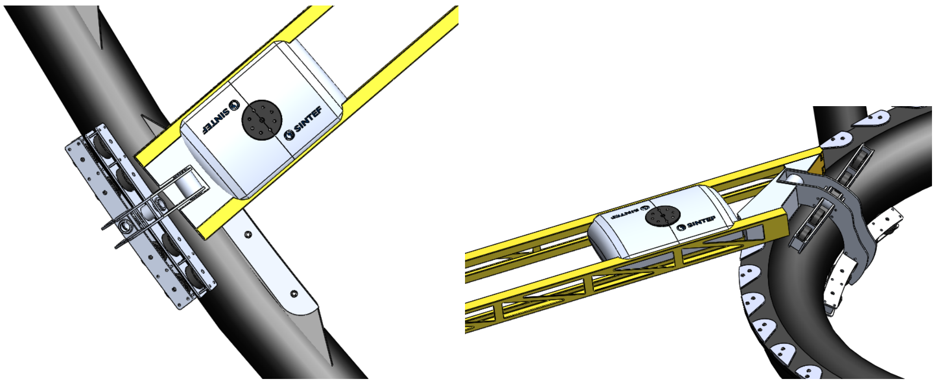


Figure 3.3: Geometric limitations of original three dimensional model.

3.2.1 Extended rail

One way for the gantry robot to access the mounting points on the ring structure is to extend the rail to allow the gantry robot to move past the mounting points for the rope. An illustration of this is shown in Figure 3.4. As the rail in this design extends past both the inner and outer ring, the grip holding the rail attached to the ring structure must be modified. As the tool now moves past the mounting points for the rope, the grip can no longer consist of one joint attached to each side of the rail. This mechanism can however be implemented using two joints on each side of the rail, as shown by the red structure in figure 3.4. Assuming each joint is controlled by a motor, this design will require a total of four motors for the grip mechanism alone. The largest downside of this is the cost of the two extra motors. The complexity of movement for this design is low, as the rope can be moved in the desired pattern only using two motors: one controlling movement radially and one controlling movement of the gantry robot. Using this design, the manipulator will have the same amount of actuators as degrees of freedom. Further, the geometry of the manipulator causes the inverse kinematics to be simple to calculate.

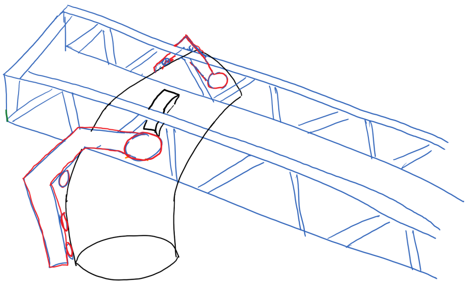


Figure 3.4: Extended rail design.

3.2.2 Robot arm on the gantry robot

A different solution to include the attachment points to the work space of the manipulator is to add an extra joint for the manipulator. This joint is attached to an arm. The tool of the manipulator will then be positioned on the end of the arm controlled by this third joint. This arm allows the system to move the rope over the connection-points on the ring-structure as illustrated in figure 3.5. Having a manipulator with three joints enables ability for more complex coordinated movements as the manipulator will be over actuated.

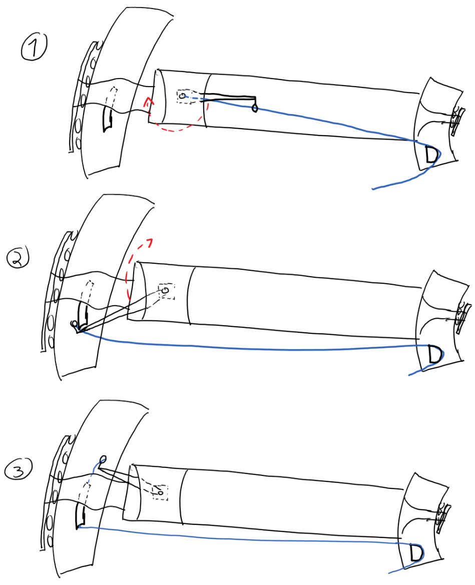


Figure 3.5: Short rail design with extra joint.

By having an arm attached to the gantry-robot, the system is over-actuated. Movement in two dimensions is controlled by three actuators. This makes the inverse kinematics more complex as there are infinitely many joint configurations that allows for any position of the tool. This design has a grip mechanism involving only two joints, similar to the original model.

3.3 Tightening rope

In order for the ropes to not be intertwined, they need to be tightened sufficiently. There are several possible designs that allow this. Three will be presented. All three includes a winch, and differs in where this winch is positioned; on the boat, on the gantry robot or on the rail.

3.3.1 Winch on boat

The winch on the boat can be used to tighten the rope attached to the spoke system. The boat will need to have a winch for pulling rope during harvest, so this design does not require any extra equipment. It does however require high tension between the boat and the ring-structure. This will require a high force output from the winch, and make the movement of the manipulator more difficult to control.

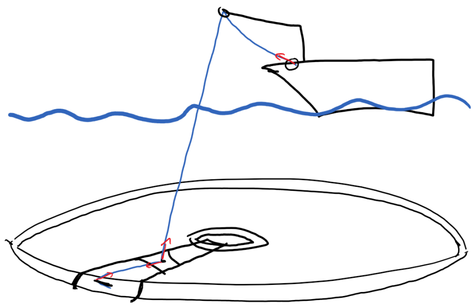


Figure 3.6: Winch positioned on the boat.

3.3.2 Winch in the gantry robot

The winch could alternatively be positioned inside the gantry robot. Two possible implementations of this in relation to the SPOKe concept has been introduced [11]. These two winch designs are illustrated in figure 3.7. In order to not damage the kelp seeds on the rope, the rope should be handled as gently as possible. In order for the winch to pull the rope, there must be large friction between the rope and the winch. As the design on the left has a larger contact area between the rope and the wheel, high friction can be achieved with lower force on the rope. This design will therefore be the one of the two damaging the kelp seeds the least.

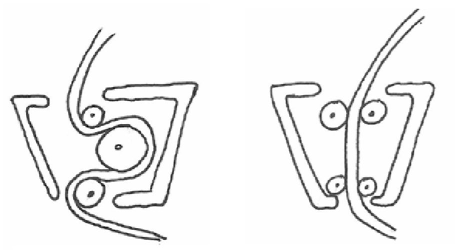


Figure 3.7: Illustration of two different winch designs in gantry robot [11].

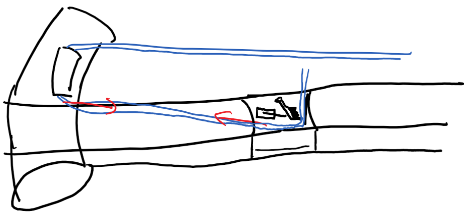


Figure 3.8: Rope lock in the gantry robot.

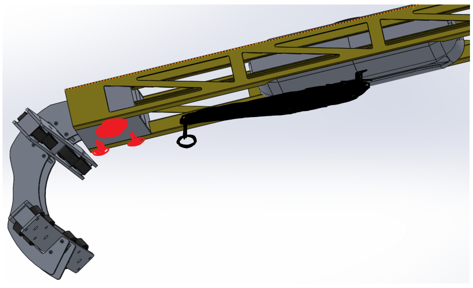
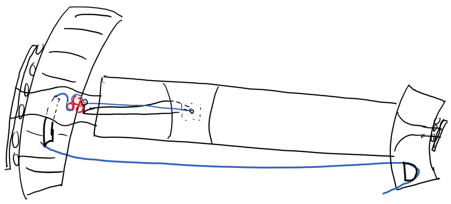
High tension on the rope connected to the manipulator affects the forces on the manipulator both in angular and radial direction [1]. The forces required by the gantry robot to have a constant position while the winch described above tightens the rope, is a function of the tension of the rope. By simply locking the rope in place in side the gantry robot and moving along the rail, the same tension will be achieved, requiring the same force from the gantry robot. This will be referred to as *rope lock*, and an example of this is illustrated in figure 3.8. This solution is simpler compared to installing a winch inside the gantry robot, as the lock can be implemented using only one motor. The lock can be controlled using only two configurations: on and off. On top of this, a lock will take up less space than a wheel based winch. By locking the rope in place like this, only the portion of the rope that is locked will experience a high force. Compared to the winch, which has to apply a force to larger portions of the rope, this solution is gentle to more of the kelp passing through the tool.

3.3.3 Winch on the rail

One winch can be placed in the centre of the rail. Alternatively, a winch can be positioned at each side of the rail, requiring two winches. A challenge with mounting the rail to the rail is connecting the rope to the winch, and moving the winch so that it does not collide with the rope as the manipulator moves along its trajectory. All the attachment points for the rope on the ring structure has the same position in z direction in a coordinate system fixed to the ring structure. This cause all the rope to have the approximately the same position in z direction, given the tension is sufficiently high. Any point of the manipulator that intersects with the rope will obstruct the movement of the manipulator. The challenge of positioning the winch on the rail is therefore to both enable it operating the rope, and not intersect with the rope. This can be done both by using a winch that is fixed, or a winch capable of moving.

Fixed winch on rail

A winch could be fixed to the rail. Because the winch is fixed to the rail, only the gantry robot can create movement relative to the winch. This causes the gantry robot to require one more joint in order to connect the rope to the winch as illustrated in Figure 3.9a. This can be solved by adding a robot arm to the gantry robot, moving horizontally similar to the one described in section 3.2.2. Mounting the rope to the winch requires coordinated movement with the arm and the gantry robot, as illustrated in figure 3.9b. As the arm rotates while the gantry robot moves along the rail, the tool can be controlled to attach the rope to the fixed rail. As the winch is fixed, it has to be positioned at a height where it does not intersect with the rope. This requires the arm capable of moving in two dimensions, one for moving the rope to the height of the winch, and one for attaching the rope to the winch. This could be designed as a cylindrical arm or polar robot, both without radial movement. With this design, the complexity of movement is increased. As the manipulator is equipped with an extra arm with two joints, it consists of a total of four joints, and moves in three degrees of freedom. This makes the manipulator over actuated, and increase complexity of control.

(a) Fixed winch and robot arm on SPOKe model.**(b)** Winch positioned on rail, with arm on gantry.

Figure 3.9: Winch fixed to rail

Movable winch on rail

An alternative to having the winch fixed to the rail, is making the winch movable. This way, the winch can move down to operate the rope, and move up when the manipulator moves. By enabling the winch to move in and out of the intersection with the rope, the manipulator itself does not need to be changed from the original model design.

One possible implementation of this is using a winch composed of two movable wheels mounted on the rail, as illustrated in Figure 3.10. The wheels are pushed against each other with the rope in between them. This creates a high tension between the rope and the wheels. By rotating the wheels, the rope gets pulled with the wheels due to the high friction. This design will require one motor in order to rotate the wheel and one motor for pushing the two wheels tightly together. Springs are used to get desired force between the wheels. In order to move the wheels away from the rope once the rope is tight, two extra joints are needed as illustrated in figure 3.11. If each of these joints are controller by a motor, this design requires four motors for each winch.

In order to tighten the rope at both ends of the rail, two winches like these are needed, one on each end of the rail. Alternatively, a single winch can be positioned in the centre of the rail. The two designs of winch on the rail rail are summarised in table 3.1.

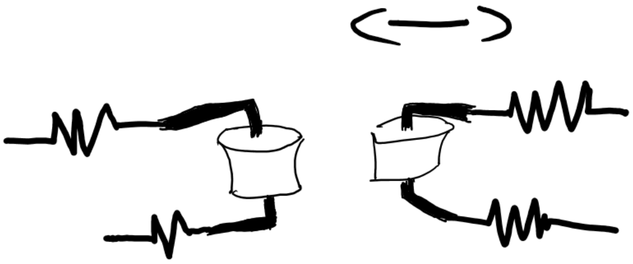
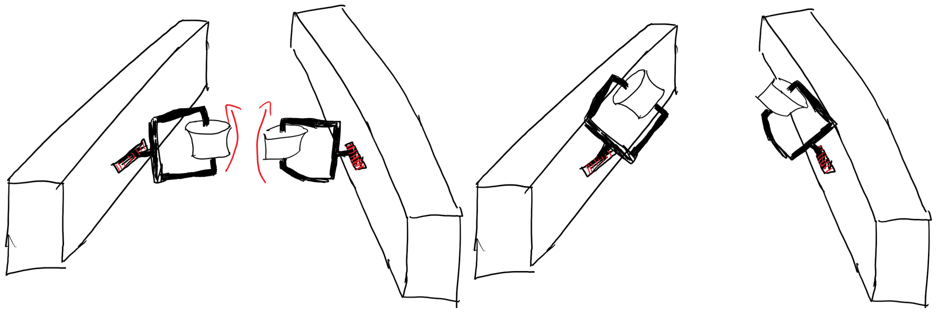


Figure 3.10: Movable winch design



(a) Winch configuration operating rope. **(b)** Winch configuration out of the way.

Figure 3.11: Movable winch on gantry

Property	Fixed winch	Movable winch
N of motors per winch	1	4
Manipulator degrees of freedom required	3	2
Manipulator joints required	4	2

Table 3.1: Properties of the three winch designs

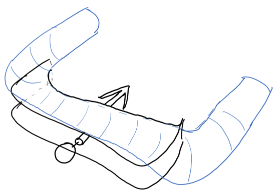
3.4 Design proposals

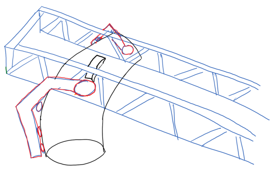
The previous three sections describes three different important design decisions that must be made. By combining the the different solutions to a more total manipulator design, it is possible to more easily compare them, and make an informed decision on what solutions should be implemented to the SPOKe concept. As there are three different design aspects each with several concepts for implementation, all possible combinations result in 20 different possible manipulator designs. This section will describe three of these, and present them as proposals for manipulator design. Among these three, one is chosen to form the basis for a physical prototype.

All three examples are based on bolt locking, as this is the only design capable of fulfilling all three requirements of the securing mechanism. As the third requirement, ”ensure that the rope does not loosen if the rope breaks at one point” is assumed critical, this is the only viable designs among the three. In all three following examples, the bolt is assumed to be attached used only on the outer ring. The motor inserting and pulling out the bolt is assumed to be positioned fixed on the rail. The concrete design of this mechanism will not be proposed here, and will have to be developed in the future.

3.4.1 Proposal 1: Extended arm, rope lock

This first configuration consists of extended rail and rope lock in the gantry robot. The combination of the extension of the rail and using the lock inside the gantry robot enables the manipulator to attach the rope to the ring structure using only two degrees of freedom.


(a) Bolt connection.


(b) Extend Rail design

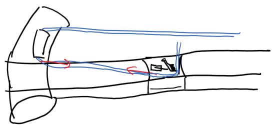

(c) Winch in ganry robot.

Figure 3.12: The three design components in example 1.

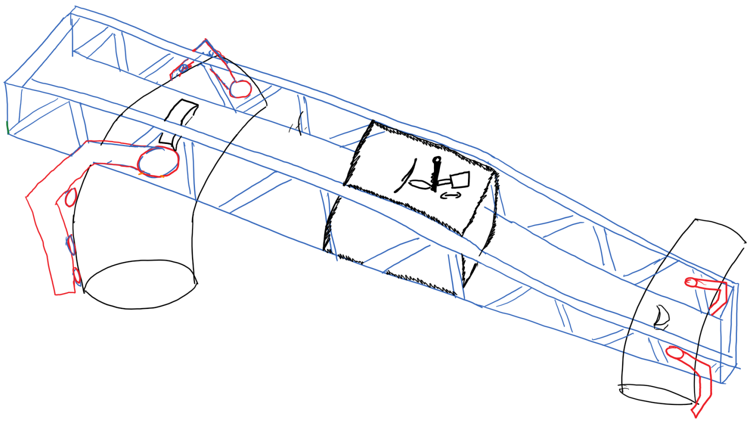
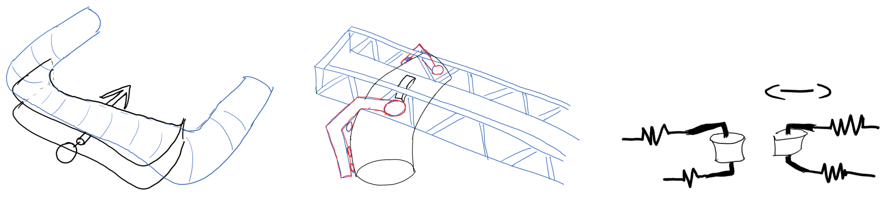


Figure 3.13: Design proposal 1 for manipulator.

3.4.2 Proposal 2: Extended arm with winch on rail

The second design proposal for the manipulator consists of extending the rail, and using a winch attached to the rail structure as illustrated in figure 3.15. In this proposal, the only task of the gantry robot is to move the rope from the inner to the outer ring. This enables the possibility for the gantry robot to be implemented as a passive component, not requiring power transfer by positioning the drive train for the movement of the gantry robot fixed to the rail. Not having power transfer to this moving component will can make implementation easier and potentially make the system easier to maintain.



(a) Bolt connection. **(b)** Extend Rail design **(c)** Movable winch design.

Figure 3.14: The three design components in example 2.

The winch used in this proposal is the movable winch illustrated in figure 3.14c. Having only one winch positioned in the centre of the rail allows the design to reduce the total number of motors by four compared to attaching one winch at each end of the rail.

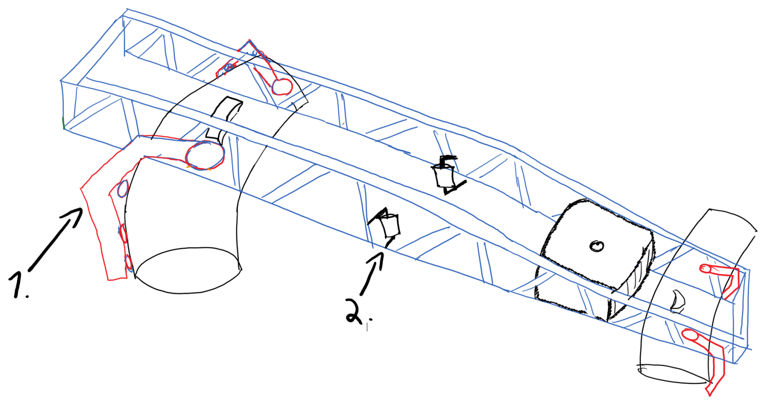
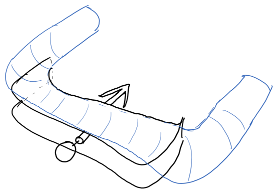


Figure 3.15: Design proposal 2 for manipulator. Only one winch, in the middle of the rail used to tighten rope in both directions.

3.4.3 Proposal 3: Extra robot-arm and fixed winch on rail

The third proposal consists of a winch fixed on the rail and an arm attached to the gantry robot. The arm has two joints, enabling it to attach rope to the winch which is positioned higher than the rope attached to the ring, in order to not intersect it. Compared to he two other proposals, this rail is easier to attach to the ring structure, as the grip about the ring only requires one joint. As the gantry-robot is equipped with an arm, this example requires power transfer to the gantry robot. The extra arm gives the total manipulator two extra joints. The manipulator is this proposal is over actuated as it has four joints moving in three dimensions. Although it is over actuated, the control of the manipulator can be simplified by describing it as a combination of several simple control states. With this design, SPOKe system can be tasked with more complex tasks in the future, that can possibly be easier to solve with an arm.


(a) Bolt connection.

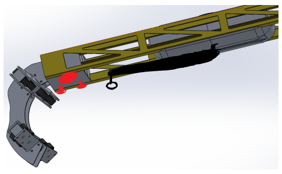

(b) Stationary winch on rail design with arm.

Figure 3.16: The three design components in example 2.

The combination of these three design choices, are combined in example 3, illustrated in Figure 3.16.

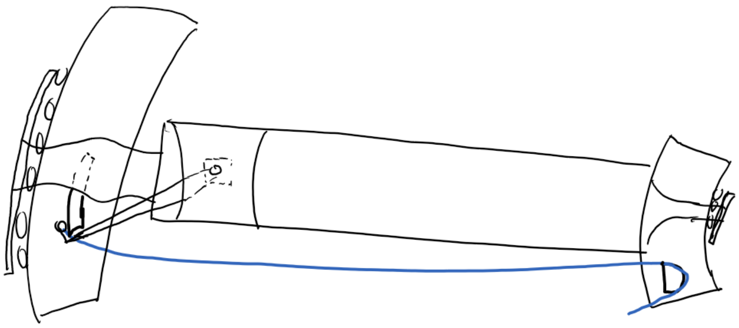


Figure 3.17: Design proposal 3 for manipulator.

3.4.4 Comparing the three proposals

The three designs are compared in Table 3.2. The table considers one motor for attaching the bolt on the outer ring, and no bolt on the inner ring.

Comparing the three designs, it becomes apparent proposal 1 is has least complexity both when it comes to number of motors and number of actuators required to do all required

Property	Example 1	Example 2	Example 3
Power needed in gantry-robot	Yes	No	Yes
N actuators needed total	8	11	8
N actuators for movement of rope	2	2	3
N actuators for tightening rope	2	4	4
Over-actuated movement	No	No	Yes

Table 3.2: Properties of the three examples

tasks. Example 2 has the highest number of motors. This is a result of this design requiring four motors for attaching the manipulator to the ring structure, and four motors for the winch design. These motors does not bring much extra complexity from a control perspective, as they only are required for mounting- and dismounting processes. Extra motors does however bring extra cost to construction and extra overhead for maintenance.

When it comes to controlling the robot, having an equally actuated manipulator is a benefit. For proposal 1 and 2, every position in configuration space corresponds only one position in joint space. In proposal 3, which is over-actuated, control becomes more difficult. Every position in configuration space corresponds to infinitely many positions in joint space.

The complexity of proposal 3 allows it to do more complex movements than the other two proposal. Among the three examples, this is the only one capable of moving in three dimensions. This extra capability enables the manipulator to do operations that the two other designs can not. With the current tasks assigned to the manipulator however, this movement is not needed. It is also not required in order to do other tasks already proposed for the manipulator in the future, such as monitoring status of growth.

The viability of proposal 1 is dependant on the gantry-robots ability to tighten the rope. The motor driving the gantry-robot will in this design require a larger force output than in the other examples. For the full scale manipulator, this motor must be able to pull the 15 m long rope to create a satisfactory tension. Assuming the mechanism for movement of the manipulator is capable of producing the required force, proposal 1 is used for the further work in this report.

3.5 Propulsion and sensory system

The choice of propulsion and sensory system is an important aspect of the design process. The main challenge is controlling the movement of the manipulator in angular direction. In this movement, the rail moves along the ring structure. This is challenging as the ring structure, being submerged in water for long periods of time potentially has a slippery and uneven surface. A second challenge is constraints on the geometry of the propulsion system, due to the geometry of the ring structure and the ropes that need to be connected to it. Movement of the gantry robot along the rail is less challenging as the rail has no obstacles. Also, as the rail is not submerged in water over longer periods of time, it will be less slippery.

The growth on the ring structure will potentially cause the ring structure to have a low friction coefficient. According to the Coulomb friction model given in 3.5, in order to get enough friction, a high force pushing the wheel to the ring is required.

$$f_{friction} = \mu P$$

Three different propulsion systems for the SPOKe manipulator has already been described and compared: Designs based on two "trilateral-wheeled robots" which are illustrated in Figure 3.19b, a "Stewart-Gough platform" illustrated in Figure 3.19a and the use of a water-jet thruster where described and compared in the project report [1]. Both the trilateral-wheeled design and the Stewart-Gough platform are based on increasing friction by using actuators to create a force towards the ring structure from several angles. The description of these three propulsion designs, are sourced from that report. Another option for propulsion compared in this section is based on a gear and a roller chain. At the end of this section, all four are compared.

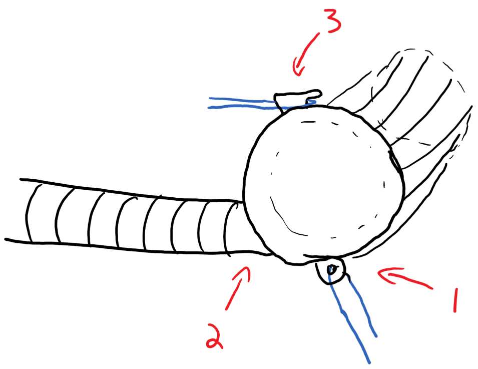
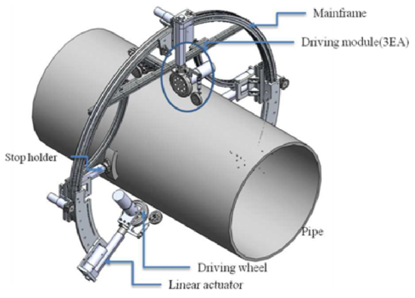


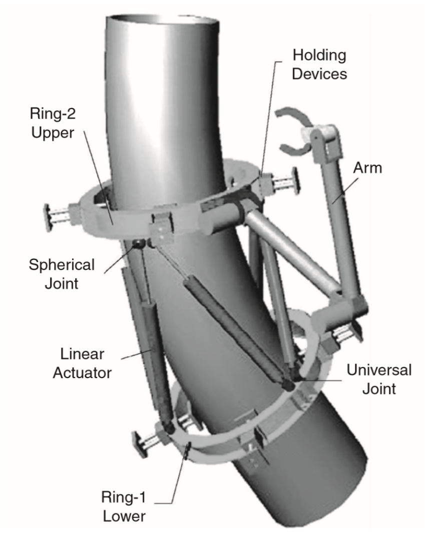
Figure 3.18: Obstacles on the outer ring of the SPOKe system

Trilateral-wheeled robot

As the trilateral-wheel design is dependant on not slipping, the rotation of the wheels can be used as a relatively accurate estimate of the position of the robot. An example of a rod-climbing robot with wheels is the trilateral-wheeled climbing robot illustrated in Figure 3.19a [17]. In this design, the wheels are placed with 120° between them. Due to the obstacles on the geometry of the outer and inner ring, implementation of the trilateral-wheeled robot for the SPOKe system will require modification. These obstacles are illustrated in Figure 3.18. One of these obstacles is the carrier rope and it's attachment point on the ring structure. The beams connecting the inner and outer ring are also obstacles that have to be avoided. The third obstacle is the connection point for the anchoring ropes of the ring structure. In order to avoid obstacles while moving, two trilateral-wheeled robots should be positioned in line.



(a) Trilateral-wheeled robot



(b) Stewart-Gough platform.

Figure 3.19: Two propulsion systems

Stewart-gough platform

An implementation of this is is the Stewart-Goughs design robot, where two rings with clamping mechanisms are moved relative to each other by linear actuators in parallel. A spesific implementation is illustrated in 3.19b [18]. Its base design is a Stewart-Gough platform, where a planes position and orientation is controlled by linear actuators in parallel. Because of this design, it is commonly named a Stewart-Gough robot (SG-robot). This design allows the robot to be locked on to the ring with pads, that can be designed to have high friction. As the robot at all times has one ring mounted at the ring, it will be able to move with high precision, as the extension of the actuators can be measured with high precision. Using this design for moving in tangential direction, the robot can use the measurements from the actuators to get an accurate position estimate.

Having the linear actuators in parallel makes the robot capable of producing a large force. This design does however suffer from the same limitation as the trilateral rod-climbing robot when it comes to obstacles. In order to pass the obstacles illustrated in 3.18, there needs to be developed a mechanism for opening one of these rings as a time. Obstacle one is the anchor points connecting the ring structure to the mooring system. Obstacle two is the support structure connecting the outer ring with the inner ring. Obstacle three is the carrier rope attached to the cleats. On top of this, the linear actuators needs to be positioned in a specific way. This means that the SG-robot needs to be developed further in order to meet the specific requirements of the SPOKe geometry. Assuming that an independent locking mechanism for the two rings can be solved simply with one rotational actuator on each ring, the SG-robot would have a total of two rotary actuators and 14 linear actuators. This robot has high potential when considering both precision and force output.

Thruster

The geometric limitations affecting the two actuator-models described previously are due to the need of producing high friction. Having an actuator that is not dependent on friction would therefore aid in avoiding this problem. An under-water thruster, like a water-jet or propeller is therefore considered. This thruster can be placed on the outer part of the robot arm. One downside with having a thruster as actuator is the fact that the position of the arm is not a simple function of actuator input and actuator state. Another downside compared to the previous two, is that the thruster does not cause the manipulator to lock on to the ring structure. These two properties then needs to be achieved by other mechanisms. By using a thruster, the arm will only need one actuator in order to move in tangential direction. If the thruster can only produce force in one direction, two actuators are needed.

Gear and roller chain

The submerging in water will potentially cause shells and seaweed to grow on the ring structure. This will cause trouble for a propulsion system based on a rack and a pinon. If there are shells or other growth between the teeth in the linear gear, the connection between the gear and the linear gear can be poor. This can ultimately cause the gear to slip, and make movement difficult.

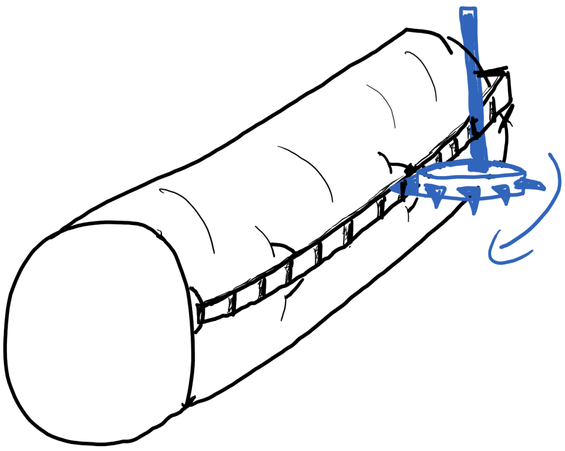


Figure 3.20: Gear and roller chain

An alternative design to using linear gear, is using a roller chain. The the roller chain is positioned along the ring structure with space between the ring and chain as illustrated in figure 3.20. When shells and seaweed grow on a roller chain, the gear will be able to operate the chain and push the growth out of it. By ensuring that there is space behind the chain, this mechanism could work even with a lot of growth. Growth pushed behind the chain will be pushed in to the space behind the roller chain and possibly detach. Using a rotary gear as propulsion system allows the position to be estimated accurately by a rotary encoder. As the gear locks the motor position to the ring, the movement can be done with high precision.

Comparison

In table 3.3, the four designs are listed and several aspects are compared. In the project report [1], using a gear with roller chain was not considered. The two designs based on clamping the ring structure (Trilateral wheel and Stewart-Gough platform), suffers from high complexity, while at the same time having a big challenge with obstacle avoidance. The report therefore concluded to proceed using a water-jet thruster as propulsion system. This will however require a complicated system of sensors for estimating the pose of the manipulator. When including the gear and roller chain in the comparison, it becomes clear that this is the alternative with the most benefits. It differs from the water-jet thruster design in that it allows for higher precision position estimates because it has an encoder locked to the gear. The movement in radial position can also be done using a gear and a roller chain.

Design	Trilateral Wheeled Robot	Stewart-Gough platform	Water-jet thruster	Gear with roller chain
Friction dependant	Yes	Yes	No	Yes
Number of actuators	14	16	1 (2)	1
Obstacle avoidance	Difficult	Difficult	Easy	Easy
Position estimate accuracy	High	High	Low	High
Complexity	High	High	Low	Low
Includes gripping mechanism	Yes	Yes	No	No

Table 3.3: Comparison of actuator models

3.6 Arm to centre of ring

As described in section 2.2, the rail is not parallel to the radial direction of the ring structure. This means that the rail is not positioned at the shortest distance from the inner- to the outer ring. If the rail was positioned strictly radially, and had length equal to the distance between the two rings, the rail would naturally keep the desired position. The rail would this way be kept stable because of geometry constraints. As the rail in reality is off-angle as illustrated in Figure 3.21, this property does not keep the position stable. Further, due to the geometry constraints illustrated in Figure 3.18, gripping all the way around the ring to keep the rail locked in the desired angle while moving will be difficult.

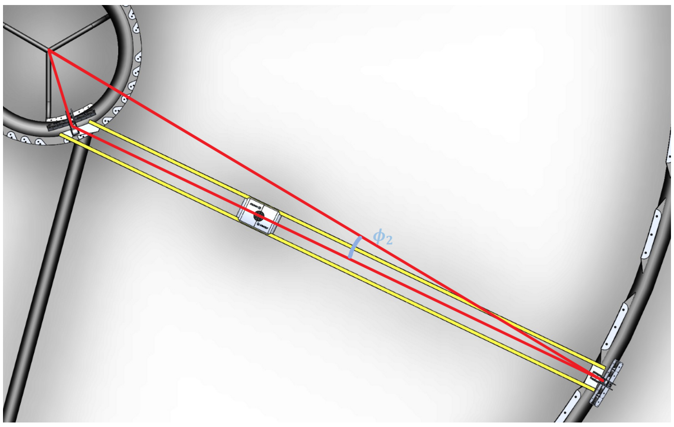


Figure 3.21: The angle offset between spoke and straight radial direction.

In order to fix this problem, the rail can be extended with an “arm” from the part of the rail close to the inner ring, to the centre of the ring. This way, the manipulator will have a point with a fixed position, and cannot move out of the desired angle. This arm can be connected to a pole fixed to the ring structure. The manipulator will still be able to move in angular direction, and the arm ensured that the manipulator keeps the desired offset angle ϕ_2. Another possible solution to this problem, is extending the arc length of the mechanism holding the rail connected to the ring structure. By having a sufficiently long arc length, the angle illustrated in Figure 3.21 will be held stable during movement about the ring structure.

Chapter 4

Prototype

A prototype was developed to implement the design choices in example 1 described in section 3.4.1. The main purpose of the prototype is to further explore the possibilities and limitations of the SPOKe concept. At this stage, it was decided that the prototype should be only a quadrant of the ring-structure. The mechanism for connecting a bolt through the rope in order to attach the rope securely is not implemented in this prototype. Instead, cleats are positioned both on the outer and inner ring. The cleats on the outer ring has teeth for holding the rope tight, while the cleats on the inner ring has no teeth. This prototype was not made to be waterproof as the cost of waterproofing would be out of budget.

4.1 Design

The prototype design is based on design considerations described in section 3.4.4, 3.5 and 3.6. A 3D model of the prototype was designed in the program Solidworks by Daniel Bogen. The size requirement for the prototype is that it should be able to fit through a normal-sized door, which is about 2.1 m tall. Restricting the radius to 2 m, the prototype has a radius six times smaller than what is suggested by Bale [6]. The number of cleats about the ring structure is equal what was used by Bale in his calculations. Many of the size-choices are based on the dimensions of the components available. Based on the 3D model, the physical prototype was also made by Danial Bogen.

The model of the prototype is illustrated in Figure 4.1. The material of both the ring and the rail is aluminium, and the wheels between the rail and ring are made up of plastic. In order to simulate the rope being fed from a vessel over the water surface, the prototype has a pole from which the rope is fed down to the tool. Power transfer to the gantry robot is connected to this pole.

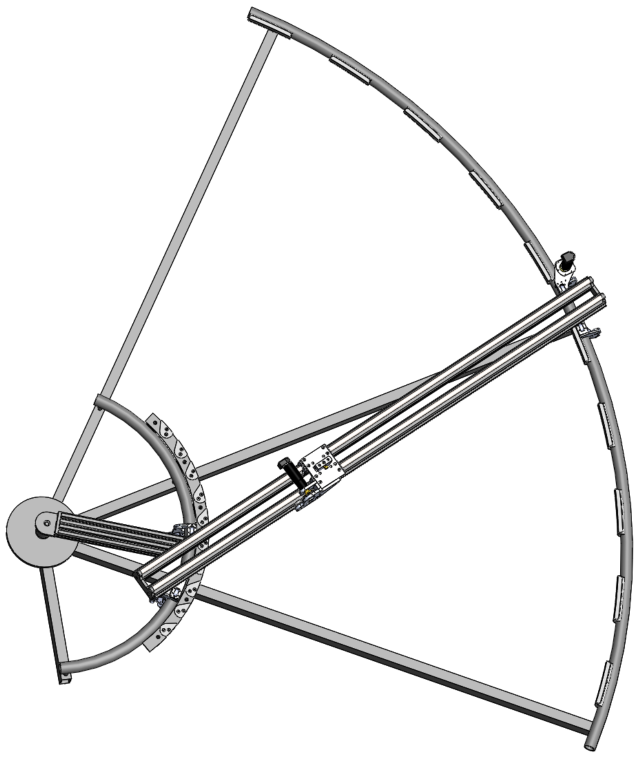


Figure 4.1: Model of the prototype.

The translation gear and roller chain mechanism is implemented for movement in both along the ring and along the rail. One roller chain is mounted along the outer ring, as shown in Figure 4.2a. The gear is positioned behind the plastic protection cover. The gear is pushed tight to the chain by springs. For movement pf the gantry robot, the roller chain is mounted along the rail. The chain is connected to the gear with plastic wheels on each side, as illustrated in Figure 4.2b, ensuring the gear and roller chain do not slip.

The mechanism for gripping the rope tight based on the design of a rope clamp with self-breaking functionality as shown in figure 4.3a. When the clamp is not closed, the rope moves though the tool with little resistance. When closed, the teeth on the clamp creates high friction to the rope, effectively slowing down movement of the rope. As the rope passes through the clamp when in locked state, the grip of the clamp tightens until the grip stops the rope from moving. This locking mechanism only locks the rope from moving in one direction and is disengaged by moving the rope in the other direction, or by pulling the clamp. The implementation of this locking mechanism is shown in figure 4.3b. A spring pulls the clamp, causing the clamp to lock by default. A stepper motor is used to push the grip to open and close.

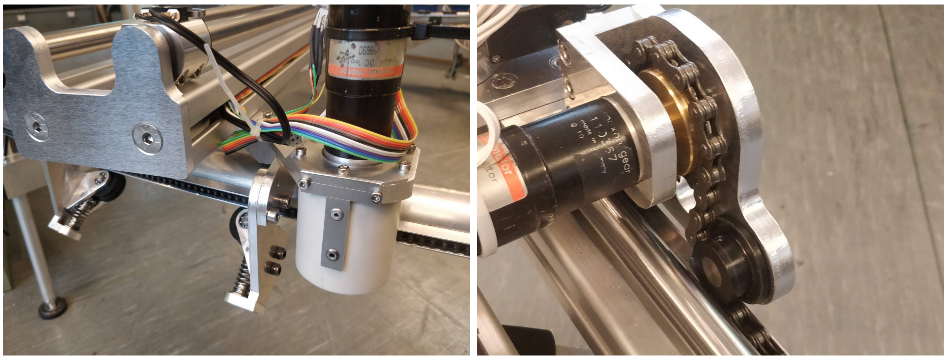

(a) Chain along outer ring. **(b)** Chain along rail, with gear.

Figure 4.2: Gear and roller chain implementations

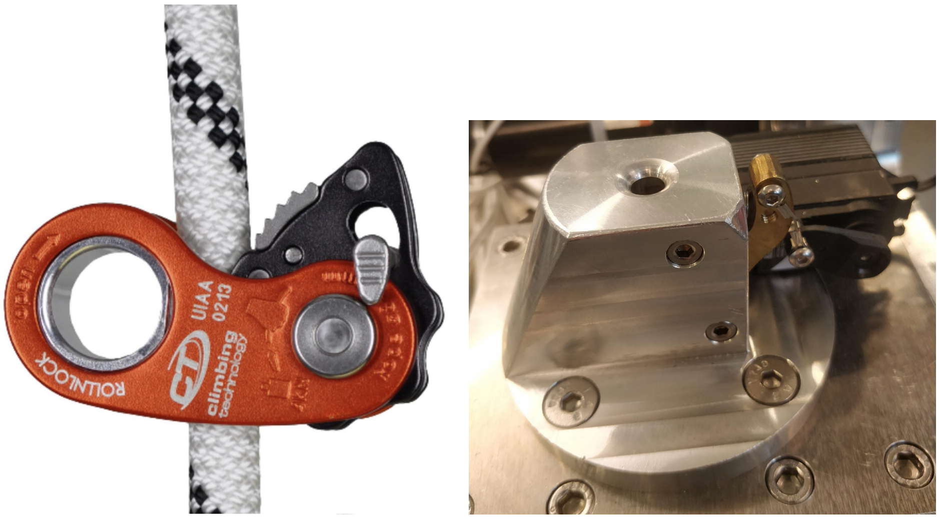

(a) Rope clamp [19] **(b)** Rope clamp on prototype

Figure 4.3: Rope clamps

The manipulator and ring structure are two separate components, and the manipulator can be taken on and off the ring structure. When the manipulator is positioned on the rail, it can be locked in place by a grip. There are four grips on the manipulator, two on each end of the rail. These grips are locked in place by a screw acting as a hinge, and a second screw locking the grip in place. By taking out the bolt, the grip can move about the hinge, and be locked in two different configurations, as shown in figure 4.4.

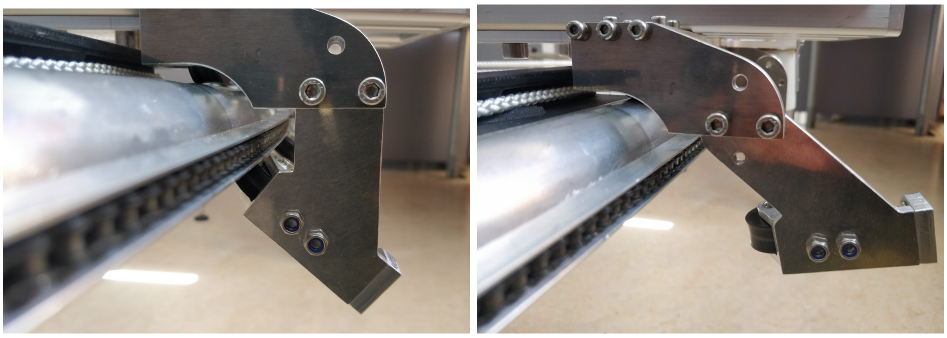

(a) Grip in closed configuration. **(b)** Caption

Figure 4.4: Prototype implementation of grip attaching manipulator to ring.

4.2 Electric hardware specification

The prototype is equipped with motors and sensors to enable controlling the movement. The electronics is powered by a 30V power supply.

4.2.1 Motor and limit switches

The motors used to control the movement of the robot both in radial and angular direction are the 90W graphite brushed motors RE 35 from Maxon motor. The motors have a nominal voltage of 30V. These motors can be controlled by controlling their input voltage. The stepper motor HSR-5990TG is used to push the grip to open and close. This supply voltage for this stepper motor is in the range 6-7.4V, and it can be controlled using a 50 Hz PWM signal. A stepper is a closed loop system with an integrated regulator. The pulse-width of the control signal for these types of motors is mapped to a specific shaft angle. At both ends of the rail, there is positioned limit switches, signalling the controller when the gantry robot is positioned at the end of the rail. There are also one limit switch on each side of the arc length. These sensors are used under initialisation of the system to get to a defined position. The limit switches are also used to ensure the controller does not destroy the manipulator by forcing movement beyond what is physically possible without breaking the prototype.

4.2.2 H-bridge

A h-bridge is used to control the voltage input of the motors. A H-bridge is an electric circuit that can control the sign of the output-voltage when given an input voltage, and two control signals. The H-bridge can also stop the voltage from flowing through it, stopping the motor. This allows the motor to be controlled to run in both directions, and to stop. In order to control the force created by the motor, the power transferred to the motor is controlled by turning the power output of the H-bridge off and on again quickly. This reduces the mean power transfer to the motor, making it possible to effectively control the force it produce. This power switching is done by the means of a Pulse Width Modulated (PWM)

A	B	M_A	M_B
1	1	0	0
0	0	0	0
1	0	VCC_M	GD_M
0	1	GD_M	VCC_M

Table 4.1: H-bridge port value table

signal. A PWM-signal illustrated in Figure 4.5, is a rectangular pulse wave, described by a frequency and a pulse width. By having a high frequency PWM-signal controlling the H-bridge, the motor behaves similar to receiving a constant power input equal to the mean power delivered by the PWM-controlled H-bridge.

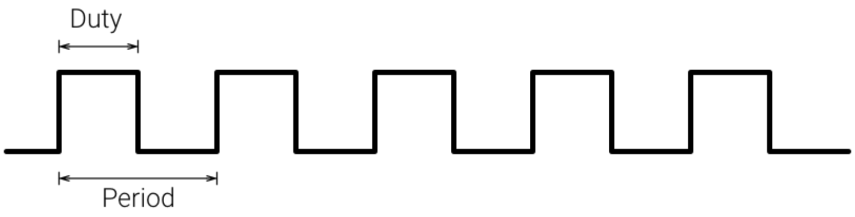
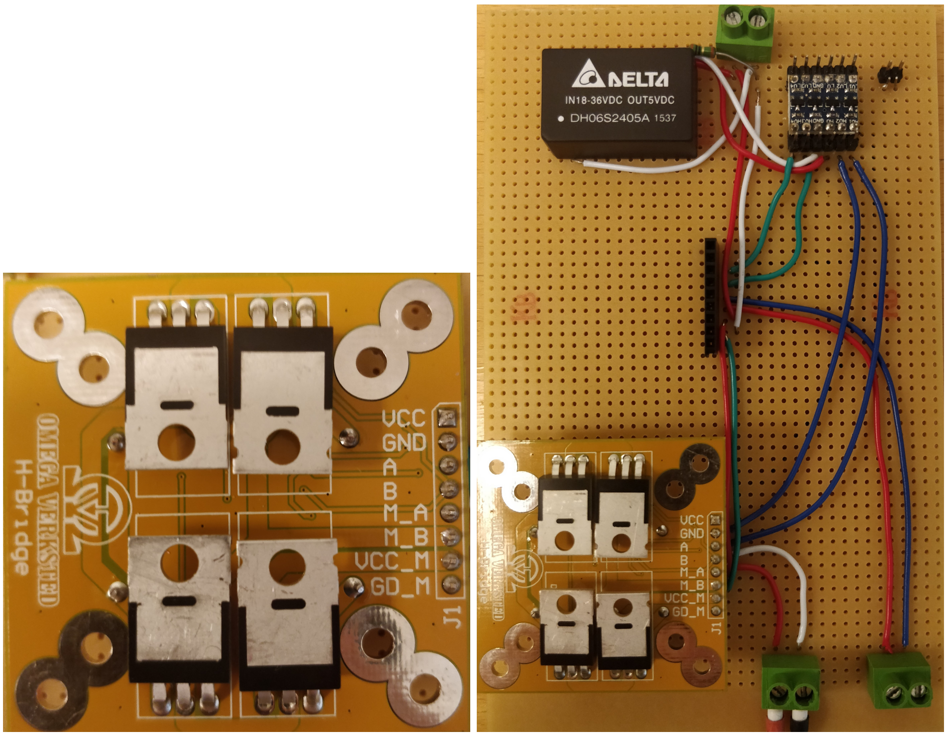


Figure 4.5: PWM signal [20], frequency = 1/period.

The H-bridge used for this prototype is the ”Omega Verksted H-bridge”, rated for 19A and 100V shown in Figure 4.6a. This H-bridge uses four MOSFETs in order to control the voltage, and two switches in order to protect against short circuiting. The high voltage signal for the motors are connected to port VCC_M and GD_M. The ports A and B control the polarity of the output signal. When both ports have equal value, the output signal is zero. The functionality of the A and B ports is of the H-bridge is given in table 4.1. The PWM-signal is connected to port VCC.

This specific H-bridge requires 5V signals for the A and B ports. As the GPIO-ports of the Raspberry Pi B+ has signals of 3.3V, the signal voltage needs to be amplified. A way to do this is using a logic-level translator. For this prototype, a bi-directional logic level translator is used. This translator takes in 3.3V, 5V, ground and is able to translate four individual signals from 3.3 V to 5 V. This is enough to control both H-bridges. The 5V power signal is produced by a 30V-5V DC-DC converter. The circuit containing these components is illustrated in Figure 4.6b

The two LED components on the H-bridge caused a voltage drop which made it impossible for this setup to properly set the A and B ports high. The two LEDs are therefore not added to the prototype, and the circuit is instead open.



(a) Omega verksted H-bridge. (b) Motor driver card.

Figure 4.6: Control cards for the two DC motors.

4.2.3 Gear and Encoder

The HEDS 5540 encoder by Maxon Motor with a resolution of 500 is mounted on the back of the motor. The gearhead "GP 32 A" from Maxon Motor, with a reduction of 23 is mounted on the motor. By counting both rising and falling edges of the encoder, this gear-encoder combination allows the system to read the motor position with a resolution of 46'000 ticks for one rotation. This resolution is higher than what is required for this system, and can be scaled down in software. The gear reduction causes the index signal Z from the encoder to be activated 23 times for each rotation of the shaft.

4.3 Dimensions

The dimensions of the prototype are listed in table 4.2. The parameters are related to the prototype as illustrated in Figure 2.4, 2.5, 2.7, 2.3a and 3.21. The gear radius R_{m2} is not measured, but is estimated using equation 2.12 and 5.15. By moving the gantry robot along the rail, the total change in shaft angle γ is estimated based on the data received from the encoders and the change in r_2 is known to be l_total. The radius of the gear moving along the ring structure was not estimated as the encoder data from that manipulator could not be received in time.

Parameter	Value	Unit
r_3	0.519	m
r_4	2.01	m
l_{total}	1.711	m
ϕ_1	2.32	rad
ϕ_4	0.63	rad
R_{m1}	0.0286	m
R_{m2}	Not estimated	m
l_1	157	mm
l_2	60	mm
α_1	0.078	rad
α_2	0.116	rad

Table 4.2: Dimensions of prototype

4.4 Controller

The hardware has several requirements of the controller: 3 PWM signals, 4 digital outputs, 6 digital inputs and the ability to read encoder signals from two rotary encoders with quadratic encoding. On top of this, it is desired to have a graphical user interface that can give the user information about the state of the manipulator. The user should also be able to start and stop the process by the push of a button.

The controller chosen for this prototype is the Raspberry Pi B+, with the Monarco HAT (Hardware Attachd on Top) extension. The Monarco HAT extension gives the Raspberry more precise input and output operations and mechanisms for handling short-circuits and reverse polarity. The raspberry is practical to use in the prototype stage of development. Other solutions considered is the use of an Arduino or a portable data acquisition module (PDAM), for use with a computer over usb. Both these two alternatives are dependant on a computer for writing, storing and compiling the program for the system. As this prototype should be as stand-alone as possible, the Raspberry Pi with the Monarco HAT was decided to be the solution most appropriate for this prototype.

4.4.1 Raspberry Pi

The raspberry Pi B+ is a small computer that among other IO-ports has USB, wireless network card, HDMI. This computer also has 40 accessible pins, including 27 general purpose input and output (GPIO) pins. The Raspberry Pi B+ also has dedicated connection points for camera and multi-touch-screen. The computer can run most common programming languages, and the GPIO ports are easy to control.

Port type	Number of ports	Description
Digital IN	4	3.5-30 VDC
Analog IN	2	0-10V / 0-20mA, 12-bit
Digital OUT	4	1 A, 5 V
Analog OUT	4	0-10 V, 20 mA
1W 5V OUT	1	

Table 4.3: Available ports on the Monarco HAT

This computer lacks some functionality required by the physical prototype. It does not support output signal modulated with Pulse Width Modulation (PWM) or quadrature signal decoding.

4.4.2 Monarco HAT

In order to extend the functionality of a Raspberry Pi computer, a standard for add-on boards has been introduced named “Hardware Attached on Top” HAT. One module using this standard is the Monarco HAT. The Monarco is equipped with its own micro-controller that enables functionality that is not available in the Raspberry Pi. This includes hardware counters that can be used for counting the signals from the encoders. The digital output ports supports signals modulated using PWM for frequencies up to 100 kHz, up to 16 bit resolution. The communication between the Raspberry Pi and the Monarco HAT is done over SPI. The open source library for the Monarco is created for the programming language C. Based on this, Bjarne Kvæstad has created a wrapper for the programming language Python. This wrapper is what is used in the software for the prototype.

The Monarco HAT communicates with the Rasperry Pi using the 40 pin header. Relevant ports available on the Monarco Hat are described in table 4.3.

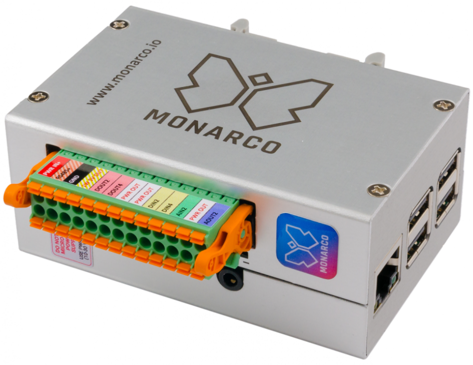


Figure 4.7: Monarco hat [21].

Not all Raspberry GPIO-ports are used by the Monarco. GPIO-Ports not used are 4, 7, 12, 13, 16-19, 22, 25 and 27. These are available for other purposes, and will be used for the H-bridge to control the voltage sign. They are also used to measure the state of the limit switches. The Monarco and Raspberry Pi ports used are listed in table 4.4. For the Raspberry pi there are two different ways of numbering the pins, either by their position on the board or by their GPIO number. The connection between the different electrical components are shown in figure 4.8.

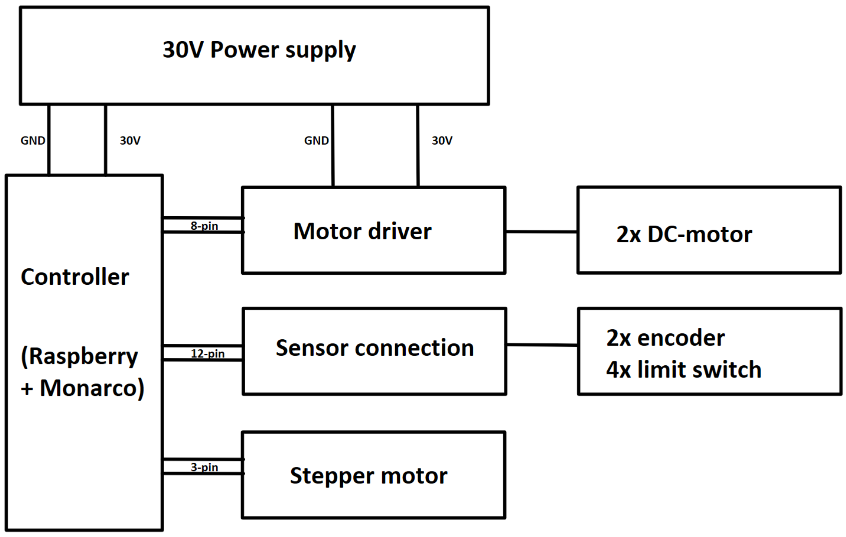


Figure 4.8: Connection of electronics scheme.

4.4.3 Raspberry Pi Touch Display and Kivy

For user input and displaying system state, the 7 inch official Raspberry Pi touch display is used. This display takes 5V from the dedicated display power pins named P4 on the monarco HAT. The communication between the screen and the Raspberry Pi B+ is done over the display serial interface (DSI) screen-port. The display has a refresh-rate of 60 fps, and has multi-touch capabilities for up to 10 inputs.

Kivy is an open source Python library developed in order to work with multi-touch displays such as the Raspberry Pi touch display. This library is is well documented and enables quick and easy interface development. The library has several modules for creating a layout on the display on which different elements can be positioned. The layout used in this prototype is the module ”BoxLayout”. The layout can be filled with different elements such as sliders and buttons for touch input, or labels and plots for displaying information.

The display on this prototype is illustrated in figure 4.9 and has a plot showing reference and estimated position for both r_2 and θ_4. One label displays the numeric values of the current estimation of r_2 and θ_4 and another label shows the current state of the system,

Hardware	Port	Usage
Monarco	+24V IN	+30V power supply
Monarco	GND IN	GND power supply
Monarco	DOUT1	Gantry motor PWM
Monarco	DOUT2	Ring motor PWM
Monarco	DOUT4	Grip motor PWM
Monarco	DIN1&DIN2	Gantry encoder A&B
Monarco	DIN3&DIN4	Ring encoder A&B
Monarco	1W 5V OUT	Grip motor VIN
Monarco	GND	Grip motor GND
Raspberry Pi	15(22)&22(25)	Gantry motor A&B
Raspberry Pi	13(27)&11(17)	Ring motor A&B
Raspberry Pi	32(12)	Limit switch 1
Raspberry Pi	33(13)	Limit switch 2
Raspberry Pi	35(16)	Limit switch 3
Raspberry Pi	36(19)	Limit switch 4
Raspberry Pi	7(4)	Gantry encoder Z
Raspberry Pi	12(18)	Ring encoder Z
Raspberry Pi	2(+5VDC)	Encoder power
Raspberry Pi	1(+3.3 VDC)	logiv-level converter LV
Raspberry Pi	6(GND)	logic-level converter GND
Raspberry Pi	9 (GND)	encoder and limit switch GND

Table 4.4: Table of ports used by the Raspberry Pi and the Monarco HAT.
For Raspberry Pi notation is BOARD(GPIO)

described further in section 5.2. The display also has two buttons, one green button acting as both initialising and start button, depending on what state of the system. The other button is a red stop button. The display also has a slider, making it possible for the operator to change the finishing time used by the trajectory planner to calculate the trajectory, as described in section 5.3.

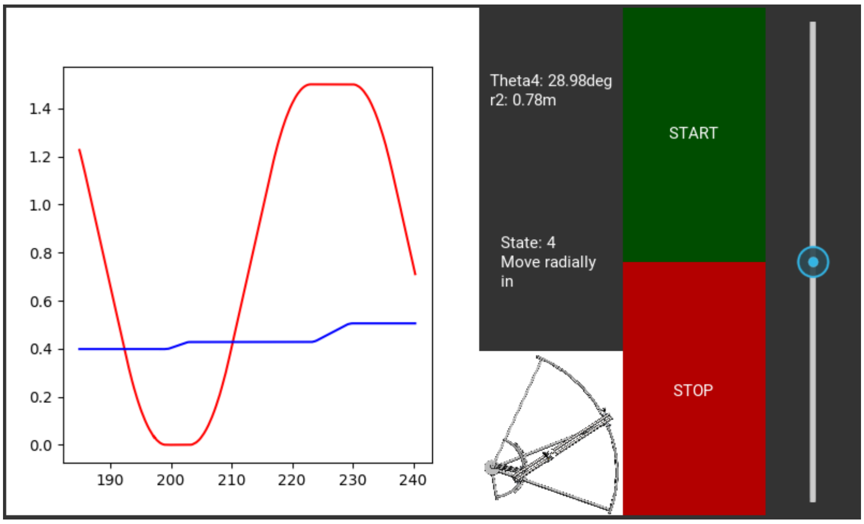


Figure 4.9: Touch display layout.

Chapter 5

Control of robot manipulator

The goal of the SPOKe concept is to deploy kelp seeded rope and harvest the full grown kelp in partly automated operations. In the full size SPOKe product, there are many operating tasks that needs to be done in order to reach this goal. The tasks required for deployment alone are summarised in the bullet-points below.

- The seeds must be attached to a carrier rope.
- The SPOKe manipulator that is stored on the boat has to be lowered and attached to the ring structure.
- The seeded rope must be fed to the manipulator.
- The two ends of the rope has to be attached securely to the ring structure.
- The manipulator has to attach the carrier rope to the cleats on the ring structure.
- The manipulator must detach from the ring structure and be lifted up to the boat.

In addition to this, when harvesting the kelp, three additional tasks must be perforemed:

- The manipulator must detach the two ends of the carrier rope.
- The kelp must be harvested from the carrier rope and stored.
- The carrier rope must be detached from the anchor points on the ring structure

This work will focus on the task of attaching the carrier rope to the cleats on the ring structure. It is assumed that the first end of the carrier rope already is attached before the operation begin. Attaching the other end of the carrier rope to the ring structure is also not considered.

5.1 Control problem in general

Before the operation starts, the robot has to be positioned in a defined state. This is achieved by moving the manipulator to a position where two limit switches both are activated. The limit switches have a fixed position and defines the state of the manipulator.

In order to attach the carrier rope to the mounting points created by the clam cleats on the ring structure, the manipulator must operate both in radial and angular direction. The tool of the manipulator has to move the carrier rope between the mounting points on the inner and outer ring without hitting them. This involves moving the tool both outwards and inwards. The manipulator also has to direct the rope around the clam cleat on the outer ring in order for the rope to connect and attach. The rope must also be moved around the cleat on the inner ring. The size of the clams must be known and represented as angle relative to the ring structure. The inner clams and outer clams will not necessarily have the same angle. In order to tighten the rope attached to the ring structure, the manipulator has to grip the rope and tighten it about the cleat.

5.2 State machine

A state machine is a method of categorising movement in tasks, forming a high abstraction level description of the operations performed by a system. The state machine describes the high level operations done by each state, and the requirements for transitioning from one state to the next. This allows for developing the details of the operations done by a state independently of the other states. It is also a tool for describing the top level functionality of a system, without requiring detailed description of implementation.

The movement described in the previous section can be considered a motion along the shortest path from one point to the next, in addition to some movement that tightens the rope about the clam cleat. The SPOKe concept is designed in such a way that the spokes are parallel to the rail when the rail is positioned at the anchor-point for the rope, illustrated in Figure 5.1. This simplifies the point to point motion in all states and allows for point to point motion in only one dimension at each state. Motion between the cleats are then represented by movement solely in r_2, while motion about the cleats can be represented by movement solely in angular direction θ_4. These operations can be described by a state machine as illustrated in Figure 5.2. Illustration of the behaviour of the six states are shown in Figure 5.3 and 5.4. In state 2, the manipulator has to a distance equal the length of the outer clam plus the diameter of the tool, in order to move past the clam. The angle θ_4 must then increase by $\alpha_1 + d_t/r_4$. As the manipulator in state 5 moves about the inner clam, the change in θ_4 must be equal $\alpha_2 + d_t/r_3$. Both in state 4 and 5, the starting point of the calculated trajectory is unknown until entering the state, as the value of r_2 is dependent on when the rope has sufficient tension.

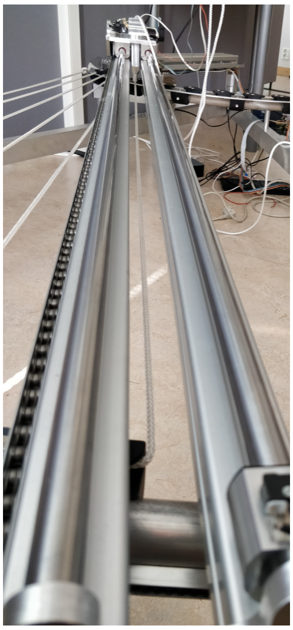


Figure 5.1: Spokes are parallel to the manipulator.

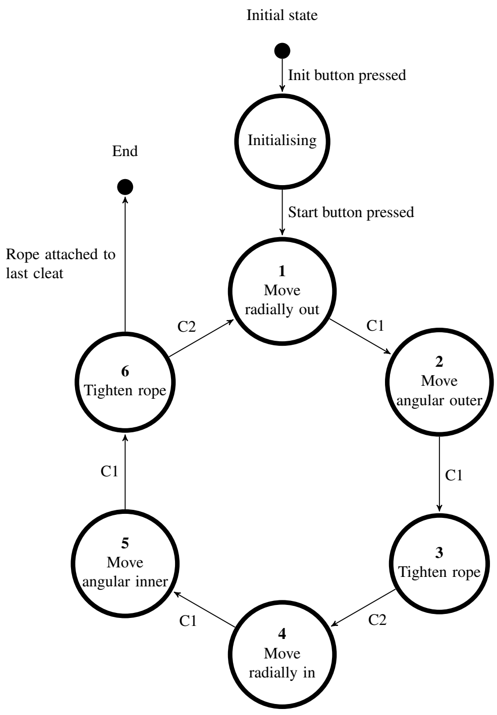


Figure 5.2: State machine for manipulator operation

Transition condition	Description
C1	The controller is at the end of the trajectory and the error on the two state variables are less than a specified threshold.
C2	The system detects that the velocity in r_2 is slowing down, while the signal to the motor is constant.

Table 5.1: State machine transitions

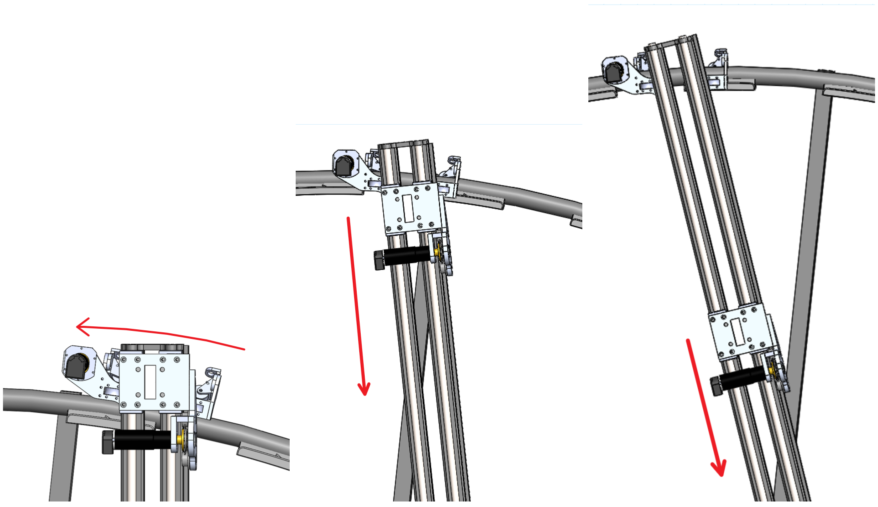

(a) State 2 **(b)** State 3 **(c)** State 4

Figure 5.3: The three states attaching the rope about the clam on outer ring.

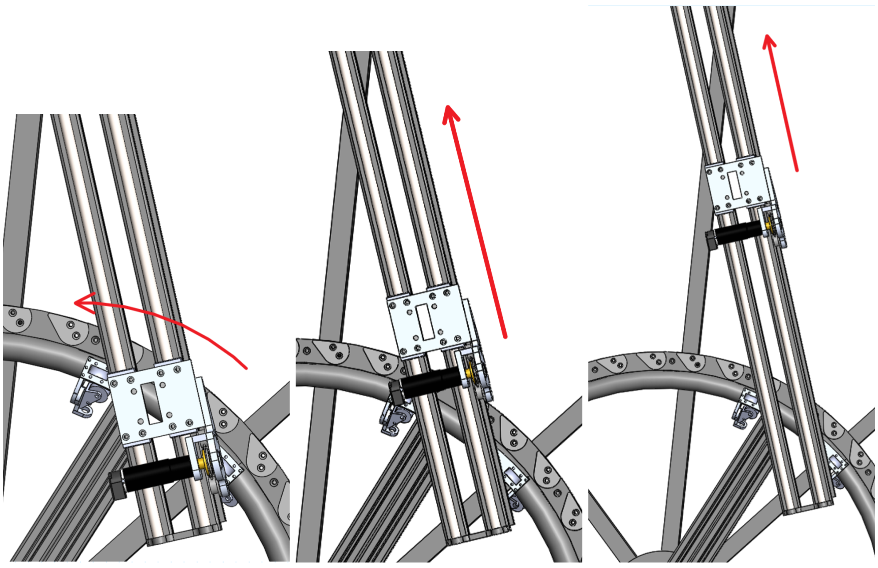

(a) State 5 (b) State 6 (c) State 1

Figure 5.4: The three states attaching the rope about the clam on inner ring.

The system starts off in an initial state where the configuration of the robot is unknown. The system will stay in this state until the initialise button is pressed. The system then goes into initialising state, where the manipulator moves to a known configuration. The manipulator remains in this configuration until the start button is pressed, which makes the state machine transition to the first state in the operational-state loop. The conditions for transition between the operational states are shown in table 5.1. When the system has moved along the entire path it transitions to the end state, where it stops.

5.2.1 Error States

During the operations of the system, there are several error states which the system can reach. In order to represent the transition between the normal operation of the system and the error states, all the states 1-6 are represented as a super-state named ”Operational”. The state machine describing the error states is illustrated in Figure 5.5. The transitions between the states are explained in table 5.2

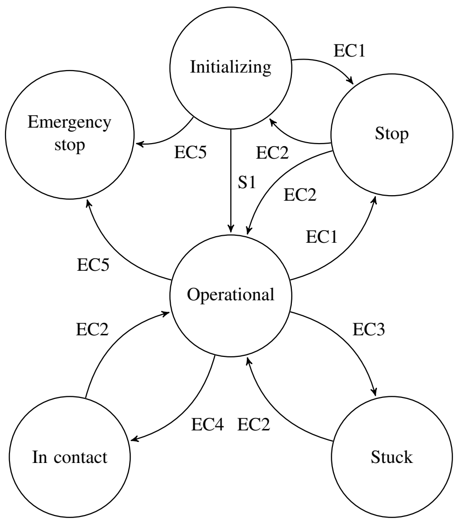


Figure 5.5: State machine for error states

Transition condition	Description
S1	Start button pressed
EC1	Stop button on touch display pressed by operator.
EC2	Start button on touch display pressed by operator.
EC3	Movement slowing down relative to control input, or current spike to motor.
EC4	Contact sensor activated.
EC5	Emergency button pressed.

Table 5.2: Transitions for error state machine

Stuck

The movement of the manipulator can be obstructed, rendering the robot unable to move in a certain direction. There can be several causes of this obstruction, for example the carrier rope may be stuck and not being fed to the robot, an object may get stuck in one of the chains or an object may be positioned in the trajectory of the robot. This condition can be detected by measuring the current flowing through the motors, and detecting a motor current above a certain threshold. If the system reach this state, the manipulator should stop, and the person operating it should find the source of the error. Once the error source is handled, the system should continue its work by a press of the start button on the display. This sensor was not implemented to the prototype. Using a hall-effect based sensor connected to the analogue input ports on the monarco is a good and non-invasive way of measuring the current to the motors.

In contact

During the operations done by the manipulator, the position is estimated based on the signals form the encoder. As the controller might miss some of the signals from the encoder, this position estimate can drift away from the actual position. This drift can at one point be significant enough to cause the manipulator to try to move the tool through the cleats. If the cleats are strong enough, this can be detected as the error state Stuck. If the cleats are weak however and printed in PLA plastic as the ones on the prototype, the cleats may be destroyed before the stuck condition is met. This state can be implemented like the stuck state, only with different a transition condition. This condition can be sensor input from a simple contact sensor positioned on the tool of the manipulator. A sensor like this is however not implemented to the prototype.

Stop

During the operation of the manipulator, the person operating the system should at all times be able to stop the movement. This can be done by pressing the ”stop” button on the touch display. When the system is in the stop state, the motors are turned off. By pressing the start button, the system transitions back to the state in which it left when the stop button was pressed.

Emergency stop

In the emergency stop state, power to the motors is stopped by a physical power switch. This button can be pressed at all times and is the most secure way for the operator to stop the manipulator movement as it is physical and involves no software.

5.3 Trajectory of Point-to-point motion

As the robot moves through the states, the movement of four of the six states can be described as point to point trajectories as described in section 1.2.7. The trajectory is calculated as the robot enters the state based on the configuration of the robot. As the two dimensions r_2 and θ_4 are linearly independent, the trajectory of the two dimensions can be calculated independently. This section will describe how the trajectory is calculated in one dimension.

By describing the trajectory as a linear segment with parabolic blends (LSPB), the manipulator can accelerate, hold a constant velocity before it decelerates to full stop. The concept of LSPB is described in section 1.2.7. The trajectory is described by three equations, one for each of the three segments of the trajectory illustrated in figure 5.6, where the segments are separated by dotted lines. When calculating the LSPB trajectory, constraints are set on initial conditions, final conditions as well as the constant velocity. When calculating the parabolic components of the trajectory, parabolic blend time needs to be defined, as it is one of the parameters used to calculate the parabolic trajectory. Blend time is the amount of time the parabolic components of the trajectory takes from start to finish. This way, the blend time dictates how quickly the trajectory accelerates. As the LSPB trajectory is

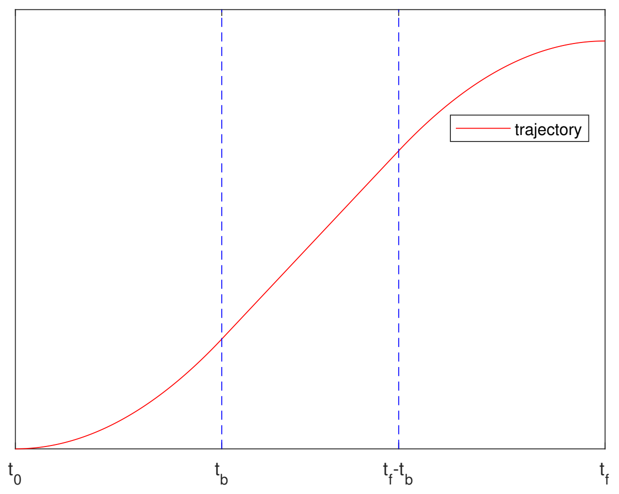


Figure 5.6: Plot of a LSPB trajectory.

defined by an initial configuration q_0, final configuration q_f, the constant velocity V as well as the start and end time, the blend time is dictated by the relation between these variables. Assuming the blend time is equal for both parabolic blends, it can be calculated by equation 5.1.

$$t_b = \frac{q_0 - q_f + V\dot{(}t_f - t_0)}{V} \tag{5.1}$$

By describing the parabolic blends as a quadratic differential function, the first blend function can be calculated by spesific values for initial position, initial velocity and the desired constant velocity after the blend. These constraints on the quadratic trajectory can be described in matrix form as shown in equation 5.2. The three variables can be solved for, using the three conditions as shown in equation 5.3 and 5.4. The trajectory can then be described by equation 5.5. In these equations, $q_0 = q(t = t_0)$ represents the positional variable at time zero.

$$\begin{bmatrix} 1 & t_0 & t_0^2 \\ 0 & 1 & 2t_0 \\ 0 & 1 & 2t_b \end{bmatrix} \begin{bmatrix} a_{10} \\ a_{11} \\ a_{12} \end{bmatrix} = \begin{bmatrix} q_0 \\ v_0 \\ v_b \end{bmatrix} \tag{5.2}$$

$$\mathbf{M_1 a_1} = \mathbf{b_1} \tag{5.3}$$

$$\mathbf{a_1} = \mathbf{M_1^{-1} b_1} \tag{5.4}$$

$$q(t) = a_{10} + a_{11}t + a_{12}t^2 \quad \text{, for t} \in [0, t_b] \tag{5.5}$$

The trajectory for the segment of linear velocity is calculated by the equations 5.6 and 5.7.

$$\mathbf{a_2} = \begin{bmatrix} a_{20} \\ a_{21} \end{bmatrix} = \begin{bmatrix} \frac{1}{2}(q_0 + q_f - Vt_f) \\ V \end{bmatrix} \tag{5.6}$$

$$q(t) = a_{20} + a_{21}t \quad \text{, for } t \in (t_b, t_f - t_b) \tag{5.7}$$

The last parabolic blend transitions the trajectory from constant velocity to a desired position and velocity. The constraints for this part of the trajectory is the constant velocity at time $t = t_f - t_b$ and a defined position and velocity at time $t = t_f$. These constraints are described in matrix form in equation 5.8 and the unknowns can be solved similar to equation 5.4. Here $v_{f-b} = v(t = t_f - t_b)$. The resulting trajectory is given in equation 5.9.

$$\begin{bmatrix} 0 & 1 & 2(t_f - t_b) \\ 1 & t_f & t_f^2 \\ 0 & 1 & 2t_f \end{bmatrix} \begin{bmatrix} a_{30} \\ a_{31} \\ a_{32} \end{bmatrix} = \begin{bmatrix} v_{f-b} \\ q_f \\ v_f \end{bmatrix} \tag{5.8}$$

$$q(t) = a_{30} + a_{31}t + a_{32}t^2 \quad \text{, for t} \in [t_f - t_b, t_f] \tag{5.9}$$

The final trajectory is a function of these three trajectories, and is given in equation 5.10. The seven constraints required to calculate this final trajectory are $q_0, v_0, t_0, q_f, v_f, t_f$ and V. An alternative to using the desired constant velocity as a constraint is using the blend time as a constraint. The desired constant velocity can then be calculated as a function of t_b using equation 5.1.

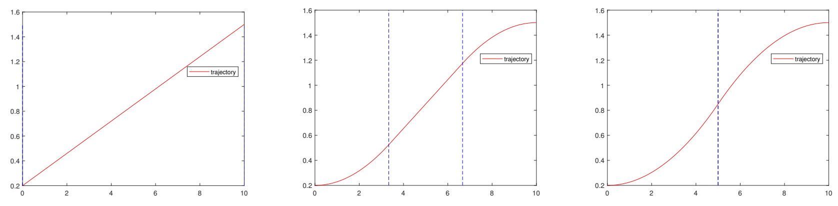
$$q(t) = \begin{cases} a_{10} + a_{11}t + a_{12}t^2 & \text{, for } t \in [0, \quad t_b] \\ a_{20} + a_{21}t & \text{, for } t \in (t_b, t_f - t_b) \\ a_{30} + a_{31}t + a_{32}t^2 & \text{, for } t \in [t_f - t_b, t_f] \end{cases} \tag{5.10}$$

Limitations for the constraints

In order to calculate a LSPB trajectory, the choice of constraints is not arbitrary. The blending time is bounded by lower and upper bounds as described by equation 5.11 [12]. If the blend time is larger than half of the total time for the trajectory, there will be no linear segment in the trajectory, and the trajectory will not be LSPB. If the blend time on the other hand is zero, the resulting trajectory is purely linear. Using equation 5.11 and 5.1, this restriction on t_b can be described as a restriction on the desired constant velocity as given in in equation 5.12. This restriction on the desired velocity is illustrated in Figure 5.7. In order for the trajectory to in fact be LSPB, the constraints therefore must be chosen according to equation 5.12.

$$0 < t_b \leq \frac{t_f - t_0}{2} \tag{5.11}$$

$$\frac{q_f - q_0}{t_f - 2t_0} < V \leq \frac{2(q_f - q_0)}{t_f - t_0} \tag{5.12}$$



(a) $V = \frac{q_f - q_0}{t_f - 2t_0}$ **(b)** $V = \frac{1.5 q_f - q_0}{t_f - t_0}$ **(c)** $V = \frac{2 q_f - q_0}{t_f - t_0}$

Figure 5.7: LSPB for three different values of V.

5.4 Controller

The controller is based on the state machine described in section 5.2. In state 1, 2, 4 and 5, a point to point trajectory is created based on initial configuration when transitioning to the state, and the desired position at the end of the state. A PID controller is implemented for controlling r_2, and a separate PID controller is created for controlling θ_4. The output from the PID controller is calculated by equation 5.13 and 5.14. This controller uses the difference in position, velocity and accumulated difference in position between the estimated position and the reference, to create a control signal for the motor.

$$[h]u(t) = K_p e(t) + K_i \int e(t)dt + K_p \frac{de}{dt} \tag{5.13}$$

$$e(t) = q(t) - q_d(t) \tag{5.14}$$

The LSPB trajectory created for point to point motion is used as the reference to the PID controllers. At all times, only one variable needs a trajectory. The other variable has a constant desired position. The PID controller is chosen as controller in order to follow the time varying trajectory and to ensure the reference is reached, even when exposed to slowly varying external forces as the force acting on the manipulator from the rope. The PID controller used in the Python program is the PID library created by Caner Durmusoglu [22].

5.4.1 Program implementation

The controller described above is implemented in Python 3 in the code and is given published to a public GitHub repository [23]. The program is divided in the seven modules illustrated in Figure 5.8. The program is made up of two processes running simultaneously, the one controlling touch inputs and updating the screen is named SPOKe_GUI. This process registers button presses, movement of the slider and updates the plot, shown in Figure 4.9. When a button is pressed or the slider is moved, this information is transferred to the second process using classes from the Python module “multiprocessing”. The second process is a loop in the module named “SPOKe_Controller”. This loop is an implementation of the state machine described in Section 5.2. The SPOKe_Controller processes

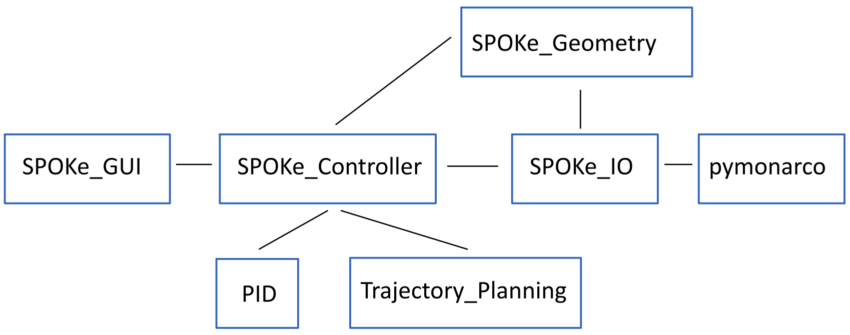


Figure 5.8: Python program architecture.

imports functions from the other modules. The SPOKe_IO module controls the ports of the Monarco HAT and the GPIO pins of the Raspberry Pi. It uses the pymonarco module to read sensor data and to control the speed and direction of the two DC motors, as well as the reference for the stepper-motor. The Geometry module contains specific geometric properties of the SPOKe prototype, and has functions for the geometric transformations described in Section 2.3. The module Trajectory_Planning is use to calculate LSPB trajectories as described in section 5.3. This module is also used to calculate the position along a given LSPB trajectory, at a given time.

Encoder implementation

As described in section 4.4.2, the Monarco HAT has functionality for decoding quadrature encoding signals. This implementation takes in the A and B signals and returns the number of counted signals using quadruple evaluation. As the encoder rotates clockwise the value increases, while it decrease as the encoder rotates counter clockwise. The value returned by the Monarco HAT is two bytes, which results in a value in the range 0 - 65'535. When the counter moves past the size limit, it continues to count from 0. As the counter counts backwards past the smallest value, it continues from 65'535. As the gear used has a reduction of 1:23, the encoder has a resolution of 500. Combined with the quadruple evaluation, each rotation of the shaft results in 46'000 ticks. This resolution makes the system able to detect position with a resolution of $360°/46'000 = 7.8° * 10^{-3}$. This resolution is a lot higher than what is needed by this system. The incoming counter value is therefore scaled down by a factor of 46, resulting in a resolution of $0.36°$.

The module "SPOKe_IO" receives the Monarco HAT counter value and use the difference in counter value between each call to determine what direction the encoder rotates, compensating for the situations where the Monarco counter overflows. The pseudo code for this behaviour is shown in algorithm 1 from line 4 to 13. In this algorithm the lines 1 to 3 describe globally accessible variables. If the absolute value of the difference between the newest value from the Monarco counter and the previously received value is greater than 23'000, corresponding to a rotation of $180°$, the most probable reason for this integer

Algorithm 1 update_counter

```
 1: previous_received_value
 2: counter_scaling_rest
 3: local_counter
 4: procedure UPDATE_COUNTER(ENCODER_NUMBER)
 5:     encoder_number
 6:     new_value = monarco.read_counter(encoder_number)
 7:     diff = new_value - previous_received_value
 8:     if abs(diff) < 23'000 then
 9:         increase = diff
10:     else if new_value < previous_received_value then
11:         increase = diff + 65'536
12:     else if new_value > previous_received_value then
13:         increase = diff - 65'536
14:
15:     counter_scaling_rest += increase % 46
16:     rest_overflow = counter_scaling_rest // 46
17:     increase += rest_overflow
18:     counter_scaling_rest -= rest_overflow * 46
19:     local_counter += increase // 46
20:     previous_received_value = new_value
```

overflow. If the new value is smaller than the previously received value, the rotation is positive and has passed the integer overflow value of 65'535. The real increase is therefore what is shown in line 11 of the algorithm. Similarly, if the new value is more than 23'000 greater than the previously received value, the counter has passed zero, counting down from 65'535. The real difference in value is therefore what is given in line 11.

Before the increase is added to the local counter in the "SPOKe_IO" module, the values are scaled down by a factor of 46. This scaling is shown in algorithm 1 in line 15 to 19. The down-scaled increase in count value is then added to a globally accessible variable named "local_counter". Finally, the algorithm stores the newly received value, so that it can be compared next time the algorithm is called.

Dictated by line 8 in the algorithm, the "Update_counter" function only works as intended if the shaft rotates less than 180° between each function call. This function must therefore be called often in order for it to work properly. As there are two encoders, one set of variables needs to be stored and used for each encoder. The accessible variable "local_counter" can be transformed to shaft angle in radians through eq. (5.15).

$$\theta_{shaft} = local\_counter * \frac{2\pi * 46}{46000} \tag{5.15}$$

As some ticks might not be read properly, an error can accumulate throughout the run-time of the system. In order to compensate for this, every time the index signal Z is high, the algorithm “index_value_high” shown in algorithm 2 is run. First the counter is updated by running algorithm 1. The variable firstZTickValue1 is equal to the value of local_counter1 the first time the index signal is triggered after initialisation. When the index signal is high while the encoder has counted a number close to a full rotation (1000/23), the value zCount1 is changed according to the direction of rotation, as shown in the lines 5 to 9 in the algorithm. This variable represents the number of full rotations for the encoder since the first time the index signal was high. This way it can be used to represent the correct number of ticks the counter should have counted at the exact time the index signal is high. If the counted values for the encoder is not close to a full rotation when z is high, the zCount1 variable remains unchanged. This is typically the case in a situation where the shaft rotation changes direction and hit the same index signal twice. Once the value for the index counter is updated, the local counter is set to the correct value for the given number of index signals. The counter scaling rest variable is also reset. As this algorithm is specific for encoder number 1, an algorithm with similar behaviour and its own variables is implemented for the second encoder.

Algorithm 2 index_value1_high

```
1:  firstZTickValue1
2:  ZCount1
3:  procedure INDEX_VALUE1_HIGH
4:      update_counter(1)
5:      diff = local_counter1 - firstZTickValue1 - zCount1 $* \frac{10'000}{23}$
6:      if diff $>$ 500/23 then
7:          zCount1 += 1
8:      else if diff $<$ -500/23 then
9:          zCount1 -= 1
10:
11:     local_counter1 = firstZTickValue1 + zCount1 $* \frac{10'000}{23}$
12:     counter_scaling_rest = 0
```

Chapter 6

Experiments

This chapter contains description of experiments that should be performed in order to validate the usability of the physical design of the prototype, usability of the sensory system, implementation of the state machine with LSPB trajectory, motor control and behaviour of the controlled system. In these tests, the threshold value for θ_4 in order to transition between states is 0.0023 radians. This corresponds to a positional accuracy of 5 mm along the outer ring. The threshold value for r_2 is set to 5mm.

6.1 Movement along trajectory without motor input

The first experiment addresses the physical movement functionalities of the prototype as well as the accuracy of the position estimation. The experiment involves moving the manipulator in angular direction from the limit switch on the right to the one on the left with r_2 constant. Next, the tool moves from the limit switch on the inner part of the rail, to the outer limit switch while the angle θ_4 is constant. During this movement, the encoder values will be used for position estimation. The resulting position estimate at the end of the two movements are recorded. These values should be the maximum possible value of θ_4 and r_2 respectively. This experiment is done 20 times, and the position estimate is reset before each experiment. As the movement is done by hand and not by motor control, the velocity along the trajectory will vary between experiments. The resulting positional estimates should not vary much, in order to be reliable enough to be used as inputs to the controller.

6.2 State machine trajectory

Assuming the controller is capable of keeping the position of the manipulator equal to the trajectory, the program should start, and create the entire trajectory required for the manipulator to attach the rope to the ring structure. The trajectory generation should be initiated by pressing a button on the touch screen. For the test to be success-full, all trajectories should be LSPB trajectories starting at the position of the manipulator when entering a new state, and ending at the desired position. For the gantry robot, this means that the trajectories will end at $r_2 = 0$ or $r_2 = l_{total}$. For the movement along the ring structure, this means that the desired position will increase with either $\alpha_1 + d_t/r_4$ or $\alpha_2 + d_t/r_3$. The trajectory should stop once the last clam has been reached, which is for $\theta = \pi/2$

6.3 Following trajectory in angular direction

This experiment involves calculating and following a trajectory varying in θ_4. This behaviour is the behaviour required by the system in state 2 and 5, as described in section 5.2. The trajectory is calculated based on the starting position and the desired final value. The total length of the trajectory is equal to the size of the outer clam cleat. Both the reference and the positional data for the system during the experiment is stored. This experiment is successful if the controller is capable of following the trajectory with little error and little over-shoot. The test ends when the transitional condition for the controller in state 2 and 5 is met, namely that the trajectory is at an end, and the error in θ_4 is below the threshold of 0.0023 radians.

6.4 Tightening rope

The third experiment involves tightening the rope, as well as moving the tool along the rail. This movement is similar to what is required for the system to operate state 6 and 1 as it is described in section 5.2. This includes locking the grip around the rope, moving the tool to tighten the rope. As the current sensor is not implemented, the rope will be tightened using a constant signal to the DC motor of 10% duty cycle for 2 seconds before the grip is released. After this, the trajectory for r_2 should be calculated from the position of the manipulator after the rope is tightened to the desired final position. The manipulator should then follow this trajectory. The desired position for the angle θ_4 is constant. The positional data of the manipulator during this operation is stored, as well as the desired trajectory. For this test to be successful, the error in θ_4 should always be below the threshold of 0.0023 radians. The grip should be able to hold the rope tight enough to create a high tension for the rope before releasing it. The error in r_2 should be low while following the trajectory, and most importantly the gantry robot should not touch the limit switch on the end of the rail. The threshold value for r_2 of 5mm should be achieved before the experiment ends.

6.5 Rope deployment

The last experiment involves the entire process for deploying the rope to the ring structure from start to end. The system starts at state “Initial state”, with one end of the rope already attached to the inner circle. By pressing the init button, the manipulator moves to the position where the limit switch on the inner side of the rail and the right side or the ring is active. Next, the start button is then pressed, starting transitioning the system to state 1. The system then iterates through the states, until the rope is tightened about the last clam cleat. After this, the system transitions to the state “End”. For this test to be successful, the whole deployment task of the manipulator must be successful. This implies that the system is capable of attaching the rope to all the clams of the ring structure. The limit switches should not be touched under the experiment, and the clams should not be touched by the tool. When the rope is attached about the last clam, the manipulator should have a constant position in both r_2 and θ_4

Chapter 7

Results

The prototype was finished one week before the deadline for delivery of this master thesis. That week the Monarco HAT got destroyed due to a short circuit when switching the GPIO pins. As there was no time to receive new hardware in time, many of the results where not possible to attain. This section presents the positional data for r_2 in the first experiment, as well as the trajectory plots for the second experiment.

7.1 Movement along trajectory without motor input

The manipulator was able to move unrestricted in both r_2 and in θ_4 direction. The resulting position estimation at the end point of the movement in r_2 are illustrated in the histogram in figure 7.1. The size of the bins in this histogram is 40 cm. During three of the tests, the counter on the Monarco reset and stopped counting. The three bins only containing a single value contains the results for these unsuccessful tests. The resulting values for the successful tests are shown in the histogram in figure 7.2. The size of the bins in this histogram is 5 mm. Not counting the three faulty results, the mean value is 1.7114 m, with a variance of $6.8 * 10^{-6}$m. Not counting the outliers shown in the figure 7.2, the mean value is 1.7123 with a variance of $4.6 * 10^{-7}$m.

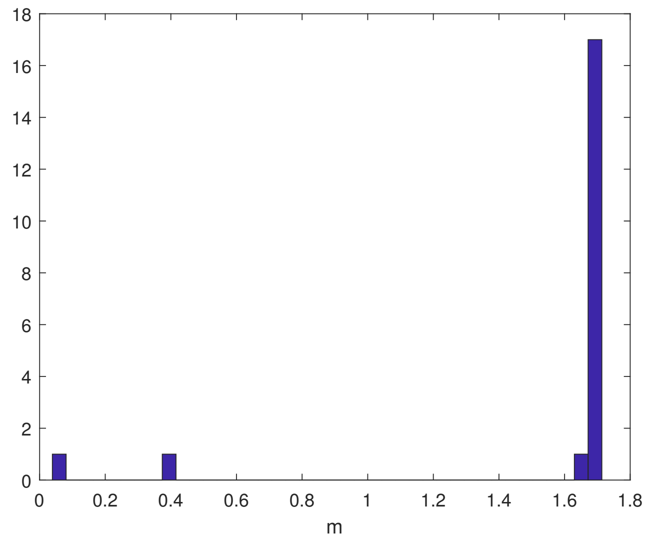


Figure 7.1: Position estimation in r_2 including results from tests with error.

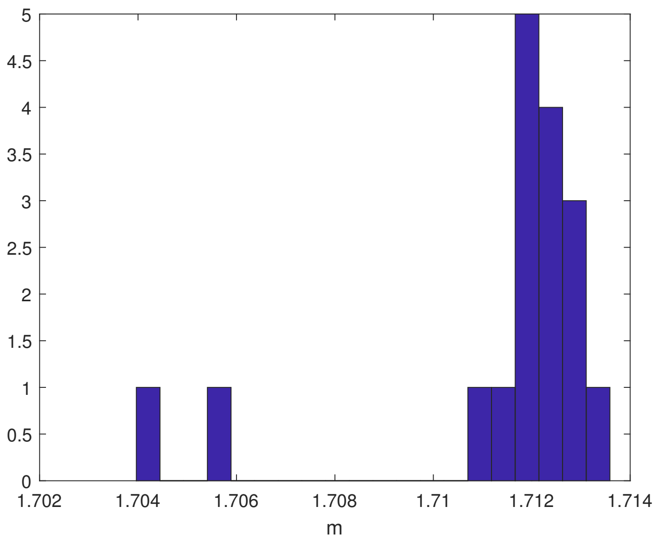


Figure 7.2: Position estimation in r_2

7.2 State machine trajectory

A plot of the trajectory for r_2 and θ_4 for the six first states is shown in figure 7.3. The trajectory for r_2 and θ_4 from start to beginning of the run-time of the program is shown in figure 7.3.

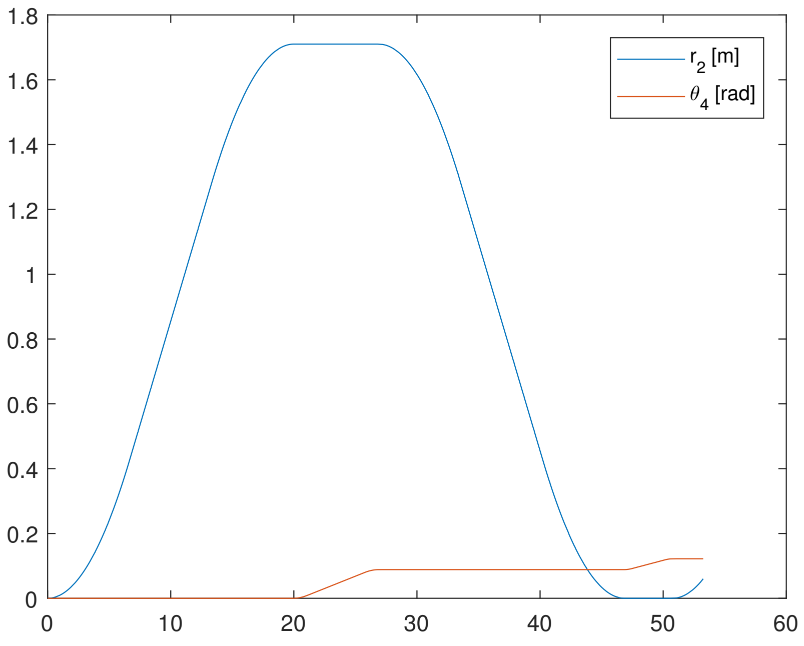


Figure 7.3: Trajectory for manipulator in r_2 and θ_4 though 4 states.

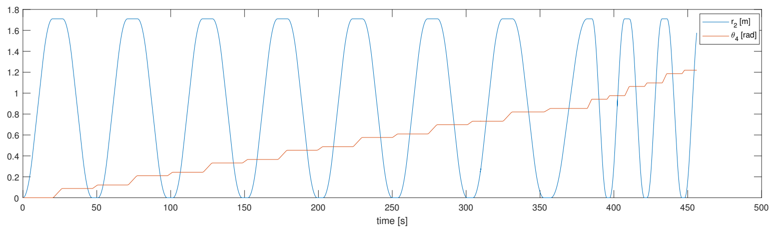


Figure 7.4: Trajectory for manipulator in r_2 and θ_4 from the beginning of the trajectory.

Chapter 8

Discussion

8.1 Prototype functionality

The prototype proved to be able to move the tool effortlessly along the rail, as well as moving the rail radially along the arc. This renders the prototype able to physically move along the desired trajectory for employing the carrier rope to the ring structure. This behaviour shows that the design of the prototype based on extending the rail with movement based on gear and roller chain is promising.

8.1.1 Encoder values

The decoding of the encoder signals resulted in an accurate estimate for the position at the end of the trajectory in r_2. The variance of the estimate was $7.2 * 10^{-3}$ mm, corresponding to 34 ticks. Among the 20 tests of the encoder, three of the tests where faulty. This was due to the counter on the Monarco HAT resetting during the run time. The counted values received from the monarco counter then ended up being the difference in counts since last function call, and not the accumulated ticks received over time. This resulted in the $SPOKe\_IO$ module not changing the position estimate after the reset, making the estimated position stuck at the last estimate calculated before the reset. Having a fail-rate of 15% is not satisfactory, and if this was to happen while the system was controlled by a controller, this would most likely result in a crash.

The encoder for θ_4 was not tested due to a short circuit destroying the Monarco HAT. A replacement Monarco was not attainable before deadline. This encoder is identical to the one used for measuring r_2 and the implementation of reading the two encoders is also identical. It is therefore reason to believe results for this encoder will be similar to the results achieved for the encoder on the gantry robot.

8.1.2 Motor control

The stepper was never tested with the 5V output of the Monarco. As the stepper motor is rated for voltages in the range 6-7.4 V, a separate power will most likely be required to control it. The 5V power supply for the stepper motor was chosen based on inaccurate information about the requirements for the motor.

When testing the H-bridge from "Omega Verksted", it became apparent that it did not work as expected. Using a PWM signal on the v_cc port did not make it possible to control the velocity of the manipulator. Instead, the motor had the same speed for all pulse widths tested for the PWM signal. By keeping the $v_c c$ port high, and using PWM signals for A and B, the H-bridge was able to both control the motor direction. As the Monarco HAT only has four digital out ports capable of sending PWM signals, and the two DC motors and the stepper motors requires a total of five PWM signals, it is not able to control all three motors. This problem can be solved in two different ways.

Switching circuit

Between the Monarco and the H-bridge, there can be positioned a switching circuit. The behaviour of this switch should be as described by table 8.1, and the switch time must be high enough to switch the PWM signal. This can be achieved using a dual Solid State Relay (SSR) as the ASSR-4128-002E illustrated in figure 8.1. This circuit requires an inverter, for example the SN74HC14AN. This way, a single PWM signal can be switched between the A and B pins on the H-bridge. As the A and B connectors, this circuit will give the behaviour required to control the H-bridge. One signal from the controller can then be used to decide the motor direction, and a single PWM channel can be used to control the motor velocity in both directions. The only change required to the Python program using this method is when controlling the motor direction; only one pin needs to be set, compared to two in the current implementation. As this is fairly easy to implement, this is the recommended solution to the PWM shortage problem.

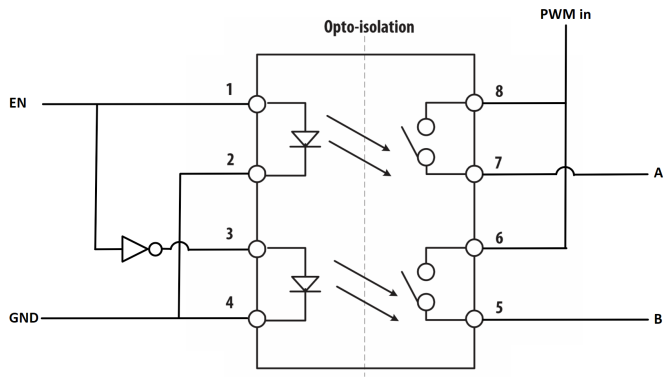


Figure 8.1: Switch circuit for control of H-bridge [24].

EN	A	B
1	PWM	0
0	0	PWM

Table 8.1: Switch behaviour

Micro-controller

Another approach for controlling the manipulators is using a micro-controller with at least four PWM-capable ports. A micro-controller capable of this is the widely used ATmega328P. Communication with the micro-controller could be made as simple as sending six signals; four for directional data for the two DC motors, and two PWM signals representing the desired pulse-width. The four directional signals can be interpreted as digital signals, and the PWM signal can be interpreted as an analogue input. This functionality can quickly be implemented in a micro-controller. A more accurate implementation would however be to use serial communication between the micro-controller and Raspberry pi. The Monarco HAT uses all the I2C capable pins of the Raspberry Pi, and these pins are advised not to be used for other purposes. Using the monarco serial communication using RS-485 is still possible. Using serial communication with the micro-controller, accurate data describing desired pulse width and motor direction can be communicated. Four PWM signals from the micro-controller can then be used to control the DC-motors. By implementing this using serial communication, code for the communication must be added to the controller running on the Raspberry Pi. The behaviour of the micro-controller must also be implemented.

Using a micro-controller for the implementation of motor control takes away one key advantage of using the Monarco HAT. Using this implementation, the only benefit of the Monarco is the decoding of the quadrature encoder signals. This can also be achieved on the micro controller. By implementing quadrature decoding on the micro-controller, there is no longer a use for the Monarco HAT in this prototype.

8.1.3 State machine

As the positional data for both r_2 and θ_4 was never received, the transitional conditions for the state machine described in section 5.2 could never be achieved and tested. By not requiring the position in r_2 and θ_4 to be below a certain threshold in order to transition between states, the trajectories created by the state machine could still be tested. As the transitional condition for state 3 and 6 is solely dependant on sensory data, the state-machine transitions away from these states as soon as it is transitioned to them. These two states therefore has no impact on the plot of the trajectory. The plot in figure 7.3 shows the desired trajectory for the first states. The plot illustrates that the trajectories are LSPB trajectories, and behaves as specified. Figure 7.4 shows that the trajectory behaves as specified for many iterations of the state loop described in section 5.2. From time $t \approx 380$, the speed of the trajectories is doubled.

8.1.4 Controller

The PID controllers for controlling the manipulator to follow the desired trajectories could not be tested. As the Monarco was short circuited and got destroyed, the control of the motors where never tested. The PID library implemented has however been tested for a motor controllable in both direction and velocity and worked as specified. Tuning the PID controller has to be done once the control of the motors is fixed. Once the control of direction and position of both motors can be achieved, simple control of the manipulator will be quick to implement as the trajectories are calculated by the state machine.

8.2 Prototype relevance for full scale

The prototype was developed in order to receive relative information about the needs and potential of a potential full sized implementation of the SPOKe system. There are however several differences to the test of the prototype compared to the proposed full sized system effecting the relevance of the prototype. These are listed below.

- The desired tension of the rope for this prototype is a lot smaller than what will be desired tension for the fully sized system. The relevance of the prototype depends on the ability to scale the propulsion system and rigidity of the system to enable the required increase in tension. The propulsion design for this prototype should be capable of scale to produce a higher force output without requiring much change.
- The rings of the prototype are made out of bent aluminium pipes. These have small irregularities compared to a perfect circle arc, and the rigidity of the aluminium cause the shape to not vary during testing. The proposed material for the rings of the full scale system is HDPE plastic with a diameter of 25 m. Depending on how these rings are reinforced, these ring shapes will deviate from a perfect circle in a varying extent. The prototype manipulator in this work is designed with springs to allow the ring structure to have some deviation from a perfect circle arc. However if the full scale system has large deviations, this design may not be satisfactory.
- The reinforcement of the prototype is made out of aluminium beams. This support structure for the SPOKe system is proposed to be made out of HDPE. The prototype was not intended to give data on the requirements of the support system for a fully sized system. When a model is made using HDPE, different considerations must be made in relation to the structural integrity of the ring structure.
- The prototype has not implemented the mechanism for attaching the rope to the cleats using a bolt as described in 3.4. Instead, the prototype uses cleats on both the inner and the outer ring, with the outer cleats being clam cleats. The prototype does therefore not give information on whether the bolt design is a viable alternative or not.
- As the ring structure of this prototype is made up of aluminium, the buoyancy forces are smaller than the gravitational forces, causing this prototype to sink. This prototype can therefore not be used to test behaviour of the system floating in water. The materials and electronics of this prototype is also not waterproof.

8.3 Test relevance for real case scenario

The test environment used in this work included many simplifications relative to a real case scenario. Below is an evaluation of the assumptions made during the test of the manipulator.

- The test scenario assumes no water and no current, meaning no external forces acting on the manipulator other than gravity. In reality, the forces acting on the manipulator due to drag on the manipulator from the current may be significant. In a scenario assuming constant current, the drag effect due to the current will vary over time. The force required by an actuator positioned 12.5 m from the centre of the ring to compensate for current of 1.5 m/s acting perpendicular to the spoke manipulator is 38 N [1]. While this is not a large force, it will affect the control of the manipulator and should be compensated for.

- The test assumes no current and no waves. This results in the ring structure having a constant orientation and position during the test. This cause the gravitational forces on the manipulator to be constant, and other external forces to be zero. In reality, both the position and the orientation of the ring structure will move relative to the seabed over time. For a manipulator to operate under these conditions, a more advanced controller might be necessary to compensate for the variation in force acting on it. To what extent the ring structure will change pose, and in what rate is not considered in this work. The effect from this factor is therefore not known.

- The propulsion system created for this prototype was made to be able to work under conditions where the ring structure is subject to algae and shell growth. The test scenarios in this work does however not put this property to a test.

- The test assumed one end of the rope already was attached to the ring structure, and did not consider the challenge of attaching the other end to the ring structure after the rope was deployed. For the real case scenario, this challenge must be solved.

- In a system with current, the rope may have to to be semi-tight during the whole process, in order to not intertwine with the already attached rope. This effect is not experienced on this prototype.

8.4 Future work

- Finish the prototype and perform the tests described in this thesis. This includes doing changes required to control all four motors and using a current sensor to measure the current flowing to the motors.

- Consider using a more complex trajectory generator creating trajectories for the manipulator to move in more than one dimension than at a time.

- Implement the bolt mechanism for securing the rope to the ring structure.

- Find a way to connect the two ends of the rope to the ring structe, one before and one after the attachment of the rope.
- Do structural integrity analysis for the plant made up of HDPE, as well as the manipulator.
- Estimate the normal un-regularities of circular HDPE structures, and create a prototype ring structure consisting of this material.
- Look in to a robust mechanism for attaching the manipulator to the ring structure.
- Create a water-proof prototype of the manipulator.
- Perform tests of the behaviour of the manipulator under water, with and without current.

Bibliography

[1] A. T. Eggesvik, “Robot for automated seaweed deployment and harvesting,” tech. rep., NTNU, 2018.

[2] A. Chapman, P. Stévant, J. Schipper, Øyvind Kråkås, B. Aspøy, and A. Stavland, “Markedsvurdering for bærekraftig algedyrking i integrert multitrofisk akvakultur (imta)-anlegg.,” tech. rep., Møreforskning, 2014.

[3] T. H. Nilsen, “Analysis of the kelp farming industry in norway with regard to conceptual design of vessles for harvesting and deployment operations.,” tech. rep., NTNU, 2018.

[4] T. Olafsen, U. Winther, Y. Olsen, and J. Skjermo, “Value creation based on productive oceans in 2050,” tech. rep., Sintef, 2012.

[5] T. Searchinger, C. Hanson, J. Ranganathan, B. Lipinski, R. Waite, R. Winterbottom, A. Dinshaw, and R. Heimlich, “Creating a sustainable food future,” tech. rep., World Recources institute, 2014.

[6] E. S. Bale, “Development of area efficient and standardized structures for large-scale macroalgae cultivation,” tech. rep., Sintef, 2017.

[7] S. E. Solutions, “Production site.” `www.seaweedenergysolutions.com`, 2018. [Online; accessed 4-December-2018].

[8] V. fiskeredskap, “Production site concept.” `vaerlandetfiskeredskap.no`, 2018. [Online; accessed 4-December-2018].

[9] At-Sea, “Production site.” `https://sioen.com/en/news/at-sea-turnkey-farm`, 2019. [Online; accessed 16-June-2019].

[10] Sintef, “Sintef - MACROSEA.” `https://www.sintef.no/projectweb/macrosea/`, 2019. [Online; accessed 27-May-2019].

[11] I. Wiik, “Concept development of details for macroalgae cultivation - an in-depth studies of development and testing of rope attachment, deploying and harvesting in

an automatized macroalgae cultivation [summer intern report, macrosea project],” tech. rep., Sintef, 2018.

[12] M. V. Mark W. Spong, Seth Hutchinson, *Robot Modeling and Control*. JOHN WILEY & SONS, INC., 2005.

[13] O. A. Olsen, *Industrielle målemetoder*. Odd Arild Olsen forlag, 2015.

[14] Hengstler, *Enkoders programme*. Hengstler, 2006.

[15] Dynapar, “QuadratureEncoding.” `https://www.dynapar.com/technology/encoder_basics/quadrature_encoder/`, 2019. [Online; accessed 2-June-2019].

[16] SVB, “Clam cleat.” `https://www.svb24.com/product/gallery/1488`, 2019. [Online; accessed 2-June-2019].

[17] S. H. Lee, “Design of the out-pipe type pipe climbing robot,” tech. rep., Andong National University, 2012.

[18] O. R. Rafael Aracil, Rouque J. Saltarén, “A climbing parallel robot,” *IEEE Robotics & Automation Magazine*, pp. 16–22, 2006.

[19] G. Climbing technology, “Rope Clamp.” `https://www.grube.eu/forestry/rope-climbing-equipment/rope-equipment/rope-clamps/3882/ct-rollnlock-rope-clamp-en-567/en-12278`, 2019. [Online; accessed 5-June-2019].

[20] AndroidDev, “PWM.” `https://developer.android.com/things/sdk/pio/pwm`, 2019. [Online; accessed 2-May-2019].

[21] M. S. Group, “Clam cleat.” `https://www.monarco.io/`, 2019. [Online; accessed 5-June-2019].

[22] C. Durmusoglu, “PID library.” `https://github.com/ivmech/ivPID/blob/master/PID.py`, 2019. [Online; accessed 15-Feb-2019].

[23] A. T. Eggesvik, “Production site.” `https://github.com/AndreasTEggesvik/SPOKeControl`, 2019. [Online; accessed 13-June-2019].

[24] m. Avago, “Avago SSR.” `https://no.mouser.com/ProductDetail/Broadcom-Avago/ASSR-4128-002E?qs=9%2FrGSQZ%252BqohR4F3xbo8K1w%3D%3D`, 2019. [Online; accessed 13-June-2019].