BEN-GURION UNIVERSITY OF THE NEGEV
FACULTY OF ENGINEERING SCIENCES
DEPARTMENT OF BIOMEDICAL ENGINEERING

DESIGN AND ANALYSIS OF A MINIMALLY ACTUATED SERIAL ROBOT FOR MEDICAL PROCEDURES

THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE M.Sc. DEGREE

By

Lior Damti

February 2017

BEN-GURION UNIVERSITY OF THE NEGEV
FACULTY OF ENGINEERING SCIENCES
DEPARTMENT OF BIOMEDICAL ENGINEERING

DESIGN AND ANALYSIS OF A MINIMALLY ACTUATED SERIAL ROBOT FOR MEDICAL PROCEDURES

THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE M.Sc. DEGREE

By: Lior Damti

Supervised by: Dr. David Zarrouk
Prof. Amnon Sintov

Author: Date: ……21/2/17………

Supervisor: Date: ……21/2/17………

Supervisor: א. סינטוב Date: ……21/2/17………

Chairman of Graduate Studies Committee: Date: ….23/2/17…...

February 2017

Abstract

Endoscopy of the gastrointestinal (GI) tract is a widely used medical procedure in many countries. However, the conventional endoscopy, which is based on a compliant endoscope manually handled by a physician, has two drawbacks: (1) It results in patient discomfort, and (2) its access to the small intestine is very limited. Accessibility is quite important for many procedures, including biopsies, control of bleeding and strictures' dilatations. Attempts made to access body vessels have utilized highly articulated serial robots, sometimes called ‘snakes’. These robots are practically impossible to navigate along the intestine due to its length, and thus require dozens or hundreds of actuators, making them extremely cumbersome to operate, and limiting the potential to minimize their diameter. Furthermore, the excessive number of actuators would result in a compliant system incapable of applying forces that may be needed for traveling inside the intestine or perform simple procedures such as a biopsy.

In this project, we propose a novel type of serial robot with minimal actuation, aka ‘MARS’ (minimally-actuated robotic snake). The robot is a serial rigid structure consisting of multiple links connected by passive joints and movable actuators. The novelty of this robot is that the actuators travel over the links to a given joint and adjust the relative angle between the two adjacent links. The joints passively preserve their angles until the actuator moves them again. This actuation can be applied to any serial robot with two or more links. This unique configuration enables the robot to undergo the same wide range of motions typically associated with hyper-redundant robots but with much fewer actuators. The robot is modular and its size and geometry can be easily changed. Besides its potential medical applications, this type of robots can also be used for industrial, agricultural, and search and rescue applications.

In this thesis, we describe the robot’s mechanical design and kinematics in detail and demonstrate its capabilities for obstacle avoidance with some simulated examples using motion planning and optimization algorithms. In addition, we show how an experimental robot fitted with a single mobile actuator can maneuver through a confined space to reach its target.

Keywords: GI endoscopy; serial robot; minimal actuation; mobile actuator.

Acknowledgments

First and foremost, I would like to express my sincere gratitude to my supervisors, Dr. David Zarrouk and Prof. Amnon Sintov, for the continuous support of my thesis and research, for their patience, understanding, motivation, and most of all, for the knowledge they shared with me. It has been a great experience being part of the Bio-inspired and Medical Robotics Laboratory, which I'm sure will continue to thrive and flourish under David's leadership for many years to come.

A very special thanks also goes to my colleagues, Mr. Gideon Tirosh, Mr. Matan Birger, Mr. Nir Dgani, Ms. Lana Waxman, Ms. Tal Dachlika, Mr. Tal Raindel and Mr. Amotz Hess, for their true friendship and support (professional and otherwise), without whom these past two years would not have been so productive and enjoyable. I would also like to thank my dear friend, Mr. Adam Kelder, for his kind advice and much needed perspective in adverse times.

Finally, I must express my very profound gratitude and appreciation to my parents, Ofer and Mazal Damti, for providing me with financial support throughout my years of study and for their unconditional love all through the years of my life. This accomplishment would not have been possible without them.

This research was supported and partially funded by a grant from the ABC (Agricultural, Biological & Cognitive) Robotics Initiative of Ben-Gurion University of the Negev and the Helmsley Charitable Trust.

Table of Contents

Nomenclature

A_i	$[-]$	Homogeneous transformation matrix of frame $o_i x_i y_i z_i$ with respect to frame $o_{i-1} x_{i-1} y_{i-1} z_{i-1}$
a	$[-]$	Approach direction vector, represents the direction of $z_i$in frame $o_{i-1} x_{i-1} y_{i-1} z_{i-1}$
a_i	$[m]$	DH parameter - link length
c_i	$[-]$	Cosine of angle θ_i
d	$[m]$	Translation vector from o_{i-1} to o_i in frame $o_{i-1} x_{i-1} y_{i-1} z_{i-1}$
d_i	$[m]$	Prismatic joint variable\ DH parameter - link offset
F	$[N]$	Force
f	$[-]$	Function
H	$[-]$	Homogeneous transformation matrix
h_{ij}	$[-]$	Elements of homogeneous transformation matrix
J	$[m]$	Jacobian
K_t	$\left[\frac{Nm}{rad}\right]$	Stiffness coefficient
L	$[m]$	Link length
l_i	$[m]$	Length of link *i*
N	$[-]$	Number of links in the chain\ Number of DOF
n	$[-]$	Normal direction vector, represents the direction of x_i in frame $o_{i-1} x_{i-1} y_{i-1} z_{i-1}$
o	$[m]$	Origin of a coordinate frame
p	$[-]$	Cost for the number of links to be moved
q_i	$[m], [rad]$	Joint variable
$\dot{q}_i$	$\left[\frac{m}{sec}\right], \left[\frac{rad}{sec}\right]$	Joint velocity
R	$[-]$	Rotation matrix
Rot	$[-]$	Basic homogeneous transformation matrix generating $SE(3)$ for rotation about the x, y, z-axes
$\mathbb{R}^3$	$[-]$	A set of real numbers in the 3D space
r_i	$[m]$	Range vector from joint *i* to the force's point of application
SE	$[-]$	Special Euclidean group
SO	$[-]$	Special Orthogonal group

s	$[-]$	Sliding direction vector, represents the direction of $y_i$in frame $o_{i-1}x_{i-1}y_{i-1}z_{i-1}$
s_i	$[-]$	Sine of angle θ_i
T_j^i	$[-]$	Homogeneous transformation matrix of frame $o_j x_j y_j z_j$ with respect to frame $o_i x_i y_i z_i$
$Trans$	$[-]$	Basic homogeneous transformation matrix generating $SE(3)$ for translation about the x, y, z-axes
v	$\left[\frac{m}{sec}\right]$	Linear velocity vector
X	$[m]$	Cartesian displacement vector\ Cartesian target coordinate
x	$[m]$	Cartesian coordinate
Y	$[m]$	Cartesian displacement vector\ Cartesian target coordinate
y	$[m]$	Cartesian coordinate
z	$[m]$	Cartesian coordinate

Greek letters

α_i	$[rad]$	DH parameter - link twist
δ	$[-]$	Differential
θ_i	$[rad]$	Revolute joint variable\ DH parameter - joint angle
ξ	$\left[\frac{m}{sec}\right]$	Cartesian velocity vector (body velocity)
τ	$[Nm]$	Torque
ω	$\left[\frac{rad}{sec}\right]$	Angular (rotational) velocity vector

Subscripts

a	Translation along x-axis
b	Translation along y-axis\ Backlash angle
c	Translation along z-axis
f	Final coordinate
fin	Final relative angle
i	Index
$init$	Initial relative angle
j	Index
k	Index

n	Index
t	Torsion angle
v	Linear velocity component
x	Axis direction
y	Axis direction
z	Axis direction
α	Rotation angle about x-axis
β	Rotation angle about y-axis
γ	Rotation angle about z-axis
ω	Angular (rotational) velocity component

Superscripts

T	Transpose of a matrix

List of Figures

List of Tables

1. Introduction

The long-term purpose of this study is to develop a mechanically innovative serial robot for operating in confined places. This robot can be used for many applications in a number of fields, such as agriculture and search-and-rescue, yet, our interest was mainly directed towards its potential medical applications. With rapid technological advancement over the past decades, there is a growing need for minimally invasive surgeries world-wide. Most of these procedures are performed using robotic instruments, as they are more accurate and therefore can minimize the risks and complications involving surgical operations. They can also be used to enhance the capabilities of surgeons performing open surgery, leading to the possibility for remote surgery (in the case of computer-controlled systems).

1.1 Literature Review

Minimally-Invasive interventions provide the patient with numerous advantages over traditional open surgery by reducing pain, tissue damage, blood loss and cosmetic damage. Overall, they are proven to greatly improve the quality of life of patients and reduce the risk of postoperative complications. One of the most commonly used minimally invasive procedures is gastrointestinal endoscopy. However, the critical limitation of gastrointestinal endoscopy is the difficulty to access the small intestine. Such access is important for many procedures, including biopsies, control of bleeding, and strictures' dilatations.

Multiple attempts were made to access biological vessels using highly articulated serial robots that are sometimes referred to as ‘snakes’ [1]. These robots are practically impossible to navigate along the intestine due to its length, and as such, snake robots usually require dozens or hundreds of actuators, making them extremely cumbersome to operate, and limiting the potential to minimize their diameter. Furthermore, the excessive number of actuators will result in a compliant system incapable of applying forces that may be needed for traveling inside the intestine or perform simple procedures such as biopsy.

While some minimally invasive external surgical robots were successful (such as Da-Vinci and Renaissance [2] [3]), all attempts to travel a large distance through biological vessels with self-propelled miniature robots [4]-[19] or snake like robots [19] [20] have not succeeded. In a recent study, Zarrouk et al. [21] modeled the locomotion of crawling inside highly elastic biological vessels and described how the compliance and varying properties and geometries of biological vessels [22] [23] stopped the advance of the robots. The models were later experimentally demonstrated [24] using a unique experimental single actuator worm robot [25].

The movement inside biological vessels presents a major challenge for medical robots because of their substantial compliance and low friction on this surface [26]. Furthermore, it requires sophisticated motility for a tiny robot, available energy source and accessories for medical use. Therefore, research groups and technology companies worldwide have failed in their attempts to produce operationally reliable micro-robots capable of moving inside biological vessels.

1.1.1 Gastrointestinal Endoscopy

Endoscopy is a minimally invasive diagnostic medical procedure in which a surgeon uses an endoscope to look inside a patient's body. The endoscope consists of a tubular probe fitted with a tiny camera and light, which is inserted to the body through a small incision. The camera transmits footage to a viewing screen that magnifies the images of the body's internal structures. The surgeon can then perform certain surgical tasks by inserting instruments through one or more small incisions in the skin [27].

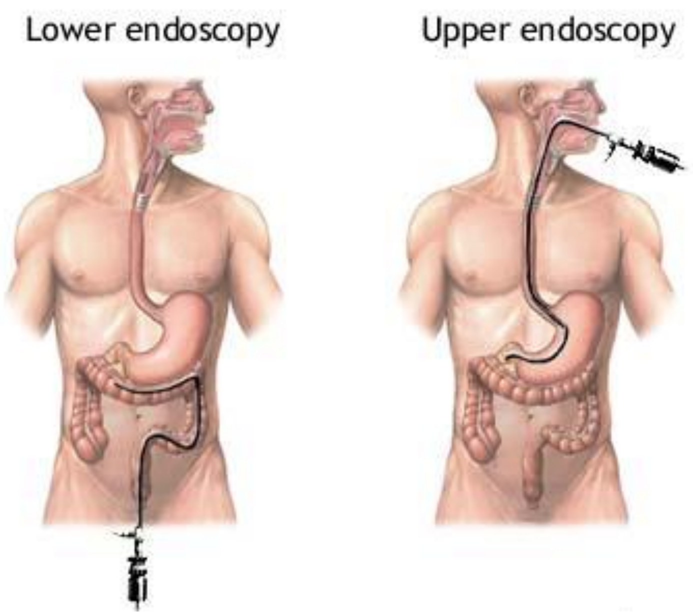


Figure 1.1: Lower and upper endoscopy of the gastrointestinal tract [27].

Most often the term ‘endoscopy’ is used to refer to an examination of the upper part of the gastrointestinal (GI) tract known as esophagogastroduodenoscopy, or EGD. Endoscopy of the lower GI tract is used to examine the intestine, most commonly known as colonoscopy (endoscopy of the large intestine), as shown in Fig. 1.1.

The gastrointestinal tract is an organ system responsible for consuming and digesting food, absorbing nutrients and expelling waste. The GI tract includes all organs between the mouth and the anus (rectum) [28] and is divided into the upper and lower GI tract. The whole digestive tract is about 9 m long, and is presented in Figure 1.2.

The upper GI tract consists of the organs between the mouth and the stomach. The process begins in the mouth and continues in a muscular tube padded mucous called the pharynx. The pharynx is connected to the esophagus, which is a fibromuscular tube approximately 25 cm long that ends in the stomach, a sack-like organ containing about 0.05 ÷ 1.5 L of fluid. The pH in the stomach is estimated between 1.5 ÷ 4, which is a very acidic environment.

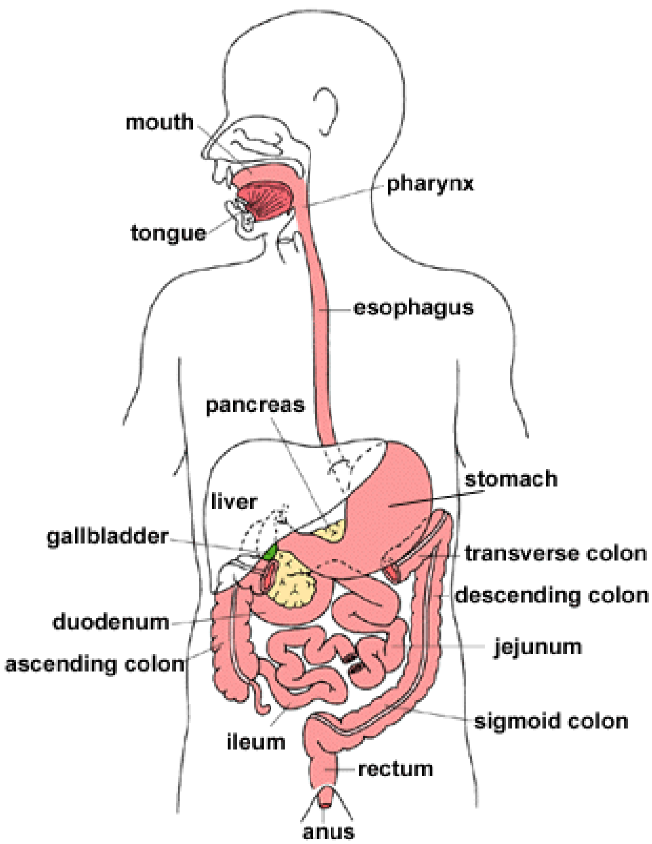


Figure 1.2: Anatomical representation of the gastrointestinal system [28].

The lower GI tract is the segment extending from the stomach to the anus, and includes most of the small intestine and the large intestine. The small intestine consists of three parts: duodenum, jejunum and ileum. Its total length is about 6 m and the average diameter is approximately 3 cm [29]. The inner walls of the jejunum and ileum are coated with bumps named "villi". This unique structure increases the surface area of the inner wall and thereby contributes to the efficiency of nutrient absorption.

The large intestine is a continuation of the small intestine. It is intermittently covered by peritoneum, which is the serous membrane that forms the lining of the abdominal cavity. The large intestine consists of four main parts: cecum, colon, rectum and anal canal. It also includes the appendix, which is attached to the cecum. The colon is the longest part in the large intestine and is about 1.5 m long. It is organized in a type of an open-square shape around the small intestine. The colon is divided into four subsections: ascending colon, transverse colon, descending colon and sigmoid colon [29].

The gastrointestinal tract has a form of general histology with some differences that reflect the specialization in functional anatomy (for more details see Appendix A). The mechanical properties of the intestine differ from person to person and frequently change depending on the health, age, gender, diet and even the time and weather. In the last decades, many developments succeeded in entering and moving inside the large intestine [30], however, at this time, no robotic development in use can enter and move freely inside the small intestine. EGD gets as far as the first segment of the small bowel, the duodenum, but the next two parts, the jejunum and ileum, require other methods.

Visualization of the small bowel has long posed a challenge to gastroenterologists, due to the physical difficulty of reaching its more distal regions. Pediatric colonoscopes which are much longer than standard gastroscopes can visualize the proximal jejunum. This technique is referred to as push enteroscopy, shown in Fig. 1.3 (A). Push forces required to advance the endoscope are usually greater than 6.6 N 40% of the time [31] [32], and can range up to 17.6 ÷ 4 N in live patients, depending on age, gender and examined part of the intestine [33]. According to a study performed on excised pig colon, the forces exerted by the colonoscope on the colon wall were about 3.0 ± 0.37 N [31]. In a different research, studying the effects of distractive forces to the small intestine, gross tissue disruption in pig and human tissue performed ex-vivo was seen at forces starting as early as 2.3 and 2.9 N, respectively; however, with in-vivo testing, blood flow to the bowel wall was reduced to undetectable levels at loads exceeding 0.98 N [34]. Due to the length of the small bowel, averaging 4-6 meters in the adult, push enteroscopy is still not effective to adequately visualize large portions of the small intestine [35] [36].

Wireless capsule endoscopy has proven to be the endoscopic investigation of choice for visualization of the entire small bowel. An 11 × 32 mm pill sized video camera is swallowed by the patient and approximately eight hours of video is transmitted wirelessly to a receiver worn by the patient. The procedure is painless, well accepted by patients and offers a very high accuracy. However, it is limited by the inability to carry out medical procedures, such as obtaining biopsies, and is therefore considered a purely diagnostic tool [30] [35] [37] [38].

Newer techniques, including single and double-balloon endoscopy (see Fig. 1.3 (B) and (C)) have been developed to overcome some of these issues, but are limited by the length of the procedure, and the need for deep sedation or general anesthesia. Spiral enteroscopy, shown in Fig. 1.3 (E), is a novel technique that utilizes an overtube with raised spirals affixed on the enteroscope that is rotated to advance the enteroscope deep into the small bowel [39]. Each of these three enteroscopy platforms offers similar accuracy and effectiveness but do not have widespread availability and are not without complications [38] [40].

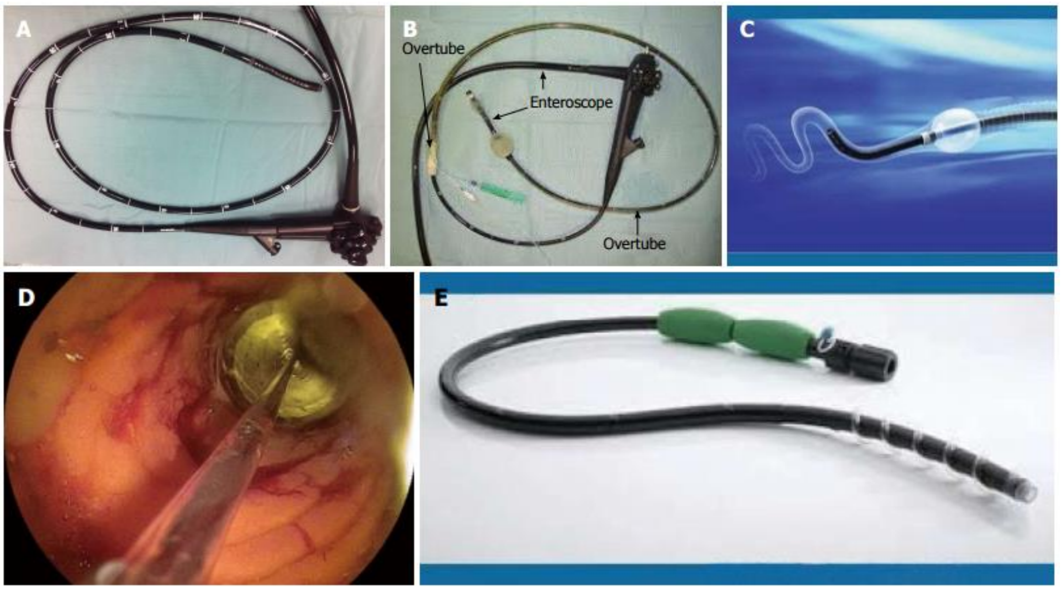


Figure 1.3: Enteroscopic devices. (A) Push enteroscope, (B) double-balloon enteroscope, (C) single-balloon enteroscope, (D) balloon dilatation of jejunal stricture, (E) spiral enteroscope [36].

For these reasons, we believe that the use of a robotic endoscope will allow the physician to better control all parts of the device, especially the distal end of the endoscope, through which the surgical operations are executed. The configurable nature of the robot will provide easier access to the intestine compared to the current tubular probe, as the orientation of each link can be controlled remotely, which will result in fewer, if any, manual maneuvers required from the physician and thereby, reducing operation time and patient discomfort. In addition, our robotic device should meet the basic demands of any endoscope: reliability, low-cost, simple to operate, and the ability to perform procedures in real-time.

1.1.2 Serial Robots

Serial robots are made of multiple links connected through actuated joints. For 6DOF (degrees-of-freedom) applications, these robots are generally made of six links and a seventh link may be added to avoid simple obstacles. Serial robots offer multiple advantages as they are accurate, quick to react and provide a large work volume [41]. Their widespread use and integration in numerous industrial applications such as pick and place and welding occurred a few years after being originally introduced in the early 1970s, with many companies offering multiple off-the-shelf prototypes [42] [43].

Over the years, the robots evolved in terms of force and precision, and current models can reach between 1 m to 3 m with a force range from 5 kg [44] to 500 kg [45] and a precision of 40 microns [44]. However, the main setback of serial robots is their force to weight ratio and inability to operate through obstacles, cavities or in pipes. To overcome this challenge, snake robots which are practically serial robots made of large number of joints, about 20 or more, were developed in the mid-1990s [26] [46] [47]. Initially, snake robots appeared to have a great potential for different applications in confined spaces, pipes and rubble, but after continuous work over two decades [48]-[54], they seem to be still facing some mechanical challenges and not ready to be used in a real-life applications due to their length, size and large weight.

1.1.3 Hyper-Redundant Robots

Hyper-redundant robots are robots with serially connected links that possess a large kinematic redundancy. As part of the robotic snake family, they are the subject of extensive research over the past several decades [55] [56] with many different configurations, mechanisms, control strategies, and motion planning algorithms being proposed over the years. The principle motivation for developing hyper-redundant robots is their ability to navigate around obstacles and in highly confined spaces.

Still, there are some serious challenges facing rigid hyper-redundant robots. Because of their large number of actuators, they are often slow-acting and consume power even when static. In addition, they contain an actuator at each joint, which renders the robot relatively weak and energy inefficient. Furthermore, their bulky design results in a low operating payload and large deflections [57].

In addition to the technical shortcomings of hyper-redundant robots, algorithms for planning their motions present a formidable challenge. Most of the standard methods developed for robot motion planning [58] [59] are intractable for the high-dimensional coordinate space of hyper-redundant robots. Early motion planners for hyper-redundant robot motion planning were developed by Gregory Chirkjian in [60]-[63]. In those works, the curvature of the robotic snake was approximated as a continuous modal function with the obstacles expressed as boundary constraints on the robot's shape. Many recent works have addressed obstacle avoidance schemes for hyper-redundant robots. State-of-the-art approaches including genetic algorithms [64] [65], variational methods [66], and probabilistic roadmaps [67] are used to plan the motions of the robots. However, these motion planners are usually time consuming and not always implementable in real-time applications.

To avoid these shortcomings while still achieving high redundancy, flexible robots have been developed as an alternative. Also known as 'soft' robots or continuum robots, they consist of a flexible continuous structure that possess, at least in theory, an infinite number of degrees-of-freedom. The advantage of flexible robots over hyper-redundant robots is their lightweight and speed. However, there is still ongoing research to improve their accuracy, control and position and sensing capabilities (see [68] and [69]). Those shortfalls render them, as of today, unsuitable for tasks requiring a relatively high degree of accuracy such as medical and agricultural applications.

1.2 The MARS: A New Robotic Concept

As previously stated, the long-term aim of this research is to develop a novel family of minimally-actuated multi-linkage serial robots (MMSR), for operating in confined spaces for light-load tasks.

In this work, we propose the Minimally Actuated Robotic Snake (MARS), which combines some characteristics and advantages from both hyper-redundant robots and compliant robots. In contrast to classical hyper-redundant robots, the MARS is a serial robot consisting of multiple links connected by passive joints and of a small number of movable actuators. The actuators translate over the links to any given joint and adjust it to the desired angular displacement. The joint passively preserves its angle until it is actuated again. The number of degrees-of-freedom is equal to the number of joints. This enables the MARS to achieve similar mobility (albeit slower) to regular hyper-redundant robots. The advantages of the MARS are its simplicity, smaller weight, higher energy density (power/mass), low cost and modularity, as the number of links and actuators can be easily changed.

The development of the under-actuated family of serial robots will allow reaching previously inaccessible areas. As such, the outcome of this research will also allow for the development of task-specific low-cost minimally invasive robots that will hopefully minimize operating room time and procedure cost. Furthermore, we expect that the ease of access to the large and small intestines will encourage doctors to develop new diagnostic and therapeutic procedures.

2. Theoretical Background

Robot manipulators are composed of nearly rigid links connected by joints to form a kinematic chain. Joints are typically rotary (revolute) or linear (prismatic), as shown in Figure 2.1 bellow, and allow relative motion between adjacent links [70]. These joints are usually instrumented with position sensors, which allow the relative position of neighboring links to be measured. In the case of rotary or revolute joints, these displacements are called joint angles. Some manipulators contain sliding (prismatic) joints, in which the relative displacement between links is a translation, sometimes called the joint offset.

At the free end of the chain of links that make up the manipulator is the end-effector. Depending on the intended application of the robot, the end-effector could be a gripper, a welding torch, an electromagnet, or another device.

In order to describe the position and orientation of a body in space, we will always attach a coordinate system, or frame, rigidly to the object. We then proceed to describe the position and orientation of this frame with respect to some reference coordinate system (see Fig. 2.2). We generally describe the position of the manipulator by giving a description of the tool frame, which is attached to the end-effector, relative to the base frame, which is attached to the nonmoving base of the manipulator [71].

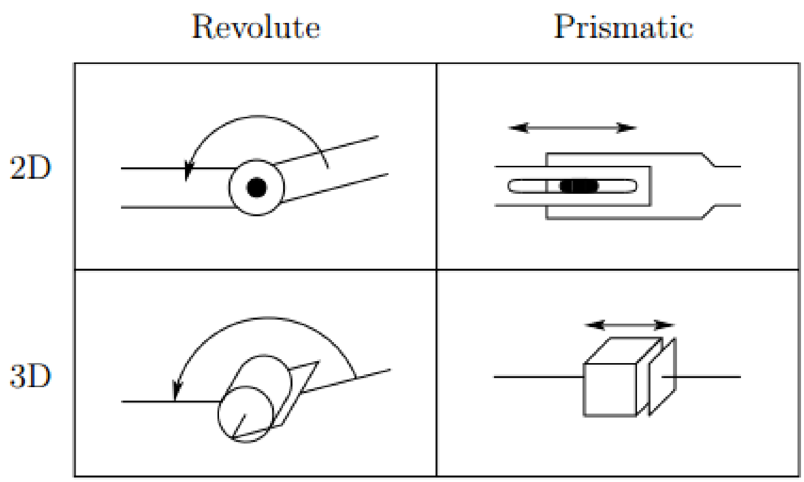


Figure 2.1: Symbolic representation of robot joints [70].

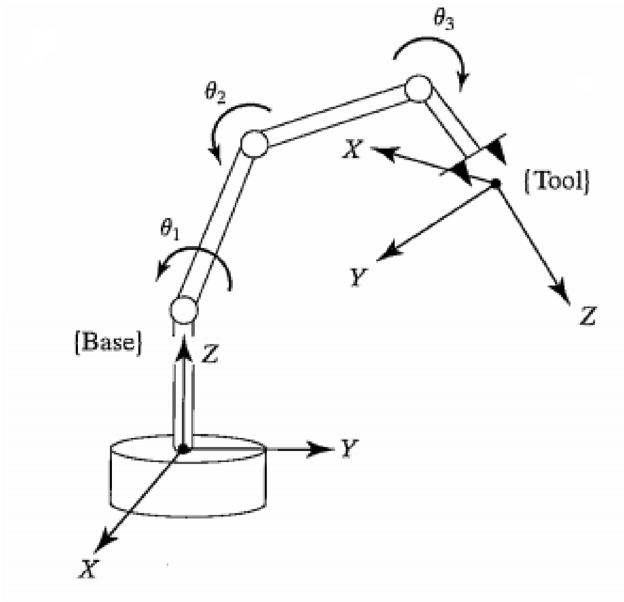


Figure 2.2: Kinematic equations describe the tool frame relative to the base frame as a function of the joint variables [71].

2.1 The Configuration Space

A configuration of a manipulator is a complete specification of the location of every point on the manipulator. The set of all possible configurations is called the configuration space. In our case, if we know the values for the joint variables (i.e., the joint angle for revolute joints, or the joint offset for prismatic joints), then it is straightforward to infer the position of any point on the manipulator, since the individual links of the manipulator are assumed to be rigid, and the base of the manipulator is assumed to be fixed. Therefore, in this text, we will represent a configuration by a set of values for the joint variables. We will denote this vector of values by q, and say that the robot is in configuration q when the joint variables take on the values $q_1,..,q_n$, with $q_i = \theta_i$ for a revolute joint and $q_i = d_i$ for a prismatic joint.

An object is said to have n degrees-of-freedom (DOF) if its configuration can be minimally specified by n parameters. Thus, the number of DOF is equal to the dimension of the configuration space. For a robot manipulator, the number of joints determines the number DOF. A rigid object in three-dimensional space has six DOF: Three for positioning and three for orientation (e.g., roll, pitch and yaw angles). Therefore, a manipulator should typically possess at least six independent DOF. With fewer than six DOF the arm cannot reach every point in its work environment with arbitrary orientation. Certain applications such as reaching around or behind obstacles may require more than six DOF. A manipulator having more than six links is referred to as a kinematically redundant manipulator. The difficulty of controlling a manipulator increases rapidly with the number of links [70].

2.2 The Workspace

The workspace of a manipulator is the total volume swept out by the end-effector as the manipulator executes all possible motions (see Figs. 2.3-2.4). The workspace is constrained by the geometry of the manipulator as well as mechanical constraints on the joints. For example, a revolute joint may be limited to less than a full 360° of motion. The workspace is often broken down into a reachable workspace and a dexterous workspace. The reachable workspace is the entire set of points in space reachable by the manipulator (in at least one orientation), whereas the dexterous workspace consists of those points that the manipulator can reach with an arbitrary orientation of the end-effector. Obviously, the dexterous workspace is a subset of the reachable workspace [70].

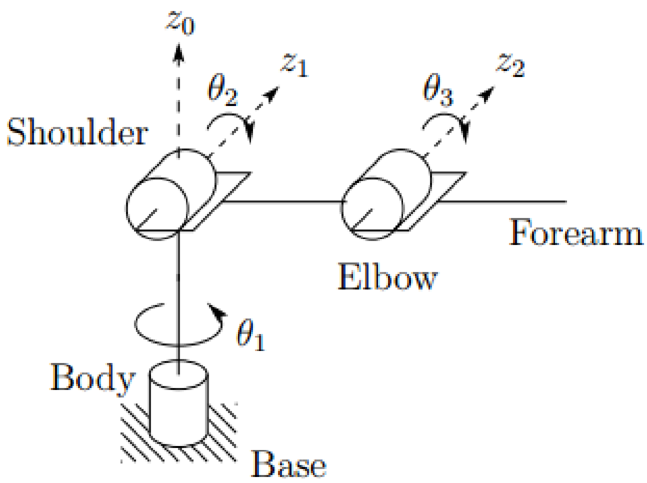


Figure 2.3: Structure of the elbow manipulator [70].

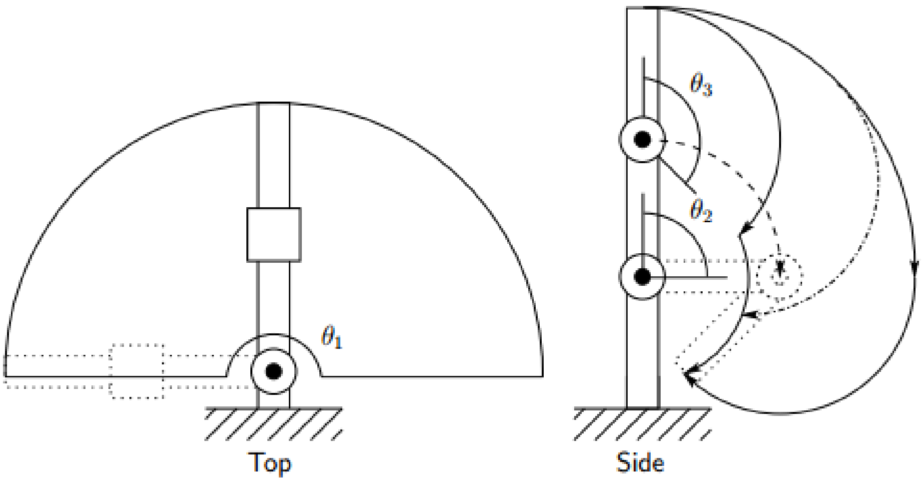


Figure 2.4: Workspace of the elbow manipulator [70].

2.3 Introduction to Robot Kinematics

Kinematics is the science of motion that treats motion without regard to the forces which cause it. Within the science of kinematics, one studies position, velocity, acceleration, and all higher order derivatives of the position variables (with respect to time or any other variable(s)). Hence, the study of the kinematics of manipulators refers to all the geometrical and time-based properties of the motion.

2.3.1 Rigid Motions and Homogeneous Transformations

A rigid motion is a combination of pure translation and pure rotation; it is defined as an ordered pair (d, R), in which $d \in \mathbb{R}^3$ and $R \in SO(3)$, where the latter represents the rotation matrix (see Appendix B.1-B.2). The group of all rigid motions is known as the Special Euclidean group and is denoted by $SE(3)$. Rigid motions can be represented in matrix form using the notion of homogeneous transformation,

$$H = \begin{bmatrix} R & d \\ 0 & 1 \end{bmatrix}; \quad R \in SO(3), \ d \in \mathbb{R}^3 \tag{2.1}$$

so that composition of rigid motions can be reduced to matrix multiplication as in the case for composition of rotations.

Homogeneous transformations combine the operations of rotation and translation into a single matrix multiplication, and are used to derive the so-called forward kinematic equations of rigid manipulators. Furthermore, homogeneous transformation matrices can be used to perform coordinate transformations, such that allow us to represent various quantities in different coordinate frames.

The most general homogeneous transformation that we will consider may be written as

$$H_1^0 = \begin{bmatrix} n_x & s_x & a_x & d_x \\ n_y & s_y & a_y & d_y \\ n_z & s_z & a_z & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} n & s & a & d \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{2.2}$$

where $n = \left(n_x, n_y, n_z\right)^T$ is a vector representing the direction of x_1 in the $o_0x_0y_0z_0$ system, the vector $s = \left(s_x, s_y, s_z\right)^T$ represents the direction of y_1, and the vector $a = \left(a_x, a_y, a_z\right)^T$ represents the direction of z_1. The vector $d = \left(d_x, d_y, d_z\right)^T$ represents the vector from the origin o_0 to the origin o_1 expressed in the frame $o_0x_0y_0z_0$ [70].

A set of basic homogeneous transformations generating $SE(3)$ is given by Eq. (2.3)-(2.5) for translation and rotation about the x, y, z-axes, respectively:

$$Trans_{x,a} = \begin{bmatrix} 1 & 0 & 0 & a \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}; \quad Rot_{x,\alpha} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c_\alpha & -s_\alpha & 0 \\ 0 & s_\alpha & c_\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.3}$$

$$Trans_{y,b} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & b \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}; \quad Rot_{y,\beta} = \begin{bmatrix} c_\beta & 0 & s_\beta & 0 \\ 0 & 1 & 0 & 0 \\ -s_\beta & 0 & c_\beta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.4}$$

$$Trans_{z,c} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & c \\ 0 & 0 & 0 & 1 \end{bmatrix}; \quad Rot_{z,\gamma} = \begin{bmatrix} c_\gamma & -s_\gamma & 0 & 0 \\ s_\gamma & c_\gamma & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{2.5}$$

The same interpretation regarding composition and ordering of transformations holds for 4 × 4 homogeneous transformations as for 3 × 3 rotations. Given a homogeneous transformation H_1^0 relating two frames, if a second rigid motion represented by $H \in SE(3)$ is performed relative to the current frame, then

$$H_2^0 = H_1^0 H, \tag{2.6}$$

whereas if the second rigid motion is performed relative to the fixed frame, then

$$H_2^0 = HH_1^0. \tag{2.7}$$

2.4 Forward Kinematics

A very basic problem in the study of mechanical manipulation is forward kinematics. This is the static geometrical problem of computing the position and orientation of the end-effector of the manipulator. Specifically, given a set of joint angles, the forward kinematic problem is to compute the position and orientation of the tool frame relative to the base frame. Sometimes, we think of this as changing the representation of manipulator position from a joint space description into a Cartesian space description; namely, the position of the point is given with three numbers representing its location defined by three axes: x, y and z, and the orientation of a body is given with three numbers representing the rotation angles about these axes [71]. The orientation of the three axes, as a whole, is arbitrary, however, the orientation of the axes relative to each other should always comply with the right-hand rule, unless specifically stated otherwise.

2.4.1 The Denavit-Hartenberg Convention

As previously stated, the forward kinematics problem is concerned with the relationship between the individual joints of the robot manipulator and the position and orientation of the tool or end-effector. The joint variables are the angles between the links in the case of revolute or rotational joints, and the link extension in the case of prismatic or sliding joints.

A set of conventions was developed in order to provide a systematic procedure for performing this analysis [70]. It is, of course, possible to carry out forward kinematics analysis even without respecting these conventions, as in the case of a two-link planar manipulator (see Appendix B.4).

A commonly used convention for selecting frames of reference in robotic applications is the Denavit-Hartenberg, or DH convention [70]. In this convention, each homogeneous transformation A_i (see Appendix B.3) is represented as a product of four basic transformations:

$$A_i = Rot_{z,\theta_i} Trans_{z,d_i} Trans_{x,a_i} Rot_{x,\alpha_i} = \tag{2.8}$$

$$= \begin{bmatrix} c_{\theta_i} & -s_{\theta_i} & 0 & 0 \\ s_{\theta_i} & c_{\theta_i} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c_{\alpha_i} & -s_{\alpha_i} & 0 \\ 0 & s_{\alpha_i} & c_{\alpha_i} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} =$$

$$= \begin{bmatrix} c_{\theta_i} & -s_{\theta_i} c_{\alpha_i} & s_{\theta_i} s_{\alpha_i} & a_i c_{\theta_i} \\ s_{\theta_i} & c_{\theta_i} c_{\alpha_i} & -c_{\theta_i} s_{\alpha_i} & a_i s_{\theta_i} \\ 0 & s_{\alpha_i} & c_{\alpha_i} & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

where the four quantities a_i, α_i d_i, θ_i are parameters associated with link i and joint i, and are generally given the names link length, link twist, link offset and joint angle, respectively. Since the matrix A_i is a function of a single variable, it turns out that three of the above four quantities are constant for a given link, while the fourth parameter, θ_i for a revolute joint and d_i for a prismatic joint, is the joint variable.

Clearly it is not possible to represent any arbitrary homogeneous transformation using only four parameters[1], as formerly discussed. Suppose we are given two frames denoted by frames 0 and 1, respectively, as illustrated in Figure 2.5, which satisfy the following two conditions:

(DH1) The axis x_1 is perpendicular to the axis z_0.

(DH2) The axis x_1 intersects the axis z_0.

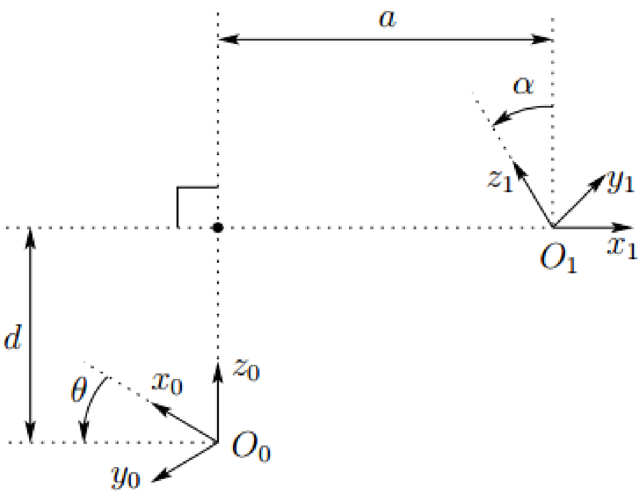


Figure 2.5: Coordinate frames satisfying assumptions DH1 and DH2 [70].

[1] Usually an arbitrary homogeneous transformation matrix is characterized by six numbers: three numbers to specify the fourth column of the matrix and three Euler angles to specify the upper left 3 × 3 rotation matrix.

Under these conditions, it can be proven (proof can be found in [70], section 3.2.1) there exists a unique homogeneous transformation matrix A that takes the coordinates from frame 1 into those of frame 0, and can be represented in the form of Eq. (2.8).

Now we can in fact give a physical interpretation to each of the four quantities in (2.8). The parameter a is the distance between the axes z_0 and z_1, and is measured along the axis x_1. The angle α is the angle between the axes z_0 and z_1, measured in a plane normal to x_1. The positive sense for α is determined from z_0 to z_1 by the right-handed rule. The parameter d is the perpendicular distance from the origin o_0 to the intersection of the x_1 axis with z_0, measured along the z_0 axis. Finally, θ is the angle between x_0 and x_1 measured in a plane normal to z_0. The positive sense for θ is determined from x_0 to x_1 by the right-handed rule [70].

For a given robot manipulator, one can always choose the frames $0, \ldots, n$ in such a way that the above two conditions are satisfied. From Eq. (2.8), it is evident that the choice of z_i is arbitrary since any direction for z_i can be obtained by choosing α_i and θ_i appropriately. Thus, axes $z_0, \ldots, z_{n-1}$ are assigned in an intuitively pleasing fashion; specifically, z_i is assigned to be the axis of actuation for joint *i* + 1. In other words, if joint *i* + 1 is revolute, z_i is the axis of revolution of joint *i* + 1; if joint *i* + 1 is prismatic, z_i is the axis of translation of joint *i* + 1. This satisfies the convention that we established earlier, namely that joint *i* is fixed with respect to frame *i*, and that when joint *i* is actuated, link *i* and its attached frame $o_i x_i y_i z_i$ experience a resulting motion.

Once we have established the z-axes for the links, we need to establish the base frame, also referred to as frame 0. The choice of a base frame is nearly arbitrary, since its origin o_0 can be chosen at any point on z_0. The choice for x_0, y_0 can be done in any convenient manner so long as the resulting frame is right-handed.

Once frame 0 has been established, we begin an iterative process in which frame *i* is defined using frame $i - 1$, starting with frame 1. This process is illustrated in Fig. 2.6 for two links with revolute joints, along with the corresponding DH parameters. In order to set up frame *i* it is necessary to consider three cases: (i) the axes z_{i-1}, z_i are not coplanar, (ii) the axes z_{i-1}, z_i are parallel, (iii) the axes z_{i-1}, z_i intersect. Each of these cases is specified below [70].

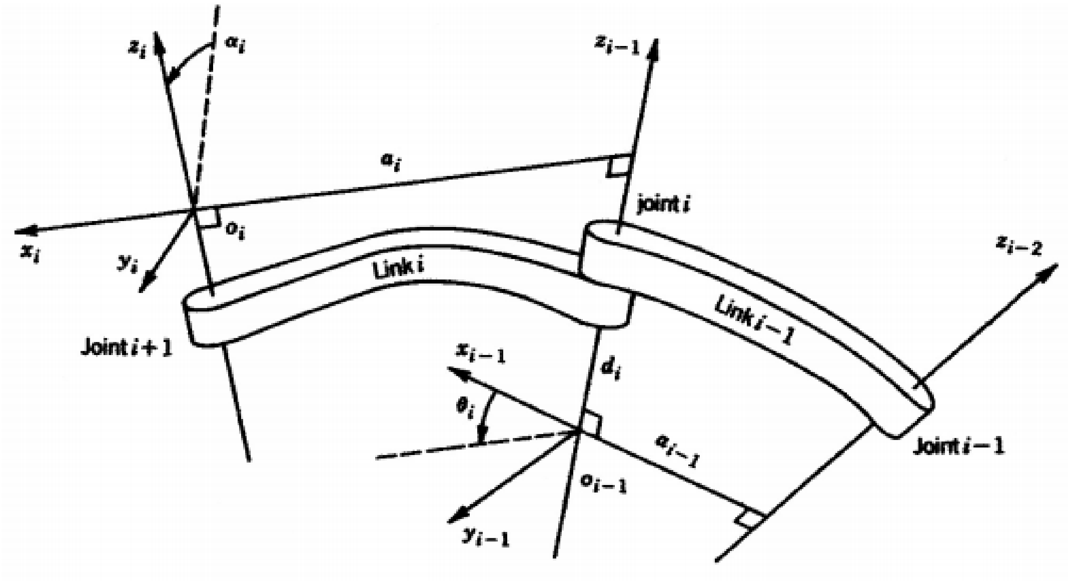


Figure 2.6: Denavit-Hartenberg frame assignment and parameters [70].

(i) z_{i-1} and z_i are not coplanar: If z_{i-1} and z_i are not coplanar, then there exists a unique line segment perpendicular to both z_{i-1} and z_i such that it connects both lines and it has minimum length. The line containing this common normal to z_{i-1} and z_i defines x_i, and the point where this line intersects z_i is the origin o_i. By construction, both conditions DH1 and DH2 are satisfied and the vector from o_{i-1} to o_i is a linear combination of z_{i-1} and x_i. The specification of frame i is completed by choosing the axis y_i to form a right-handed frame. Since assumptions DH1 and DH2 are satisfied, the homogeneous transformation matrix A_i is of the form of Eq. (2.8).

(ii) z_{i-1} is parallel to z_i: If the axes z_{i-1} and z_i are parallel, then there are infinitely many common normals between them and condition DH1 does not specify x_i completely. In this case the origin o_i can be chosen anywhere along z_i, preferably in a manner which simplifies the resulting equations. The axis x_i is then chosen either to be directed from o_i toward z_{i-1}, along the common normal, or as the opposite of this vector. A common method for choosing o_i is to choose the normal that passes through o_{i-1} as the x_i axis; o_i is then the point at which this normal intersects z_i. In this case, d_i would be equal to zero. Once x_i is fixed, y_i is determined, as usual, by the right hand rule. Since the axes z_{i-1} and z_i are parallel, α_i will be zero in this case.

(iii) z_{i-1} *Intersects* z_i*:* In this case x_i is chosen normal to the plane formed by z_i and z_{i-1}. The positive direction of x_i is arbitrary. The most natural choice for the origin o_i in this case is at the point of intersection of z_i and z_{i-1}, however, any convenient point along the axis z_i suffices. Note that in this case the parameter a_i equals 0.

This constructive procedure works for frames $0, \ldots, n-1$ in an *n*-link robot. To complete the construction, it is necessary to specify frame *n*. The final coordinate system $o_n x_n y_n z_n$ is commonly referred to as the end-effector or tool frame. The origin o_n is most often placed symmetrically between the fingers of the gripper, as shown in Fig. 2.7.

The unit vectors along the x_n, y_n and z_n axes are labeled as *n*, *s*, and *a*, respectively. The terminology arises from the fact that the direction *a* is the approach direction, in the sense that the gripper typically approaches an object along the *a* direction. Similarly, the *s* direction is the sliding direction, the direction along which the fingers of the gripper slide to open and close, and *n* is the direction normal to the plane formed by *a* and *s* [70].

In most contemporary robots the final joint motion is a rotation of the end-effector by θ_n and the final two joint axes, z_{n-1} and z_n, coincide. In this case, the transformation between the final two coordinate frames is a translation along z_{n-1} by a distance d_n followed (or preceded) by a rotation of θ_n about z_{n-1}. This observation will simplify the computation of the inverse kinematics, which will be addressed in the next section.

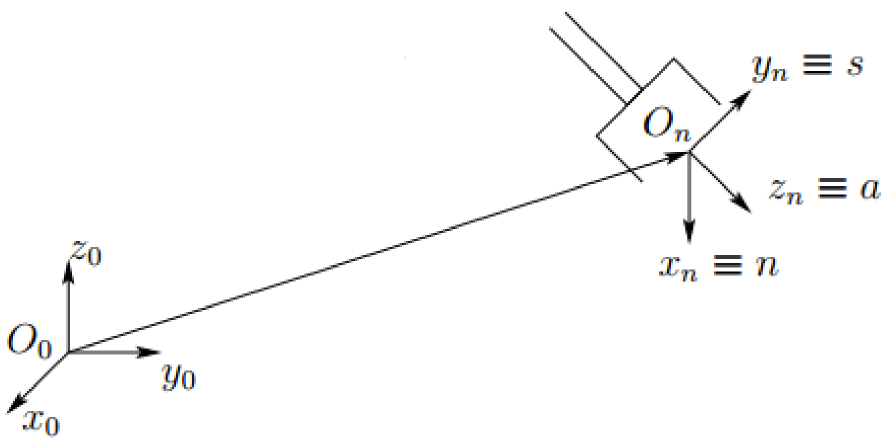


Figure 2.7: Tool frame assignment for a 3D gripper [70].

In all cases, whether the joint in question is revolute or prismatic, the quantities d_i and α_i are always constant for all i and are characteristic of the manipulator. If joint i is prismatic, then θ_i is also a constant, while a_i is the i-th joint variable. Similarly, if joint i is revolute, then a_i is constant and θ_i is the i-th joint variable.

2.5 Inverse Kinematics

In the previous section we considered the problem of computing the position and orientation of the end-effector when given the joint variables of the manipulator. This section is concerned with the more difficult converse problem: Given the desired position and orientation of the end-effector, how do we compute the set of joint variables which will achieve this desired result?

Solving this problem requires first to formulate the general inverse kinematics problem: Given a 4×4 homogeneous transformation H, as defined in Eq. (2.11), find (one or all) solutions of the equation

$$T_n^0(q_1, \dots, q_n) = H \tag{2.9}$$

where

$$T_n^0(q_1, \dots, q_n) = A_1(q_1) \cdots A_n(q_n). \tag{2.10}$$

Here, H represents the desired position and orientation of the end-effector, and the objective is to find the values for the joint variables $q_1, \dots, q_n$ so that Eq. (2.9) applies.

Since the bottom row of both T_n^0 and H is $(0,0,0,1)$, four of the sixteen equations represented by Eq. (2.9) are trivial. Hence, the solution for Eq. (2.9) results in twelve nonlinear equations in n unknown variables, which can be written as

$$T_{ij}(q_1, \dots, q_n) = h_{ij}, \qquad i = 1,2,3, \quad j = 1, \dots ,4 \tag{2.11}$$

where T_{ij}, h_{ij} refer to the twelve nontrivial entries of T_n^0 and H, respectively.

Whereas the forward kinematics problem always has a unique solution that can be obtained simply by evaluating the forward equations, the inverse kinematics problem may or may not have a solution. Even if a solution exists, it may or may not be unique. Furthermore, because these forward kinematic equations are in general complicated nonlinear functions of the joint variables, the solutions may be difficult to obtain even when they exist.

For example, in the case of a two-link planar mechanism there may be no solution if the given (x, y) coordinates are out of reach of the manipulator. If the given (x, y) coordinates are within the manipulator's reach there may be two solutions as shown in Figure 2.8 below, the so-called 'elbow-up' and 'elbow-down' configurations, or there may be exactly one solution if the manipulator must be fully extended to reach the point. There may even be an infinite number of solutions in some cases.

In solving the inverse kinematics problem we are most interested in finding a closed form solution of the equations rather than a numerical solution. Finding a closed form solution means finding an explicit relationship:

$$q_k = f_k(h_{11}, \dots, h_{34}), \qquad k = 1, \dots, n. \tag{2.12}$$

Having closed form solutions allows one to develop rules for choosing a particular solution among several. In certain applications, where the equations must be solved at a rapid rate, having closed form expressions rather than an iterative search is a practical necessity.

In most cases, the inverse kinematic equations are much too difficult to solve directly in closed form. Therefore, we need to use efficient and systematic techniques that exploit the particular kinematic structure of the manipulator (see Appendix B.5-B.7) [70].

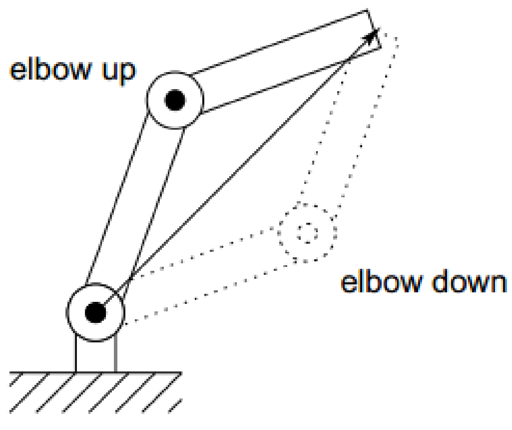


Figure 2.8: Multiple inverse kinematic solutions for a two-link planar mechanism [70].

The practical question of the existence of solutions to the inverse kinematics problem depends on engineering as well as mathematical considerations. Once a solution to the mathematical equations is identified, it must be further checked to see whether or not it satisfies all constraints on the ranges of possible joint motions.

2.6 Velocity Kinematics - The Manipulator Jacobian

In the previous sections we discussed the forward and inverse position equations relating joint positions to the end-effector positions and orientations. In this chapter we derive the velocity relationships, relating the linear and angular velocities of the end-effector to the joint velocities.

Mathematically, the forward kinematic equations define a function between the space of Cartesian positions and orientations and the space of joint positions. The velocity relationships are then determined by the Jacobian of this function. The Jacobian is a matrix that can be thought of as the vector version of the ordinary derivative of a scalar function. The Jacobian is one of the most important quantities in the analysis and control of robot motion. It arises in virtually every aspect of robotic manipulation, such as the planning and execution of smooth trajectories, the determination of singular configurations and in the derivation of the dynamic equations of motion. In this paper it is mainly used in the transformation of forces and torques from the end-effector to the manipulator's joints, and in the transformation between the *n*-vector of joint velocities and the six-vector consisting of the linear and angular velocities of the end-effector [70].

2.6.1 Jacoboians

As mentioned above, the Jacobian is a multidimensional form of the derivative. Suppose we have six functions, each of which is a function of six independent variables:

$$\begin{gathered} y_1 = f_1(x_1, x_2, x_3, x_4, x_5, x_6) \\ y_2 = f_2(x_1, x_2, x_3, x_4, x_5, x_6) \\ \vdots \\ y_6 = f_6(x_1, x_2, x_3, x_4, x_5, x_6). \end{gathered} \tag{2.13}$$

If we wish to calculate the differentials of y_i as a function of differentials of x_j, we simply use the chain rule and we get

$$\begin{aligned}
\delta y_1 &= \frac{\partial f_1}{\partial x_1}\delta x_1 + \frac{\partial f_1}{\partial x_2}\delta x_2 + \cdots + \frac{\partial f_1}{\partial x_6}\delta x_6 \\
\delta y_2 &= \frac{\partial f_2}{\partial x_1}\delta x_1 + \frac{\partial f_2}{\partial x_2}\delta x_2 + \cdots + \frac{\partial f_2}{\partial x_6}\delta x_6 \\
&\vdots \\
\delta y_6 &= \frac{\partial f_6}{\partial x_1}\delta x_1 + \frac{\partial f_6}{\partial x_2}\delta x_2 + \cdots + \frac{\partial f_6}{\partial x_6}\delta x_6.
\end{aligned} \tag{2.14}$$

These equations could also be written using vector notation:

$$\vec{Y} = F(\vec{X}) \tag{2.15}$$

$$\delta\vec{Y} = \frac{\partial F}{\partial \vec{X}}\delta\vec{X}. \tag{2.16}$$

The 6 × 6 matrix of partial derivatives in (2.14) is called the Manipulator Jacobian, *J*, or Jacobian for short [71]. Note that if the functions $f_1(\vec{X})$ through $f_6(\vec{X})$ are nonlinear, then the partial derivatives are a function of the x_i so the following notation can be used:

$$\delta\vec{Y} = J(\vec{X})\delta\vec{X} \tag{2.17}$$

By dividing both sides by the differential time element, the Jacobian defines a mapping between the velocities in $\vec{X}$ to those in $\vec{Y}$:

$$\dot{\vec{Y}} = J(\vec{X})\dot{\vec{X}} \tag{2.18}$$

At any particular instant, $\vec{X}$ has a certain value, and $J(\vec{X})$ is a linear transformation. At each new time instant, $\vec{X}$ has changed and therefore, so has the linear transformation. Jacobians are time-varying linear transformations.

In the field of robotics, we generally use Jacobians that relate joint velocities to Cartesian velocities of the end-effector. For example,

$$\xi = J\dot{q} \tag{2.19}$$

where q is the vector of joint angles of the manipulator and ξ is a vector of Cartesian velocities. For the general case of an *n*-jointed manipulator, the Jacobian is 6 × *n*, $\dot{q}$

is $n \times 1$, and ξ is 6×1. This 6×1 Cartesian velocity vector[2] is the 3×1 linear velocity vector and the 3×1 rotational velocity vector stacked together, which correspond to the linear and rotational components of the Jacobian, given by

$$\xi = \begin{bmatrix} v_n \\ \omega_n \end{bmatrix} \text{ and } J = \begin{bmatrix} J_v \\ J_\omega \end{bmatrix}. \tag{2.20}$$

Jacobians of any dimension (including non-square) can be defined. The number of rows equals the number of degrees-of-freedom in the Cartesian space being considered. The number of columns in a Jacobian is equal to the number of joints of the manipulator. In dealing with a planar arm, for example, there is no reason for the Jacobian to have more than three rows, although for redundant planar manipulators, there could be arbitrarily many columns (one for each joint).

As shown in Eq. (2.20), the upper half of the Jacobian J_v is given as

$$J_v = [J_{v_1} \quad \cdots \quad J_{v_n}] \tag{2.21}$$

where the *i*-th column J_{v_i} is

$$J_{v_i} = \begin{cases} z_{i-1} \times (o_n - o_{i-1}) & \text{for revolute joint } i \\ z_{i-1} & \text{for prismatic joint } i \end{cases} \tag{2.22}$$

and the lower half of the Jacobian is given as

$$J_\omega = [J_{\omega_1} \quad \cdots \quad J_{\omega_n}] \tag{2.23}$$

where the *i*-th column J_{ω_i} is

$$J_{\omega_i} = \begin{cases} z_{i-1} & \text{for revolute joint } i \\ 0 & \text{for prismatic joint } i \end{cases} \tag{2.24}$$

Proof for Eqns. (2.22) and (2.24) can be found in [70], sections 4.6.2 and 4.6.1, respectively.

Putting the upper and lower halves of the Jacobian together, the Jacobian for an *n*-link manipulator is of the form

---

[2] In considering the motions of robotic links, we will always use link frame {0} as our reference frame. Hence, v_i is the linear velocity of the origin of link frame {i} and ω_i is the angular velocity of link frame {i}.

$$J = [J_1 \quad \cdots \quad J_n] \tag{2.25}$$

where the i-th column J_i is given by

$$J_i = \begin{bmatrix} z_{i-1} \times (o_n - o_{i-1}) \\ z_{i-1} \end{bmatrix} \tag{2.26}$$

if joint i is revolute and

$$J_i = \begin{bmatrix} z_{i-1} \\ 0 \end{bmatrix} \tag{2.27}$$

if joint i is prismatic.

Figure 2.9 illustrates a second interpretation of Eq. (2.26). As can be seen in the figure, $o_n - o_{i-1} = r$ and $z_{i-1} = \omega$ in the familiar expression $v = \omega \times r$. The above formulas simplify the determination of the Jacobian of any manipulator since all of the quantities needed are available once the forward kinematics are worked out. The only quantities needed to compute the Jacobian are the unit vectors z_i with respect to the base, which are given by the first three elements in the third column of T_i^0, and the coordinates of the origins o_i, which are given by the first three elements of the fourth column of T_i^0.

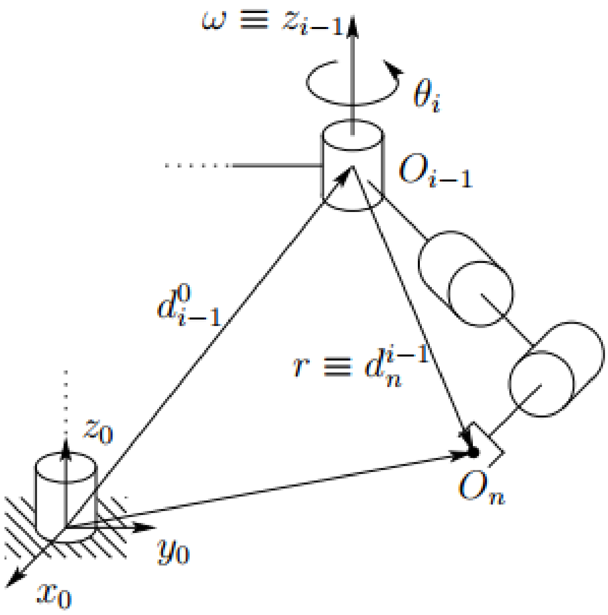


Figure 2.9: Motion of the end-effector due to link *i* [70].

2.6.2 Jacoboians in the Force Domain

The chainlike nature of a manipulator leads us quite naturally to consider how forces and moments "propagate" from one link to the next. Typically, the robot is pushing on something in the environment with the chain's free end (the end-effector) or is perhaps supporting a load at the tip. We wish to solve for the joint torques that must be acting to keep the system in static equilibrium.

When forces act on a mechanism, work (in the technical sense) is done if the mechanism moves through a displacement. Work is defined as a force acting through a distance and is a scalar with units of energy. The principle of “virtual work” allows us to make certain statements about the static case by allowing the amount of this displacement to go to an infinitesimal. Work has the units of energy, so it must be the same measured in any set of generalized coordinates. Specifically, we can equate the work done in Cartesian terms with the work done in joint-space terms. In the multidimensional case, work is the dot product of a vector force or torque and a vector displacement. Thus, we have

$$\vec{F} \cdot \delta\vec{X} = \vec{\tau} \cdot \delta\vec{\theta} \tag{2.28}$$

where $\vec{F}$ is a 6 × 1 Cartesian force-moment vector acting at the end-effector, $\delta\vec{X}$ is a 6 × 1 infinitesimal Cartesian displacement of the end-effector, $\vec{\tau}$ is a 6 × 1 vector of torques at the joints, and $\delta\vec{\theta}$ is a 6 × 1 vector of infinitesimal joint displacements. Expression (2.28) can also be written as

$$\vec{F}^T \delta\vec{X} = \vec{\tau}^T \delta\vec{\theta}. \tag{2.29}$$

Using the definition of the Jacobian from Eq. (2.17), where in this case

$$\delta\vec{X} = J\delta\vec{\theta}, \tag{2.30}$$

and substituting into (2.29) yields the following expression

$$\vec{F}^T J\delta\vec{\theta} = \vec{\tau}^T \delta\vec{\theta} \tag{2.31}$$

which must hold for all $\delta\vec{\theta}$; hence, we get

$$\vec{F}^T J = \vec{\tau}^T. \tag{2.32}$$

Transposing both sides yields this result:

$$\vec{\tau} = J^T \vec{F}. \tag{2.33}$$

Eq. (2.33) verifies in general what was stated at the beginning of the section: The Jacobian transpose maps Cartesian forces acting at the end-effector into equivalent joint torques. This relationship is very useful in a many aspects of robotic manipulation as it allows us to convert a Cartesian quantity into a joint-space quantity without calculating any inverse kinematic functions [71].

3. Robot Description and Design

One of the most crucial elements in the general mechanical design of a robot is to determine the minimal actuation requirements to perform a specific task. The minimalistic approach allows us to simplify the geometry, minimize the robot size and reduce the costs altogether while increasing the accuracy of the operation and the forces it can apply.

Our novel robot system is composed of N links connected through passive joints, mobile actuators that travel over the links, and an end-effector. The passivity of the joints is defined by there being no motors in between them, while the angle between adjacent links is preserved. The number of links and mobile actuators can be easily varied depending on the proposed task; the current design is composed of one mobile actuator and ten links (10 DOF), where the last link is unique and functions as the end-effector. The mobile actuator consists of two motors: One is required for translation, and the second for rotation. When the mobile actuator travels over the links it can rotate the desired joint, thereby changing the relative angle between the links by a desired angle of up to 45 degrees in each direction. The base is where the robot is connected to a constant support or a mobile platform, and is referred to as link 0, according to the DH convention.

All of the robot's parts are 3D printed using Object Connex 350 with nominal accuracy of nearly 50 microns using 'VeroGray' (prototype 'A') or 'VeroWhite' (prototype 'C') material, which possess similar mechanical properties.

3.1 Prototype 'A'

The initial version of the robot was developed during the first year, and our main objective was to show proof of concept. For this reason, as well as the simplicity of the design, it was decided to limit the robot's capabilities to operate in a two-dimensional workspace. The basic requirements which needed to be answered are as follows:

- Modular structure of the links - the links must be easily connected (and disconnected) to each other so that the robot's size could be changed on demand.
- Structural rigidity - the links must be rigid enough to sustain self-weight.

- Each link could be rotated in a relative angle of up to 45° clockwise (CW) and counterclockwise (CCW) with respect to the link it is attached to.
- Each link must have a locking mechanism in order to maintain its desired orientation.
- The robot will consist of only one actuator with the ability to travel freely over the links, forward and backward, and rotate each link in the plane (CW and CCW).
- The mobile actuator must travel over curved joints without changing their orientation.

3.1.1 Mechanical Design

In this version, the links are connected to each other by revolute joints and the joint angle is passively locked by a spring applying a friction force. The maximum relative angle between the links is 45 degrees, as shown in Fig. 3.1(a). At their bottom, the links have a track (see Fig. 3.1(b)) which allows the mobile actuator to travel along them to reach and actuate a desired joint. Each of the links is 2.5 cm wide and 5 cm long, giving the active section of the snake robot a total length of 50 cm.

To increase the friction force we glued sand papers to the links and inserted a metal screw to the clamp. Using this mechanism, the friction torque required to move the links is nearly 50 mNm. As a result, a half meter long 3D printed robot can apply nearly 0.1 N only at its tip without slipping.

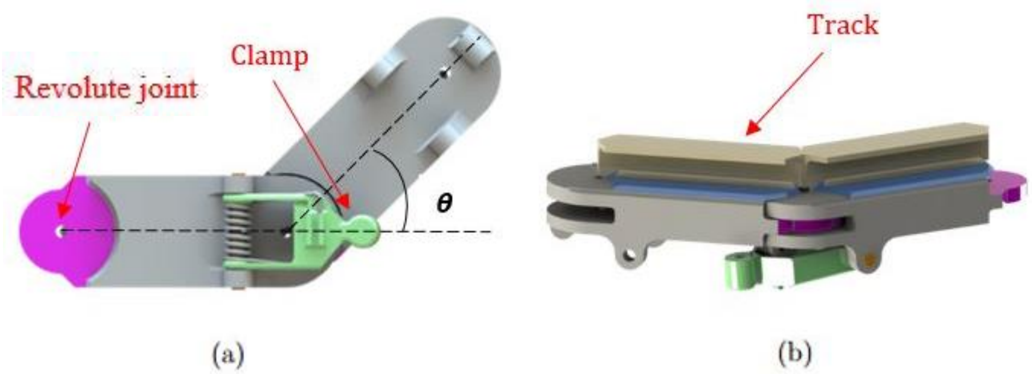


Figure 3.1: A top and bottom view of two adjacent links. (a) The relative orientaion between the links, given by the angle θ, is passively fixed by the clamp (shown in green). (b) At the bottom of each link there is a track for the mobile actuator.

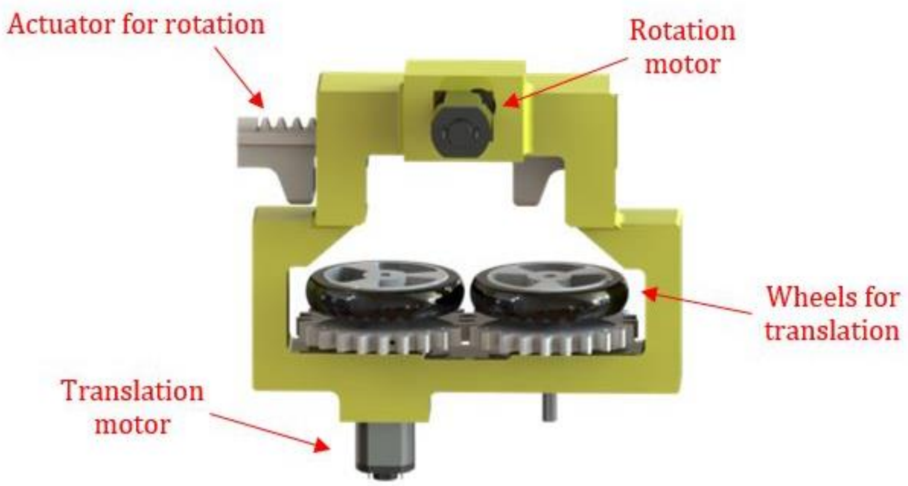


Figure 3.2: The mobile actuator consists of two motors - one motor to travel over the links, and a second motor to rotate the links.

The mobile actuator, presented in Figure 3.2, has two motors: One motor actuates the wheels to drive the mobile actuator along the tracks of the links, and a second motor is used to rotate the links. The rotational motor is attached to a linear gear mechanism, allowing the actuator to disconnect from the links when the translation motor is activated, or push them (the head of the clamp) for rotation.

The final version of the design is shown in Fig. 3.3 below. This prototype was used in the preliminary experiments described in the next section, and was the subject of a paper submitted to the journal *Robotica* of Cambridge University Press.

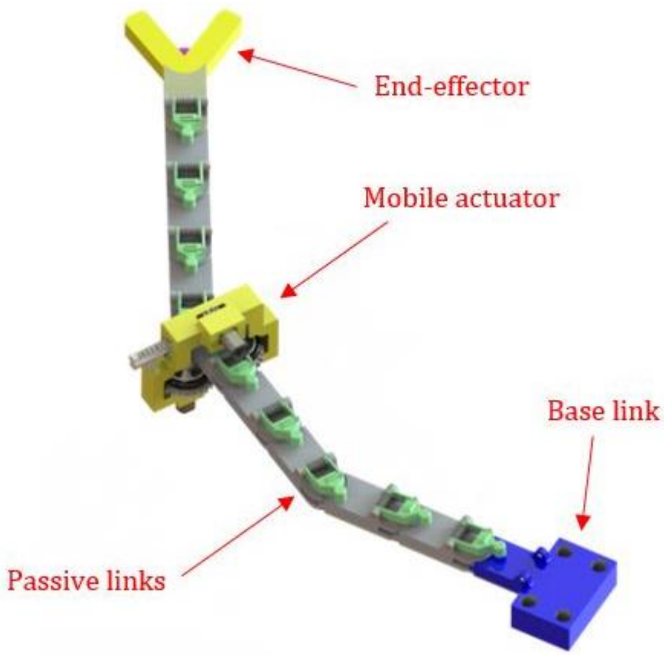


Figure 3.3: Prototype 'A' of the MARS, composed of ten links (including the end-effector), a base link and one mobile actuator.

3.1.2 Experiments

To verify our results and prove the feasibility of the MARS, we performed a few experiments using this prototype. During all of the experiments, the mobile actuator was remotely controlled by a human operator. The operator had a two channel joystick: One channel is used to drive the mobile actuator forward and backward along the links, and the other to rotate the links CW or CCW.

The robot’s modularity is demonstrated in the following experiments. We used a 4 Volts Lithium-ion battery to actuate the motors of the mobile actuator. The speed of locomotion is nearly 2.5 cm/s and the rotational speed is approximately 15 degrees/s. We used motors with 1000:1 gear ratio which can produce 0.9 Nm of torque at 32 RPM [72]. This torque is necessary to overcome the friction torque between the different links and other external forces to produce motion.

The basic experiment involved a chain of five links, as shown in Fig. 3.4. The mobile actuator was tested going towards the end of the chain and returning back to the base link, with and without rotating the links. Starting at (a), the robot advances towards its tip (b)-(c), then returns to the center (d). Next, the robot rotates the links clockwise (e) and counterclockwise (f). The robot then travels over the curved joint (g) and rotates its tip clockwise (h) and counterclockwise (i).

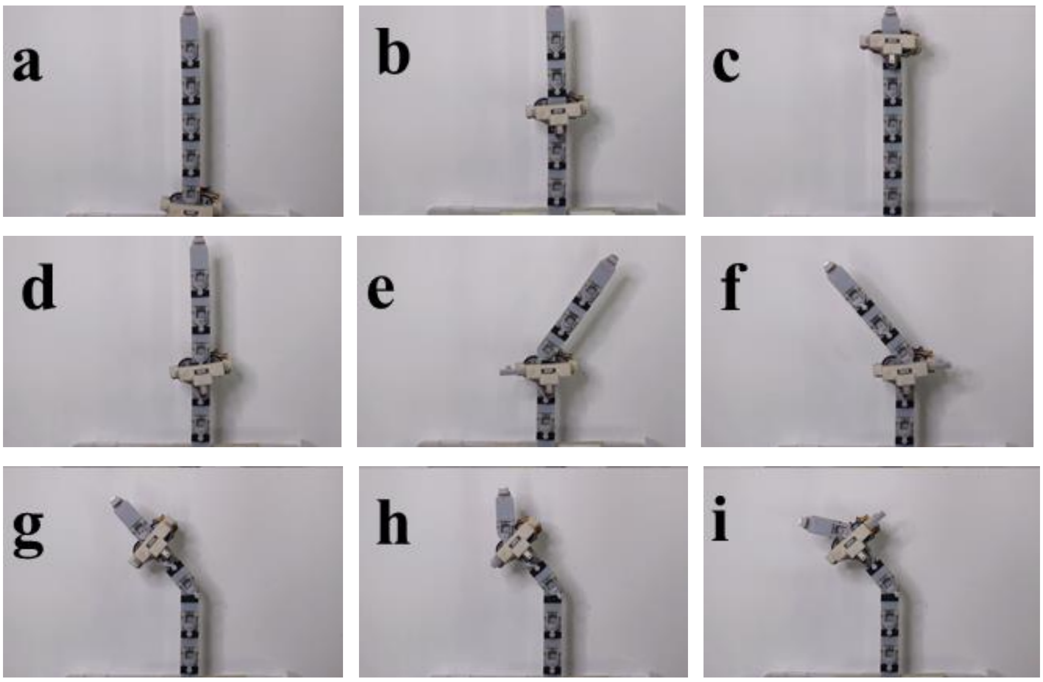


Figure 3.4: The mobile actuator travels forward and backward over the links without changing their orientation, and activates them on demand (CW and CCW).

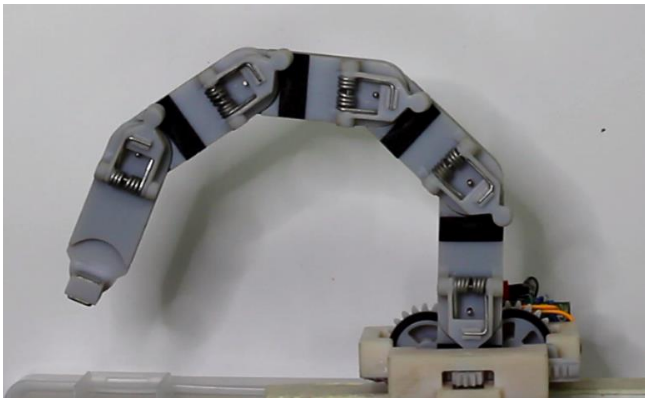


Figure 3.5: By rotating the four links CCW, the robot gets a ‘C’ shape.

The weight of the mobile actuator is 102 grams, whereas the average weight of each link including the clamp and joint is nearly 25 grams. We attached a 1 cm^3 magnetic cube to the tip of the last link in order to grasp our target, as a form of an end-effector.

As the joints can be rotated 45 degrees in each direction, the robot can make a ‘C’ shape (half a circle) by rotating four links in the same direction. This experiment is illustrated in Fig. 3.5. In both of these experiments, the robot had no difficulty travelling over the links or rotate them in either direction.

In the following experiments we added five more links to the robot (ten in total). This operation required nearly two minutes thanks to the modular structure of the links. The additional links enlarged the robot’s configuration space, thereby enabling us to perform more diverse and complex tasks. Fig. 3.6 shows the MARS forming an ‘S’ shape by rotating links 6-9 clockwise, and links 1-4 counterclockwise.

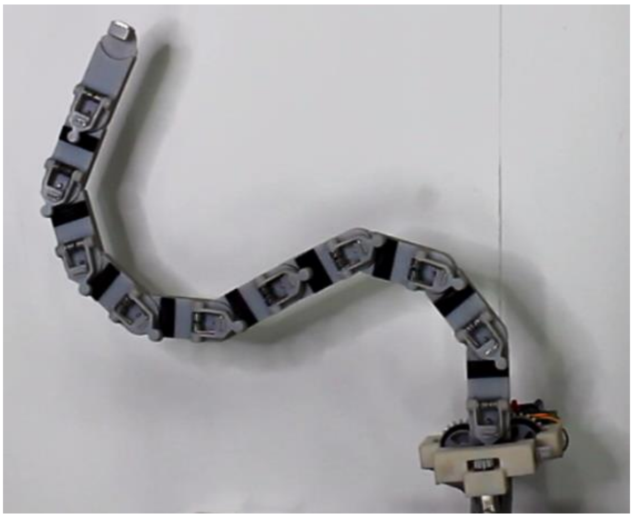


Figure 3.6: By rotating links 6-9 CW and links 1-4 CCW, the robot gets an ‘S’ shape.

With the longer version, we performed another experiment in order to test the robot's maneuverability in an environment which contains obstacles. We simulated a motion planning situation (with obstacles) as summarized in Fig. 3.7. The goal of the robot is to grab the round object (red ball) and bring it back to the robot's original configuration. In this section, the planning was performed by the human operator.

This task is composed of two main challenges. The first is going through the narrow pass of 15 mm, and the second is reaching the target with the small section of the robot that went through the opening. Throughout the whole task, the robot must avoid colliding with the obstacles.

The robot accomplished this task by having the actuator translate and adjust the angles of the joints one at a time. The robot first passes through the narrow pass by transforming its second half into an arc-like shape. Then, the mobile actuator travels along the links through the pass and then rotates the top links to reach the target. Since four joints and links went through the pass, the robot had four degrees-of-freedom to reach its target (only three are required in a 2D space to reach location and orientation). Only eight translational steps were required for the mobile actuator in each direction, demonstrating the dexterity and maneuverability of the MARS.

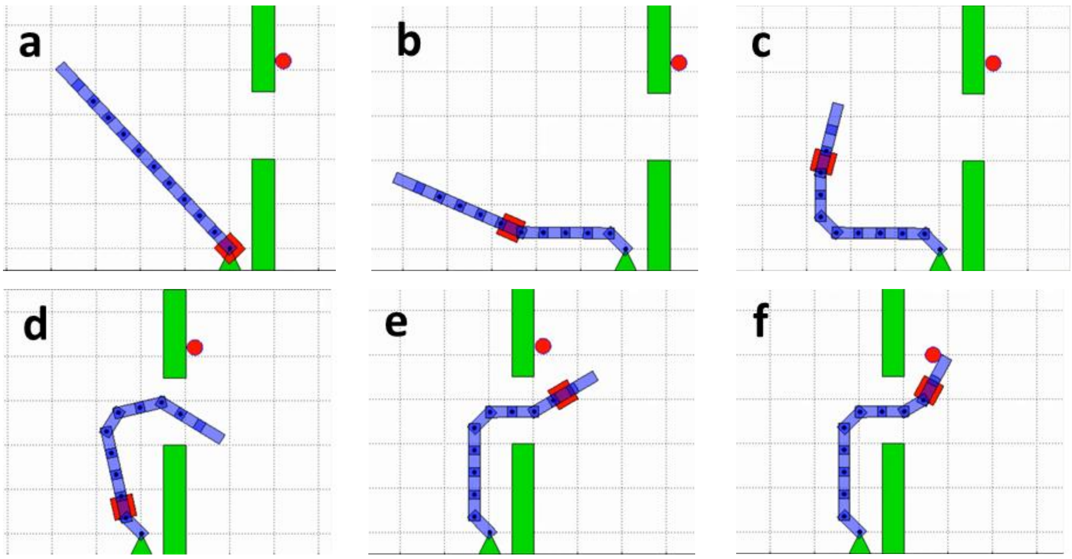


Figure 3.7: An animation of MARS, equipped with a single mobile actuator, reaching its target. The mobile actuator rotates the base link (a) and then advances to the center (b). At (c), the mobile actuator rotates the six top links to make an arc shape and then advances to the base (d) to rotate the second link and penetrate through the small cavity. The actuator travels again to the top links to rotate them towards the target (e). After reaching its target (f), the robot makes the inverse plan of a-b-c-d-e to return to its original configuration.

Table 3.1: Motion summary of the MARS, as presented in the animation and the experiment. During each step, the mobile actuator advances from one joint (start) to another (end) and rotates the specific joint by an angle θ.

STEP no.	Translation (start-end)	Rotation [degrees]	
		joint	angle
Reaching the target			
1	(1-1)	1	+45
2	(1-2)	2	+45
3	(2-6)	6	-45
4	(6-7)	7	-45
5	(7-9)	9	-45
6	(9-2)	2	-45
7	(2-9)	9	+75
8	(9-10)	10	+45
Returning to initial configuration			
9	(10-10)	10	-75
10	(10-9)	9	-60
11	(9-2)	2	+90
12	(2-10)	10	+30
13	(10-9)	9	+30
14	(9-7)	7	+45
15	(7-6)	6	+45
16	(6-2)	2	-45
17	(2-1)	1	-45

As shown in Table 3.1, each stage of motion consists of rotating the given joint by the turning angle, then translating the actuator to the desired joint, and repeating the process. There are a total of eight actions required to reach the object, one action to grasp it, and another eight actions required to return to its initial state with the grasped object in hand.

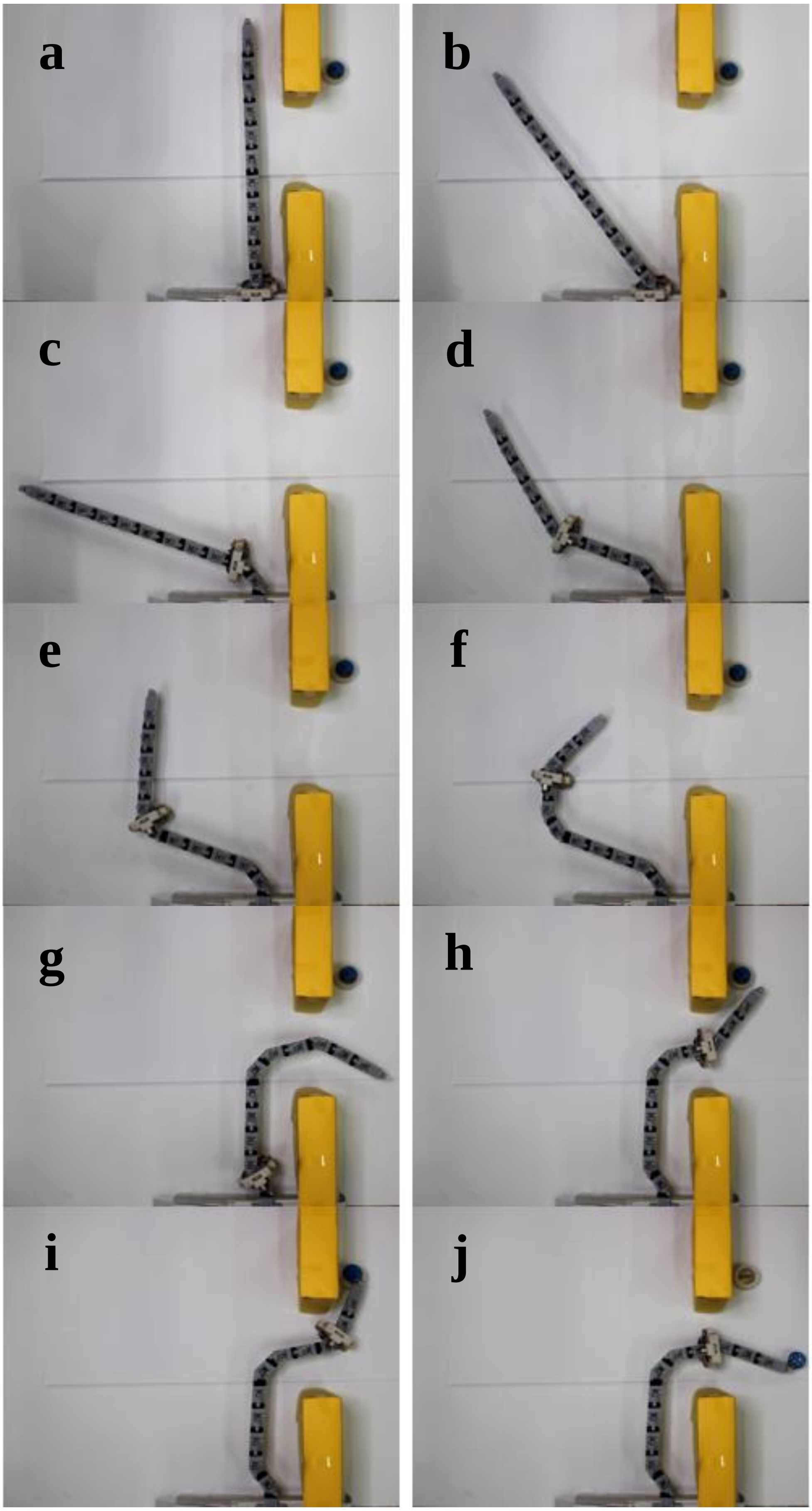


Figure 3.8: An experiment showing the MARS maneuverability in an environment with obstacles.

Following the same algorithm described in Table 3.1, the robot successfully reached its desired target and retrieved the object (3D printed blue ball), as shown in Fig. 3.8. However, we found that since the robot is made of printed material, it slightly curved downwards by nearly 1 cm. Even though the weight of the robot is larger than in the previous experiments, and the torque acting on the links substantially increased, the links remained locked during the entire session.

3.2 Prototype ‘C’

During the second year, our main focus was to improve the original design of the MARS in order to achieve a fully-working 2D prototype, which will be mechanically stronger and efficient. Based on the challenges we encountered with prototype ‘A’, new design requirements were formed to overcome the shortcomings of the initial version, whilst preserving the robot’s strengths and without “violating” the basic requirements:

- Stiffer links - in the longer version of the robot, where the chain consists of ten links, we detected a "sinking" problem due to the fact that the links are made of printed material (this challenge was resolved during the development of prototype ‘B’).
- Locking mechanism - the current mechanism which allows the links to remain in a desired orientation is based completely on friction (passive lock), and therefore cannot maintain its position when great forces are applied at its direction.
- Precision - the rotation angle of the joints is currently determined based on visual estimation of the human operator, which limits the robot’s abilities to maneuver in a confined space containing various obstacles. Integrating a simple control system would allow the MARS to gain higher precision in its motion and increase its automation.

3.2.1 Mechanical Design

In the course of developing the second version of the MARS (prototype ‘B’), we designed stiffer links in order to overcome the robot's "sinking" problem. The general structure and size of the links remained unchanged apart from the design of the track; the bottom of the track was altered to fit the structure of the joint, so that the metal pin (2 mm diameter) connecting the two links will pass through the joint entirely, as shown in Fig. 3.9(a). This amendment increased the rigidity of the joints significantly, thereby eliminating the curving of the whole chain.

In addition, we integrated a more "active" locking mechanism between the links; the previous lock was based completely on friction (passive lock), and therefore cannot maintain its position when great forces are applied at its direction. As presented in Fig. 3.9(b), the new mechanism consists of a worm drive transmission, where the worm gear (shown in green) is printed as part of the link and a 0.5 module worm is placed in the posterior part of each link. This mechanism prevents the links from sliding across one another and ensures they remain locked in the desired orientation. Furthermore, the gear transmission allows us to determine the exact angle of rotation between the links, a feature which was missing from the initial prototype.

Altering the locking machanism required a different design of the mobile actuator. To maintain the robot’s mechanical uniqness, which is manifasted in the minimal actuation demand, a number of design alternatives were tested. The final version of the mobile actuator is presented in Fig. 3.10; this design was proven to be most efficient for both operations required from the actuator, namely, translation along the chain and rotation of the links.

The mobile actuator consists of two gear transmissions, each connected to one motor. The actuator advances along the links using three wheels, two of which are actively spinning due to the motor located on the bottom left side of the structure,

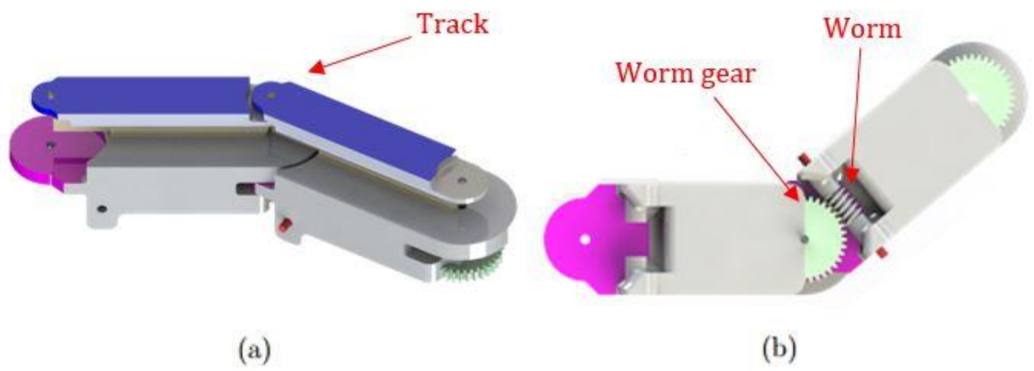


Figure 3.9: A top and bottom view of two adjacent links. (a) The bottom of the track (painted in blue) is similar to the structure of the joint (painted in pink) so that the connecting pin could pass through the entire joint. (b) The links are firmly locked at a relative angle of 45 degrees by the worm drive transmission.

and a horizontal gear transmission that allows for the two wheels to spin in the same direction. The third wheel, which is slightly larger in diameter, is located on the opposite side of the link between the two "active" wheels (see Fig. 3.10(a)). As a result of the force applied by the torsion spring on the right, the "passive" wheel is pressed against the track across from the "active" wheels, allowing the actuator to travel forward and backward over the links. Once the actuator reaches the desired link, the second motor activates the vertical gear transmission (see Fig. 3.10(b)), causing the worm to spin the worm gear, consequently rotating the adjacent link.

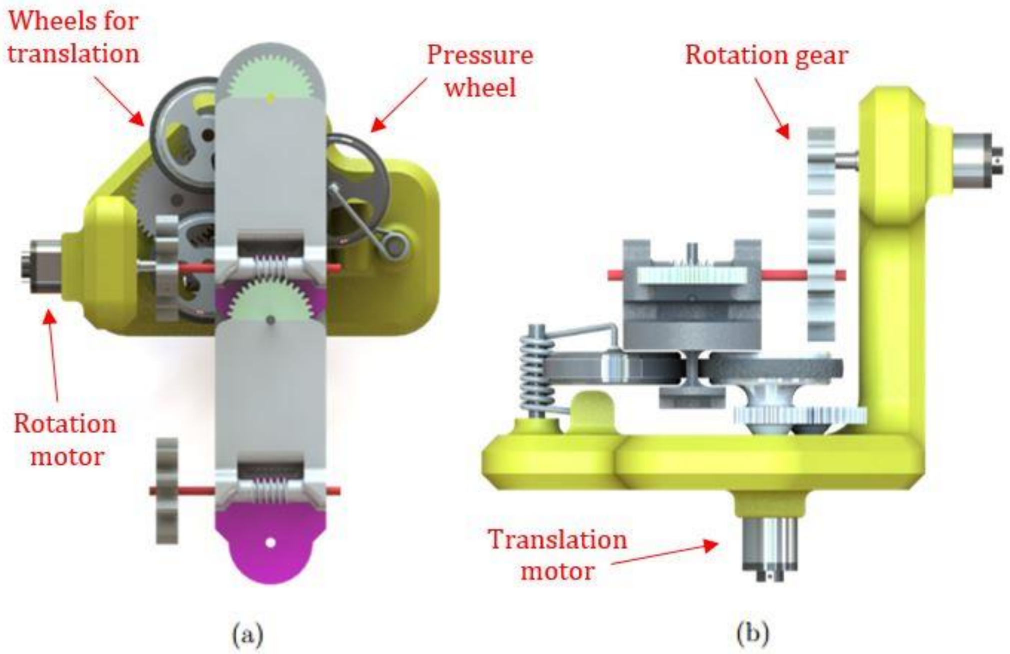


Figure 3.10: A top and front view of the mobile actuator. (a) The bottom motor actuates the wheels using a gear transmission (left side) and a pressure wheel which is forced by the spring (right side). (b) The upper motor actuates a second gear transmission that connects to the worm drive transmission of a specific link and rotates it upon demand (CW and CCW).

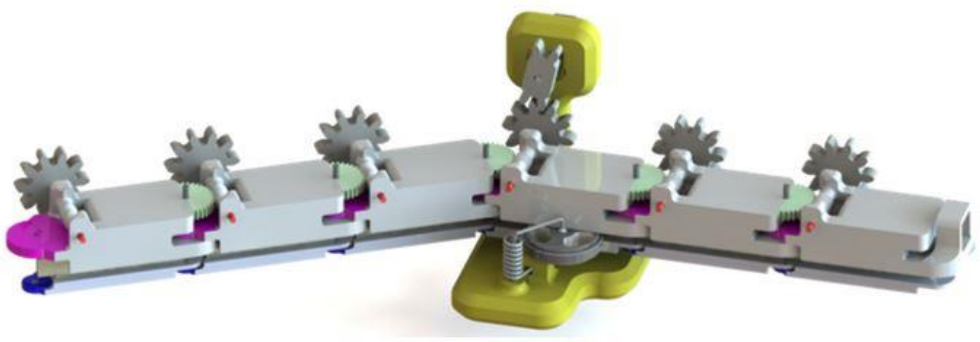


Figure 3.11: The mobile actuator rotating the eighth joint 30 degrees CW, shown from a side view.

Figure 3.11 shows the rotation gear transmission from a side view. The upper gear, which is connected directly to the rotation motor, is identical in size and module to the lower gear. However, six teeth were cut from the upper gear in order to prevent the two gears from integrating as the actuator advances along the chain. This allows us to control the relative rotation angle between two adjacent links with a precision of up to 2 degrees (see Appendix C.1).

3.2.2 Experiments

To prove the feasibility of the new prototype, we performed a few experiments using the latest version of the MARS combined with the control system. Integrating the control system allows the mobile actuator to operate autonomously by executing the commands given via the computer.

The programming code, which was written in C++, enables the human operator to insert as input the current location of the actuator (joint no.), the next location (joint) it needs to be, the desired relative angle of the new joint (expressed in the number of spins executed by the motor) and its direction (CW/CCW).

We used a 7.4 Volts Lithium-ion battary to actuate both motors (rotation - 100:1 [73], translation - 1000:1) and the control system, which consists of four main electronic components including a micro-processor (Teensy 3.1), H-bridge and two electro-optical sesnsors: an encoder and a reflective IR sensor (for technical specifications see Appendix C). The IR sensor is placed at the bottom of the actuator, directly beneath the track of the links. The bottom of the tracks were painted black except for one small area which was painted in white, as presented in Fig. 3.12.

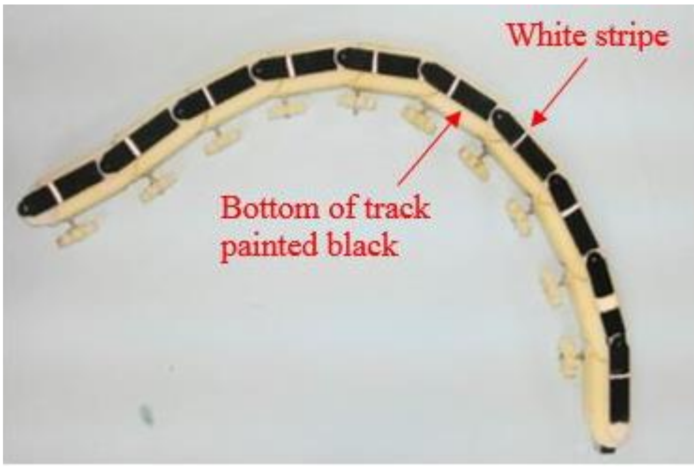


Figure 3.12: The bottom of the tracks were painted black & white in order for the IR sensor to get different readings.

The white stripe was meant for the IR sensor to detect the optimal spot the actuator needs to be positioned along the link, so that the two gears of the rotation transmission will be intagrated successfully.

As the actuator travels a distance of X links, the IR sensor counts the number of white stripes it passes under. Once the actuator reaches the desired joint, the controller (micro-processor) sends the command to the translation motor through the H-bridge to stop. Next, the rotation motor is activated as the encoder counts the number of spins executed by the motor. Once the desired angle is achieved, the controller sends the command to the rotation motor (through the H- bridge) to stop.

The matrix upon which we formed the electric circle connecting all of the components (schematic is shown in Appendix C.2), was placed at the bottom right side of the actuator to maintain its balance (see Fig. 3.13).

Table 3.2: Motion summary of the MARS with the control system, as presented in the experiment. The robot receives the commands through the computer, which is connected via a cable to the control board attached to the mobile actuator.

STEP no.	Translation (start-end)	Rotation [degrees]	
		joint	angle
Reaching the target			
1	(1-2)	2	+45
2	(2-6)	6	-45
3	(6-7)	7	-45
4	(7-2)	2	-45
5	(2-8)	8	-30
6	(8-2)	2	-15
Returning to initial configuration			
7	(2-8)	8	+30
8	(8-2)	2	+45
9	(2-7)	7	+45
10	(7-6)	6	+45
11	(6-2)	2	-30
12	(2-1)	1	0

To demonstrate the maneuverability of the new MARS with the implemented control system, we performed an experiment similar to the one shown in Fig. 3.8. The goal is to retrieve the object (3D printed red minion), which is hidden behind one of the obstacles, and bring it back to the robot's original configuration without colliding with the obstacles. The motion planning algorithm and the experiment (all twelve stages) are presented in Table 3.2 and Fig. 3.13, respectively.

As can be seen from Fig. 3.13, the MARS successfully reached its target, which proves once again that this concept is viable.

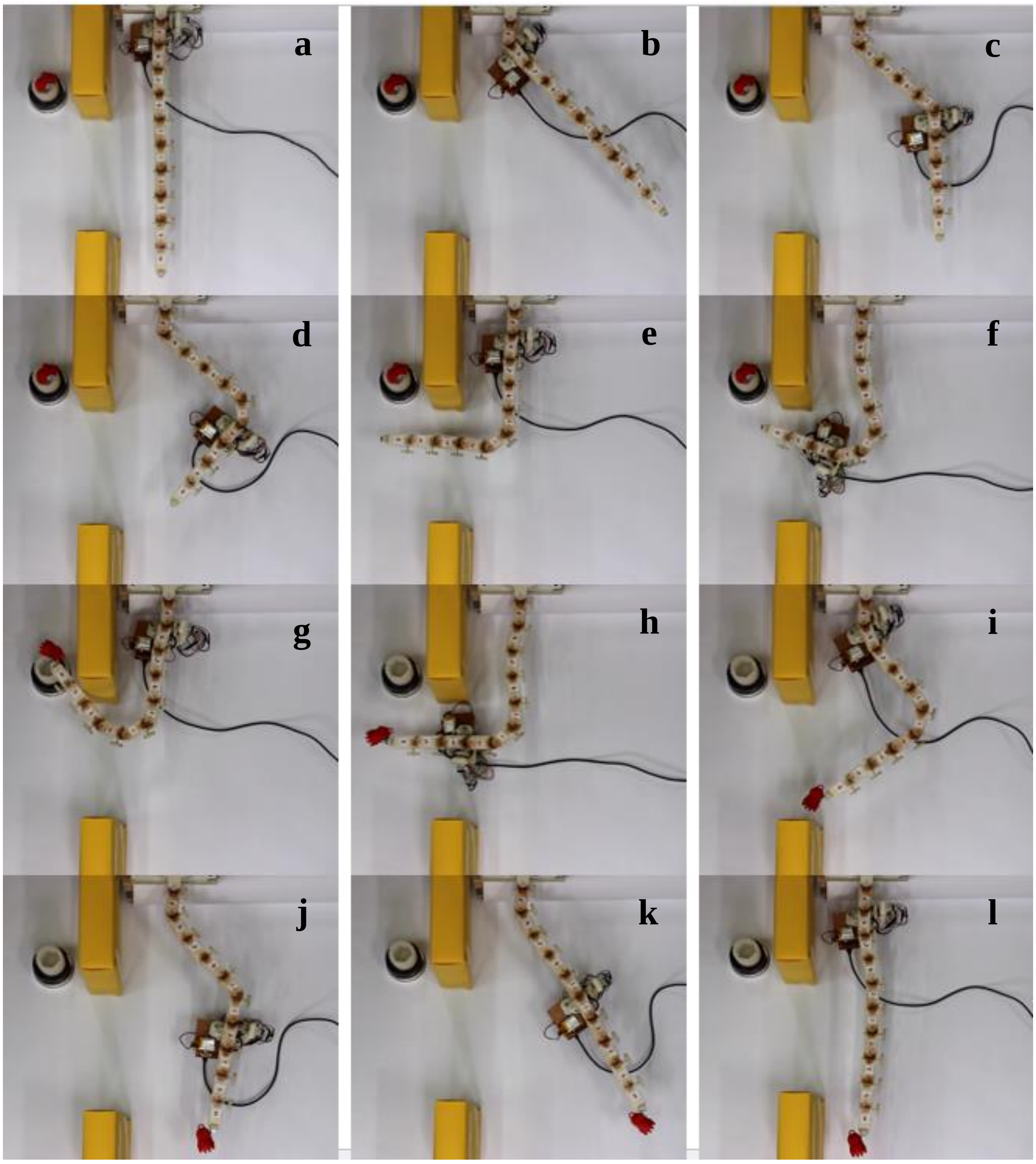


Figure 3.13: The latest experiment showing the MARS with the implemented control system, retrieving the minion (g) and returning to the initial configuration (l).

4. Robot Analysis

This chapter will discuss the four main analyses performed on the MARS in order to investigate its motion and unique structure in operating in a two-dimensional workspace.

4.1 Forward Kinematics

In order to present the configuration of the robot's links in the analyses performed in this chapter, we used the DH convention to form the homogenous transformation matrices as given in Eq. (2.8). Figure 4.1 shows the coordinate frames attached to the joints of the robot (prototype 'A'), where the z-axis is pointed out of the page according to the positive direction of θ, determined by the right-hand rule.

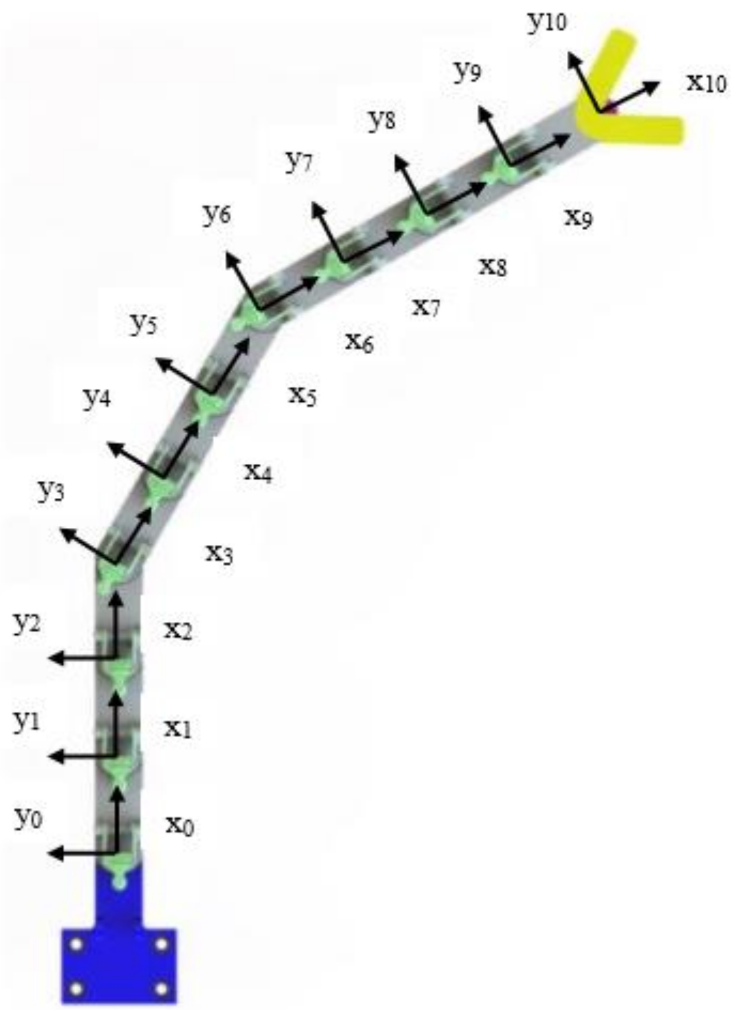


Figure 4.1: Coordinate frames attached to the joints of the MARS, according to the DH convention. The z-all point out of the page and are not shown in the figure.

Since the MARS is a planar robot, consists of revolute joints with one DOF θ_i, the general transformation matrix for each link is of the form:

$$A_i = \begin{bmatrix} c_{\theta_i} & -s_{\theta_i} & 0 & lc_{\theta_i} \\ s_{\theta_i} & c_{\theta_i} & 0 & ls_{\theta_i} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \qquad (4.1)$$

where *l* represents the length of the links (DH parameter - a_i), or the distance between the joints, which is constant. The other parameters, α_i and d_i, are zero according to the DH convention.

In each analysis, the initial configuration of the robot was chosen randomly, where the links (joints) can be rotated in one of the following angles: 0°, ±30°, ±45°, relative to the preceding link (θ_i is the angle between joint $i-1$ and joint *i*). The code for this algorithm, as well as the other analyses performed in this chapter, can be found in Appendix E.

4.2 Motion Planning Algorithm (Inverce Kinematics)

Within the framework of this project, we also explored the robot's capability to maneuver in a two-dimensional workspace. A two-stage algorithm was written in MATLAB® software, for the purpose of finding the optimal configuration of the robot in order to reach any point in its workspace with minimal operations of the mobile actuator. The motion planning algorithm presented in this section was performed in C-space, due to the unique structure and capabilities of the MARS.

As the MARS is constructed of multiple links while the end-effector has only two endpoint coordinates (x, y), it has *N*-2 redundant DOF. There are many different techniques for resolving joint redundancy and different objectives for their resolution. One method to resolve this redundancy is by selecting the joint angles so as to maximize the determinant of J^TJ, where *J* is the Jacobian, while constraining the endpoints to stay on target. This method was applied in the paper "Minimally Actuated Serial Robot", which was co-written with Dr. Moshe Mann and is currently pending acceptance for the prestigious journal *Robotica* of Cambridge University Press. This method was chosen because it is a standard objective in robotics that yields the maximum manipulability, or the ability to exert any desired motion at the manipulator's end-effector.

The algorithm presented in this section was developed in a slightly different method than the one mentioned above. Here, the optimization was accomplished in two stages: First, by using the fmincon© function in the MATLAB™ Optimization Toolbox (which was also used in the paper but with different constraints) to find the optimal final configuration of the robot, and second, by adjusting the results to the current design with the two control systems used in the experiments. This algorithm provides the optimal path within minimal time period (the time required to perform each action of the actuator) and maximal precision. It should be noted that the motion planning algorithm presented here models the robot's motion in an obstacle-free environment, while the robot consists of ten links and only one mobile actuator.

4.2.1 STAGE 1: Optimization Algorithm

Given the initial (random) configuration of the robot, the objective is to find the final configuration of the links in order for the end-effector to reach the target with maximal precision. Since our prototype consists of a large number of joints (N=10), this problem can't be solved using the conventional inverse kinematics equations, as presented in sections B.5-B.7. In order to resolve the robot's redundancy problem we examined several known optimization tools, from which the function fmincon© was found to be conveniently suitable for this purpose.

Based on design demands, the links of the MARS can operate in a range of 90 degrees; meaning, the maximal (relative) rotation angle between two adjacent links is 45 degrees in each direction (CW/CCW). Therefore, the constraint was applied to the joints, where the upper and lower bounds for each DOF are $\pi/4$ and $-\pi/4$, respectively. The cost function was composed of two functions, intended to meet the two main requirements,

$$f_1 = \sum_{i=1}^{N} \left|\theta_{i,fin} - \theta_{i,init}\right|^p \tag{4.2}$$

$$f_2 = \left[\left(L \cdot \sum_{i=k}^{N} \cos\left(\sum_{i=1}^{k} \theta_{i,f}\right)\right) - X_f\right]^2 + \left[\left(L \cdot \sum_{i=k}^{N} \sin\left(\sum_{i=1}^{k} \theta_{i,f}\right)\right) - Y_f\right]^2 \tag{4.3}$$

where:

L - length of the link [m]

N - number of links in the chain

$\theta_{i,init}$ - initial relative angle of link i [rad]

$\theta_{i,fin}$ - final relative angle of link i [rad]

(X_f, Y_f) - target coordinates

p - cost for the number of links to be moved

The function f_1 is designed to minimize the change in the robot's configuration from its initial to final state, which is manifested in the orientation of the links. The function f_2 is designed to minimize the error in the desired positon of the end-effector, thereby ensuring the robot reaches the target with maximal precision. As both demands are equally important to our purpose, the two functions were given equal weight (lambda=1), therefore the total cost function is the sum of f_1 and f_2.

Three cases were examined and compared, where the value of p was chosen to be greater, smaller and equal to 1. In the first case, p=2, f_1 is a parabolic function, meaning there will be an increased additional cost for a larger difference in the links' configuration. In the second case, p=0.5, f_1 is a square-root function, providing decreased additional cost for a larger difference in the orientation of the links. In the third case, p=1, f_1 is a linear function, which means the cost is proportional to the change in the configuration of the links.

This algorithm finds the optimal configuration of the MARS according to the chosen cost function, whether by minimizing the number of links that must be moved (p<1) or by minimizing the displacement angle of each link (p>1). In each case, the end-effector reaches the target with maximum precision.

Figures 4.2-4.4 and Tables 4.1-4.3 present the optimization results based on the cost function combined of Eqns. (4.2) and (4.3), where p equals 2, 0.5 and 1, respectively. The initial configuration (in green) and the target coordinates (marked with a red 'X') were chosen arbitrarily. The final configuration is shown in blue.

As shown in Table 4.1 below, almost all the joints had shifted from their initial positions, except for joint 5. The movements are quite small, between 1 and 3 degrees, which is expected since the target is not very far from the initial position of the end-effector, but mainly due to the choice of p.

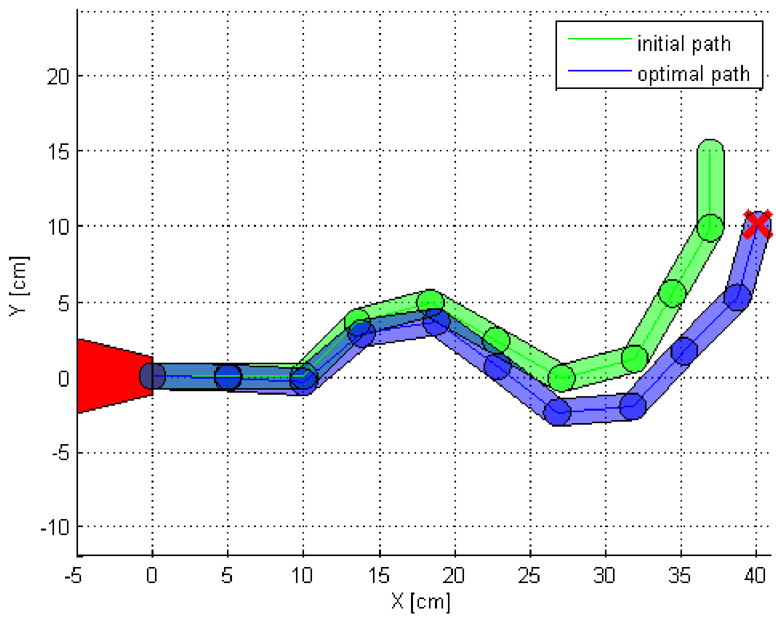


Figure 4.2: Optimization results for the case where f_1 is parabolic (p=2), achieved in 31 iterations.

Table 4.1: Change in configuration of the joints, resulted from the first optimization (p=2).

Link	Initial angle [degrees]	Final angle [degrees]	Angle disp. [degrees]
1	0	-2	-2
2	0	-2	-2
3	+45	+43	-2
4	-30	-31	-1
5	-45	-45	0
6	0	-2	-2
7	+45	+42	-3
8	+45	+42	-3
9	0	-2	-2
10	+30	+29	-1

For the case where p was chosen to be 0.5, it is enticipated to see larger movements of the joints which consequently occur in fewer joints, as the cost decreases for a larger change in the orientation of the already actuated joints. The results shown in Table 4.2 clearly support this assumption since only joint 6 has moved, and the change in its orientation is significantly larger than the one occurred in the first case.

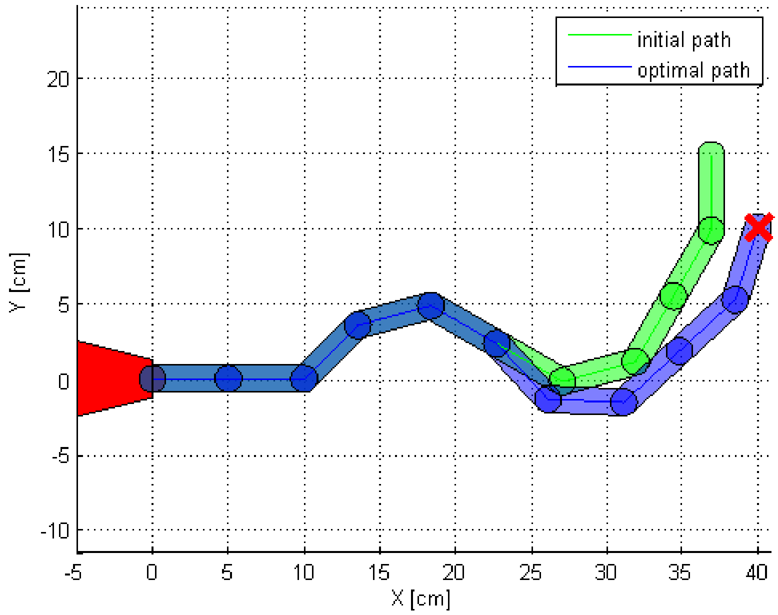


Figure 4.3: Optimization results for the case where f_1 is square-root (p=0.5), achieved in 63 iterations.

Table 4.2: Change in configuration of the joints, resulted from the second optimization (p=0.5).

Link	Initial angle [degrees]	Final angle [degrees]	Angle disp. [degrees]
1	0	0	0
2	0	0	0
3	+45	+45	0
4	-30	-30	0
5	-45	-45	0
6	0	-17	-17
7	+45	+45	0
8	+45	+45	0
9	0	0	0
10	+30	+30	0

In the last case, we chose f_1 to be a linear function, meaning the cost of changing the orientation of the joints is proportional to the number of actuated links. Therefore, in comparison to the previous case, the number of actuated links is expected to increase while the change in orientation is expected decrease. This is also evident from the results presented in Table 4.3.

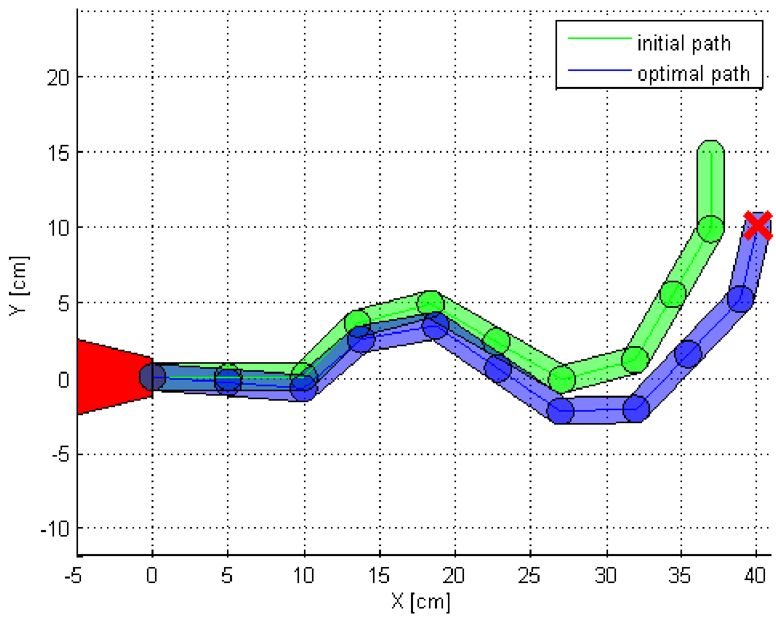


Figure 4.4: Optimization results for the case where f_1 is linear (p=1), achieved in 60 iterations.

Table 4.3: Change in configuration of the joints, resulted from the third optimization (p=1).

Link	Initial angle [degrees]	Final angle [degrees]	Angle disp. [degrees]
1	0	-4	-4
2	0	0	0
3	+45	+45	0
4	-30	-30	0
5	-45	-45	0
6	0	0	0
7	+45	+36	-9
8	+45	+45	0
9	0	0	0
10	+30	+30	0

Note that in all three cases the function f_2 was not changed as the demand for maximum precision is crucial and could not be compromised.

Next, we can calculate the time required for the robot to reach the target in each case, thereby determining the optimal path out of the three options presented above. The mobile actuator has two operations, travelling forward and backward over the links, and rotating the joints CW and CCW. The time to perform each operation was measured according to the speed of the motors used in the latest experiment with prototype 'C', shown in Fig. 3.13.

The average time in which the actuator passes one link is about 2 seconds, and one spin of the rotation motor (4 degrees) takes approximately 2.5 seconds. Under these conditions, the algorithm produces the exact time period required for each path and provides the user with the optimal choice. In this case, for the chosen initial configuration of the robot and the location of the target, the output is:

```
Choose option 3(p=1): Optimal path is achieved within 20 seconds
```

This result makes sense: Option 3 is the reasonable choice since the actuator has to travel a total of seven links and rotate only two joints (no. 1 and 7), as opposed to the first case shown in Table 4.1, in which the actuator has to travel over the entire chain and rotate all joints but one (t_1=29.25 sec). This option is also preferable to the second case because it requires fewer spins of the rotation motor, as the total displacement angle is 13 degrees compared to the 17 degrees required for option 2 (see Table 4.2). Having said that, the end result is quite similar; there is a minor difference between the time periods calculated for each of these options, which is about 0.5 seconds (t_2=20.625 Vs. t_3=20.125 sec).

The first part of the motion planning algorithm provides accurate results for the optimization process, however, it does not take into consideration current design constraints and limitations that arise from the use of different control systems. This issue will be addressed in the following section.

4.2.2 STAGE 2: Adjustments to Control Systems

In order for the motion planning algorithm to be viable, it must produce results which are compatible with the physical features of the robot, such as its mechanical design, so that it could be implemented with the current control systems.

As mentioned in sub-section 3.2.1, the latest prototype of the MARS consists of a gear transmissions which allows the user to control the relative rotation angle between two adjacent links with a precision of up to 2 degrees, where one spin of the upper gear rotates the joints 4 degrees in the desired direction. During the development of prototype ‘C’, two control systems were used to operate the robot. Initially, the robot was controlled manually by the operator via a remote, a method which was also used in the experiments performed with prototype ‘A’. Later, an algorithm was written in C++ which allows the operator to control the robot via the computer, enabling the MARS to move autonomously. Although this method proved to be quite successful, for the moment, the control of the rotation motor is programmed according to the number of full spins executed by the upper gear; this means that the relative rotation angle of the joints is determined to a resolution of 4 degrees instead of 2.

For this reason, the results of the optimization performed in the previous section should be adjusted to the control system of choice. Therefore, the second part of the motion planning algorithm requires the user to first choose the way in which the MARS is operated:

```
Enter 1 for Automated control or 2 for Manual control: 1
```

Figures 4.5-4.7 present the new configuration of the robot calculated for each of the cost functions, where *p* equals 2, 0.5 and 1, respectively. The final configuration of the robot is shown in pink.

The values of the original (accurate) displacement angles of the joints were rounded to the nearest values that are multiples of 4, which is the resolution of the automated control system. These adjustments are shown in Tables 4.4-4.6. Note that in the case of multiple intermediate values, where the residual of the quotient is 2 (e.g., Table 4.4), the results are rounded up or down intermittently to the nearest value that is a multiple of 4.

Since the displacement angles were altered to fit the practical needs of the system, it is expected that the end-effector would not be able to reach the target with the same precision as before. Therefore, we determined a threshold for the acceptable error of the end-effector, above which the new path will not be chosen.

The threshold was chosen to be a distance of 0.5 cm from the target, which is 1% of the entire length of the robot. The error is calculated using Pythagoras theorem and indicated in the bottom of each figure. Observing Figures 4.5-4.7, one can notice there is indeed a deviation of the end-effector from the location of the target, as predicted.

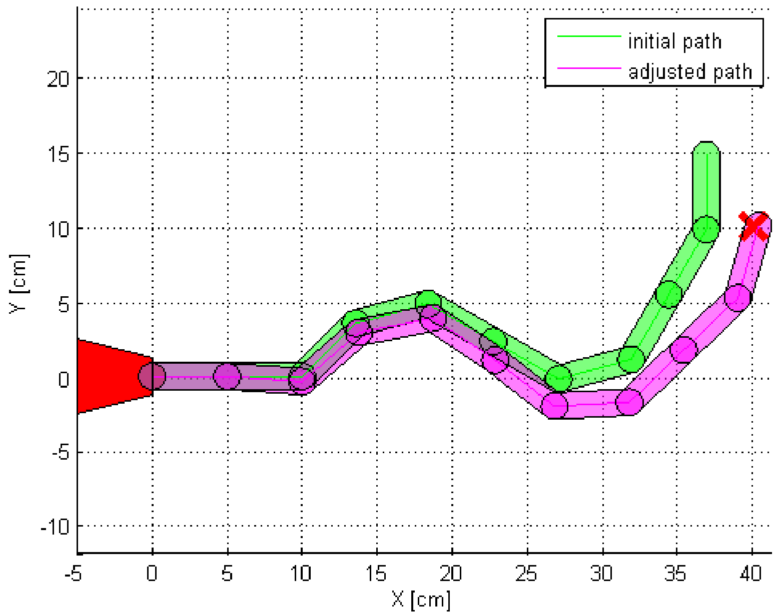


Figure 4.5: Adjusted results to the automated control system, in the case where f_1 is parabolic (p=2), error is 0.3205 cm.

Table 4.4: Change in configuration of the joints for the automated control system, in the case where f_1 is parabolic (p=2).

Link	Initial angle [degrees]	Optimal angle disp. [degrees]	Adjusted angle disp. [degrees]	Adjusted final angle [degrees]
1	0	-2	0	0
2	0	-2	-4	-4
3	+45	-2	0	+45
4	-30	-1	0	-30
5	-45	0	0	-45
6	0	-2	-4	-4
7	+45	-3	-4	+41
8	+45	-3	-4	+41
9	0	-2	0	0
10	+30	-1	0	+30

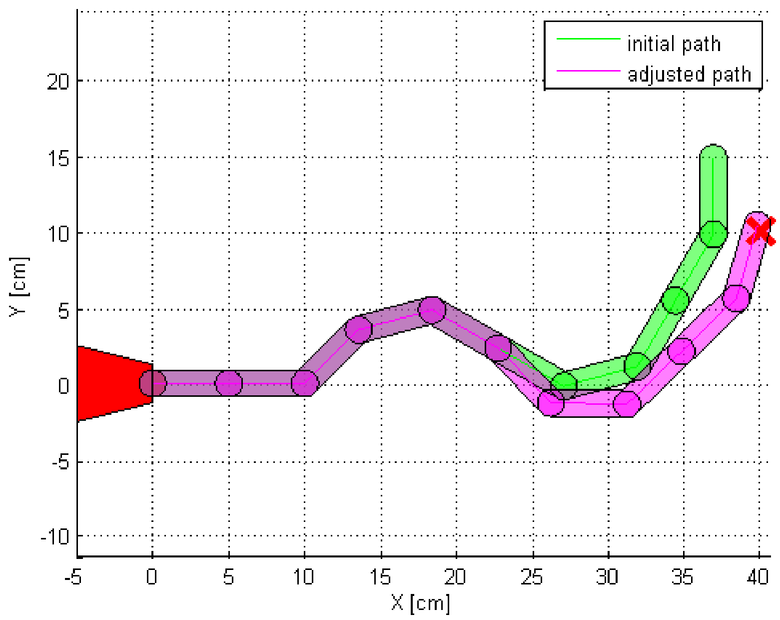


Figure 4.6: Adjusted results to the automated control system, in the case where f_1 is square-root (p=0.5), error is 0.4762 cm.

Table 4.5: Change in configuration of the joints for the automated control system, in the case where f_1 is square-root (p=0.5).

Link	Initial angle [degrees]	Optimal angle disp. [degrees]	Adjusted angle disp. [degrees]	Adjusted final angle [degrees]
1	0	0	0	0
2	0	0	0	0
3	+45	0	0	+45
4	-30	0	0	-30
5	-45	0	0	-45
6	0	-17	-16	-16
7	+45	0	0	+45
8	+45	0	0	+45
9	0	0	0	0
10	+30	0	0	+30

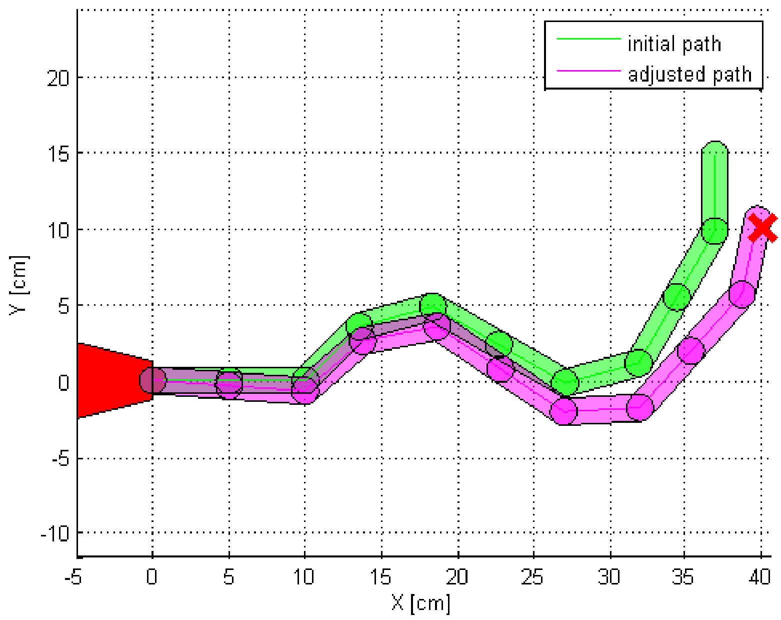


Figure 4.7: Adjusted results to the automated control system, in the case where f_1 is linear (p=1), error is 0.6223 cm.

Table 4.6: Change in configuration of the joints for the automated control system, in the case where f_1 is linear (p=1).

Link	Initial angle [degrees]	Optimal angle disp. [degrees]	Adjusted angle disp. [degrees]	Adjusted final angle [degrees]
1	0	-4	-4	-4
2	0	0	0	0
3	+45	0	0	+45
4	-30	0	0	-30
5	-45	0	0	-45
6	0	0	0	0
7	+45	-9	-8	+37
8	+45	0	0	+45
9	0	0	0	0
10	+30	0	0	+30

From the results presented above, the conclusion is that the optimal configuration of the robot should be executed according to option 1 (p=2) or 2 (p=0.5). The algorithm alerts the user in case the calculated error exceeds the determined threshold. For the aforementioned example the output is:

```
Option3(p=1): Error=0.62234, End-effector position Exceeds acceptable Error(0.5cm)
```

Now we can calculated the time required for the actuator to achieve the new configurations in each case. Although option 3 (p=1) is the least preferable choice, as it produces the largest error out of the three, it is in fact the best option time-wise: t_3=19.5 sec. However, as mentioned above, this option does not meet the accuracy demand and therefore will not be chosen.

Comparing the two viable options, 1 and 2, the difference between the time periods is 4 seconds, where t_1=24 sec and t_2=20 sec, which means option 2 (p=0.5) best serves our purpose.

In the case where the control is executed manually by the operator:

```
Enter 1 for Automated control or 2 for Manual control: 2
```

Here, we can achieve a resolution of 2 degrees in rotation, which means fewer adjustments are needed to be performed on the results obtained from the optimization process. Consequently, the new configurations of the robot for the three cases examined are expected to be more similar to the original results shown in Figures 4.2-4.4. The final configurations are presented in Figures 4.8-4.10.

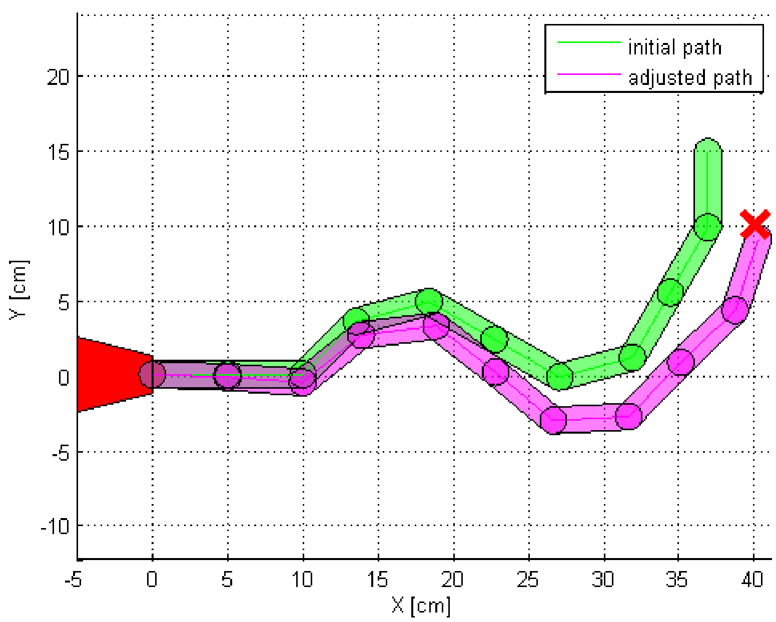


Figure 4.8: Adjusted results to the manual control system, in the case where f_1 is parabolic (p=2), error is 0.9954 cm.

Table 4.7: Change in configuration of the joints for the manual control system, in the case where f_1 is parabolic (p=2).

Link	Initial angle [degrees]	Optimal angle disp. [degrees]	Adjusted angle disp. [degrees]	Adjusted final angle [degrees]
1	0	-2	-2	-2
2	0	-2	-2	-2
3	+45	-2	-2	+43
4	-30	-1	-2	-32
5	-45	0	0	-45
6	0	-2	-2	-2
7	+45	-3	-2	+43
8	+45	-3	-2	+43
9	0	-2	-2	-2
10	+30	-1	-2	+28

The alterations made to the displacement angles of the joints included only the odd values, which were rounded up or down intermittently to the nearest even value. These results are displayed in Tables 4.7-4.9 for the three cases, respectively.

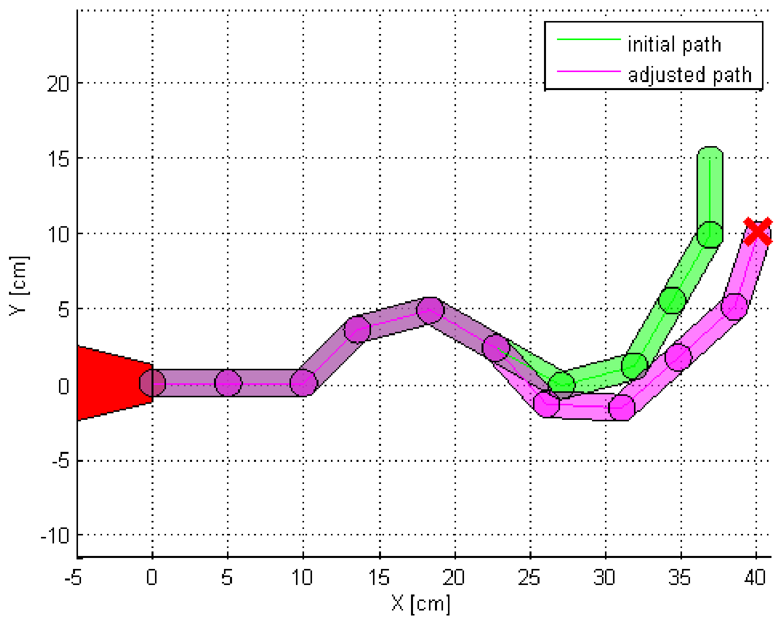


Figure 4.9: Adjusted results to the manual control system, in the case where f_1 is square-root (p=0.5), error is 0.2015 cm.

Table 4.8: Change in configuration of the joints for the manual control system, in the case where f_1 is square-root (p=0.5).

Link	Initial angle [degrees]	Optimal angle disp. [degrees]	Adjusted angle disp. [degrees]	Adjusted final angle [degrees]
1	0	0	0	0
2	0	0	0	0
3	+45	0	0	+45
4	-30	0	0	-30
5	-45	0	0	-45
6	0	-17	-18	-18
7	+45	0	0	+45
8	+45	0	0	+45
9	0	0	0	0
10	+30	0	0	+30

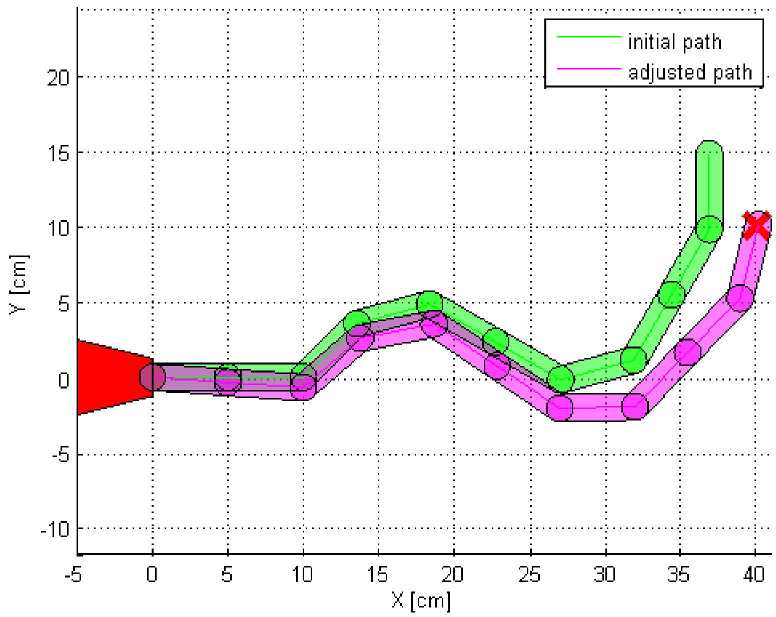


Figure 4.10: Adjusted results to the manual control system, in the case where f_1 is linear (p=1), error is 0.1287 cm.

Table 4.9: Change in configuration of the joints for the manual control system, in the case where f_1 is linear (p=1).

Link	Initial angle [degrees]	Optimal angle disp. [degrees]	Adjusted angle disp. [degrees]	Adjusted final angle [degrees]
1	0	-4	-4	-4
2	0	0	0	0
3	+45	0	0	+45
4	-30	0	0	-30
5	-45	0	0	-45
6	0	0	0	0
7	+45	-9	-10	+35
8	+45	0	0	+45
9	0	0	0	0
10	+30	0	0	+30

As shown in Figures 4.8-4.10, the largest deviation of the end-effector from the target occurred in the first case (p=2), as opposed to the results presented in Figure 4.5, where the calculated error was the smallest out of the three options. However, for the second and third case, the calculated error was significantly smaller compared to the adjustments made for the automated control system. Since the first option was the only one of the three with an error above the threshold, the output is:

```
Option1(p=2): Error=0.9954, End-effector position Exceeds acceptable Error(0.5cm)
```

From the results presented above, there are two viable options for the optimal path of the robot. Now we can choose the best option according to the time required for the actuator to perform the task in each case. After calculation of the time periods, the algorithm produces this output:

```
Choose option 3(p=1): Optimal path is achieved within 21 seconds
```

Option 3 is indeed the best choice in this case, both time-wise and precision-wise, though the difference between this option and the second one (p=0.5) is negligible, as t_2=21.25 sec and t_3=20.75 sec.

In conclusion, the adjustments of the optimization results to the current control systems yield different outcomes in each of the three cases examined that could not have been predicted by the human operator. It is evident that the slightest change made to original values, such as an addition or subtraction of one degree from the orientation of a single joint, will result in a significantly different outcome, which will ultimately affect the operator's choice in determining the optimal path of the MARS.

4.3 Velocity Kinematics

Given the velocity of the joints, we can calculate the velocity of the end-effector from Eq. (2.19) using the Jacobian. As the current design of the MARS consists of one mobile actuator, only one link can be actuated at each stage of the robot's motion. Therefore, we examined two scenarios in which the velocity of the end-effector was calculated at six stages of the experiment performed with prototype 'A', as listed in Table 3.1 (stages 1-5 and 8). Figure 4.11 illustrates the transition from one configuration to the next at each step, indicating the joint number and the relative rotation angle. The Jacobian was recalculated at each step of the robot's motion.

First, the end-effector velocity was calculated for the case where all joints have the same angular velocity of 32 RPM (3.35 rad/sec), which is the actual speed of the rotation motor used in the experiment.

Table 4.10: The initial velocity vector of the end-effector at six stages of the experiment. All the joints were given the same angular velocity - 32 RPM (speed of the rotation motor).

STEP no.	Rotation [degrees]		Initial linear velocity [m/sec]				Angular velocity [rad/sec]		
	joint	angle	v_x	v_y	v_z	$\lvert v_{tot} \rvert$	ω_x	ω_y	ω_z
1	1	+45	0	1.6750	0	1.6750	0	0	3.35
2	2	+45	-1.0660	1.0660	0	1.5075	0	0	3.35
3	6	-45	-0.8375	0	0	0.8375	0	0	3.35
4	7	-45	-0.4738	0.4738	0	0.6700	0	0	3.35
5	9	-45	0	0.3350	0	0.3350	0	0	3.35
8	10	+45	0.1184	0.1184	0	0.1675	0	0	3.35

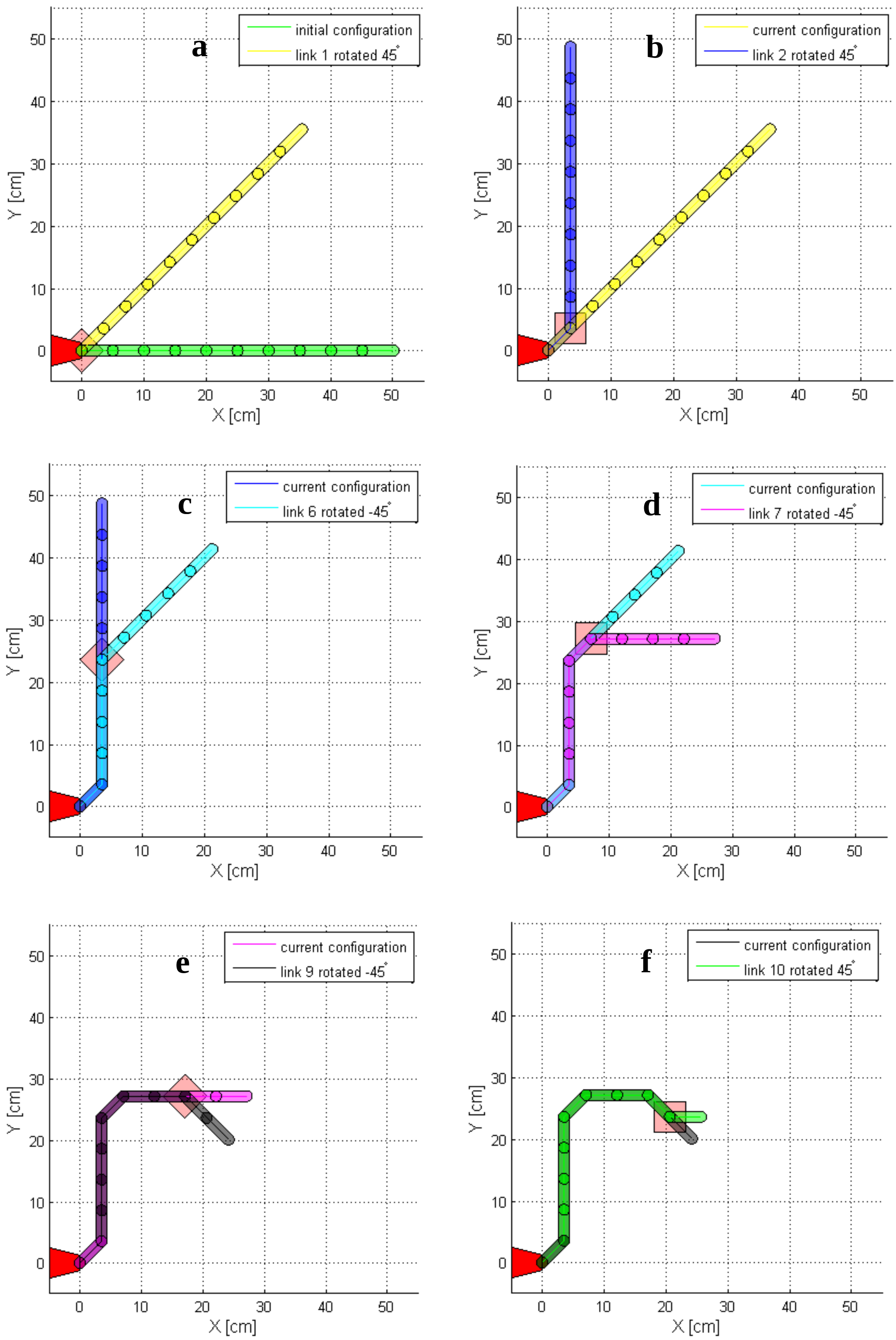


Figure 4.11: The six stages demonstrated in the experiment performed with prototype 'A', showing the change in the configuration of the links from (a)-(f) as listed in Table 4.10.

In the second (theoretical) scenario, each joint was given a different angular velocity, which was chosen randomly from this set of values: 1.5, 3 and 4 rad/sec. Tables 4.10 and 4.11 display the initial velocity vector of the end-effector for the two scenarios discribed above, respectively. Table 4.10 clearly shows that when given the same angular velocity in all the joints, the total linear velocity of the end-effector decreases as the actuated joint is closer to the tip of the chain. This is expected since the linear velocity components of the Jacobian are calculated based on the distance from the joints to the end-effector (see Eqns. (4.14) and (4.16) in section 4.4).

Figures 4.12 and 4.13 display the linear velocity components, v_x and v_y, and the total linear velocity of the end-effector for the six joints as a function of time, where the speed of the rotation motor is 15 degrees/sec, according to the experiment. Figure 4.12 shows there is a clear trend in the values of the end-effector velocity, as the total linear velocity (shown in cyan) decreases with time as the actuated joint is closer to the tip of the chain.

In the case where the joints were given different angular velocities, the same consistency applies as before. The results presented in Table 4.11 clearly shows that for two adjacent links with the same angular velocity, the total liner velocity of the end-effector decreases as the actuated joint is closer to the tip, due to the use of the Jacobian.

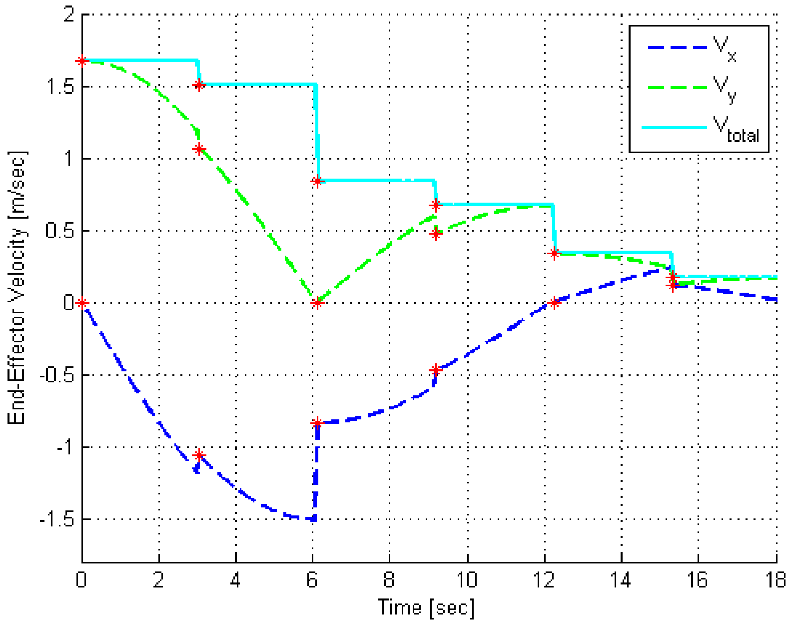


Figure 4.12: Linear velocity components, v_x (in blue) and v_y (in green), of the end-effector for the six stages demonstrated in the experiment, where each joint was given the same angular velocity - 32 RPM (3.35 rad/sec). The total linear velocity is shown in cyan, and the velocity values at the beginning of each stage are marked with red asterisks.

Table 4.11: The initial velocity vector of the end-effector at six stages of the experiment. Each joint was given a different angular velocity, chosen randomly from the values 1.5, 3 and 4 rad/sec.

STEP no.	Rotation [degrees]		Initial linear velocity [m/sec]				Angular velocity [rad/sec]		
	joint	angle	v_x	v_y	v_z	$\lvert v_{tot} \rvert$	ω_x	ω_y	ω_z
1	1	+45	0	0.7500	0	0.7500	0	0	1.5
2	2	+45	-0.4773	0.4773	0	0.6750	0	0	1.5
3	6	-45	-0.7500	0	0	0.7500	0	0	3
4	7	-45	-0.4243	0.4243	0	0.6000	0	0	3
5	9	-45	0	0.4000	0	0.4000	0	0	4
8	10	+45	0.1414	0.1414	0	0.2000	0	0	4

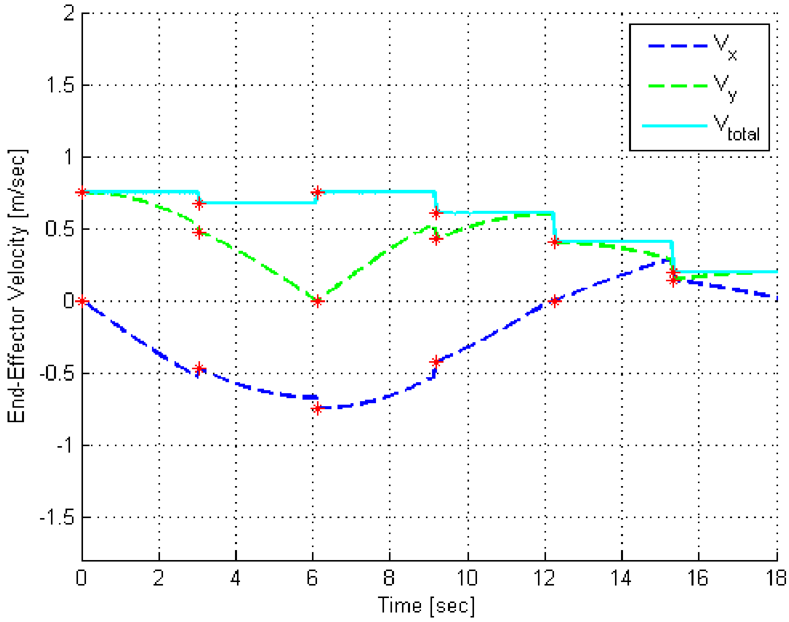


Figure 4.13: Linear velocity components, v_x (in blue) and v_y (in green), of the end-effector for the six stages demonstrated in the experiment, where each joint was given a different angular velocity, chosen randomly from the values 1.5, 3 and 4 rad/sec. The total linear velocity is shown in cyan, and the velocity values at the beginning of each stage are marked with red asterisks.

However, the magnitude of the angular velocity also plays a significant role in determining the velocity of the end-effector, as evident from the results for joints 2 and 6. Although joint 6 is much closer to the tip of the chain, the total linear velocity end-effector is greater than the one calculated when joint 2 is actuated, because the angular velocity of joint 6 was doubled. This increase in the end-effector's velocity can also be observed in Fig. 4.13.

The results presented in Tables 4.10 and 4.11 show there is a linear dependency between the angular velocity given in a specific joint and the total linear velocity of the end-effector, which corresponds to the basic dynamic equation:

$$\vec{v} = \vec{\omega} \times \vec{r} \tag{4.4}$$

where $\vec{r}$ is the vector from joint *i* to the end-effector.

4.4 Structural Rigidity

To examine the structural rigidity of the MARS, we performed a strength analysis of the robot by referring to the chain of links as a robotic arm (open kinematic chain). The goal is to calculate the displacement of the links as a result of an external force applied to the robot at its tip (the end-effector).

In considering static forces in a manipulator, we first lock all the joints so that the manipulator becomes a structure. We then consider each link in this structure and write a force-moment balance relationship in terms of the link frames. Finally, we compute what static torque must be acting about the joint axis in order for the manipulator to be in static equilibrium. In this way, we solve for the set of joint torques needed to support a static load acting at the end-effector.

In this section, we will not be considering the force acting on the links due to gravity. The static forces and torques we are considering at the joints are those caused by a static force acting on the last link; for example, as when the manipulator has its end-effector in contact with the environment.

Figure 4.14 shows the initial configuration of the robot, in which the links are locked in one of the five possible angles, as mentioned in the beginning of the chapter (section 4.1). The blue arrow in the figure represents the external force applied to the end-effector, with a magnitude of 5 N directed +45 degrees with respect to the x-axis (these values were also chosen arbitrarily).

From the experiments performed on the robot, it is evident that applying external force to the end-effector forms small perturbations along the chain of links.

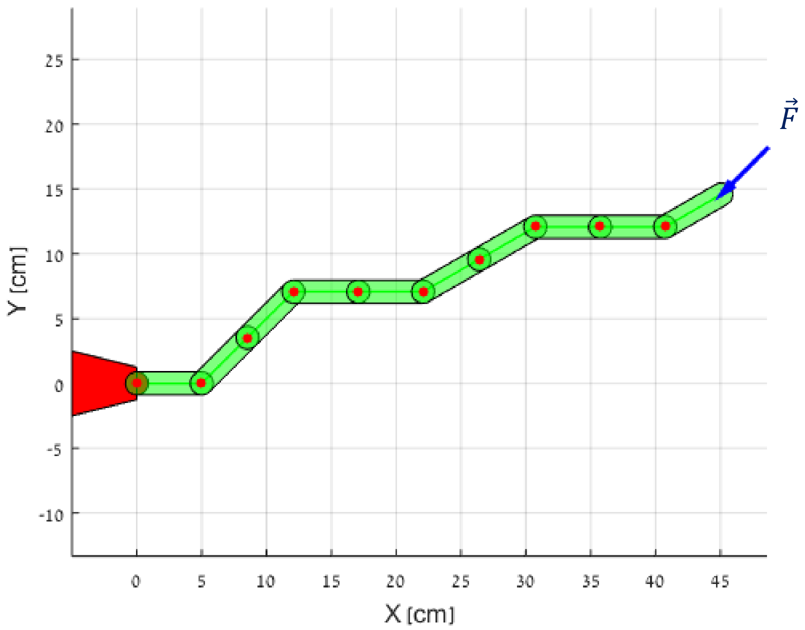


Figure 4.14: Initial configuration of the links (in green), chosen randomly. The blue arrow represents the force vector (F=5 N at +45°) applied to the end-effector, and the red dots represent the location of the joints.

Some perturbations are more significant than others, depending on the initial angle in which the link is locked and the distance between the joint and the position of the force, which affects the torque created in the joint.

This analysis is based on the latest design of the robot (prototype 'C'), in which the locking mechanism consists of a worm drive transmission, as opposed to the earlier version, where the lock was based completely on friction. Hence, the two main factors that contribute to the formation of these perturbations are the backlash of the transmission, and the stiffness of the joint (measured in an experiment conducted on the links of the current prototype, found in Appendix D).

Fig. 4.15 shows the backlash of the worm drive transmission, as measured in the SolidWorks software. The "freedom" between the worm and the worm gear is illustrated in Fig. 4.15(a) and (b), marked with a red circle.

It is evident from Fig. 4.15(c) there exists a small shift of 0.87 mm between the joints of two adjacent links, which results directly from the backlash. The center distance between the joints is constant and equals to 5 cm, therefore the backlash angle can be calculated using simple trigonometry:

$$\sin(\theta_b) = \frac{0.87}{50} \;\Rightarrow\; \theta_b = 0.997° \cong 1°. \tag{4.5}$$

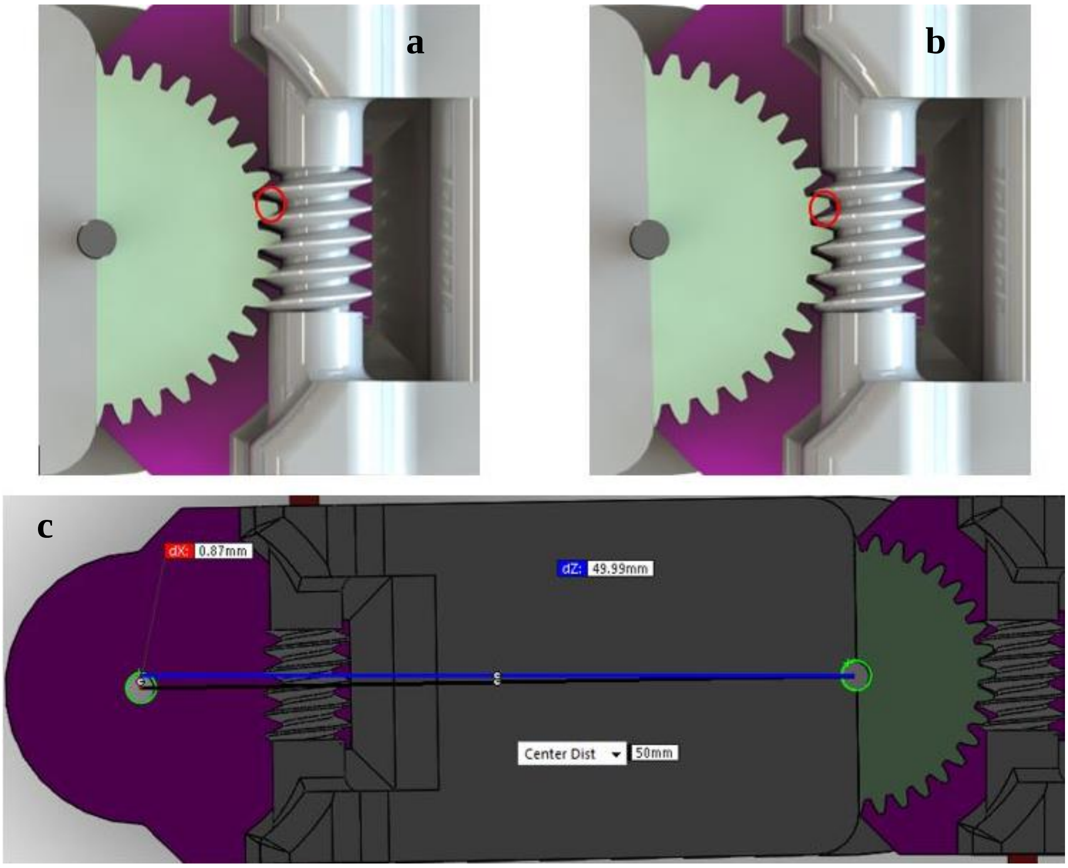


Figure 4.15: The backlash of the worm drive transmission between two adjacent links. The red circle shows the space created between one tooth of the worm gear and the worm (a) when the links are aligned, and (b) when the preceding link is slightly tilted, so that the same tooth touches the helix of the worm from its other side. (c) The distance between the joints of the links resulted from the backlash, as measured in SolidWorks.

In order to develop the model, we first examined the simplest case where only one link is connected to the base link, and a horizontal force is applied to the tip of the link. This case is illustrated in Fig. 4.16 below.

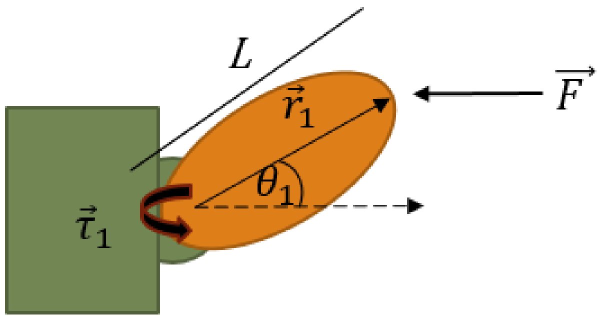


Figure 4.16: Force-moment analysis of one link under the influence of a horizontal force $\vec{F}$ applied at the tip.

The parameters used in this analysis are as follows:

L - length of the link [m]

$\vec{F}$ - external force applied to the tip of the link [N]

$\vec{r}_i$ - range vector from the joint to the point of application [m]

$\vec{\tau}_i$ - torque created in the joint due to the force [Nm]

θ_{init} - initial angle in which the link is locked [rad]

θ_b - backlash angle (estimated at 1 degree, according to Eq. (4.5)) [rad]

K_t - stiffness/torsion coefficient of the joint (estimated at 2 Nm/degree, based on the experiment) [Nm/rad]

θ_t - torsion angle created by the torque [rad]

The tourque created in the joint due to the force $\vec{F}$ is defined as

$$\vec{\tau}_1 = \vec{r}_1 \times \vec{F} \tag{4.6}$$

and the relation between the torque and the torsion angle is given by

$$\tau = K_t \cdot \delta\theta_t \tag{4.7}$$

according to Hooke's law for torsion springs.

Therefore, the torsion angle can be expressed from Eq. (4.6) and (4.7) as

$$\delta\theta_{t_1} = \frac{\tau_1}{K_t} = \frac{\vec{r}_1 \times \vec{F}}{K_t}. \tag{4.8}$$

The total angle displacement of each link is influenced by the backlash of the worm drive transmission and the torque generated in the joint, so that

$$\delta\theta = \delta\theta_b + \delta\theta_t \tag{4.9}$$

as the direction of the backlash angle (CW/CCW) depends on the direction of the torque:

$$\delta\theta_{b_1} = sign(\tau_1) \cdot |\delta\theta_b|. \tag{4.10}$$

Substituting Eq. (4.8) and (4.10) into (4.9) yields the expression for the total angle displacement,

$$\delta\theta_1 = sign(\tau_1) \cdot |\delta\theta_b| + \frac{\vec{r}_1 \times \vec{F}}{K_t} \tag{4.11}$$

and the final angle is:

$$\theta_{fin_1} = \theta_{init_1} + sign(\tau_1) \cdot |\delta\theta_b| + \frac{\vec{r}_1 \times \vec{F}}{K_t} \tag{4.12}$$

where θ_{init_1} is the initial angle of the link (prior to the application of the force).

This analysis was also performed for the case where the robot consists of two links, and then for three links, as illustrated in Figure 4.17 below (for the complete analysis, see Appendix D).

The anlysis was based on the assumptions that the force acting at the tip is relatively small, the movements occur slowly and the accelerations in the joints are negligible. Therefore, the torques created in the joints are calculated according to Eq. (4.6), where $\vec{r}_i$ is the range vector from each joint to the end-effector (point of application). Also, under the assumption of small perturbations, the absolute angle of the final link can be calculated using superposition, since this is a linear system.

For a serial robot composed of *N* links, the final (absolute) angle of the end-effector is calculated according to Eq. (4.13):

$$\theta_{end,fin} = \sum_{k=1}^{N} \theta_{init_k} + |\delta\theta_b| \cdot \sum_{k=1}^{N} sign(\tau_k) + \frac{1}{K_t}\left(\sum_{k=1}^{N} k \cdot \vec{r}_k\right) \times \vec{F} \tag{4.13}$$

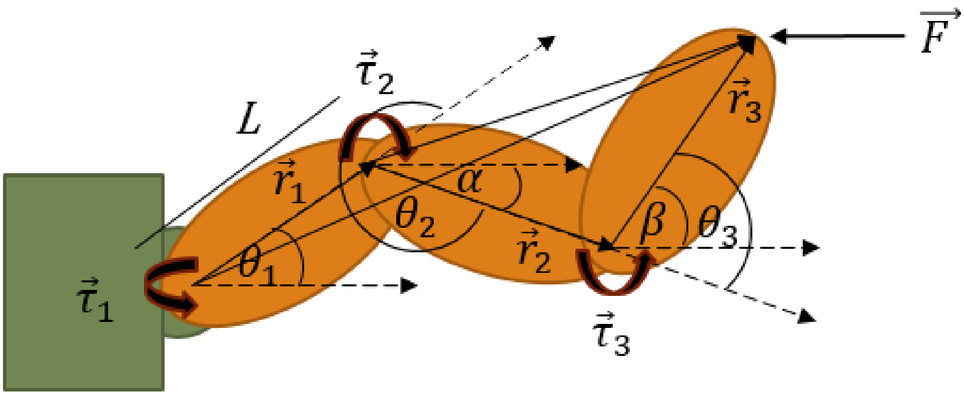


Figure 4.17: Force-moment analysis of three links under the influence of a horizontal force applied at the tip.

The final angle of each link is calculated in the same manner, as the summation limit (upper bound) varies according to the joint.

The result of this analysis is illustrated in Figure 4.18. As expected in the case of the chosen initial configuration, the most significant change in the location of the joints manifested in the last link, which resulted from the perturbations occurred in each link along the chain.

Table 4.12 presents the torques created in the joints due to the application of the force and the total angle displacement of each link, as calculated from the model. It is evident there is a clear trend in the values of the torques and the total angle displacement depending on the distance of the joint from the point of application. The largest torque is created in the first joint, which connects the first link to the base link, and the smallest torque is created in the last link, as it is closest to the location of the force. The total displacement angles correspond to the values of the torques, since this calculation includes the torsion angles (see Eq. (4.9)), however, they are mainley affected by the backlash angle, which is approximately 1 degree.

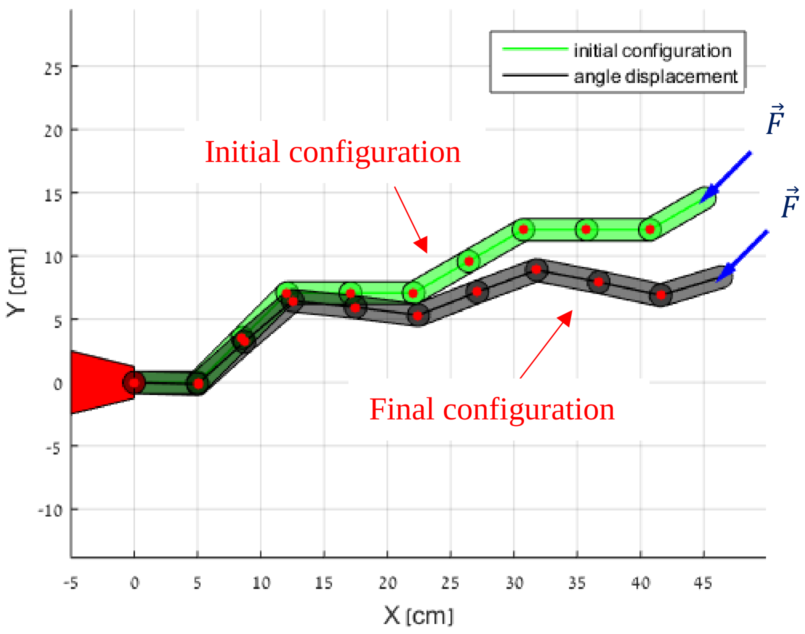


Figure 4.18: Configuration of the links prior to and following the application of the force (green vs. black), according to the model. The blue arrows represents the force vector (F=5 N at +45°) applied to the end-effector, and the red dots represent the location of the joints.

Table 4.12: The torques created in the joints and the total displacement angle of each link as a result of the external force acting on the end-effector, calculated according to the model.

Link	Torque [Nm]	Total disp. angle [degrees]
1	-1.0780	-1.5371
2	-0.9012	-1.4490
3	-0.9012	-1.4490
4	-0.9012	-1.4490
5	-0.7244	-1.3609
6	-0.5477	-1.2729
7	-0.4830	-1.2406
8	-0.4183	-1.2084
9	-0.2415	-1.1203
10	-0.0647	-1.0322

For comparison, the torques created in the joints were also calculated according to Eq. (2.33), which results from the principal of virtual work. The Jacobian was calculated according to Eq. (2.26),

$$J_i = \begin{bmatrix} J_{v_i} \\ J_{w_i} \end{bmatrix} = \begin{bmatrix} z_{i-1} \times (o_N - o_{i-1}) \\ z_{i-1} \end{bmatrix}. \tag{4.14}$$

where the *i*-th column J_i corresponds to the *i*-th joint of the robot. Since the MARS is a planar robot consists of revolute joints, the z-axes all point in the same direction (out of the page) according to the DH convention, as shown in Figure 4.1. Therefore, the angular velocity vector for each joint is given by

$$z_{i-1} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}. \tag{4.15}$$

In order to find the linear velocity components of the Jacobian matrix, we calculated the distance from the end-effector to the joints,

$$d_{i-1}^{N} = o_N - o_{i-1} \tag{4.16}$$

where o_N is the position of the end-effector.

The position of the coordinate frames, as well as the end-effector, can be derived from the fourth column of the homogenous transformation matrices.

The Jacobian matrix[3] for the initial configuration presented in Figure 4.14 is

$$J=\begin{bmatrix} -14.57 & -14.57 & -11.04 & -7.5 & -7.5 & -7.5 & -5 & -2.5 & -2.5 & -2.5 \\ 45.06 & 40.06 & 36.53 & 32.99 & 27.99 & 22.99 & 18.66 & 14.33 & 9.33 & 4.33 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

and the force was chosen to be

$$\vec{F}=\begin{bmatrix} -3.5355 \\ -3.5355 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}\text{N.}$$

Hence, the torques in the joints are

$$\tau=J^{T}F=\begin{bmatrix} -1.0780 \\ -0.9012 \\ -0.9012 \\ -0.9012 \\ -0.7244 \\ -0.5477 \\ -0.4830 \\ -0.4183 \\ -0.2415 \\ -0.0647 \end{bmatrix}\text{Nm.} \tag{4.17}$$

Eq. (4.17) clearly validates the results presented in Table 4.12, which were calculated from the force-moment model.

Another interesting point for discussion regarding this analysis is finding the maximal force (size and direction) which will cause failure to the entire structure of the robot, depending on the initial configuration of the links. This can be accomplished using optimization methods, similar to those discussed in section 4.2 but with different contraints, cost functions and boundary conditions. Unfortunately, due to time duress, this point was not further investigated within the scope of this work.

[3] Data values of the Jacobian are given in centimeters instead of meters due to lack of space.

5. Summary and Conclusions

This work has introduced a minimally actuated robotic snake (MARS). The MARS can execute complex motions with a small number of actuators. It consists of a mobile actuator that shifts its position along the joints of the robot. This enables the actuator to shape the robot to any desired position by incrementally adjusting all of its joints. The capabilities of the MARS were first tested theoretically and shown by an example in which the robot successfully manipulates an object while maneuvering around obstacles, using a computerized animation. We have described the unique kinematics of the MARS and demonstrated how it can duplicate the motion of a fully actuated robot to within any desired degree of accuracy.

The robot is suitable for applications in a complex and confined environment with low payload that do not require rapid deployment. While the robot cannot hold large weights, it is a "rigid" mechanism (not compliant) in the sense that it is not meant to deform due to performance of its tasks. The robot is also very modular - the number of links and mobile actuators can be changed in a matter of minutes to adjust it to a specific task.

We built an experimental robot with ten links and one mobile actuator. We developed two main prototypes designed to show how by using a single mobile actuator, it is possible to control the ten joints of our robot and penetrate through a confined space and reach the target. We found that the control is simple and intuitive, and only a few minutes are required for a human operator to learn how to actuate the robot. We were able to perform simple tasks that included going through a small pass and reaching a target. Moreover, we demonstrated the robot's capabilities in achieving different configurations, such as a 'C' shape or an 'S' shape.

Further research and development of the MARS is ongoing. New improved designs are being developed for the physical actuating mechanism that will yield a more rigid structure (by producing metal links) and smoother motions in 3D, as well as reduce errors and malfunctions by fitting the mobile actuator with a more accurate controller and sensors. In our future work we aim at developing a comprehensive general motion planning algorithm to yield optimal motions for the MARS in an obstacle-embedded environment for one or more actuators. In addition, ex-vivo experiments using pig intestine should be performed in order to test the forces exerted by the robot (a minimized version) as it moves through biological vessels.

Appendix A - Histology of the Gastrointestinal Tract

The GI tract can be divided into four concentric layers in the following order: mucosa, submucosa, muscular layer and serosa or adventitia, depending on whether the tissue is intraperitoneal or retroperitoneal, respectively. The mucosa is the innermost layer of the GI tract, surrounding the lumen. This layer comes in direct contact with digested food and it is highly specialized in each organ of the GI tract to deal with the different conditions. The submucosa consists of a dense irregular layer of connective tissue with large blood vessels, lymphatics, and nerves branching into the mucosa and muscularis externa. The muscular layer consists of an inner layer, which prevents food from traveling backward, and an outer layer which shortens the tract. The coordinated contractions of these layers is called peristalsis and propels the food through the tract; peristalsis is controlled by the myenteric plexus, located between the two muscle layers [29].

The outermost layer of the GI tract consists of several layers of connective tissue. The intraperitoneal parts of the GI tract, which are located within the abdominal cavity, are covered with serosa and have a mesentery. These include the small intestine, appendix, cecum, transverse and sigmoid colon, rectum, first part of the duodenum and most of the stomach. In these sections there is clear boundary between the gut and the surrounding tissue. The retroperitoneal parts, which are the structures in the abdominal cavity that are located behind the intraperitoneal space, are covered with adventitia. They blend into the surrounding tissue and are fixed in position. These include the esophagus, pylorus of the stomach (connects the stomach to the duodenum), distal duodenum, ascending and descending colon and anal canal.

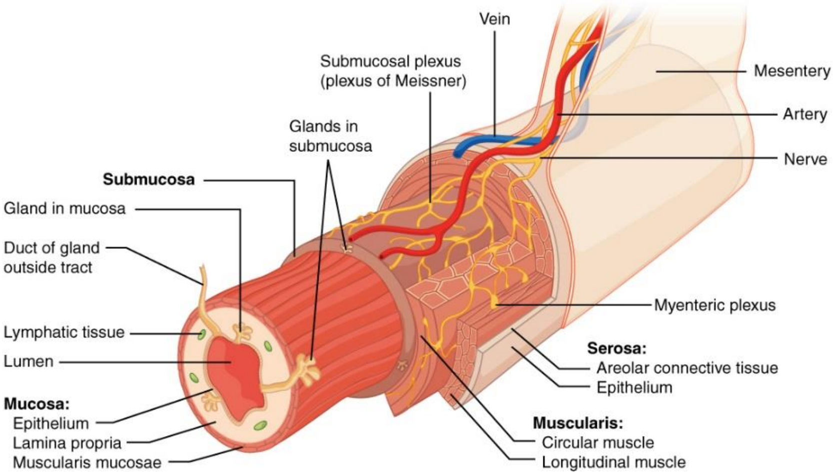


Figure A.1: General structure of the intestinal wall [29].

Appendix B - Theoretical Background

B.1 Spatial Descriptions: Positions, Orientations and Frames

A large part of robot kinematics is concerned with the establishment of various coordinate systems to represent the positions and orientations of rigid objects, and with transformations among these coordinate systems. Indeed, the geometry of three-dimensional space and of rigid motions plays a central role in all aspects of robotic manipulation. To define and manipulate mathematical quantities that represent position and orientation, we must define coordinate systems and develop conventions for representation. Many of the ideas presented here in the context of position and orientation will form a basis for our later consideration of linear and rotational velocities, forces, and torques [70] [71].

All positions and orientations will be described with respect to a universe coordinate system or with respect to other Cartesian coordinate systems that are (or could be) defined relative to the universe system.

Once a coordinate system is established, we can locate any point in the universe with a 3 × 1 position vector. Because we will often define many coordinate systems in addition to the universe coordinate system, vectors must be tagged with information identifying which coordinate system they are defined within; for example, the components of the vector P^A have numerical values that indicate distances along the axes of $\{A\}$. Each of these distances along an axis can be thought of as the result of projecting the vector onto the corresponding axis. Individual elements of a vector are given the subscripts *x*, *y*, and *z*:

$$P^A = \begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix}. \qquad (B.1)$$

In order to describe the orientation of a body, we will attach a coordinate system to the body and then give a description of this coordinate system relative to the reference system. In Fig. B.1, coordinate system $\{B\}$ has been attached to the body in a known way. A description of $\{B\}$ relative to $\{A\}$ now suffices to give the orientation of the body.

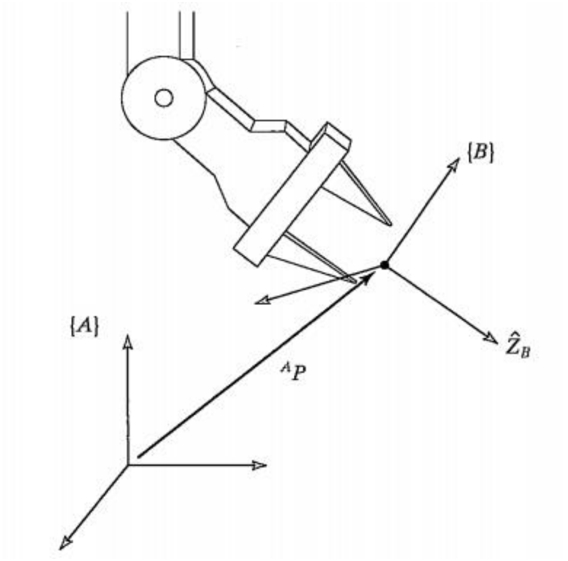


Figure B.1: Locating an object in position and orientation [71].

One way to describe the body-attached coordinate system, $\{B\}$, is to write the unit vectors of its three principal axes[4] in terms of the coordinate system $\{A\}$. We denote the unit vectors giving the principal directions of coordinate system $\{B\}$ as $\hat{x}_B$, $\hat{y}_B$ and $\hat{z}_B$. When written in terms of coordinate system $\{A\}$, they are called $\hat{x}_B^A$, $\hat{y}_B^A$ and $\hat{z}_B^A$.

It will be convenient if we stack these three unit vectors together as the columns of a 3 × 3 matrix - this matrix is called a rotation matrix. Because this particular rotation matrix describes $\{B\}$ relative to $\{A\}$, we name it with the notation R_B^A:

$$R_B^A = [\hat{x}_B^A \quad \hat{y}_B^A \quad \hat{z}_B^A] = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}. \qquad (B.2)$$

We can give expressions for the scalars r_{ij} in Eq. (B.2) by noting that the components of any vector are simply the projections of that vector onto the unit directions of its reference frame [71]. Hence, each component of R_B^A in (B.2) can be written as the dot product of a pair of unit vectors:

[4] It is often convenient to use three, although any two would suffice (the third can always be recovered by taking the cross product of the two given).

$$R_B^A = [\hat{x}_B^A \quad \hat{y}_B^A \quad \hat{z}_B^A] = \begin{bmatrix} \hat{x}_B \cdot \hat{x}_A & \hat{y}_B \cdot \hat{x}_A & \hat{z}_B \cdot \hat{x}_A \\ \hat{x}_B \cdot \hat{y}_A & \hat{y}_B \cdot \hat{y}_A & \hat{z}_B \cdot \hat{y}_A \\ \hat{x}_B \cdot \hat{z}_A & \hat{y}_B \cdot \hat{z}_A & \hat{z}_B \cdot \hat{z}_A \end{bmatrix}. \qquad (B.3)$$

The dot product of two unit vectors yields the cosine of the angle between them, so it is clear why the components of rotation matrices are often referred to as direction cosines. Since the inner product is commutative, (i.e. $\hat{x}_B \cdot \hat{y}_A = \hat{y}_B \cdot \hat{x}_A$), we see that

$$R_A^B = (R_B^A)^T. \qquad (B.4)$$

In a geometric sense, the orientation of $\{A\}$ with respect to the frame $\{B\}$ is the inverse of the orientation of $\{B\}$ with respect to the frame $\{A\}$. Algebraically, using the fact that coordinate axes are always mutually orthogonal, it can readily be seen that

$$(R_B^A)^T = (R_B^A)^{-1}. \qquad (B.5)$$

The column vectors of R_B^A are of unit length and mutually orthogonal. Such a matrix is said to be orthogonal. It can also be shown that $\det(R_B^A) = \pm 1$. In keeping with conventional practice, if we restrict ourselves to right-handed coordinate systems, then $\det(R_B^A) = +1$. It is customary to refer to the set of all such n × n matrices by the symbol *SO*(n), which denotes the Special Orthogonal group of order *n*. The properties of such matrices are summarized in Table B.1.

The basic rotation matrix of frame $o_1x_1y_1z_1$ relative to a coordinate frame $o_0x_0y_0z_0$ about the z-axis can be easily calculated using Eq. (B.3), where the positive sense for the angle θ is given by the right hand rule. From Fig. B.2 we see that

$$x_1 \cdot x_0 = \cos\theta, \qquad y_1 \cdot x_0 = -\sin\theta$$

$$x_1 \cdot y_0 = \sin\theta, \qquad y_1 \cdot y_0 = \cos\theta$$

and

$$z_1 \cdot z_0 = 1,$$

while all other dot products are zero.

Table B.1: Properties of the matrix group *SO*(n) [70].

• $R \in SO(n)$
• $R^{-1} \in SO(n)$
• $R^{-1} = R^T$
• The columns (and therefore the rows) of R are mutually orthogonal
• Each column (and therefore each row) of R is a unit vector
• $\det(R) = 1$

Thus, the rotation matrix R_1^0 has a particularly simple form in this case, namely

$$R_1^0 = R_{z,\theta} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}. \qquad (B.6)$$

Similarly the basic rotation matrices representing rotations about the x and y-axes are given as

$$R_{x,\theta} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix} \qquad (B.7)$$

$$R_{y,\theta} = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}. \qquad (B.8)$$

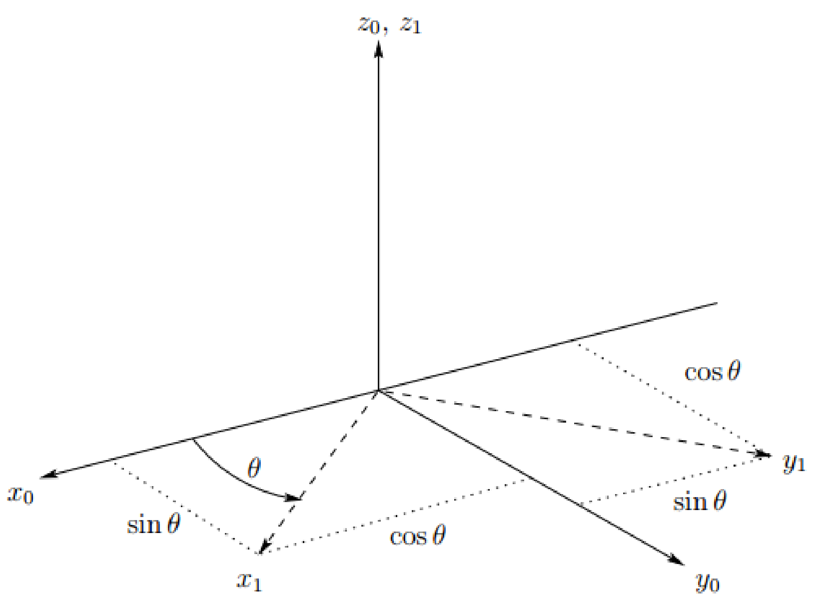


Figure B.2: Rotation about z_0 by an angle θ [70].

The rotation matrix R_1^0 can be used not only to represent the orientation of coordinate frame $o_1x_1y_1z_1$ with respect to frame $o_0x_0y_0z_0$, but also to transform the coordinates of a point from one frame to another. If a given point is expressed relative to $o_1x_1y_1z_1$ by coordinates P^1, then $R_1^0P^1$ represents the same point expressed relative to the frame $o_0x_0y_0z_0$. We can also use rotation matrices to represent rigid motions that correspond to pure rotation [70].

Often it is desired to perform a sequence of rotations, whether it is about a given fixed coordinate frame, or about successive current[5] frames. The order in which a sequence of rotations are carried out, and consequently the order in which the rotation matrices are multiplied together, is crucial since the composition law of rotational transformations is different in each case.

Given a fixed frame $o_0x_0y_0z_0$ and a current frame $o_1x_1y_1z_1$, together with rotation matrix R_1^0 relating them, if a third frame $o_2x_2y_2z_2$ is obtained by a rotation R performed relative to the current frame then R_1^0 should be multiplied by $R = R_2^1$, yielding the expression

$$R_2^0 = R_1^0R_2^1. \qquad (B.9)$$

However, if the second rotation is performed relative to the fixed frame then we represent the rotation by R in order to avoid confusion.
In this case the order in which the rotation matrices are multiplied is as follows:

$$R_2^0 = RR_1^0. \qquad (B.10)$$

In each case R_2^0 represents the transformation between the frames $o_0x_0y_0z_0$ and $o_2x_2y_2z_2$, although the frame $o_2x_2y_2z_2$ that results in Eq. (B.9) will be different from that resulting from Eq. (B.10) [70].

B.2 Parametrizations of Rotations

The nine elements r_{ij} in a general rotational transformation R are not independent quantities. Indeed a rigid body possesses at most three rotational degrees-of-freedom and thus at most three quantities are required to specify its orientation [70].

[5] The term ‘current frame’ is referred to the frame relative to which the rotation occurs.

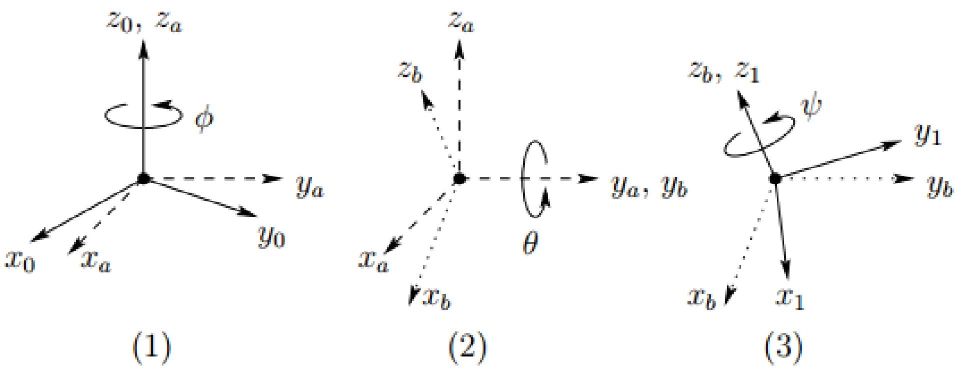


Figure B.3: Euler Angle representation. (1) Frame $o_a x_a y_a z_a$ represents the new coordinate frame after the rotation by ϕ, (2) frame $o_b x_b y_b z_b$ represents the new coordinate frame after the rotation by θ, and (3) frame $o_1 x_1 y_1 z_1$ represents the final frame, after the rotation by ψ [70].

A common method of specifying a rotation matrix in terms of three independent quantities is to use the Euler Angle representation. Consider the fixed coordinate frame $o_0 x_0 y_0 z_0$ and the rotated frame $o_1 x_1 y_1 z_1$ shown in Figure B.3. We can specify the orientation of the frame $o_1 x_1 y_1 z_1$ relative to the frame $o_0 x_0 y_0 z_0$ by three angles (ϕ, θ, ψ), known as Euler Angles, and obtained by three successive rotations as follows: First, a rotation about the z-axis by the angle ϕ, next, a rotation about the current y-axis by the angle θ, and finally a rotation about the current z-axis by the angle ψ. Frames $o_a x_a y_a z_a$ and $o_b x_b y_b z_b$ are shown in the figure to help visualize the rotations.

In terms of the basic rotation matrices the resulting rotational transformation R_1^0 can be generated as the product

$$R_{ZYZ} = R_{z,\phi} R_{y,\theta} R_{z,\psi} = \qquad (B.11)$$

$$= \begin{bmatrix} c_\phi & -s_\phi & 0 \\ s_\phi & c_\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c_\theta & 0 & s_\theta \\ 0 & 1 & 0 \\ -s_\theta & 0 & c_\theta \end{bmatrix} \begin{bmatrix} c_\psi & -s_\psi & 0 \\ s_\psi & c_\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} =$$

$$= \begin{bmatrix} c_\phi c_\theta c_\psi - s_\phi s_\psi & -c_\phi c_\theta s_\psi - s_\phi c_\psi & c_\phi s_\theta \\ s_\phi c_\theta c_\psi + c_\phi s_\psi & -s_\phi c_\theta s_\psi + c_\phi c_\psi & s_\phi s_\theta \\ -s_\theta c_\psi & s_\theta s_\psi & c_\theta \end{bmatrix}.$$

The matrix R_{ZYZ} in Eq. (B.11) is called the ZYZ-Euler Angle Transformation. The three angles, ϕ, θ, ψ, can be obtained for a given rotation matrix $R \in SO(3)$ using trigonometry and the properties of rotation matrices, which were discussed in the previous section. This problem will be important later when we address the inverse kinematics problem for manipulators.

A rotation matrix R can also be described as a product of successive rotations about the principal coordinate axes x_0, y_0 and z_0 taken in a specific order. These rotations define the roll, pitch, and yaw angles, as shown in Figure B.4, which are also denoted by ϕ, θ, ψ.

We specify the order of rotation as x – y − z, in other words, first a yaw about x_0 through an angle ψ, then pitch about the y_0 by an angle θ, and finally roll about the z_0 by an angle ϕ. Since the successive rotations are relative to the fixed frame, the resulting transformation matrix is given by

$$R_{XYZ} = R_{z,\phi} R_{y,\theta} R_{x,\psi} = \tag{B.12}$$

$$= \begin{bmatrix} c_\phi & -s_\phi & 0 \\ s_\phi & c_\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c_\theta & 0 & s_\theta \\ 0 & 1 & 0 \\ -s_\theta & 0 & c_\theta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & c_\psi & -s_\psi \\ 0 & s_\psi & c_\psi \end{bmatrix} =$$

$$= \begin{bmatrix} c_\phi c_\theta & -s_\phi c_\psi + c_\phi s_\theta s_\psi & s_\phi s_\psi + c_\phi s_\theta c_\psi \\ s_\phi c_\theta & -c_\phi c_\psi + s_\phi s_\theta s_\psi & -c_\phi s_\psi + s_\phi s_\theta c_\psi \\ -s_\theta & c_\theta s_\psi & c_\theta c_\psi \end{bmatrix}.$$

Of course, instead of yaw-pitch-roll relative to the fixed frames we could also interpret the above transformation as roll-pitch-yaw, in that order, each taken with respect to the current frame. The end result is the same matrix as in Eq. (B.12).

Another way in which an arbitrary rotation can be represented using only three independent quantities is the axis/angle representation, which can be found in [70], section 2.5.3.

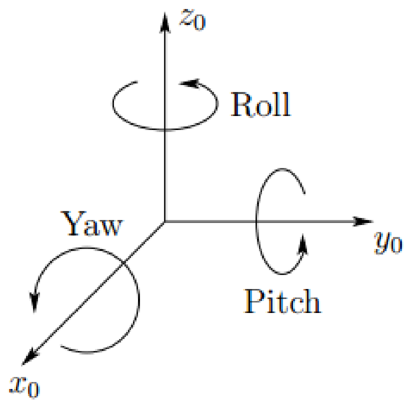


Figure B.4: Roll-pitch-yaw angle representation [70].

B.3 Kinematic Chains

As described in the beginning of chapter 2, a robot manipulator is composed of a set of links connected together by joints. The term ‘lower pair’ is used to describe the connection between a pair of bodies (or links) when the relative motion is characterized by two surfaces sliding over one another. Figure B.5 below shows the six possible lower pair joints.

Mechanical-design considerations favor manipulators' generally being constructed from joints that exhibit just one degree-of-freedom, so most manipulators have either revolute or prismatic joints. In the rare case that a mechanism is built with a joint having n degrees-of-freedom, it can be modeled as a succession of single degree-of-freedom joints with links of length zero in between. Therefore, without loss of generality, we will consider only manipulators that have joints with a single degree-of-freedom [71].

A robot manipulator with n joints will have $n + 1$ links, since each joint connects two links. We number the joints from 1 to n, and we number the links from 0 to n, starting from the base. By this convention, joint i connects link $i - 1$ to link i. We will consider the location of joint i to be fixed with respect to link $i - 1$. When joint i is actuated, link i moves. Therefore, link 0 (the first link) is fixed, and does not move when the joints are actuated.

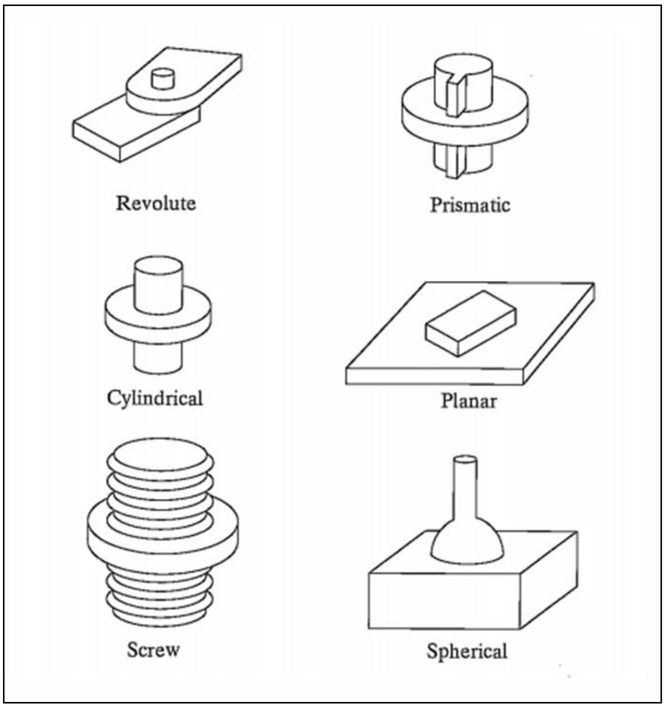


Figure B.5: The six possible lower-pair joints [71].

With the i-th joint, we associate a joint variable, denoted by q_i. In the case of a revolute joint, q_i is the angle of rotation, and in the case of a prismatic joint, q_i is the joint displacement:

$$q_i = \begin{cases} \theta_i & \text{if joint } i \text{ is revolute} \\ d_i & \text{if joint } i \text{ is prismatic} \end{cases} \qquad (B.13)$$

To perform the kinematic analysis, we attach a coordinate frame rigidly to each link. In particular, frame $o_i x_i y_i z_i$ is attached to link i. This means that whatever motion the robot executes, the coordinates of each point on link i are constant when expressed in the i-th coordinate frame. Furthermore, when joint i is actuated, link i and its attached frame experience a resulting motion. The frame $o_0 x_0 y_0 z_0$, which is attached to the robot base, is referred to as the inertial frame. Figure B.6 illustrates the idea of attaching frames rigidly to links in the case of an elbow manipulator [70].

Suppose A_i is the homogeneous transformation matrix that expresses the position and orientation of $o_i x_i y_i z_i$ with respect to $o_{i-1} x_{i-1} y_{i-1} z_{i-1}$. The matrix A_i is not constant, but varies as the configuration of the robot is changed. However, the assumption that all joints are either revolute or prismatic means that A_i is a function of only a single joint variable, namely q_i. In other words,

$$A_i = A_i(q_i). \qquad (B.14)$$

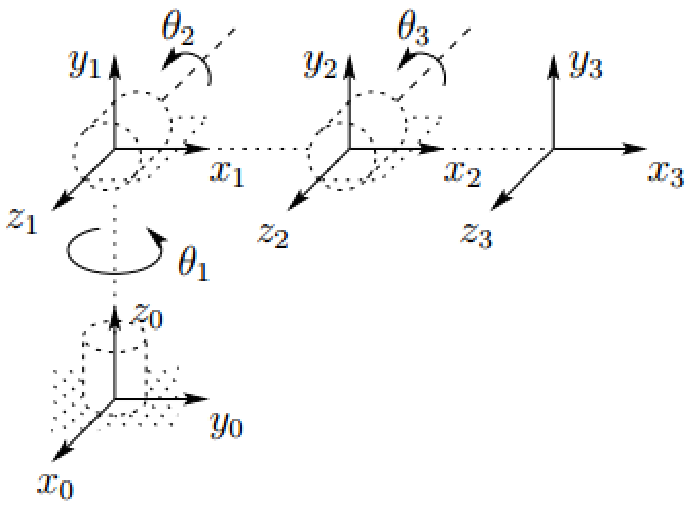


Figure B.6: Coordinate frames attached to elbow manipulator [70].

The homogeneous transformation matrix that expresses the position and orientation of $o_j x_j y_j z_j$ with respect to $o_i x_i y_i z_i$ is called, by convention, a transformation matrix, and is denoted by T_j^i. From section 2.3 we see that

$$T_j^i = \begin{cases} A_{i+1} A_{i+2} \cdots A_{j-1} A_j & if\ i < j \\ I & if\ i = j \\ \left(T_i^j\right)^{-1} & if\ i > j \end{cases} \qquad (B.15)$$

By the manner in which we have rigidly attached the various frames to the corresponding links, it follows that the position of any point on the end-effector, when expressed in frame *n*, is a constant independent of the configuration of the robot [70]. Denote the position and orientation of the end-effector with respect to the inertial or base frame by a three-dimensional vector o_n^0 (which gives the coordinates of the origin of the end-effector frame with respect to the base frame) and the 3 × 3 rotation matrix R_n^0, and define the homogeneous transformation matrix:

$$H = \begin{bmatrix} R_n^0 & o_n^0 \\ 0 & 1 \end{bmatrix}. \qquad (B.16)$$

Then the position and orientation of the end-effector in the inertial frame are given by

$$H = T_n^0 = A_1(q_1) \cdots A_n(q_n) \qquad (B.17)$$

where each homogeneous transformation A_i is of the form

$$A_i = \begin{bmatrix} R_i^{i-1} & o_i^{i-1} \\ 0 & 1 \end{bmatrix}. \qquad (B.18)$$

Hence

$$T_j^i = A_{i+1} \cdots A_j = \begin{bmatrix} R_j^i & o_j^i \\ 0 & 1 \end{bmatrix} \qquad (B.19)$$

where the matrix R_j^i expresses the orientation of $o_j x_j y_j z_j$ relative to $o_i x_i y_i z_i$ and is given by the rotational parts of the A-matrices as

$$R_j^i = R_{i+1}^i \cdots R_j^{j-1}. \qquad (B.20)$$

The coordinate vectors o_j^i are given recursively by the formula

$$o_j^i = o_{j-1}^i + R_{j-1}^i o_j^{j-1} \qquad (B.21)$$

where o_j^{j-1} is the vector form o_{j-1} to o_j in o_{j-1} coordinate system.

B.4 Forward Kinematic: Analysis of a Two-Link Planar Manipulator

Suppose we wish to move the manipulator from its home position to position *A*, from which point the robot is to follow the contour of the surface *S* to the point *B*, at constant velocity, while maintaining a prescribed force *F* normal to the surface (see Figure B.7(a)). In doing so the robot will cut or grind the surface according to a predetermined specification. For this reason we first need to describe both the position of the tool and the locations *A* and *B* (and most likely the entire surface *S*) with respect to a common coordinate system.

Typically, the manipulator will be able to sense its own position in some manner using internal sensors (position encoders located at joints 1 and 2) that can measure directly the joint angles θ_1 and θ_2. Therefore, we also need to express the positions *A* and *B* in terms of these joint angles.

Here, we establish the base coordinate frame $o_0x_0y_0z_0$ at the base of the robot (see Figure B.7(b)), and the coordinates (x, y) of the tool are expressed in this coordinate frame as

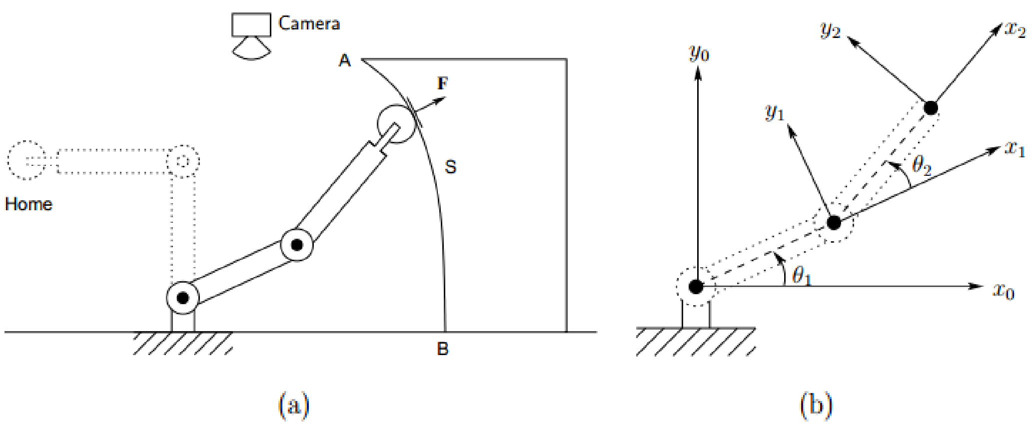


Figure B.7: Industrial two-link planar robot manipulator. (a) The manipulator is shown with a grinding tool that it must use to remove a certain amount of metal from a surface. (b) Coordinate frames attached to the manipulator. The z-axes all point out of the page and are not shown. [70].

$$x = x_2 = l_1 \cos\theta_1 + l_2 \cos(\theta_1 + \theta_2) \qquad (B.22)$$

$$y = y_2 = l_1 \sin\theta_1 + l_2 \sin(\theta_1 + \theta_2) \qquad (B.23)$$

in which l_1 and l_2 are the lengths of the two links, respectively. Also the orientation of the tool frame relative to the base frame is given by the direction cosines of the x_2 and y_2 axes relative to the x_0 and y_0 axes, that is,

$$x_2 \cdot x_0 = \cos(\theta_1 + \theta_2)\,; \qquad y_2 \cdot x_0 = -\sin(\theta_1 + \theta_2)$$

$$x_2 \cdot y_0 = \sin(\theta_1 + \theta_2)\,; \qquad y_2 \cdot y_0 = \cos(\theta_1 + \theta_2)$$

which can be combined into an orientation matrix:

$$\begin{bmatrix} x_2 \cdot x_0 & y_2 \cdot x_0 \\ x_2 \cdot y_0 & y_2 \cdot y_0 \end{bmatrix} = \begin{bmatrix} \cos(\theta_1 + \theta_2) & -\sin(\theta_1 + \theta_2) \\ \sin(\theta_1 + \theta_2) & \cos(\theta_1 + \theta_2) \end{bmatrix}. \qquad (B.24)$$

Equations (B.22)-(B.24) are called the forward kinematic equations for this arm, and can be derived quite easily due to its simple geometric structure. However, the kinematic analysis of an *n*-link manipulator with six degrees-of-freedom (or more) can be extremely complex and the conventions introduced below simplify the analysis considerably. Moreover, they give rise to a universal language with which robot engineers can communicate.

B.5 Inverse Kinematics: Kinematic Decoupling

Although the general problem of inverse kinematics is quite difficult, it turns out that for manipulators having six joints, with the last three joints intersecting at a point (such as the Stanford Manipulator), it is possible to decouple the inverse kinematics problem into two simpler problems, known respectively as inverse position kinematics, and inverse orientation kinematics. To put it another way, for a six-DOF manipulator with a spherical wrist, the inverse kinematics problem may be separated into two simpler problems, namely first finding the position of the intersection of the wrist axes, hereafter called the wrist center, and then finding the orientation of the wrist [70].

Under these conditions, Eq. (2.9) can be expressed as two sets of equations representing the rotational and positional equations:

$$R_6^0(q_1, \dots, q_6) = R \qquad (B.25)$$

$$o_6^0(q_1, \dots, q_6) = o \qquad (B.26)$$

where o and R are the desired position and orientation of the tool frame, expressed with respect to the world coordinate system. Thus, given o and R, the inverse kinematics problem is to solve for $q_1, \dots, q_6$.

The assumption of a spherical wrist means that the axes z_3, z_4 and z_5 intersect at one point o_c, and hence the origins o_4 and o_5 assigned by the DH convention will always be at the wrist center o_c, as shown in Figure B.8 (often o_3 will also be at o_c, but this is not necessary for the subsequent development). The important point of this assumption is that motion of the final three links about these axes will not change the position of o_c, and thus, the position of the wrist center is a function of only the first three joint variables.

The origin of the tool frame (whose desired coordinates are given by o) is simply obtained by a translation of distance d_6 along z_5 from o_c (see Table B.2 and Fig. B.9). In this case, z_5 and z_6 are the same axis, and the third column of R expresses the direction of z_6 with respect to the base frame.

Table B.2: DH parameters for spherical wrist (* represents a variable) [70].

Link	a_i	α_i	d_i	θ_i
4	0	-90	0	θ_4^*
5	0	90	0	θ_5^*
6	0	0	d_6	θ_6^*

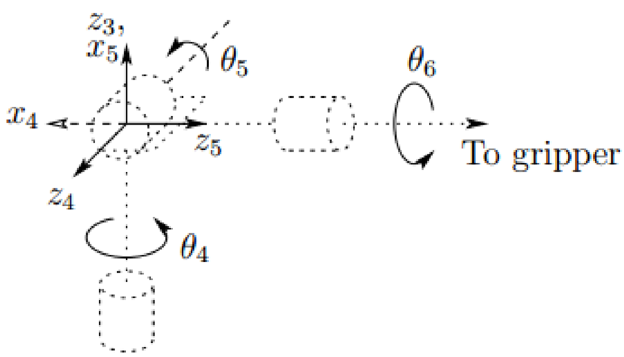


Figure B.8: The spherical wrist frame assignment in which the joint axes z_3, z_4 and z_5 intersect at o_c [70].

Therefore, we have

$$o = o_c^0 + d_6 R \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}. \qquad (B.27)$$

If the components of the end-effector position o are denoted o_x, o_y, o_z and the components of the wrist center o_c^0 are denoted x_c, y_c, z_c, then from Eq. (B.27) we can extract the following relationship:

$$\begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = \begin{bmatrix} o_x - d_6 r_{13} \\ o_y - d_6 r_{23} \\ o_z - d_6 r_{33} \end{bmatrix}. \qquad (B.28)$$

Using Eq. (B.28) we can find the values of the first three joint variables, which determines the orientation transformation R_3^0 since it depends only on these first three joint variables. We can now determine the orientation of the end-effector relative to the frame $o_3 x_3 y_3 z_3$ from the expression

$$R = R_3^0 R_6^3 \qquad (B.29)$$

$$R_6^3 = (R_3^0)^{-1} R = (R_3^0)^T R. \qquad (B.30)$$

The final three joint angles can then be found as a set of Euler angles corresponding to R_6^3.

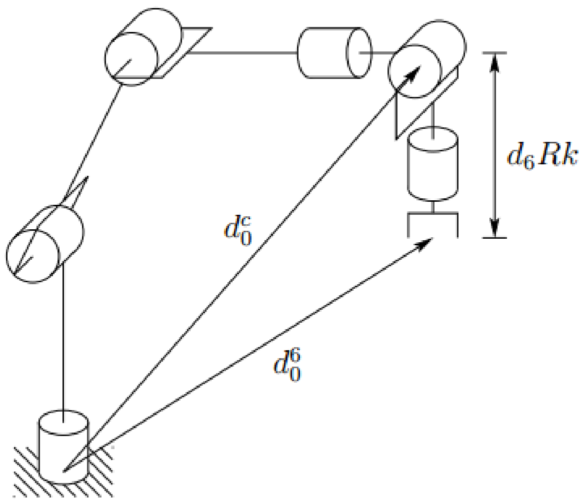


Figure B.9: Kinematic decoupling of a manipulator with a spherical wrist [70].

B.6 Inverse Position: A Geometric Approach

For the common kinematic arrangements that we consider, we can use a geometric approach to find the variables q_1, q_2, q_3 corresponding to o_c^0 given by Eq. (B.28). In general, the complexity of the inverse kinematics problem increases with the number of nonzero link parameters, based on the DH convention. For most manipulators, many of the a_i, d_i are zero, the α_i are 0 or ±π/2, etc. In these cases especially, a geometric approach is the simplest and most natural.

The general idea of the geometric approach is to solve for joint variable q_i by projecting the manipulator onto the $x_{i-1} - y_{i-1}$ plane and solving a simple trigonometry problem [70]. For example, consider the diagram of Figure B.10 for the two-link manipulator shown in Figure B.7. Using the Law of Cosines, angle θ_2 is given by

$$\cos\theta_2 = \frac{x^2 + y^2 - \alpha_1{}^2 - \alpha_2{}^2}{2\alpha_1\alpha_2} := D \qquad (B.31)$$

We could now determine θ_2 as

$$\theta_2 = \cos^{-1}(D). \qquad (B.32)$$

However, a better way to find θ_2 is to notice that if $\cos\theta_2$ is given by Eq. (B.31) then $\sin\theta_2$ is given as

$$\sin\theta_2 = \pm\sqrt{1 - D^2} \qquad (B.33)$$

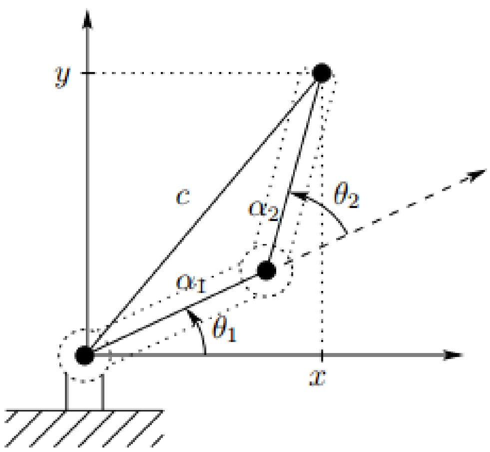


Figure B.10: Solving for the joint angles of a two-link planar arm [70].

and, hence, θ_2 can be found by

$$\theta_2 = \tan^{-1}\left(\frac{\pm\sqrt{1-D^2}}{D}\right). \qquad (B.34)$$

The advantage of this latter approach is that both the elbow-up and elbow-down solutions are recovered by choosing the positive and negative signs in Eq. (B.34), respectively.

Now it can be shown that θ_1 is given as

$$\theta_1 = \tan^{-1}\left(\frac{y}{x}\right) - \tan^{-1}\left(\frac{\alpha_2 \sin\theta_2}{\alpha_1 + \alpha_2 \cos\theta_2}\right). \qquad (B.35)$$

It is evident that the angle θ_1 depends on θ_2. This makes sense physically since we would expect to require a different value for θ_1, depending on which solution is chosen for θ_2.

A more comprehensive analysis of this approach can be found in [70], section 3.3.3, for the elbow manipulator shown in Figure 2.3.

B.7 Inverse Orientation

In the previous section we solved the inverse positon problem, thus attaining the values of the first three joint variables corresponding to a given position of the wrist origin. The inverse orientation problem is now one of finding the values of the final three joint variables corresponding to a given orientation with respect to the frame $o_3x_3y_3z_3$. For a spherical wrist, this can be interpreted as the problem of finding a set of Euler angles corresponding to a given rotation matrix R.

The DH parameters for the frame assignment shown in Figure 2.3 are summarized in Table B.3. Multiplying the corresponding A_i matrices presented in the form of Eq. (2.8) gives the transformation matrix T_3^0 for the articulated or elbow manipulator as

$$T_3^0 = A_1A_2A_3 = \begin{bmatrix} R_3^0 & o_3^0 \\ 0 & 1 \end{bmatrix} \qquad (B.36)$$

Table B.3: DH parameters for the elbow manipulator of Fig. 2.3 (* represents a variable) [70].

Link	a_i	α_i	d_i	θ_i
1	0	90	d_1	θ_1^*
2	a_2	0	0	θ_2^*
3	a_3	0	0	θ_3^*

where the rotational part R_3^0 is

$$R_3^0 = \begin{bmatrix} c_1 c_{23} & -c_1 s_{23} & s_1 \\ s_1 c_{23} & -s_1 s_{23} & -c_1 \\ s_{23} & c_{23} & 0 \end{bmatrix}. \qquad (B.37)$$

For the spherical wrist configuration shown in Figure B.8, the transformation matrix $T_6^3 = A_4 A_5 A_6$ is computed using the DH parameters shown in Table B.2. The rotational part of this matrix is given as

$$R_6^3 = \begin{bmatrix} c_4 c_5 c_6 - s_4 s_6 & -c_4 c_5 s_6 - s_4 c_6 & c_4 s_5 \\ s_4 c_5 c_6 + c_4 s_6 & -s_4 c_5 s_6 + c_4 c_6 & s_4 s_5 \\ -s_5 c_6 & s_5 s_6 & c_5 \end{bmatrix}. \qquad (B.38)$$

Equation (B.38) has the same form as the rotation matrix obtained for the Euler transformation, given in (B.11).

Hence, the final three joint variables θ_4, θ_5, θ_6 can indeed be identified as the Euler angles ϕ, θ and ψ (with respect to the coordinate frame $o_3 x_3 y_3 z_3$), and the Euler angle solution can be applied to Eq. (B.30). In this case, the three equations given by the third column in the above matrix equation are as follows:

$$c_4 s_5 = c_1 c_{23} r_{13} + s_1 c_{23} r_{23} + s_{23} r_{33} \qquad (B.39)$$

$$s_4 s_5 = -c_1 c_{23} r_{13} - s_1 s_{23} r_{23} + c_{23} r_{33} \qquad (B.40)$$

$$c_5 = s_1 r_{13} - c_1 r_{23}. \qquad (B.41)$$

If not both of the expressions (B.39), (B.40) are zero, meaning $s_5 \neq 0$, then θ_5 can be obtained from Eq. (B.41) as

$$\theta_5 = \mathrm{atan2}\left(s_1 r_{13} - c_1 r_{23}, \pm\sqrt{1 - (s_1 r_{13} - c_1 r_{23})^2}\right), \qquad (B.42)$$

where $\mathrm{atan2}(x, y)$ denotes the two-argument arctangent function, in which x and y are the cosine and sine, respectively, of the angle θ_5. This function uses the signs of x and y to select the appropriate quadrant for the angle θ_5.

If the positive square root is chosen in (B.42), then θ_4 is given by Eqns. (B.39) and (B.40) as

$$\begin{aligned}\theta_4 = \mathrm{atan2}(&c_1 c_{23} r_{13} + s_1 c_{23} r_{23} + s_{23} r_{33}\,, \\ &-c_1 c_{23} r_{13} - s_1 s_{23} r_{23} + c_{23} r_{33})\end{aligned} \qquad (B.43)$$

and θ_6 is given by the third row of Eq. (B.38):

$$\theta_6 = \mathrm{atan2}(-s_1 r_{11} + c_1 r_{21}, s_1 r_{12} - c_1 r_{22}). \qquad (B.44)$$

The other solutions are obtained analogously.

If $s_5 = 0$, then joint axes z_3 and z_5 are collinear. This is a singular configuration and only the sum $\theta_4 + \theta_6$ can be determined, as Eq. (B.38) becomes

$$R_6^3 = \begin{bmatrix} c_4 c_6 - s_4 s_6 & -c_4 s_6 - s_4 c_6 & 0 \\ s_4 c_6 + c_4 s_6 & -s_4 s_6 + c_4 c_6 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} c_{4+6} & -s_{4+6} & 0 \\ s_{4+6} & c_{4+6} & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad (B.45)$$

if $c_5 = +1$ (and $\theta_5 = 0$) , or

$$R_6^3 = \begin{bmatrix} -c_4 c_6 - s_4 s_6 & c_4 s_6 - s_4 c_6 & 0 \\ -s_4 c_6 + c_4 s_6 & s_4 s_6 + c_4 c_6 & 0 \\ 0 & 0 & -1 \end{bmatrix} = \begin{bmatrix} -c_{4-6} & -s_{4-6} & 0 \\ -s_{4-6} & c_{4-6} & 0 \\ 0 & 0 & -1 \end{bmatrix} \qquad (B.46)$$

if $c_5 = -1$ (and $\theta_5 = \pi$) .

One solution is to choose θ_4 arbitrarily and then determine θ_6 using the following expressions:

$$\theta_4 + \theta_6 = \mathrm{atan2}(r_{11}, r_{21}) = \mathrm{atan2}(r_{22}, -r_{12}) \qquad (B.47)$$

$$\theta_4 - \theta_6 = \mathrm{atan2}(-r_{11}, -r_{12}) = \mathrm{atan2}(r_{22}, -r_{21}). \qquad (B.48)$$

In each case, there are infinitely many solutions [70].

Appendix C - Prototype ‘C’

C.1 Mechanical Design: Transmission Ratio of Rotation Motor

To achieve a precision of 2 degrees in rotation, the following calculations were made: Since the upper (input) and lower (output) gears of the vertical transmission are identical originally, they have the same module

$$m_{input} = m_{output} = 2. \tag{C.1}$$

However, six teeth were cut from the upper gear (see Fig. C.1 below) in order to prevent the two gears from integrating as the mobile actuator advances along the links. Therefore, the transmission ratio between the two gears is calculated according to the ratio between the number of teeth, Z:

$$i_{vertical} = \frac{Z_{input}}{Z_{output}} = \frac{4}{10} = 2:5. \tag{C.2}$$

Because the output gear and the worm are fixated on the same axis (2 mm metal pin), they spin together. Hence, the transmission ratio between the input gear and the worm is

$$i_{input-worm} = i_{vertical} \cdot \underbrace{\frac{n_{output}}{n_{worm}}}_{=1} = 2:5. \tag{C.3}$$

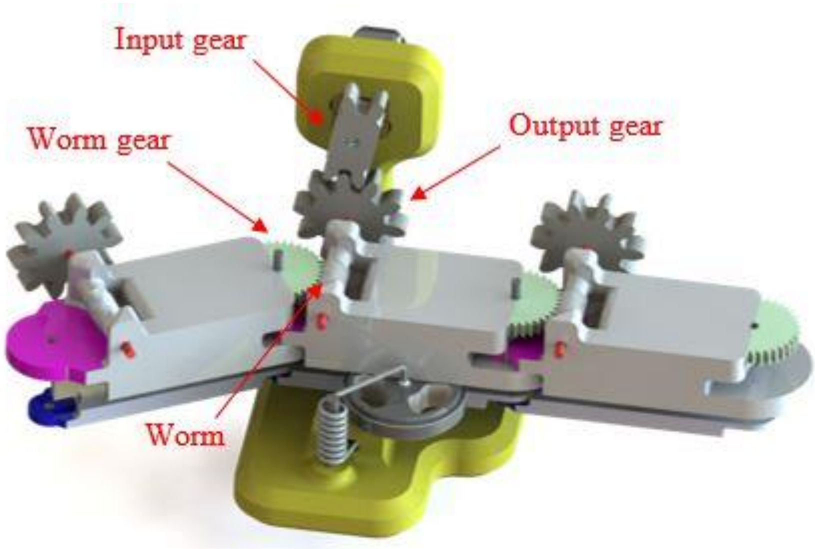


Figure C.1: The rotation mechanism consists of a vertical transmission activated by the motor (input and output gears) and a worm drive transmission (worm and worm gear) rotating each link.

Using a 0.5 module worm (as part of the locking mechanism) required us to design the worm gear with 36 teeth, in order to maintain the same module within the design constraints of the link. Therefore, the worm drive transmission ratio is

$$i_{worm\ drive} = \frac{Z_{worm}}{Z_{worm\ gear}} = 1 : 36. \qquad (C.4)$$

The transmission ratio of the entire mechanism can be calculated by multiplying Eq. (C.3) and (C.4):

$$i_{rotation} = i_{input-worm} \cdot i_{worm\ drive} = 1 : 90. \qquad (C.5)$$

From Eq. (C.5) we derive the relative angle of rotation for every spin of the motor:

$$\theta = i_{rotation} \cdot 2\pi = \frac{\pi}{45} = 4\ \text{deg}. \qquad (C.6)$$

As the input and output gears can be disconnected from each other twice over the course of one spin, we can in fact gain control over 2 degrees of rotation between two adjacent links.

C.2 The Control System

The control system described in sub-section 3.2.2 is comprised of the electrical circuit presented in Figure C.2. The components of the electrical circuit used in our experiments are shown below in the following order: The controller (Teensy 3.1), H-bridge, optical encoder and reflective IR sensor. The complete technical specifications of each component can be found online.

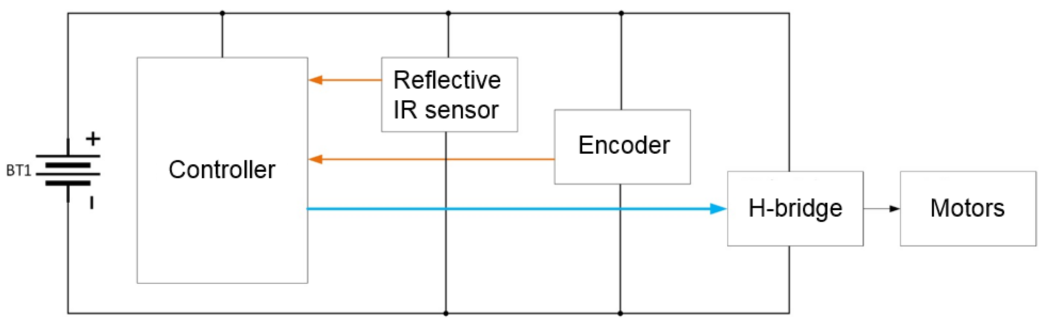


Figure C.2: A schematic diagram of the electrical circuit of the control system.

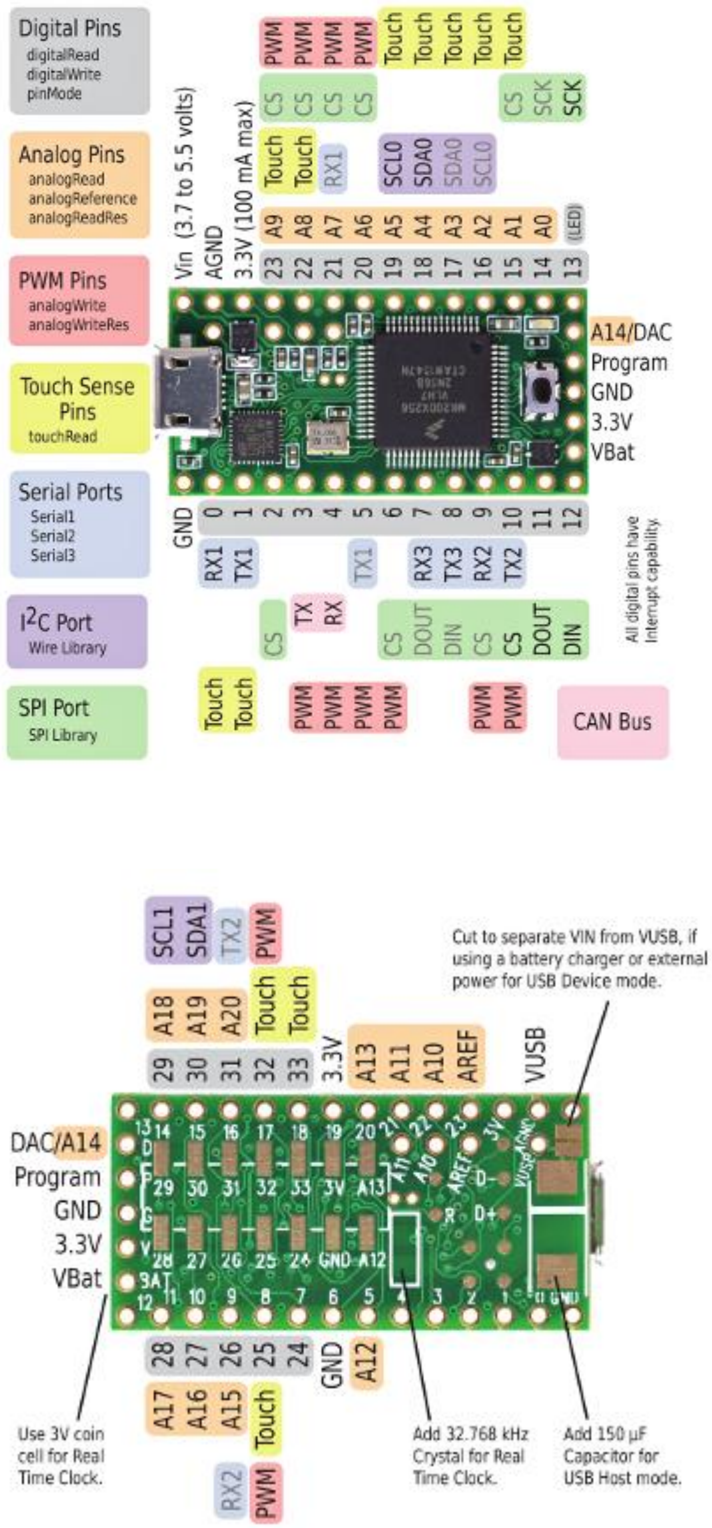


Figure C.3: Micro-processor Teensy 3.1 shown from the front (upper image) and back side (lower image) [74].

Features

- Dual-H-bridge motor driver: can drive two DC motors or one bipolar stepper motor
- Operating voltage: 2.7 V to 10.8 V
- Output current: 1.2 A continuous (2 A peak) per motor
- Motor outputs can be paralleled to deliver 2.4 A continuous (4 A peak) to a single motor
- Inputs are 3V- and 5V-compatible
- Under-voltage lockout and protection against over-current and over-temperature
- Reverse-voltage protection circuit
- Current limiting can be enabled by adding sense resistors (not included)

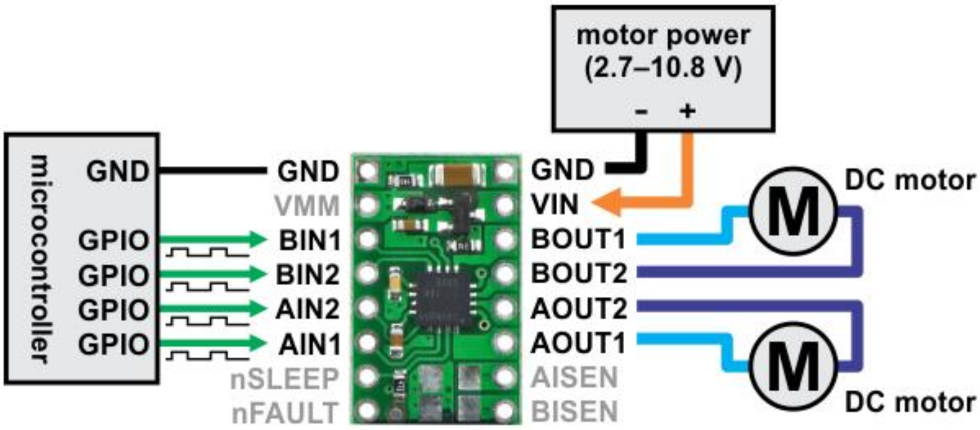


Figure C.4: Minimal wiring diagram for connecting a micro-controller to a DRV8833 dual-H-bridge motor driver carrier [75].

Dimensions

Size:	9.6 mm × 11.6 mm
Weight:	0.7 g[1]

General specifications

Voltage:	5 V
Average input current:	12 mA

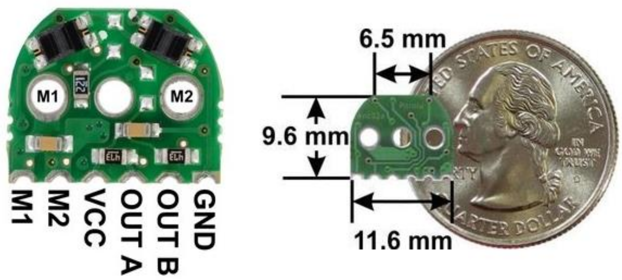
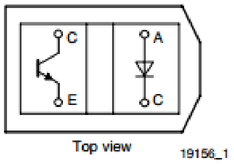


Figure C.5: Reflective optical encoder for micro metal gearmotors pinout (on the right), and from a bottom view with dimensions [76].

Figure C.6: Installed micro metal gearmotor reflective optical encoder with 5-tooth wheel, side view [76].



DESCRIPTION

The TCRT5000 and TCRT5000L are reflective sensors which include an infrared emitter and phototransistor in a leaded package which blocks visible light. The package includes two mounting clips. TCRT5000L is the long lead version.

FEATURES

- Package type: leaded
- Detector type: phototransistor
- Dimensions (L x W x H in mm): 10.2 x 5.8 x 7
- Peak operating distance: 2.5 mm
- Operating range within > 20 % relative collector current: 0.2 mm to 15 mm
- Typical output current under test: I_C = 1 mA
- Daylight blocking filter
- Emitter wavelength: 950 nm
- Lead (Pb)-free soldering released
- Compliant to RoHS directive 2002/95/EC and in accordance to WEEE 2002/96/EC

Pb-free
e2
RoHS
COMPLIANT

APPLICATIONS

- Position sensor for shaft encoder
- Detection of reflective material such as paper, IBM cards, magnetic tapes etc.
- Limit switch for mechanical motions in VCR
- General purpose - wherever the space is limited

Model	TCRT5000
Quantity	10
Color	Black + blue
Material	Plastic + iron
Specification	Detecting range: 0.591" (15mm); Detecting method: Reflective; Collector Emitter Voltage: 70V; Collector Current: 100mA; Forward Current: 60mA; Output type: Phototransistor; Working temperature: -25~85'C
Features	Compact construction, sense the presence of an object by using the reflective IR beam from the object
Application	Perfect for line tracking robot car DIY project
Packing List	1 x TCRT5000 module

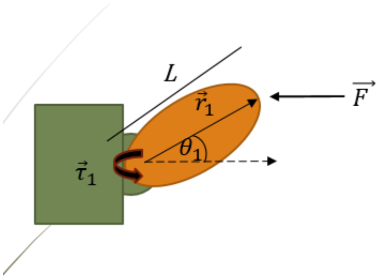
Dimensions: 0.79 in x 0.39 in x 0.20 in (2.0 cm x 1.0 cm x 0.5 cm)
Weight: 0.18 oz (5 g)

Figure C.7: TCRT5000 Reflective infrared sensor with photoelectric switches [77].

Appendix D - Strength Analysis

D.1 Structural Rigidity: Complete Analysis

MARS - Strength Analysis (one link)



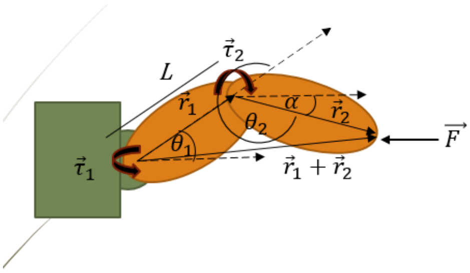
θ_{init} - initial link angle
θ_b - backlash angle
θ_t - torsion angle
K_t - stiffness coefficient

$$\vec{\tau}_1 = \vec{r}_1 \times \vec{F}\ ; \quad \tau = K_t \cdot \delta\theta_t\ ; \quad \delta\theta = \delta\theta_b + \delta\theta_t \quad \Rightarrow$$

$$\delta\theta_{t_1} = \frac{\tau_1}{K_t} = \frac{\vec{r}_1 \times \vec{F}}{K_t}\ ; \quad \delta\theta_{b_1} = sgn(\vec{\tau}_1) \cdot |\delta\theta_b|\ ; \quad \Rightarrow \ \delta\theta_1 = sgn(\vec{\tau}_1) \cdot |\delta\theta_b| + \frac{\vec{r}_1 \times \vec{F}}{K_t}$$

$$\theta = \theta_{init} + \delta\theta \quad \Rightarrow \quad \theta = \theta_{init_1} + sgn(\vec{\tau}_1) \cdot |\delta\theta_b| + \frac{\vec{r}_1 \times \vec{F}}{K_t}$$

MARS - Strength Analysis (two links)



θ_{init} - initial link angle
θ_b - backlash angle
θ_t - torsion angle
K_t - stiffness coefficient

$$\vec{\tau}_1 = (\vec{r}_1 + \vec{r}_2) \times \vec{F}\ ; \quad \vec{\tau}_2 = \vec{r}_2 \times \vec{F}\ ; \quad \tau = K_t \cdot \delta\theta_t\ ; \quad \delta\theta = \delta\theta_b + \delta\theta_t \quad \Rightarrow$$

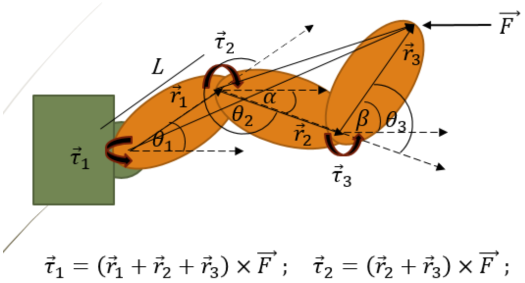
$$\delta\theta_1 = \delta\theta_{b_1} + \frac{(\vec{r}_1 + \vec{r}_2) \times \vec{F}}{K_t}\ ; \quad \delta\theta_{b_1} = sgn(\vec{\tau}_1) \cdot |\delta\theta_b|\ ;$$

$$\delta\theta_2 = \delta\theta_{b_2} + \frac{\vec{r}_2 \times \vec{F}}{K_t}\ ; \quad \delta\theta_{b_2} = sgn(\vec{\tau}_2) \cdot |\delta\theta_b|\ ; \quad \alpha = 2\pi - (\theta_1 + \theta_2)\ ;$$

$$\theta = \theta_{init} + \delta\theta \quad \Rightarrow$$

$$\theta = \theta_{init_1} + \theta_{init_2} + |\delta\theta_b| \cdot \big(sgn(\vec{\tau}_1) + sgn(\vec{\tau}_2)\big) + \frac{FL(\sin\theta_1 + \sin(\theta_1 + \theta_2))}{K_t} + \frac{FL\sin(\theta_1 + \theta_2)}{K_t}$$

MARS - Strength Analysis (three links)



θ_{init} - initial link angle
θ_b - backlash angle
θ_t - torsion angle
K_t - stiffness coefficient

$$\vec{\tau}_1 = (\vec{r}_1 + \vec{r}_2 + \vec{r}_3) \times \vec{F}\,; \quad \vec{\tau}_2 = (\vec{r}_2 + \vec{r}_3) \times \vec{F}\,; \quad \vec{\tau}_3 = \vec{r}_3 \times \vec{F}\,;$$

$$\tau = K_t \cdot \delta\theta_t\,; \quad \delta\theta = \delta\theta_b + \delta\theta_t \quad \Rightarrow$$

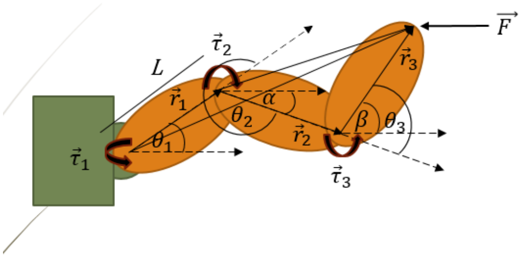
$$\delta\theta_1 = \delta\theta_{b_1} + \frac{(\vec{r}_1 + \vec{r}_2 + \vec{r}_3) \times \vec{F}}{K_t}\,; \quad \delta\theta_{b_1} = sgn(\vec{\tau}_1) \cdot |\delta\theta_b|\,;$$

$$\delta\theta_2 = \delta\theta_{b_2} + \frac{(\vec{r}_2 + \vec{r}_3) \times \vec{F}}{K_t}\,; \quad \delta\theta_{b_2} = sgn(\vec{\tau}_2) \cdot |\delta\theta_b|\,; \quad \alpha = 2\pi - (\theta_1 + \theta_2)\,;$$

$$\delta\theta_3 = \delta\theta_{b_3} + \frac{\vec{r}_3 \times \vec{F}}{K_t}\,; \quad \delta\theta_{b_3} = sgn(\vec{\tau}_3) \cdot |\delta\theta_b|\,; \quad \beta = \theta_3 - \alpha = \theta_3 - 2\pi + \theta_1 + \theta_2\,;$$

$$\theta = \theta_{init} + \delta\theta \quad \Rightarrow \quad \theta = \sum_{k=1}^{3} \theta_{init_k} + \sum_{k=1}^{3} |\delta\theta_b| \cdot sgn(\vec{\tau}_k) + \frac{(\vec{r}_1 + \vec{r}_2 + \vec{r}_3) \times \vec{F}}{K_t} + \frac{(\vec{r}_2 + \vec{r}_3) \times \vec{F}}{K_t} + \frac{\vec{r}_3 \times \vec{F}}{K_t}$$

MARS - Strength Analysis (N links)



θ_{init} - initial link angle
θ_b - backlash angle
θ_t - torsion angle
K_t - stiffness coefficient

In the event that the MARS consists of N links, the total angle displacement is:

$$\theta = \theta_{init} + \delta\theta\,; \quad \delta\theta = \delta\theta_b + \delta\theta_t \quad \Rightarrow$$

$$\theta = \sum_{k=1}^{N} \theta_{init_k} + \sum_{k=1}^{N} |\delta\theta_b| \cdot sgn(\vec{\tau}_k) + \frac{1}{K_t}\Big(N \cdot (\vec{r}_N \times \vec{F}) + (N-1) \cdot (\vec{r}_{N-1} \times \vec{F}) + \cdots + 1 \cdot (\vec{r}_1 \times \vec{F})\Big) \quad \Rightarrow$$

$$\theta = \sum_{k=1}^{N} \theta_{init_k} + |\delta\theta_b| \cdot \sum_{k=1}^{N} sgn(\vec{\tau}_k) + \frac{1}{K_t}\left(\sum_{k=1}^{N} k \cdot \vec{r}_k\right) \times \vec{F}$$

The stiffness coefficient K_t was estimated from the experiment conducted on two adjacent links of the latest design (prototype 'C'). One link was fixated using a metalworking vise (shown in Fig. D.1), while the other link was gradually loaded with different weights, ranging from 50 g to 1 kg. The experiment was conducted three times, until a small deformation occurred in the joint. The average weight which caused a deformation of 1 degree was about 4 kg.

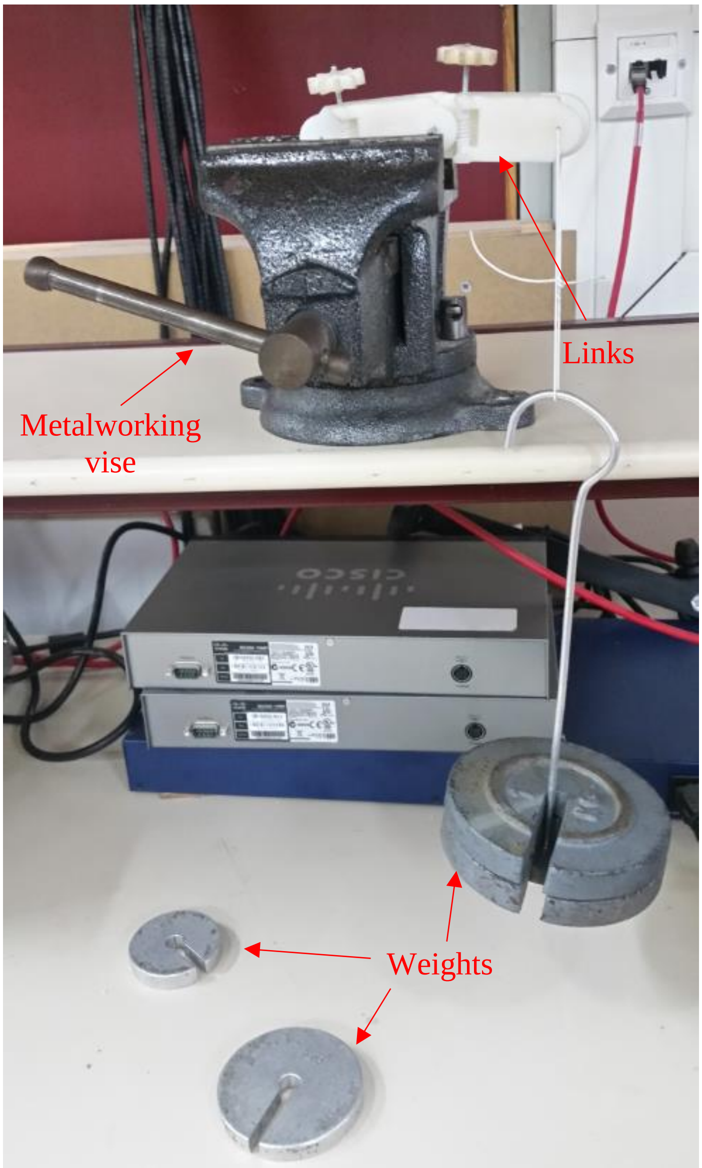


Figure D.1: The experiment conducted on two adjacent links, 3D printed using 'VeroWhite' material. One link is fixated using a metalworking vise, while the other link is gradually loaded with different weights.

As the force was applied at the free joint of the link, the torque which caused the deformation can be calculated from the following equation:

$$\vec{M} = \vec{r} \times \vec{F_g} \qquad (D.1)$$

where $\vec{r}$ is the range vector from the fixed joint to the point of application. In this case, $\vec{r}$ is the distance between the two joints of the link and is equal to the length of the link, L=5 cm.

$\vec{F}_g$ is the gravitational force, which according to the results of the experiment equals to

$$\vec{F}_g = m\vec{g} \tag{D.2}$$

$$\vec{F}_g = 4 \cdot 9.81 = 39.24\ N. \tag{D.3}$$

Hence, the maximum torque is

$$M = 0.05 \cdot 39.24 = 1.96 \cong 2\ Nm, \tag{D.4}$$

and the stiffness coefficient can be calculated from Eq. (4.7),

$$K_t = \frac{M}{\theta_t} = 2\ \frac{Nm}{deg}. \tag{D.5}$$

Converting the result given by Eq. (D.5) to the appropriate units (degrees to radians), the estimated stiffness coefficient of the joint is

$$K_t = 2 \cdot \frac{180}{\pi} \cong 115\ \frac{Nm}{rad}. \tag{D.6}$$

Appendix E - MATLAB codes

OptimizationUpdated

```
% This program finds the optimal path of the MARS to a specific (arbitrary) point in the 2D
% workspace according to the chosen cost functions.

close all;
clear; clc

N = 10;                                  % number of links
l = 5;                                   % length of each link [cm]
angle = [0 pi/6 pi/4 -pi/6 -pi/4]';      % angle between two links
thetaInit = datasample(angle,N);         % initial angle of each link
%thetaInit = [0 0 pi/4 -pi/6 -pi/4 0 pi/4 pi/4 0 pi/6]';
clr = cell(1,4);                         % path colors
clr{1} = 'g'; clr{2} = 'b'; clr{3} = 'c'; clr{4} = 'm';
xyFinal = [40 10];                       % target coordinates [cm]
error = (N*l)/100;                       % error deviation allowed (1% of MARS length)
[MARS_path,~,~] = plotPath(thetaInit,l,clr{1});
plot(xyFinal(1),xyFinal(2),'xr','linewidth',3,'markersize',15)

% check if target is in workspace
if sqrt(xyFinal(1)^2+xyFinal(2)^2) > N*l
   disp('Target is Not in Workspace')
   return
end

%% optimization process
lb = -pi/4*ones(N,1);                    % lower bound of theta
ub = pi/4*ones(N,1);                     % upper bound of theta
lambda = 1;
power = [2 0.5 1];                       % cost function f1
diff_angle = zeros(N,7);                 % pre-allocation
diff_angle(:,1) = round(rad2deg(thetaInit));

for i = 1:length(power)
   figure(i)
   [MARS_path,~,~] = plotPath(thetaInit,l,clr{1});
   plot(xyFinal(1),xyFinal(2),'xr','linewidth',3,'markersize',15)
   x0 = zeros(N,1);                      % initial guess for theta (1st iteration)
   fun = @(theta)costFun(theta,thetaInit,l,lambda,xyFinal,power(i));
   options = optimset('Display','iter');     % iterations of the solution
   %options = optimoptions('fmincon','Algorithm','trust-region-reflective','GradObj','on');
   [thetaFinal,fval] = fmincon(fun,x0,[],[],[],[],lb,ub,[],options);
   [MARS_path_opt,~,~] = plotPath(thetaFinal,l,clr{2});

   % plots end-effector trajectory
   xf = zeros(N+1,1); yf = zeros(N+1,1);
   temp = thetaInit;
   xf(1) = l*sum(cos(cumsum(temp)));
   yf(1) = l*sum(sin(cumsum(temp)));
   for k = 1:length(thetaFinal)
      temp(k) = thetaFinal(k);
      xf(k+1) = l*sum(cos(cumsum(temp)));
      yf(k+1) = l*sum(sin(cumsum(temp)));
```

```
  end
  trj = plot(xf,yf,'sk','linewidth',1,'markersize',8,'markerfacecolor','c');
  axis equal;  xlim([-5 45]);
  legend([MARS_path,MARS_path_opt,trj],'initial configuration','optimal configuration','end-
          effector trajectory'...
          ,'location','southwest');
  title(['Optimal Path of 10-links MARS, cost function: power = ',num2str(power(i))]
        ,'FontSize',12);
  xlabel('X [cm]');  ylabel('Y [cm]');

  % change in orientation of each link
  diff_angle(:,2*i) = round(rad2deg(thetaFinal));
  diff_angle(:,2*i+1) = round(rad2deg(thetaFinal-thetaInit));

  % check results
  if sqrt((l*sum(cos(cumsum(thetaFinal)))-xyFinal(1)).^2 +
        (l*sum(sin(cumsum(thetaFinal)))-xyFinal(2)).^2) > error
    disp(['End-effector position Exceeds acceptable Error, power = ',num2str(power(i))])
    x0 = thetaInit;
    figure(i)
    fun = @(theta)costFun(theta,thetaInit,l,lambda,xyFinal,power(i));
    options = optimset('Display','iter');          % iterations of the solution
    [thetaFinal,fval] = fmincon(fun,x0,[],[],[],[],lb,ub,[],options);
    [MARS_path_opt,~,~] = plotPath(thetaFinal,l,clr{3});
    legend([MARS_path,MARS_path_opt],'initial path','optimal path');
    title(['Optimal Path of 10-links MARS, cost function: power = ',num2str(power(i))],
          'FontSize',12);
    % change in orientation of each link
    diff_angle(:,2*i) = round(rad2deg(thetaFinal));
    diff_angle(:,2*i+1) = round(rad2deg(thetaFinal-thetaInit));
  end
  if sqrt((l*sum(cos(cumsum(thetaFinal)))-xyFinal(1)).^2 +
(l*sum(sin(cumsum(thetaFinal)))-xyFinal(2)).^2) > error
    title(['Target is Not in Workspace (power = ',num2str(power(i)),')'],'FontSize',12);
  end
end

%% choose optimal path for minimum time
t_trans = 2;              % time to pass one link [sec]
t_rot = 2.5;              % time to rotate one link 4 deg [sec]
% pre-allocation
time = zeros(1,3);
pos_rot = zeros(1,3);   % number of links to be rotated

for j = 1:length(power)
  pos_rot(j) = find(diff_angle(:,2*j+1),1,'last');
  counter = 0;
  t1 = 0;
  for k = 1:pos_rot(j)
    counter = counter+1;
    if diff_angle(k,2*j+1)      % computes rotation time
      t1 = t1+t_rot*abs(diff_angle(k,2*j+1)/4);
    end
  end
  t2 = t_trans*(counter-1);   % computes translation time
  time(j) = t1+t2;
end
[val,ind] = min(time);          % finds minimum time
```

```
disp(['Choose option ',num2str(ind),'(p=',num2str(power(ind)),'): Optimal path is achieved
within ',num2str(round(val)),...
   ' seconds (about ',num2str(val/60),' minutes)'])

%% adjusting results to current design with different control systems:
% case 1 - Automated control, via algorithm (control of 4 degrees in rotation)
% case 2 - Manual control, via joystick (control of 2 degrees in rotation)

% pre-allocation
adj_rot = zeros(N,3);
adj_ang = zeros(N,3);
thetaFinal_new = zeros(N,3);
tFinal_new = zeros(N,3);
Err = zeros(1,3);
diff_new = zeros(N,3);

operator = input('Enter 1 for Automated control or 2 for Manual control: ');
switch operator
  case 1
    for j = 1:length(power)
      temp = find(mod(diff_angle(:,2*j+1),4)==2);
      adj_rot(:,j) = round(diff_angle(:,2*j+1)/4);      % number of upper gear spins
      for i = 1:2:length(temp)
        adj_rot(temp(i),j) = adj_rot(temp(i),j)-1*sign(adj_rot(temp(i),j));
      end
      thetaFinal_new(:,j) = adj_rot(:,j)*4+diff_angle(:,1);
      joint = find(abs(thetaFinal_new(:,j)) > 45);
      for k = 1:length(joint)
        adj_rot(joint(k),j) = adj_rot(joint(k),j)-1*sign(adj_rot(joint(k),j));
        thetaFinal_new(joint(k),j) = adj_rot(joint(k),j)*4+diff_angle(joint(k),1);
      end
      % check error
      tFinal_new(:,j) = deg2rad(thetaFinal_new(:,j));
      Err(j) = sqrt((l*sum(cos(cumsum(tFinal_new(:,j))))-xyFinal(1)).^2 +
                    (l*sum(sin(cumsum(tFinal_new(:,j))))-xyFinal(2)).^2);
      if Err(j) > error
        disp(['Option',num2str(j),'(p=',num2str(power(j)),'): Error=',num2str(Err(j)),...
            ', End-effector position Exceeds acceptable Error(0.5cm)']);
      end
      figure(j+3)
      [MARS_path,~,~] = plotPath(thetaInit,l,clr{1});
      [path_new,~,~] = plotPath(tFinal_new(:,j),l,clr{4});
      plot(xyFinal(1),xyFinal(2),'xr','linewidth',3,'markersize',15)

      % plots end-effector trajectory
      xf = zeros(N+1,1);  yf = zeros(N+1,1);
      temp = thetaInit;
      xf(1) = l*sum(cos(cumsum(temp)));
      yf(1) = l*sum(sin(cumsum(temp)));
      for k = 1:length(thetaFinal)
        temp(k) = tFinal_new(k,j);
        xf(k+1) = l*sum(cos(cumsum(temp)));
        yf(k+1) = l*sum(sin(cumsum(temp)));
      end
      trj = plot(xf,yf,'sk','linewidth',1,'markersize',8,'markerfacecolor','c');
      axis equal;  xlim([-5 45]);
      legend([MARS_path,path_new,trj],'initial configuration','adjusted configuration','end-
            effector trajectory'...
            ,'location','southwest');
```

```
        xlabel('X [cm]');  ylabel('Y [cm]');
        title(['Adjusted Path, cost function: power = ',num2str(power(j)),', Error =
            ',num2str(Err(j))],'FontSize',12);
        diff_new(:,j) = thetaFinal_new(:,j)-diff_angle(:,1);
    end
case 2
    for j = 1:length(power)
        adj_ang(:,j) = diff_angle(:,2*j+1);
        odd = find(mod(adj_ang(:,j),2));       % indices of odd angle displacement values
        temp = zeros(1,length(odd));
        for k = 1:length(odd)
            temp(k) = adj_ang(odd(k),j);
        end
        val = median(temp);
        for k = 1:length(temp)
            if temp(k)<val
                adj_ang(odd(k),j) = adj_ang(odd(k),j)+1;
            else
                adj_ang(odd(k),j) = adj_ang(odd(k),j)-1;
            end
        end
        thetaFinal_new(:,j) = adj_ang(:,j)+diff_angle(:,1);
        joint = find(abs(thetaFinal_new(:,j)) > 45);
        for k = 1:length(joint)
            adj_ang(joint(k),j) = adj_ang(joint(k),j)-2*sign(adj_ang(joint(k),j));
            thetaFinal_new(joint(k),j) = adj_ang(joint(k),j)+diff_angle(joint(k),1);
        end
        % check error
        tFinal_new(:,j) = deg2rad(thetaFinal_new(:,j));
        Err(j) = sqrt((l*sum(cos(cumsum(tFinal_new(:,j))))-xyFinal(1)).^2 +
                      (l*sum(sin(cumsum(tFinal_new(:,j))))-xyFinal(2)).^2);
        if Err(j) > error
            disp(['Option',num2str(j),'(p=',num2str(power(j)),'): Error=',num2str(Err(j)),...
                ', End-effector position Exceeds acceptable Error(0.5cm)']);
        end
        figure(j+3)
        [MARS_path,~,~] = plotPath(thetaInit,l,clr{1});
        [path_new,~,~] = plotPath(tFinal_new(:,j),l,clr{4});
        plot(xyFinal(1),xyFinal(2),'xr','linewidth',3,'markersize',15)

       % plots end-effector trajectory
        xf = zeros(N+1,1);  yf = zeros(N+1,1);
        temp = thetaInit;
        xf(1) = l*sum(cos(cumsum(temp)));
        yf(1) = l*sum(sin(cumsum(temp)));
        for k = 1:length(thetaFinal)
            temp(k) = tFinal_new(k,j);
            xf(k+1) = l*sum(cos(cumsum(temp)));
            yf(k+1) = l*sum(sin(cumsum(temp)));
        end
        trj = plot(xf,yf,'sk','linewidth',1,'markersize',8,'markerfacecolor','c');
        axis equal;  xlim([-5 45]);
        legend([MARS_path,path_new,trj],'initial configuration','adjusted configuration','end-
                effector trajectory'...
                ,'location','southwest');
        xlabel('X [cm]');  ylabel('Y [cm]');
        title(['Adjusted Path, cost function: power = ',num2str(power(j)),', Error =
            ',num2str(Err(j))],'FontSize',12);
        diff_new(:,j) = thetaFinal_new(:,j)-diff_angle(:,1);
```

```
    end
end

%% choose new optimal path for minimum time
% pre-allocation
time = zeros(1,3);
pos_rot = zeros(1,3);          % number of links to be rotated

for j = 1:length(power)
  pos_rot(j) = find(diff_new(:,j),1,'last');
  counter = 0;
  t1 = 0;
  for k = 1:pos_rot(j)
    counter = counter+1;
    if diff_new(k,j)           % computes rotation time
      t1 = t1+t_rot*abs(diff_new(k,j)/4);
    end
  end
  t2 = t_trans*(counter-1);   % computes translation time
  time(j) = t1+t2;
end
%[val,ind] = min(time);       % finds minimum time

for i = 1:length(time)        % finds minimum time according to threshold
  if Err(i) > error
    time(i) = NaN;
  end
end
[val,ind] = min(time);
disp(['Choose option ',num2str(ind),'(p=',num2str(power(ind)),'): Optimal path is achieved
within ',num2str(round(val)),...
   ' seconds (about ',num2str(val/60),' minutes)'])
```

plotPath

```
function [data,x,y] = plotPath(theta,l,color)
% This function plots the MARS path (links configuration), given the length of each link (l)
% and the relative angles between the links (theta).

axis equal; grid on;
hold on;

w = 2.5;        % width of each link [cm]
rd = 1.25;      % radius of link's head [cm]
N = numel(theta);
abs_angle = cumsum(theta);

% pre-allocation
x = zeros(1,length(N+1));
y = zeros(1,length(N+1));
x(1) = 0;  y(1) = 0;
mat = zeros(4,4,N);

% DH transformation matrices
T_tot = 1;
for j = 2:N+1
  mat(:,:,j) = [cos(theta(j-1)) -sin(theta(j-1)) 0 l*cos(theta(j-1));
```

```
                    sin(theta(j-1)) cos(theta(j-1)) 0 l*sin(theta(j-1));
                    0 0 1 0;
                    0 0 0 1];
   T_tot = T_tot*mat(:,:,j);
   x(j) = T_tot(1,4);
   y(j) = T_tot(2,4);
   data = line(x,y,'linewidth',1,'color',color);       %  links configuration
end

% creating a chain of links - the MARS path
t1 = -pi/2:0.01:pi/2;
t2 = pi/2:0.01:3*pi/2;
x1 = (rd/sqrt(2))*cos(t1);
y1 = (rd/sqrt(2))*sin(t1);
base = patch([-l 0 x1 0 -l],[w rd y1 -rd -w],'r');   %  base link
x1 = 0;  y1 = 0;
x2 = 0;  y2 = 0;
ang1 = 0;  ang2 = 0;
for i = 1:N
   ang1 = t1+abs_angle(i);
   ang2 = t2+abs_angle(i);
   x1 = (rd/sqrt(2))*cos(ang1)+x(i);
   y1 = (rd/sqrt(2))*sin(ang1)+y(i);
   x2 = (rd/sqrt(2))*cos(ang2)+x(i);
   y2 = (rd/sqrt(2))*sin(ang2)+y(i);
   xdata = [x1+l*cos(abs_angle(i)) x2];
   ydata = [y1+l*sin(abs_angle(i)) y2];
   patch(xdata,ydata,color,'FaceAlpha',.5);
end
end
```

costFun

```
function [f] = costFun(theta,thetaInit,l,lambda,xyFinal,pow)
% This function describes the cost of the combined functions f1 & f2 designed to find the
% optimal path from the initial position to the target ('X').
% Input: thetaInit - initial relative angle; l - link length;  xyFinal - target destination
%            lambda - cost of f2 relative to f1;  theta - relative angle after optimization.

% pow=1 - proportional cost (absolute value)
% pow>1 - increased additional cost for larger difference (prabolic)
% pow<1 - decreased additional cost for larger difference (square root)

% f1 - minimizes the number of links that must be moved
f1 = sum((abs(theta - thetaInit)).^pow);
% f2 - maximizes the precision of the robot (all links) in order to reach the target
f2 = (l*sum(cos(cumsum(theta)))-xyFinal(1)).^2 + (l*sum(sin(cumsum(theta)))-
        xyFinal(2)).^2;

f = lambda*f1 + f2;
end
```

EndVelocity

```
% This program calculates the velocity of the end-effector, given random and constant angular
% velocities in the joints.
```

```
close all;
clear; clc

init_ang = zeros(1,10);                        % initial relative angle of displacement
N = numel(init_ang);                           % number of links
l=5;                                           % length of each link [cm]
ng = [1  2  6  7  9  10];                      % link to move
alpg = [pi/4 pi/4 -pi/4 -pi/4 -pi/4 pi/4];     % angle to change
abs_angle = cumsum(alpg);                      % absolute angle of displacement
velocity = [3.35 3.35 3.35 3.35 3.35 3.35;     % angular velocity of our motor - 32 RPM [rad/sec]
             1.5 1.5 3 3 4 4];                 % optional angular velocities [rad/sec]
%single_joint_velocity = datasample(velocity(2,:),length(ng));          % choose velocities
single_joint_velocity = velocity(1,:);         % choose case
end_velocity = zeros(length(ng),6);            % velocity of end-effector
MARS_path = zeros(7,length(init_ang));
MARS_path(1,:) = init_ang;
w = 2.5;                                       % width of mobile actuator [cm]

path_color = ['g' 'y' 'b' 'c' 'm' 'k' 'g'];
for k = 1:length(ng)
  joint_velocity = zeros(N,1);                         % pre-allocation
  joint_velocity(ng(k)) = single_joint_velocity(k);    % choose velocity
  jacob = Jacobian(l,init_ang);
  end_velocity(:,k) = jacob*joint_velocity;
  init_ang(ng(k)) = init_ang(ng(k))+alpg(k);
  MARS_path(k+1,:) = init_ang;
  figure(k)
  xlabel('X [cm]','FontSize',12);  ylabel('Y [cm]','FontSize',12);
  hold on
  p1 = plotPath(MARS_path(k,:),l,path_color(k));
  [p2,x,y] = plotPath(MARS_path(k+1,:),l,path_color(k+1));
  drawMotor([x(ng(k)) y(ng(k))],abs_angle(k),w);
  if k==1
    legend([p1,p2],'initial configuration',['link ' num2str(ng(k)) ' rotated '
            num2str(rad2deg(alpg(k))) '^\circ']);
  else
    legend([p1,p2],'current configuration',['link ' num2str(ng(k)) ' rotated '
            num2str(rad2deg(alpg(k))) '^\circ']);
  end
  axis([-5 55 -5 55]);
end
% calculating the absolute linear velocity of each link
abs_vel = sqrt(end_velocity(1,:).^2+end_velocity(2,:).^2);

% display results
figure(k+1)
y1 = end_velocity(1,:);
y2 = end_velocity(2,:);
hold on;  grid on
axis([0.999 6 -1.5 2])
Vx = plot(y1,'--b','linewidth',1.5);  plot(y1,'r*');
Vy = plot(y2,'--g','linewidth',1.5);  plot(y2,'r*');
Vtot = plot(abs_vel,'c','linewidth',1.5);  plot(abs_vel,'r*');
set(gca,'XTickLabel',{'1','','2','','6','','7','','9','','10'})
xlabel('Joint No.');  ylabel('End-Effector Velocity [m/sec]');
legend([Vx,Vy,Vtot],'V_x','V_y','V_t_o_t_a_l')

%% calculating velocity components of the end-effector with 1 degree increments
deg = pi/180;
```

```
inc = (pi/4)/deg+1;       % number of increments in each section
inc_end_vel = zeros(6,inc,length(ng));
% pre-allocation
v_x = zeros(1,length(ng)*inc);
v_y = zeros(1,length(ng)*inc);
v_x_init = ones(1,length(ng)*inc);
v_y_init = ones(1,length(ng)*inc);
init_ang = zeros(1,N);

for k = 1:length(alpg)
  joint_velocity = zeros(N,1);                          % pre-allocation
  joint_velocity(ng(k)) = single_joint_velocity(k);     % choose velocity
  for i = 1:inc
    jacob = Jacobian(l,init_ang);
    inc_end_vel(:,i,k) = jacob*joint_velocity;
    if i==inc
      j = inc*k-45;
      v_x(j:j+45) = inc_end_vel(1,:,k);
      v_x_init(j) = inc_end_vel(1,1,k);
      v_y(j:j+45) = inc_end_vel(2,:,k);
      v_y_init(j) = inc_end_vel(2,1,k);
      continue;
    else
      init_ang(ng(k)) = init_ang(ng(k))+deg*sign(alpg(k));
    end
  end
end
v_tot = sqrt(v_x.^2+v_y.^2);

% display results
figure(k+2)
hold on;  grid on
v_x_ind = find(v_x_init~=1);
v_y_ind = find(v_y_init~=1);
t_rot = 3;                 % time to rotate one link 45 deg [sec]
time_inc = t_rot/45;       % time increment - time to rotate one link 1 deg [sec]
t_deg = 0:time_inc:time_inc*(inc*k-1);
Vx = plot(t_deg,v_x,'--b','linewidth',1.5);     plot(t_deg(v_x_ind),v_x(v_x_ind),'*r');
Vy = plot(t_deg,v_y,'--g','linewidth',1.5);     plot(t_deg(v_y_ind),v_y(v_y_ind),'*r');
Vtot = plot(t_deg,v_tot,'c','linewidth',1.5);  plot(t_deg(v_y_ind),v_tot(v_y_ind),'*r');
xlabel('Time [sec]');  ylabel('End-Effector Velocity [m/sec]');
axis([0 18 -1.8 2]);
legend([Vx,Vy,Vtot],'V_x','V_y','V_t_o_t_a_l')
```

drawMotor

```
function drawMotor (center,ang,w)
% This function plots the mobile actuator given the location of the joint (center=[x y]),
% the absolute angle of rotation (ang) and the width of the actuator (w).

h = fill([(center(1)-w),(center(1)+w),(center(1)+w),(center(1)-w),(center(1)-w)],...
         [(center(2)-w),(center(2)-w),(center(2)+w),(center(2)+w),(center(2)-w)],'r',
         'FaceAlpha',0.3);
ang = rad2deg(ang);
rotate(h,[0 0 1],ang,[center 1])
end
```

StrengthAnalysis

```
% This program finds the new configuration of the MARS due to a small external force applied
% to the end-effector (small perturbations) and calculates the torques in the joints.

close all;
clear; clc

N = 10;                                 % number of links
l = 5;                                  % length of each link [cm]
w = 2.5;                                % width of each link [cm]
rd = 1.25;                              % radius of link's head [cm]
theta = [0 pi/6 pi/4 -pi/6 -pi/4];      % possible angles between 2 links
%init_ang = datasample(theta,N);
init_ang = [0 pi/4 0 -pi/4 0 pi/6 0 -pi/6 0 pi/6];   % initial relative angle of displacement
abs_angle = cumsum(init_ang);           % initial absolute angle of displacement
clr = cell(1,2);                        % path colors
clr{1} = 'g';  clr{2} = 'k';

figure
[MARS_config,x,y] = plotPath(init_ang,l,clr{1});    % initial configuration
xlabel('X [cm]', 'FontSize',12);  ylabel('Y [cm]', 'FontSize',12);
% plot force vector
fx = -5*cos(pi/4);  fy = -5*sin(pi/4);    % F=5N, for example
F_ext = [fx,fy,0];                         % external force applied to the end-effector
drawArrow = @(x,y) quiver(x(1),y(1),x(2)-x(1),y(2)-y(1),0,'LineWidth',2.1,
'MaxHeadSize',0.8,'color','b');
xf = [(x(N+1)-fx) x(N+1)];
yf = [(y(N+1)-fy) y(N+1)];
drawArrow(xf,yf);

%% kinematic analysis - angle displacement
theta_b = pi/180;                       % backlash angle - 1 degree
Kt = 115;                               % stiffness coefficient (from experiment with VeroWhite)
r = zeros(length(N),3);                 % pre-allocation
for i = 1:N
   r(i,:) = [x(i+1)-x(i),y(i+1)-y(i),0];        % range vector
end
% pre-allocation
M_ext = zeros(N,3);
theta_t = zeros(N,1);
theta_b_link = zeros(N,1);
d_theta = zeros(N,1);
new_theta = zeros(N,1);

sum_r = [zeros(1,3); cumsum(r)];
for j = 1:N
   M_ext(j,:) = cross(sum_r(N+1,:)-sum_r(j,:),F_ext);     % torque due to external force
   theta_t(j) = M_ext(j,3)/Kt;                            % torsion angle
   theta_b_link(j) = sign(M_ext(j,3))*theta_b;            % backlash angle
   d_theta(j) = theta_t(j)+theta_b_link(j);               % angle displacement of each link
   new_theta(j) = init_ang(j)+d_theta(j);                 % relative (new) angle of each link
end

[MARS_config_new,x_new,y_new] = plotPath(new_theta,l,clr{2});  % new configuration after
applying the force
% plot force vector
drawArrow = @(x,y) quiver(x(1),y(1),x(2)-x(1),y(2)-y(1),0,'LineWidth',2.1,
'MaxHeadSize',0.8,'color','b');
```

```
xf_new = [(x_new(N+1)-fx) x_new(N+1)];
yf_new = [(y_new(N+1)-fy) y_new(N+1)];
drawArrow(xf_new,yf_new);

for i=1:N
  plot(x(i),y(i),'.r','MarkerSize',16)
  plot(x_new(i),y_new(i),'.r','MarkerSize',16)
end
legend([MARS_config,MARS_config_new],'initial configuration','angle displacement');
title('Kinematic Analysis of 10-links MARS - Angle Displacement due to External Force (F=5N)','FontSize',12);
xlabel('X [cm]', 'FontSize',12);  ylabel('Y [cm]', 'FontSize',12);

jacob = Jacobian(l,init_ang);
Torque = jacob'*[F_ext 0 0 0]';
dist = sqrt((xf(2)-xf_new(2))^2+(yf(2)-yf_new(2))^2);    % distance the end-effector moved
```

Jacobian

```
function [jacob] = Jacobian(l,init_ang)
% This function creates the Jacobian matrix of the MARS given the length of the links (l) and
% the initial relative angles between the links (init_ang).

N = numel(init_ang);           % number of links
a=l/100*ones(N);               % length of each link [m]

% pre-allocation
mat = zeros(4,4,N);
x_link = zeros(N,1);
y_link = zeros(N,1);
z_link = zeros(N,1);
o_link = zeros(N,3);
jacob = zeros(6,N);

T_tot = 1;                     % initial value of transformation matrix
for j = 2:N+1
  mat(:,:,j) = [cos(init_ang(j-1)) -sin(init_ang(j-1)) 0 a(j-1)*cos(init_ang(j-1));
                sin(init_ang(j-1)) cos(init_ang(j-1)) 0 a(j-1)*sin(init_ang(j-1));
                0 0 1 0;
                0 0 0 1];
  T_tot = T_tot*mat(:,:,j);
  x_link(j) = T_tot(1,4);
  y_link(j) = T_tot(2,4);
  z_link(j) = T_tot(3,4);
  o_link(j,:) = [x_link(j) y_link(j) z_link(j)];
end

z_axis = [0 0 1];
for i = 1:N
  d = o_link(N+1,:)-o_link(i,:);
  jacob(1:3,i) = cross(z_axis,d);   % linear velocity component
  jacob(4:6,i) = [0 0 1];           % angular velocity component
end
end
```

References

[1] A. Degani, H. Choset, A. Wolf, M. A. Zenati, “Highly Articulated Robotic Probe for Minimally Invasive Surgery”, in *IEEE International Conference on Robotics and Automation*, 2006.

[2] "da Vinci® Surgery: Minimally Invasive Surgery", 11 2015. [Online]. http://www.davincisurgery.com/

[3] "Mazor Robotics", 11 2016. [Online]. www.mazorrobotics.com/

[4] P. Dario, P. Ciarletta, A. Menciassi, and B. Kim, “Modeling and experimental validation of the locomotion of endoscopic robots in the colon”, *International Journal of Robotics Research*, vol. 23, no. 4-5, pp. 549-556, 2004.

[5] I. Kassim, L. Phee, W. S. Ng, G. Feng, P. Dario and C.A. Mosse, “Locomotion techniques for robotic colonoscopy”, *IEEE Engineering in Medicine and Biology Magazine*, vol. 25, no. 3, pp. 49-56, 2006.

[6] L. Phee, A. Menciassi, S. Gorini, G. Pernorio, A. Arena, and P. Dario, “An innovative locomotion principle for minirobots moving in the gastrointestinal tract”, *Proceedings IEEE International Conference on Robotics and Automation*, vol. 2, pp. 1125-1130, 2002.

[7] L. Phee, D. Accoto, A. Menciassi, C. Stefanini, M. C. Carrozza and P. Dario, “Analysis and development of locomotion devices for the gastrointestinal tract”, *IEEE Transactions on Biomedical Engineering*, vol. 49, no. 6, pp. 613-616, 2002.

[8] A. S. Boxerbaum, K.M. Shaw, H. J. Chiel and R. D. Quinn, “Continuous wave peristaltic motion in a robot”, *International Journal of Robotics Research,* vol. 31, pp. 302-318, 2012.

[9] L. Phee, A. Menciassi, D. Accoto, C. Stefanini and P. Dario ,“Analysis of robotic locomotion devices for the gastrointestinal tract”, in *Robotics Research*, vol. 6, pp. 467-483, Springer, 2003.

[10] A. Menciassi ,S. Gorini ,G. Pernorio and P. Dario, “A SMA actuated artificial earthworm”, *Proceedings IEEE International Conference on Robotics and Automation*, pp. 3282-3287, 2004.

[11] B. Kim, S. Lee and J. Park, “Design and fabrication of a locomotive mechanism for capsule-type endoscopes using shape memory alloys (SMAs)”, *IEEE/ASME Transactions on Mechatronics*, vol. 10, no.1, pp. 77-86, 2005.

[12] B. Kim, H. Y. Lim and J. H. Park, “Inchworm like colonoscopic robot with hollow body and steering device”, *JSME International Journal, Series C,* vol. 49, no. 1, pp. 205-212, 2006.

[13] J. Lim, H. Park, J. An, Y. S. Hong, B. Kim and B. J. Yi, “One pneumatic line based inchworm-like micro robot for half-inch pipe inspection”, *Mechatronics*, vol. 18, pp. 315–322, 2008.

[14] J. Li, R. Sedaghati, J. Dargahi and D.Waechter, “Design and development of a new piezoelectric linear Inchworm actuator”, *Mechatronics,* vol. 15, pp. 651–681, 2005.

[15] K. Wang, G. Yan, P. Jiang and D. Ye, “A wireless robotic endoscope for gastrointestine”, *IEEE Transactions on Robotics,* vol. 24, no. 1, pp. 206-210, 2008.

[16] K. Wang, G. Yan, G. Ma and D. Ye, “An earthworm-like robotic endoscope system for human intestine: Design, analysis and experiment,” *Annals of Biomedical Engineering,* vol. 37, no. 1, pp. 210-221, 2009.

[17] N. Saga, T. Seto, H. Takanashi and N.Saito, “Development of a peristaltic crawling robot using planar link mechanisms,” *IEEJ Transactions on Electrical and Electronic Engineering,* vol. 3 no. 1, pp. 72-78, 2008.

[18] J. Dietrich, P. Meier, S. Oberthur, R. Preub, D. Voges, and K. Zimmermann, “Development of a peristaltically actuated device for the minimal invasive surgery with a haptic sensor array”, available online: http://www.docstoc.com/docs/50288015/development-of-a-peristatically-actuated-device-for-the-minimal

[19] D. Glozman, N. Hassidov, M. Senesh, and M. Shoham, “A self-propelled inflatable earthworm-like endoscope actuated by single supply line”, *IEEE Transactions on Biomedical Engineering*, vol. 57, no. 6, pp. 1264-1272, 2010.

[20] H. Howie, A. Wolf, M. A. Zenati, “Steerable, Follow the leader device”, patent application, WO/2006/083306 A3, 2008.

[21] D. Zarrouk, I. Sharf, and M. Shoham, “Analysis of worm-like robotic locomotion on compliant surfaces”, *IEEE Transactions On Biomedical Engineering,* vol. 58, no. 2, Feb. 2011, pp. 301-309.

[22] N. K. Baek, I. H. Sung and D. E. Kim, “Frictional resistance characteristics of a capsule inside the intestine for microendoscope design”, *Proceeding of the Institution of Mechanical Engineers*, vol. 218, Part H, Journal of Engineering in Medicine, pp. 193-201, 2004.

[23] Y. C. Fung, “Biomechanics, Mechanical Properties of Living Tissues”, Springer-Verlag New York, Inc. 1993.

[24] D. Zarrouk and M. Shoham, “Design and Analysis of a One DOF Multi Cell Worm-Like Robot”, ASME, Journal of Mechanical design, vol. 134 (2), 021010, Feb. 2012.

[25] D. Zarrouk, I. Sharf and M. Shoham, “Conditions for worm-robot locomotion in flexible environment: Theory and experiments”, *IEEE Transactions on Biomedical Engineering,* vol. 59, no. 4, pp. 1057-1067, 2012.

[26] G. Chirikjian and J. W. Burdick, “The Kinematics of Hyper-Redundant Robot Locomotion”, *IEEE Transactions On Robotics and Automation*, vol. 11, no. 6, 1995.

[27] R. Poulin III, "Thai Medical News", 22 April 2008. [Online]. http://www.thaimedicalnews.com/medical-tourism-thailand/what-is-endoscopy-gastrointestinal-digestion-disorder-exam/

[28] Hussainz, "Hussainz: Exploring New Hights", 1 January 2012. [Online]. http://pharmaworld.pk.cws3.my-hosting-panel.com/BodySystemDetail.asp?tId=63

[29] H. Gray, "Anatomy of the Human Body", New York: Philadelphia: Lea & Febiger, 1918.

[30] A. L. Faulx *et al.*, "Guidelines for privileging, credentialing, and proctoring to perform GI endoscopy", *Gastrointestinal Endoscopy,* vol. 85, no. 2, pp. 273-281, 2017.

[31] G. Long, A. Fritscher-Ravens, C. A. Mosse, T. Mills and P. Swain, "The Cath-Cam: a new concept in colonoscopy", *Gastrointestinal Endoscopy,* vol. 64, no. 6, pp. 997-1001, 2006.

[32] P. Joice, P. D. Ross, D. Wang, E. W. Abel and P. S. White, "Measurement of osteotomy force during endoscopic sinus surgery", *Allergy & Rhinology,* vol. 3, no. 2, pp. 61-65, 2012.

[33] M. N. Appleyard, C. A. Mosse, T. N. Mills, G. D. Bell, F. D. Castillo and C. P. Swain, "The measurement of forces exerted during colonoscopy", *Gastrointestinal Endoscopy,* vol. 52, no. 2, pp. 237-240, 2000.

[34] E. A. Miyasaka, M. Okawada, B. Utter, H. Mustafa-Maria, J. Luntz, D. Brei and D. H. Teitelbaum, "Application of Distractive Forces to the Small Intestine: Defining Safe Limits", *Journal of Surgical Research,* vol. 163, no. 2, pp. 169-175, 2010.

[35] B. R. Upchurch and J. J. Vargo, "Small bowel enteroscopy", *Reviews in Gastroenterological Disorders,* vol. 8, no. 3, pp. 169-177, 2008.

[36] B. Tharian, G. Caddy and T. C. K. Tham, "Enteroscopy in small bowel Crohn's disease: A review", *World Journal of Gastrointestinal Endoscopy,* vol. 5, no. 10, pp. 476-486, 2013.

[37] G. E. Tontini *et al.*, "Small-bowel capsule endoscopy with panoramic view: results of the first multicenter, observational study", *Gastrointestinal Endoscopy,* vol. 85, no. 2, pp. 401-408, 2017.

[38] D. Rolniak and D. G. Adler, "Complications of small bowel endoscopy", *Techniques in Gastrointestinal Endoscopy,* vol. 14, no. 2, p. 117–122, 2012.

[39] P. A. Akerman and D. Cantero, "Spiral Enteroscopy and Push Enteroscopy", *Gastrointestinal Endoscopy Clinics of North America*, vol. 19, no. 3, p. 357–369, 2009.

[40] B. R. Upchurch, M. R. Sanaka, A. Rocio Lopez and J. J. Vargo, "The clinical utility of single-balloon enteroscopy: a single-center experience of 172 procedures", *Gastrointestinal Endoscopy,* vol. 71, no. 7, p. 1218–1223, 2010.

[41] L.W. Tsai, “Robot analysis: The mechanics of serial and parallel manipulators”, *Engineering & Materials Science,* Wiley-Interscience, 1st edition, 1999.

[42] "KUKA Robotics Corporation". [Online]. http://www.kuka-robotics.com/en/

[43] "ABB Technologies Ltd.", 12 2016. [Online]. http://new.abb.com/products/robotics

[44] "KUKA Roboter GmbH", 2016. [Online]. http://www.kuka-robotics.com/en/products/industrial_robots/low/kr5_arc/start.htm

[45] "KUKA Roboter GmbH", 2016. [Online]. http://www.kuka-robotics.com/en/products/industrial_robots/heavy/kr-600-fortec

[46] J. Ostrowski and J. Burdick, “Gait kinematics for a Serpentine Robot”, *Proceedings of the International Conference on Robotics and Automation*, 1996.

[47] J. W. Burdick, J. Radford and G. S. Chirikjian, “Sidewinging Locomotion Gait for Hyper redundant robots”, *Adavanced robotics,* vol. 9, no. 3, pp 195-216, 1995.

[48] Z. Lu, S. Ma, B. Li and Y. Wang, “3D Locomotion of a Snake-Like robot Controlled by Cyclic Inhibitory CPG Model”, *Proceedings of the International Conference on Intelligent robots and Systems,* 2006.

[49] S. Hirose and E. F. Fukushima, “Snakes and Strings: New Robotics components for Rescue Operations”, *International Journal of Robotics Research,* vol. 23, pp. 341-349, 2004.

[50] A. Wolf, H. Choset, H. B. Brown and R. Casciola, “Design and Control of a Mobile Hyper-Redundant Urban Search and Rescue Robot”, *International Journal of Advanced Robotics,* vol. 19 (8), pp. 221-248, 2005.

[51] M. Mori and S. Hirose, “Locomotion of 3D Snake-Like robots – Shifting and rolling Control of Active Cord Mechanism ACM-R3”, *Journal of Robotics and Mechatronics,* vol. 18, no. 5, 2006.

[52] H. Kimura, K. Shimizu, and S. Hirose, “Development of Genbu: Active-Wheel Passive-Joint Snake-Like Mobile Robot”, *Journal of Robotics and Mechatronics,* vol. 16, no. 3, 2004.

[53] F. Enner, D. Rollinson and H. Choset, “Estimating Motion of Snake Robots in Straight Pipes”, *Proceedings of the IEEE International Conference on Robotics and Automation*, Karlsruhe, DE, May, 2013.

[54] F. Enner, D. Rollinson and H. Choset, “Simplified Motion Modeling for Snake Robots”, *Proceedings of the IEEE International Conference on Robotics and Automation*, St. Paul, MN, May, 2012.

[55] P. Liljeback, K. Pettersen, O. Stavdahl and J. Gravdahl, “A review on modelling, implementation, and control of snake robots”, *Robotics and Autonmous Systems*, vol. 60, pp. 29-40, 2012.

[56] P. Liljeback, K. Pettersen, O. Stavdahl and J. Gravdahl, “Snake Robots - Modelling, Mechatronics, and Control”, Springer, 2013.

[57] P. Singh and C. Krishna, “Continuum Arm Robotic Manipulator: A Review”, *Universal Journal of Mechanical Engineerings*, vol. 2, no. 6, pp. 193-198, 2014.

[58] H. Choset, K. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. Kavraki and S. Thrun, “Principles of Robot Motion - Theory, Algorithms, and Implementation”, MIT Press, 2005.

[59] S. Lavalle, “Planning Algorithms”, Cambridge University Press, 2006.

[60] G. Chirikjian, “Theory and Applications of Hyper-Redundant Robotic Manipulators”, 1992.

[61] G. Chirikjian and J. Burdick, “An obstacle avoidance algorithm for hyper-redundant manipulators”, Cincinnatti, OH, 1990.

[62] G. Chirikjian and J. Burdick, “Design and experiments with a 30 DOF robot”, Atlanta, GA, 1993.

[63] G. Chirikjian, “Hyper-Redundant Robot Mechanisms and Their Applications”, Osaka, Japan, 1991.

[64] M. D. Graca and J. T. Machado, “An evolutionary approach for the motion planning of redundant and hyper-redundant manipulators”, *Nonlinear Dynamics*, pp. 115-129, 2010.

[65] E. Xidias and N. Aspragathos, “Time Sub-Optimal Path Planning for Hyper Redundant Manipulators Amidst Narrow Passages in 3D Workspaces”, in

Advances on Theory and Practice of Robots and Manipulators, vol. 22, springer, 2014, pp. 445-452.

[66] A. Shukla, E. Singla, P. Wahi and B. Dasgupta, “A direct variational method for planning monotonically optimal paths for redundant manipulators in constrained workspaces”, *Robotics and Autonomous Systems,* pp. 209-220, 2013.

[67] N. Shvalb, B. B. Moshe and O. Medina, “A real-time motion planning algorithm for a hyper-redundant set of mechanisms”, *Robotica*, pp. 1327-1335, 2013.

[68] D. Trivedi, C. Rahn, W. Kier and I. Walker, “Soft robotics: Biological inspiration, state of the art, and future research”, *Applied Bionics and Biomechanics,* vol. 5, no. 3, pp. 99-117, 2008.

[69] I. Walker, “Continuous Backbone 'Continuum' Robot Manipulators”, *ISRN Robotics,* vol. 2013, 2013.

[70] M. W. Spong, S. Hutchinson and M. Vidyasagar, Robot Modeling and Control, 1st edition, New York: John Wiley & Sons, Inc., 2005.

[71] J. J.Craig, Introduction to Robotics: Mechanics and Control 3rd edition, New Jersey: Pearson Education, Inc., 2005.

[72] "4P", [Online]. https://www.4project.co.il/product/micro-metal-gearmotor-strong-32rpm. [Accessed 2015].

[73] "4P", [Online]. https://www.4project.co.il/product/micro-metal-gearmotor-strong-320rpm. [Accessed 2015].

[74] "PJRC", Electronic Projects Components Available Worldwide. [Online]. https://www.pjrc.com/teensy/teensy31.html. [Accessed 2016].

[75] "Pololu Robotics & Electronics", [Online]. https://www.pololu.com/product/2130. [Accessed 2016].

[76] "Pololu Robotics & Electronics", [Online]. https://www.pololu.com/product/2590/resources. [Accessed 2016].

[77] "DX, deal extream", [Online]. http://www.dx.com/p/tcrt5000-reflective-infrared-sensor-photoelectric-switches-10-pcs-163251#.WKsnwW996Ul. [Accessed 2016].

תקציר

אנדוסקופיה של מערכת העיכול הינו הליך רפואי הנמצא בשימוש נרחב במדינות רבות. עם זאת, הטכניקה הקונבנציונלית, המבוססת על החדרה ידנית של צינור פלסטי גמיש על ידי הרופא, בעלת שני חסרונות עיקריים: (1) תהליך התאוששות המטופל לאחר ההליך מלווה באי נוחות משמעותית, ו- (2) נגישותו למעי הדק היא מוגבלת מאוד. נגישות זו חשובה למדי עבור הליכים רפואיים רבים, ביניהם: ביופסיות, הרחבת הצירויות של כלי דם ושליטה בדימום הנוצר בזמן ביצוע ההליך ולאחריו (בזמן ההתאוששות). מרבית ניסיונות הגישה אל כלי דם כללו שימוש ברובוטים טוריים מרובי מפרקים, המכונים לעתים "נחשים". רובוטים אלו הם כמעט בלתי אפשריים לניווט לאורך המעי בשל אורכו, ועל כן נדרשים עשרות או מאות אקטואטורים להפעלתם, מה שהופך אותם מסורבלים מאוד לתפעול ומגבילים את היכולת למזעורם. יתר על כן, השימוש במספר רב של אקטואטורים עלול להניב מערכת גמישה למדי, אשר אינה מסוגלת להפעיל את הכוחות הנדרשים למעבר בתוך המעי או לביצוע הליכים פשוטים כגון ביופסיה.

בפרויקט זה אנו מציעים סוג חדש של רובוט טורי בעל הנעה מינימלית, המכונה "MARS" (Minimally Actuated Robotic Snake). רובוט זה מורכב ממספר חוליות קשיחות המחוברות זו לזו באמצעות מפרקים סיבוביים פסיביים (חסרי הנעה), ובעל אקטואטורים ניידים. החידוש של הרובוט טבוע בכך שהאקטואטור נע על גבי החוליות אל המפרק הרצוי ומסובב אותו ליצירת זווית יחסית בין אותן שתי חוליות סמוכות. המפרקים, ובהתאם לכך גם החוליות, משמרים את האוריינטציה שלהם בצורה פסיבית, עד אשר האקטואטור מניע אותם שוב. מנגנון הנעה זה ניתן ליישום על כל רובוט טורי בעל שתי חוליות או יותר. קונפיגורציה ייחודית זו מאפשרת לרובוט טווח תנועה רחב, בדומה לרובוטים בעלי יתירות גבוהה, אבל עם מעט אקטואטורים. הרובוט הוא מודולרי, שכן הגודל והגיאומטריה שלו ניתנים לשינוי בקלות, כמו גם כמות החוליות והאקטואטורים. מלבד האפליקציות הרפואיות, כמו ביצוע הליכים זעיר-פולשניים, רובוטים מסוג זה הם בעלי פוטנציאל גדול ליישום במגוון תחומים, למשל בתעשייה, בחקלאות, ובמשימות חיפוש והצלה.

התכן המכני של הרובוט (עבור שני אבי-הטיפוס העיקריים) והאנליזות הקינמטיות מתוארים בגוף העבודה, בנוסף לניסויים המדגימים את יכולות הרובוט בביצוע משימות בסביבה מרובת מכשולים תוך שימוש בסימולציות ובאלגוריתמים לתכנון תנועה המבוססים על אופטימיזציה. שני אבי הטיפוס בהם השתמשנו לביצוע הניסויים והאנליזות כוללים עשר חוליות ואקטואטור נייד אחד, אם כי כמות החוליות והאקטואטורים ניתנים לשינוי בהתאם למשימה של הרובוט, כפי שנאמר קודם לכן.

מילות מפתח: אנדוסקופיה של מערכת העיכול; רובוט טורי; הנעה מינימלית; אקטואטור נייד.

אוניברסיטת בן-גוריון בנגב

הפקולטה למדעי ההנדסה

המחלקה להנדסה ביו-רפואית

תכנון ואנליזה של רובוט טורי מינימליסטי לביצוע הליכים רפואיים

חיבור זה מהווה חלק מהדרישות לקבלת תואר מגיסטר בהנדסה

מאת : ליאור דמתי

מנחים : ד״ר דוד זרוק

פרופ׳ אמנון סינטוב

חתימת המחבר תאריך ....21/2/17.......

אישור המנחה תאריך ....21/2/17.......

אישור המנחה תאריך ....21/2/17.......

אישור יו״ר ועדת תואר שני מחלקתית תאריך ....23/2/17.......

שבט ה׳תשע״ז פברואר 2017

אוניברסיטת בן-גוריון בנגב
הפקולטה למדעי ההנדסה
המחלקה להנדסה ביו-רפואית

תכנון ואנליזה של רובוט טורי מינימליסטי לביצוע הליכים רפואיים

חיבור זה מהווה חלק מהדרישות לקבלת תואר מגיסטר בהנדסה

מאת

ליאור דמתי

שבט ה'תשע"ז

פברואר 2017