DESIGN OF A LOW-COST SWARM ROBOTIC SYSTEM FOR FLOCKING

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

ÇAĞRI ATA DEMİR

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
MECHANICAL ENGINEERING

NOVEMBER 2019

Approval of the thesis:

DESIGN OF A LOW-COST SWARM ROBOTIC SYSTEM FOR FLOCKING

submitted by **ÇAĞRI ATA DEMİR** in partial fulfillment of the requirements for the degree of **Master of Science in Mechanical Engineering Department, Middle East Technical University** by,

Prof. Dr. Halil Kalıpçılar
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. Sahir Arıkan
Head of Department, **Mechanical Engineering** _____

Assist. Prof. Dr. Ali Emre Turgut
Supervisor, **Mechanical Engineering, METU** _____

Assist. Prof. Dr. Kutluk Bilge Arıkan
Co-Supervisor, **Mechanical Engineering, TED University** _____

Examining Committee Members:

Assoc. Prof. Dr. Mehmet Bülent Özer
Mechanical Engineering, METU _____

Assist. Prof. Dr. Ali Emre Turgut
Mechanical Engineering, METU _____

Assist. Prof. Dr. Kutluk Bilge Arıkan
Mechanical Engineering, TED University _____

Assoc. Prof. Dr. Ulaş Yaman
Mechanical Engineering, METU _____

Assist. Prof. Dr. Selçuk Himmetoğlu
Mechanical Engineering, Hacettepe University _____

Date: 28.11.2019

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Surname: Çağrı Ata DEMİR

Signature:

ABSTRACT

DESIGN OF A LOW-COST SWARM ROBOTIC SYSTEM FOR FLOCKING

DEMİR, Çağrı Ata
Master of Science, Mechanical Engineering
Supervisor: Assist. Prof. Dr. Ali Emre Turgut
Co-Supervisor: Assist. Prof. Dr. Kutluk Bilge Arıkan

November 2019, 85 pages

Swarm robotics is an approach to the coordination of large numbers of robots. The main motivation of this thesis is to study a robotic system designed to do flocking both indoors and outdoors. A walking robot is designed parallel to this purpose. In the first part of thesis, a leg is designed to minimize the displacement of center of mass of robot in vertical axis to eliminate mechanical noise. Mechanism analysis and Matlab optimization tools are utilized in this process. Then, electronic components of robot are determined and mechanical design of robot, which is applicable to leg designed in previous stage and selected electronics, is done as first prototype in Solidworks platform. The controller of robot is written in Arduino platform. Parts are produced by using Ultimaker 2+ 3D printer and assembled afterwards. Assembled robot is run to test its obstacle avoidance and light detection skills.In the second part of thesis, the first prototype is improved and a smaller robot is designed mainly to decrease its production time. An encoder embedded motor is utilized in second prototype to implement different gaits. In second prototype, robot reaches to a size and mobility for flocking.

Keywords: Leg Design, Walking Robot, Optimization, 3D Printer

ÖZ

SÜRÜ HALİNDE HAREKET İÇİN DÜŞÜK BÜTÇELİ ROBOT TOPLULUĞU SİSTEMİ TASARIMI

DEMİR, Çağrı Ata
Yüksek Lisans, Makina Mühendisliği
Tez Danışmanı: Dr. Öğr. Üyesi Ali Emre Turgut
Ortak Tez Danışmanı: Dr. Öğr. Üyesi Kutluk Bilge Arıkan

Kasım 2019, 85 sayfa

Robot topluluğu bilimi, çok sayıda robotun koordinasyonunu inceler. Hem iç hem de dış mekanlarda sürü halinde hareket etmek için tasarlanmış bir robot sistemini incelemesi bu tezin ana motivasyonunu oluşturuyor. Bu amaç doğrultusunda bir yürüyen robot sistemi geliştirildi. Tezin ilk bölümünde, mekanik aksaklıkları ortadan kaldırmak için, robotun kütle merkezinin düşey eksende yer değiştirmesini en aza indirecek bir bacak tasarlandı. Bu süreçte, mekanizma analizi ve Matlab optimizasyon araçları kullanıldı. Daha sonra, robotun elektronik bileşenleri belirlendi ve önceki aşamada tasarlanan ayağa ve monte edilecek elektronik ekipmana uygun bir mekanik tasarım Solidworks platformunda prototip olarak yapıldı. Robotun kontrolcüsü Arduino platformunda yazıldı. Parçalar Ultimaker 2+ 3 boyutlu yazıcı kullanılarak üretildi ve sonrasında robotun montajı yapıldı. Testlerde robotun engellerden kaçma ve ışık algılama becerileri incelendi. Tezin ikinci bölümünde, ilk prototip geliştirildi ve üretim süresini kısaltmak için daha küçük bir robot tasarlandı. Farklı yürüme biçimlerinin denenebilmesi için ikinci prototipte konum algılayıcılı motor kullanıldı. İkinci prototipte robot toplu şekilde hareket için uygun boyutlara ve hareket kabiliyetine ulaştı.

Anahtar Kelimeler: Bacak Tasarımı, Yürüyen Robot, Optimizasyon, 3 Boyutlu Yazıcı

To my lovely family

ACKNOWLEDGEMENTS

First of all, I would like to thank to my advisor, Ali Emre Turgut, who gave me the chance to begin my graduate journey and always there for me whenever I was struggling with my thesis with sympathy and understanding. I also would like to thank to my co-advisor, Kutluk Bilge Arıkan, for sharing his complete knowledge with me in my hardest times. Without him, this study would not shine brightly. Special thanks go to Eres Söylemez. Even if I was not his student, he accepted to help me at a critical checkpoint in my thesis and accelerated my study. This thesis would not be complete without Onur Can Yağmur. He helped me a lot especially in production stage. I am very grateful to him. I would like to thank to my lead engineer in office, Bengi Demirçivi, who was always supportive for three years. She never said no when I asked permission to go to department. Without my mother and father, Elçin and Zafer Ziya Demir, I would accomplish nothing. Whenever things go wrong, they are the ones encouraging me to keep fighting. I owe everything to them. Sincere thanks coming from bottom of my heart to Ece Tavukçuoğlu, who has become my main motivation not only in my thesis process, but also in my entire life. She makes me feel stronger in every single stage of my life. She becomes my light when things seem dark. No word is enough to thank to her. I am so glad to have her. I have a second family since last year. I would like to express my thanks to Büşra and Hakan Tavukçuoğlu, for being so kind and affectionate to me. Their love raised my self-esteem and eased my stress in my thesis writing process. The hero of my thesis: Bulut Tavukçoğlu who is the cat of Tavukçuoğlu family. My robot, Bulubot, takes its name from him. I owe him a special thank for being such an inspiration.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

CHAPTER 1

INTRODUCTION

1.1. Motivation

In swarm robotics, behavior of insects, such as ants and termites, have been the major inspiration in the development of coordinated motion of robots. The main principle laying behind the insect behavior is the decision process done collectively.[1]

A swarm robotic system should embrace three major characteristics. These are robustness, flexibility and scalability. *Robustness* means that organization of a swarm robotic system should not be affected by any kind of external disturbances or individual failures. *Flexibility* means that a swarm should be able to arrange its behavior depending on the requirements of different tasks such as foraging or aggregation. *Scalability* means that changes in the swarm size should not affect the performance of the overall robotic system. [1,2]

As far as coordinated motion is concerned, there are several problems studied in swarm robotics: *Self-deployment, aggregation, foraging, self-assembly, pattern formation,* and *coordinated movement. Self-deployment,* which is also called *dispersion,* underlines a swarm robotic system covering an area as much as possible. *Aggregation* is the behavior, in which randomly distributed robots form the largest possible aggregate. *Foraging* is a behavior that robots search for the best food source in the environment like ants. *Self-assembly* is a behavior that robots assemble to form certain structures based on the needs of a certain task. *Pattern formation* is forming certain patterns by local interactions among robots. *Coordinated movement*, in other words *flocking*, is the movement of robots to a certain direction. [3]

The focus of this thesis is the flocking behavior. In nature, flocking is a highly encountered behavior in certain groups of animals such as fish, birds and locusts. Animals benefit from flocking, or coordinated motion, by increasing their survival rate in nature, navigating in a more precise manner and consuming less energy than an individual effort. [4] As for autonomous robots, coordinated motion not only tenders a collective behavior with a minimum amount of collision between agents, but also improves perception of the environment of the swarm. [5]

The pioneering study on coordinated motion is done by Reynolds in computer graphics [6]. He proposed a framework in which flocking behavior observed in birds is to be generated in an artificial manner by combining simple behaviors obeying local rules. In his model, he successfully made agents perform separation, alignment and cohesion behavior in Figure 1.1.

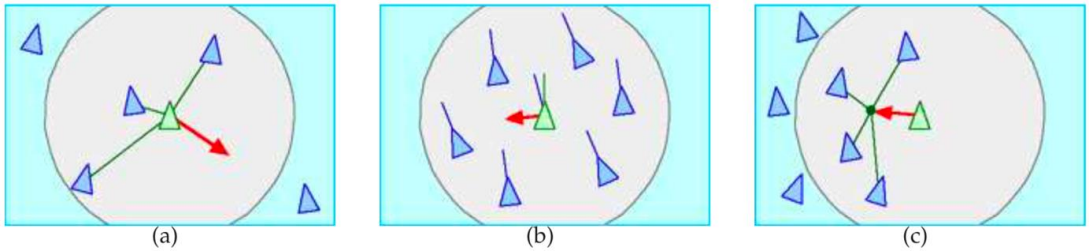

Figure 1.1. Three main behaviors of Reynold's model (a) Seperation. (b) Alignment. (c) Cohesion. [6]

1.2. Problem

The aim of this thesis is to design a legged robot that can be used in flocking. Robot should consist of simple parts that can be easily manufactured to keep its cost low. It should also have a compact design for flocking.

In the following section, an extensive literature research including wheeled, legged and 3D printed robots is presented. In the third chapter, methods used in leg and robot design are discussed. In chapter four, applications of methods are described. Mechanical and electronic design of robotic system are detailed. In the course of chapter five, robot experiments are explained alongside the results obtained from

simulations and real time applications. In the sixth chapter, results of experiments are discussed. In the final chapter, conclusion of thesis is stated.

CHAPTER 2

LITERATURE SURVEY ON SWARM INTELLIGENCE

2.1. Introduction

Swarm intelligence is linked to biology. Difficulties and complicated problems in nature have been struggled by biological systems through collective decision making for millions of years [7]. The pioneering work of the swarm robotics was accomplished by Beni [8], whose work includes cellular robotic systems. The author discusses the conceptual basis for the theory and engineering of a new type of robotic system. The system is composed of autonomous robotic units which accomplish tasks in cooperation. After describing the relevance of this system and contrasting it with cellular automata and neural networks, the author establishes the fundamental properties of the system and their consequences for the structure of the robotic units, the space on which they operate, and the algorithms by which they accomplish the global tasks. The significance of the concept of cellular robotic systems for distributed computing, molecular computing, self-organization, and reliability is explained. Swarm robotics emerged from swarm intelligence as a promising approach to solve multi-robot coordinated problems[2]. Swarm robotics is defined as:

"The study of how a large number of relatively simple physically embodied agents can be designed such that a desired collective behavior emerges from the local interactions among the agents and between the agents and the environment."[1]

Robots in a swarm robotic system are expected to satisfy the following requirements[1,2] :

- *Sensing and signaling:* Robots should not be interfered by neighbors' sensing system and environmental signals.

- *Communication:* . Robots must support wireless communication.
- *Physical Interaction:* Individuals in swarm shall interact with each other and with the environment for certain tasks.
- *Power:* A long battery life is necessary to complete task.
- *Cost:* Swarm platforms should be cost effective.
- *Size:* Robots should be small enough not to increase the size of test arena, yet big enough not to limit the mobility of the robot.

2.2. Wheeled Mobile Robots for Swarms

Several robotic platforms are designed and used for a variety of tasks. In this section, wheeled robotic systems are discussed.

- s-bot [9] is a round robot with a 116 mm diameter. Its locomotion is provided via two DC gear head motors driving its wheels. Robots have grippers for self-assembly. There are 15 infrared proximity sensors. 4 infrared sensors located beneath robot. Torque sensors on wheels, a three-axis accelerometer, an omni directional camera and a force sensor are additional electronics of the s-bot. Signaling between robots is realized by 8 red-green-blue LED. Wireless communication is adopted by a WI-FI module. 64 MB RAM, 32 MB memory and 400 MHz XScale CPU Board are equipment of s-bot. A Li-Ion battery is used and it delivers one-hour of battery life. SwarmBot3D is its simulation environment.
- Alice [10] mobile robot has a rectangular shape and it is 22x21 mm in size. For locomotion, two SWATCH motors are used. A PIC16F877 microcontroller having 8K words flash EPROM program memory is used. Four infrared proximity sensors provide obstacle detection and communication module, which are effective in short range. It is claimed that it can work autonomously by using two button batteries for 10 hours. Battery life can be improved by using a LiPoly battery increasing battery life to 20 hours. Webots simulation platform is used as the simulation platform.

- Swarmbot [11] is a four wheeled square robot with a 130 mm edge length. DC gear-head motors are connected to each wheel for loccomotion. Its electronic equipment consists of a camera, eight bump sensors, FPGA, ARM Thumb CPU and four light sensors. It has an infrared system called ISIS sensing orientation, distance and bearing of its neighbor robots. To be able to program the robot, an RF communication unit is used.
- e-Puck [12] is a circular robot having a diameter of 70 mm. Its locomotion is achieved via two stepper motors. A dsPIC30F6014 microcontroller having 8KB RAM and 144 KB memory is used as the controller. The robot has a speaker for audible feedback and three directional microphones for sound source localization. For wireless programming and communication, a Bluetooth module is used. For signaling, a couple of LEDs are placed. The battery of the robot is a 5 Wh Li-Ion type that provides 3 hours of autonomy. Webots is used as simulator.
- Another circular robot is Kobot [13] with a diameter of 120 mm. It has two gear-head DC motors for locomotion. Obstacle detection is done by a modulated infrared system to reduce the effects of ambient lighting conditions. As wireless communication channel, IEEE 802.15.4/ZigBee module is utilized. A LiPoly battery gives robot 10 hours long functionality. Its simulator is custom made.
- Sambot [14] has a size of 80x80x120 mm. 2 differentially driven wheels are used in system. A control unit with an ARM series STM32 microprocessor and ATMega8 microcontroller is used. Actuator unit consist of four micro DC motors. No battery and simulation information are reported.
- Jasmine [15] has been designed as a rectangular robot with a size of 23x23 mm. Two gear-head motors are used in Jasmine for locomotion. Infrared sensors are used for proximity sensing and communication with neighbors. It has one infrared LED. It has an autonomy of two hours with LiPoly batteries. LaRoSim simulation environment is used for simulations.
- Kilobots [16] are mainly designed to test collective algorithms. For locomotion, it uses two vibrational motors. Its controller is Atmega 328 microprocessor running

at 8 Mhz and having 32K of memory.. In order to communicate with other Kilobots, each Kilobot has an infrared LED transmitter and infrared photodiode receiver. Each robot can be active for 3-24 hours with a lithium-ion battery on them. The most intriguing property of Kilobot is its low cost. It does not have its own simulation platform but it can be simulated through other platforms.

- R-one robot [17] is a circular 2-wheeled robot with 100 mm of diameter weighing 230 grams. Locomotion is done via 2 DC gear-head motors. Control of the robot is done via Texas Instruments Stellaris LM3S8962 microcontroller. ARM Cortex-M3 is its CPU Core with 64KB of SRAM and 256 KB flash memory. Eight infrared transmitters and receivers are used as proximity sensors. A LiPoly battery is used as power source, which can maintain its operation for 4 hours.
- Weemik [18] has a cylindrical body whose diameter is 120 mm and height is 160 mm. Its locomotion is provided by 3 omnidirectional wheels, DC gear-head motors and shaft encoders. An 8bit Atmel AVR atmega2560 is used as the main control unit. It has 8 infrared sensors for proximity control. 4 Li-ION Batteries are utilized, but battery life is not reported.
- UB Robot [19] is developed by RISC Lab. DC motors, geared DC motors and servomotors are used for locomotion. Its control unit consists of PIC32 and Arduino Uno microcontrollers. Ultrasonic and infrared sensors are used to avoid obstacles and keep a certain distance between robots. To obtain a cooperative task, one robot in the swarm is equipped with GPS/GPRS/GSM module shield, while others have encoders and vision navigation for sharing position information with host and swarm. Rechargeable NiMH and LiPoly batteries are utilized, which provides approximately 3 hours of autonomy.
- Mona [20] is a circular robot with an 80 mm of diameter. Actuation is provided by using 2 DC gear-head motors. The main processor of Mona is ATmega 328 microcontroller with 32 KB flash memory, 2KB SRAM and 1 KB EEPROM. 5 infrared sensors are available on Mona for proximity control. In addition, there are expansion boards on which extra sensors can be added. A 3.7 Volts, 250 mAh

battery gives Mona 1.2 hours of continuous operation in the case of maximum motor speed and continuous sensing. Stage is used as the main simulation platform for Mona.

2.3. Legged Mobile Robots for Swarms

Robots mentioned in the previous section are all wheeled ones. Legged robots designed for swarm robotics are introduced in this section.

- The Harvard Ambulatory MicroRobot(HAMR) [21] is a 5.7 cm robot having six legs. Its fabrication is done via Smart Composite Microstructure technique. For its actuation, piezoelectric actuators are preferred to obtain higher scalability and easy integration. Spherical five-bar linkage is used in its leg.
- Hexapod [22] is a foldable robot fabricated from a single sheet of plastic. A laser cutter extracts the pattern to be folded from the sheet. ATMEL ATtiny2313 microcontroller is used, which have a central role in controlling two DC gear-head motors. To command it wirelessly, an Xbee module is utilized. As the power source, a LiPoly battery is used in the robot. Legs of the robot are constructed via two four bar mechanisms.
- Tribot [23] is an origami inspired six legged robot designed for swarm applications. As controller, an Atmel Atmega81 microcontroller unit is used. The leg design of Tribot is based on six bar Hooken linkage. Its production stage is conducted over folding a single sheet of polyester after extracting the correct pattern via a laser cutter. LiPoly batteries are used as the power source.

2.4. Legged Small Sized Mobile Robots

Legged robots introduced in this section include robots are not intended for swarm applications.

- TURTLE [24] is a relatively bigger robot having a length of 500 mm, width of 330 mm and height of 380 mm. Two DC servomotors are used for locomotion. An innovative leg mechanism called ASTBALLEM, which results in a straight-line

motion, is adopted. A microcomputer is used as the main controller. In its electronic design, 8 channels potentiometer and tachometer generator are utilized.

- SCOUT [25] is a simple legged robot which can walk, climb and run with dimensions of 200 mm length and 190 mm wide. It has 4 RC servo motors per each leg for locomotion. It is connected to a stable power source rather than a battery.
- Mini-Whegs [26] is a 90 mm long, 68 mm wide robot, which can run and jump besides walking. It has a Maxon DC motor for locomotion. Its legs are whegs with 36 mm radius. An RC transmitter and a sub-micro receiver manage its control. Steering motion is obtained with a micro-servo. Two CR2 lithium batteries are used as the power source.
- iSprawl [27] is a six-legged robot with a size of 155x116x70 mm. It has a DC motor and an RC servomotor for locomotion. Manufacturing process is conducted by Shape Deposition Process, which is a multi-material rapid prototyping process. As for leg design, a double crank-slider mechanism is embraced. It has a six-pack LiPoly battery as the power source.
- A Biomimetic Quadruped Robot [28] designed by Konkuk University has a length, width and height 120 mm, 115 mm and 75 mm respectively. A different type of actuator called Lightweight Piezoceramic Composite Curve Actuator (LIPCA) is designed to obtain two types of locomotion gaits. Hip joint is the only joint of the leg. With a crank connection, one piece of LIPCA actuates two legs. For simulation, ADAMS software is utilized. Its battery life is not reported.
- Another robot manufactured by Smart Composite Manufacturing technique is RoACH [29] whose length is 30 mm long and weight is 2.4 grams. SMA wires actuate robot. A parallel four bar configuration is embraced in its leg design. For its electronic circuit, a FR4 fiberglass board is preferred. A 10Mips PIC LF2520 microcontroller is used to consume less power. It has red and green LED’s to show battery voltage. It has a LiPoly battery as the power source that helps the robot to operate for approximately 9 minutes.

- DASH [30] is a rectangular legged robot with length of 100 mm, wide of 100 mm and height of 50 mm. DC motors transfer their power to legs. The electronic design of the robot is not reported. The production process of DASH is done by Smart Composite Manufacturing technique. With full power, DASH continuously move 40 minutes long with its LiPoly battery.
- FireAnt [31] is another foldable hexapod. Servomotors actuate its legs. It has on-board power, computation and wireless communication. On its custom printed circuit board, a Gumstix Overo Computer-On-Module interface, a low level gait coordinator and servomotors are present. Six legs of robot are extracted from a 2D paper assembled to body afterwards. It has a Li-Poly battery having 39 minutes long life.
- MinIAQ [32] having dimensions 60 mm in width, 12 mm in length and 43 mm in height is another origami inspired robot whose pattern is extracted from an A4 PET film. Arduino Pro is its microcontroller. Lightweight DC motors, which are driven by two L293D H-bridge motor drivers, are used for locomotion. Its leg design is based on a four-bar mechanism to control speed and position of legs. Small analog infrared sensors are utilized. It has a LiPoly battery lasting 30 minutes.

2.5. 3D Printed Small Sized Mobile Robots

Especially in the last decade, 3D printers became very popular in prototyping industry. 3D printer technology is very useful for robotic studies. Since keeping the cost of robots low is primary concern in most of the robotic studies, 3D printed robots or robots having 3D-printed subparts are very common. In this section, 3D printed robots are discussed.

- Rat-like robot [33] having 173.5 mm of length, 68 mm of width and 55 mm of height consists of 4 legs. 8 of its 12 degree of freedoms are driven by 8 DC servomotors and the rest is driven by RC servo motors. Two types of links constitute the robot leg, the proximal link and the distal link. Proximal link is

directly connected to DC servomotors while the transmission of motion is obtained in distal link by a wire and pulley mechanism. The entire body of robot is manufactured by 3D printing. As microcontroller, PIC18F252 model is used.

- The inverted Mini-Whegs [34] is a 46 mm length robot with its legs made out of Mushroom-Shaped Adhesive Microstructure (MSAMS) tapes for walking on ceilings. Gear-head motors provide locomotion. Its drivetrain is completely comprised of 8 3D-printed parts. A LiPoly battery is used as power source.
- Aracna [35] is an open source robot. Its parts are manufactured by 3D printing. 2 four bar mechanisms are driven by 2 RC servomotors 4 legs. It has Arbotix microcontroller, an Xbee module providing wireless communication with an external computer, AX-18A servomotors and a single LiPoly battery.
- A quadruped robot [36] is developed, which can be manufactured by 3D-printing. Two RC servomotors are used for locomotion. The leg mechanism of robot is based on Jansen's linkage. In order to model the 3D assembly and parts, a CAD software called OpenSCAD is adopted. As for processing unit, an Arduino UNO microcontroller is used. As power source, 4 AA batteries are used.
- With its maximum length of 525 mm and maximum height of 145 mm, Hexabot [37] is a six-legged 3D-printed walking robot. RC servomotors are utilized for locomotion. The legs of Hexabot are produced with polyactic acid filament. In order to analyze disturbances stemming from walking behavior of Hexabot, a motion capture system is used.

2.6. Contributions

Throughout the literature review, several types of robots are underlined and it is observed that there are mainly two types of robots as far as locomotion is concerned. Wheeled robots are more common in swarm robotics. Legged robots are more complex than wheeled robots due to the increase of number of actuators. However, legged robots have some definite advantages. Walking robots provide better mobility on rough terrains. They can step over obstacles rather than avoiding them. Walking

robots are energy efficient especially on soft grounds. Therefore, a walking robot is designed in the thesis, to which a flocking algorithm can be directly implemented.

The contributions of the thesis are:

1) Design of a quadruped walking robot eligible for flocking.
2) Design of a 3D-printed robot that can be produced in a short time.
3) A low-cost robot with its electronic equipment and simple production technique.
4) Design of a novel leg mechanism.

CHAPTER 3

METHODS USED IN LEG DESIGN AND FLOCKING BEHAVIOR

3.1. Introduction

This chapter consists of two sub-sections. In the first sub-section, useful background information about both mechanisms and Matlab optimization tools is introduced. In the second sub-section, the methods developed for the thesis are discussed.

3.2. Background

Mechanism is defined as a rigid body assembly whose linkages (links or bars) are connected to each other by joints, in order to provide force and motion transmission. In order for such a kinematic chain to be counted as a mechanism, it needs to have at least one fixed link and two mobile links.

Mechanisms are classified in three groups based on the tasks they are meant to achieve. Motion generation deals with the entire coupler motion. Path generation deals with the path of tracer point following. Function generators deal with the relative motion between two grounded links. The methods used in this thesis embrace path generation task.

In this section, two different approaches employed in leg design process are introduced. First method is kinematic synthesis of mechanisms and position analysis of mechanisms is the second one.

3.2.1. Four Position Synthesis of Mechanisms

Kinematic synthesis is a process, in which mechanism dimensions (fixed joint coordinates, link lengths and link positions) are determined in order to achieve a predefined output motion. Displacement angles, path points and link positions constitute the output parameters obtained as a result of kinematic synthesis [38]. In this thesis, quantitative methods are adopted. [39]

The principle behind position synthesis of mechanisms is illustrated by investigating Figure 3.1. By taking the vector sums of each vector, two loop closure equations are obtained.

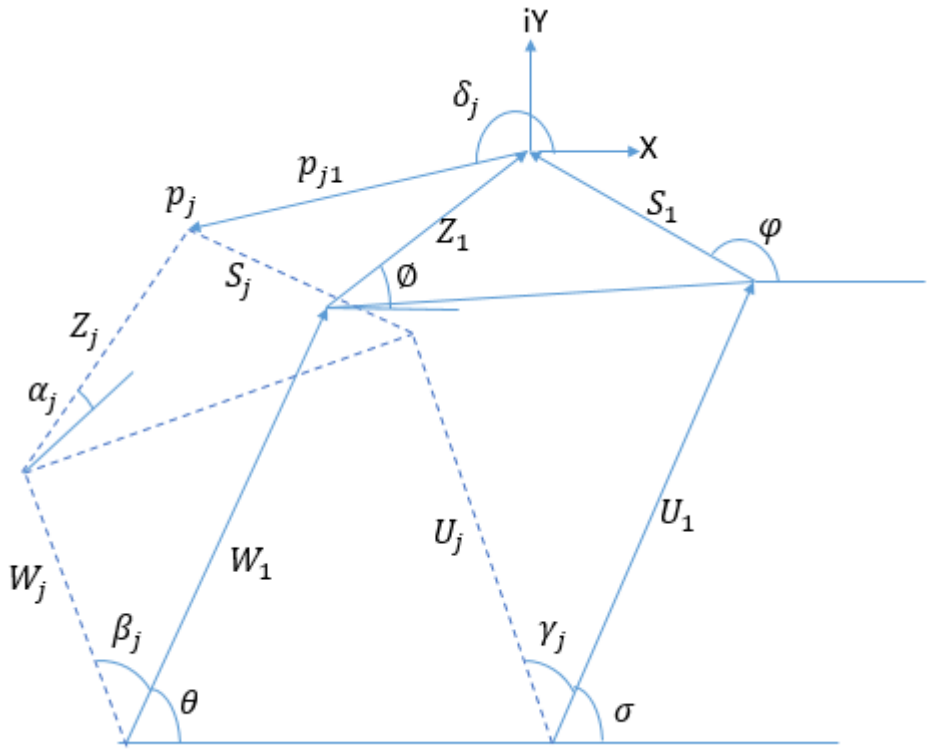

Figure 3.1. Initial position and displaced position (dashed) of a four-bar mechanism

$$W_1 e^{i\theta} + Z_1 e^{i\phi} + P_{j1} e^{i\delta_j} - Z_1 e^{i(\phi + a_j)} - W_1 e^{i(\theta + \beta_j)} = 0 \qquad (1)$$

$$U_1 e^{i\sigma} + S_1 e^{i\psi} + P_{j1} e^{i\delta_j} - S_1 e^{i(\psi + a_j)} - U_1 e^{i(\sigma + \beta_j)} = 0 \qquad (2)$$

where $P_{j1}e^{i\delta_j} = p_j - p_1$

After factorization, the loop closure equations become;

$$W_1 e^{i\theta}\left(e^{i\beta_j} - 1\right) + Z_1 e^{i\phi}\left(e^{ia_j} - 1\right) = P_{j1}e^{i\delta_j} \qquad (3)$$

$$U_1 e^{i\sigma}\left(e^{i\gamma_j} - 1\right) + S_1 e^{i\psi}\left(e^{ia_j} - 1\right) = P_{j1}e^{i\delta_j} \qquad (4)$$

In the case of four-position synthesis, there exists 3 different crank and coupler angles. After writing these equations for each position, matrix given below is obtained:

$$\begin{bmatrix} e^{i\beta_2} - 1 & e^{ia_2} - 1 \\ e^{i\beta_3} - 1 & e^{ia_3} - 1 \\ e^{i\beta_4} - 1 & e^{ia_4} - 1 \end{bmatrix} \begin{bmatrix} W_1 e^{i\theta} \\ Z_1 e^{i\phi} \end{bmatrix} = \begin{bmatrix} P_{21}e^{i\delta_2} \\ P_{31}e^{i\delta_3} \\ P_{41}e^{i\delta_4} \end{bmatrix} \qquad (5)$$

The goal in four position synthesis is to determine the W_1 and Z_1 vectors, which are also dimensions of mechanism searched for. In order to obtain these vectors, unknown displacement angles β_2, β_3, β_4 are found by using user-prescribed parameters a_j, P_{j1} and δ_j (j = 1,2,3). For W_1 and Z_1 to have solutions, the following condition should be satisfied:

$$\left| \begin{Bmatrix} e^{i\beta_2} - 1 & e^{ia_2} - 1 \\ e^{i\beta_3} - 1 & e^{ia_3} - 1 \\ e^{i\beta_4} - 1 & e^{ia_4} - 1 \end{Bmatrix} \begin{Bmatrix} P_{21}e^{i\delta_2} \\ P_{31}e^{i\delta_3} \\ P_{41}e^{i\delta_4} \end{Bmatrix} \right| = 0 \qquad (6)$$

If the determinant is expanded:

$$\Delta_2 e^{i\beta_2} + \Delta_3 e^{i\beta_3} + \Delta_4 e^{i\beta_4} + \Delta_1 = 0 \qquad (7)$$

where;

$$\Delta_2 = \begin{vmatrix} e^{ia_3} - 1 & P_{31}e^{i\delta_3} \\ e^{ia_4} - 1 & P_{41}e^{i\delta_4} \end{vmatrix}$$

$$\Delta_3 = \begin{vmatrix} e^{ia_2} - 1 & P_{21}e^{i\delta_2} \\ e^{ia_4} - 1 & P_{41}e^{i\delta_4} \end{vmatrix}$$

$$\Delta_4 = \begin{vmatrix} e^{ia_2} - 1 & P_{21}e^{i\delta_2} \\ e^{ia_3} - 1 & P_{31}e^{i\delta_3} \end{vmatrix}$$

$$\Delta_1 = -\Delta_2 - \Delta_3 - \Delta_4$$

Defined parameters can be utilized to find input angles of both chains by using the following algorithm [40]:

$$\Delta = \Delta_1 + \Delta_2 e^{i\beta_2}$$

$$\cos\theta_3 = \frac{|\Delta_4|^2 - |\Delta_3|^2 - |\Delta|^2}{2|\Delta_3||\Delta|}$$

$$\sin\theta_3 = \left|\sqrt{1 - (\cos\theta_3)^2}\right| \geq 0$$

$$\theta_3 = ATAN2(\sin\theta_3, \cos\theta_3) \quad \text{where} \quad 0 \leq \theta_3 \leq \pi$$

$$\beta_3 = arg\Delta + \theta_3 - arg\Delta_3$$

$$\tilde{\theta}_3 = 2\pi - \theta_3$$

$$\tilde{\beta}_3 = arg\Delta + \tilde{\theta}_3 - arg\Delta_3$$

$$\cos\theta_4 = \frac{|\Delta_3|^2 - |\Delta_4|^2 - |\Delta|^2}{2|\Delta_4||\Delta|}$$

$$\sin\theta_4 = \left|\sqrt{1 - (\cos\theta_4)^2}\right| \geq 0$$

$$\theta_4 = ATAN2(\sin\theta_4, \cos\theta_4) \quad \text{where} \quad 0 \leq \theta_4 \leq \pi \quad \text{and} \quad \tilde{\theta}_4 = -\theta_4$$

$$\beta_4 = arg\Delta - \theta_4 - arg\Delta_4$$

$$\tilde{\beta}_4 = arg\Delta + \theta_4 - arg\Delta_4 + \pi$$

The algorithm results in two of three positions' displacement and starting angles. Besides, it gives two different branches of mechanism passing through the same 4 positions as illustrated in Figure 3.2. $\beta_3, \beta_4, \theta_3, \theta_4$ correspond to the parameters of one branch of mechanism while $\tilde{\beta}_3, \tilde{\beta}_4, \tilde{\theta}_3, \tilde{\theta}_4$ correspond to the other

one. After obtaining all required angles, these values are plugged in equation (5) to find W_1 and Z_1 vectors. By implementing exactly the same procedure to remaining of the mechanism, U_1 and S_1 vectors can be obtained too.

In order to initialize the algorithm, β_2 value should be specified. By determining a range for β_2 between 0 and 2π, a curve called Burmester's curve is plotted, on which two center and circle points constituting a four-bar mechanism passing through the prescribed four positions are selected.

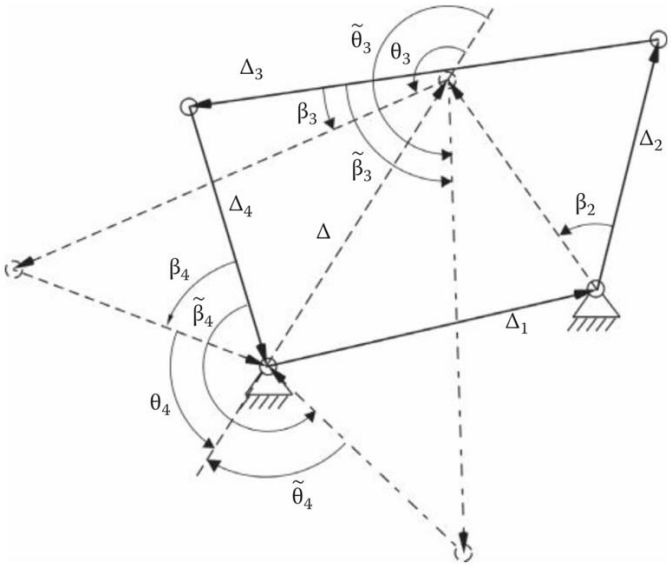

Figure 3.2. Both branches of single mechanism with their starting and displacement angles [40]

Although mechanisms suggested by Burmester's curve guarantee required path generation, some of the selected mechanisms might have branch defect or order defect. Former states that mechanism may disassemble and reassemble from one to another during motion in order to achieve passings directly or approximately through the prescribed positions. An illustration is given in Figure 3.3. Figure shows that it is not possible to reach 1*-1-2-3 positions continuously with single assembly. In the case of latter, mechanism may disobey the order of settled path and may follow the path with a different order. As given in Figure 3.4 mechanism may draw an eight path rather than an ellipse. [40]

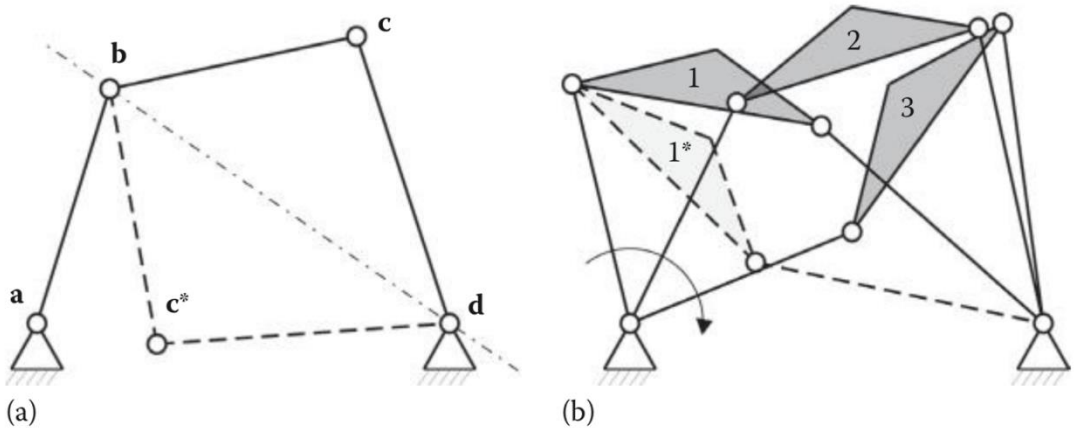

Figure 3.3. a) Both branches of four bar b) Branch defect [40]

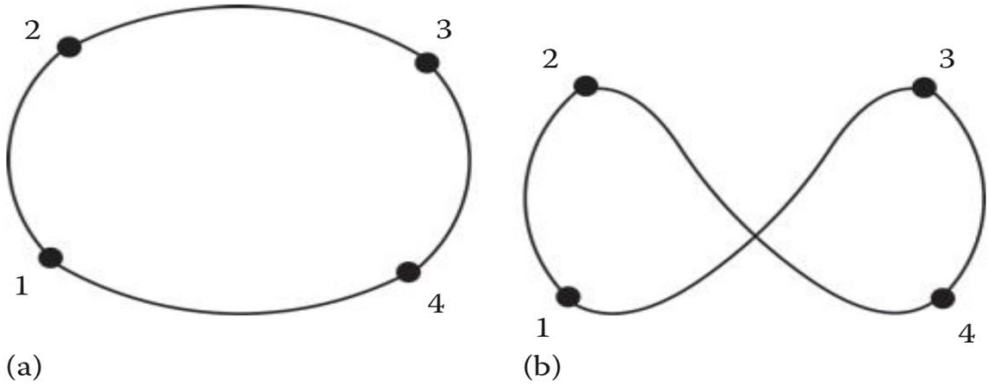

Figure 3.4. a) Path required to be followed b) Path mechanism follows [40]

3.2.2. Position Analysis of Four Bar Mechanism

The four bar mechanism analysis is based on Freudenstein's equation, which investigates loop closure equation of four bar. A representative four bar is given in Figure 3.5.

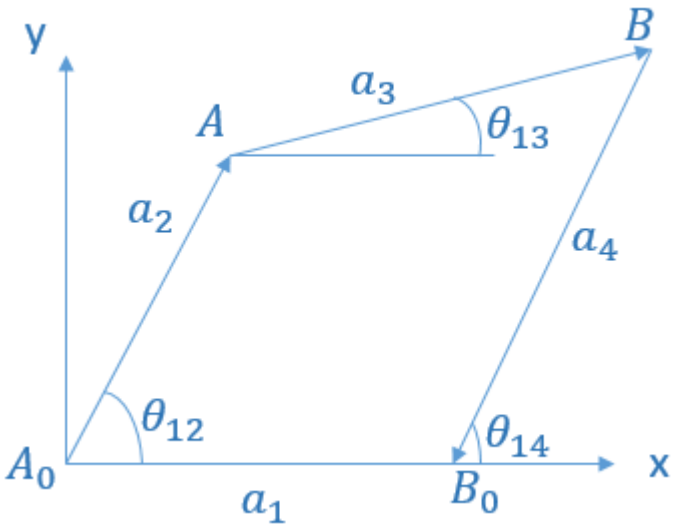

Figure 3.5. Four Bar Mechanism

The loop closure equation can be written such that [41]:

$$A_0A + AB = A_0B_0 + B_0B \quad (8)$$

If complex numbers are plugged in the equation:

$$a_2e^{i\theta_{12}} + a_3e^{i\theta_{13}} = a_1 + a_4e^{i\theta_{14}} \quad (9)$$

In the case of complex numbers, the conjugate of complex equations is also true, hence it is possible to write another loop closure equation such that:

$$a_2e^{-i\theta_{12}} + a_3e^{-i\theta_{13}} = a_1 + a_4e^{-i\theta_{14}} \quad (10)$$

After multiplication of equations (9) and (10) and some arrangements to resulting equation, the equation given below is obtained:

$$K_1 \cos\theta_{14} - K_2 \cos\theta_{12} + K_3 = \cos(\theta_{14} - \theta_{12}) \quad (11)$$

where, $K_1 = \frac{a_1}{a_2}$, $K_2 = \frac{a_1}{a_4}$, $K_3 = \frac{(a_1{}^2 + a_2{}^2 - a_3{}^2 + a_4{}^2}{2a_4a_2}$

Equation (11) is known as Freudenstein's Equation, which lies behind both mechanism analysis and synthesis principles.

Half-tangent representations of sine and cosine functions can be written such that:

$$\sin\theta_{14} = \frac{2\tan(\frac{1}{2}\theta_{14})}{\left[1+(\tan(\frac{1}{2}\theta_{14}))^2\right]}$$

$$\cos\theta_{14} = \frac{\left[1-(\tan(\frac{1}{2}\theta_{14}))^2\right]}{\left[1+(\tan(\frac{1}{2}\theta_{14}))^2\right]}$$

Introducing half-tangent equations into Freudenstein's equation results in:

$$A\,(\tan(\tfrac{\theta_{14}}{2}))^2 + B\tan\left(\tfrac{\theta_{14}}{2}\right) + C = 0 \quad (12)$$

where $A = \cos\theta_{12}(1-K_2) + K_3 - K_1$, $B = -2\sin\theta_{12}$, $C = \cos\theta_{12}(1+K_2) + K_3 + K_1$

Since equation (12) is a quadratic equation, it can be solved by second order quadratic equation solution methods:

$$\tan\left(\frac{\theta_{14}}{2}\right) = \frac{-B \pm \sqrt{(B^2-4AC)}}{2A}$$

$$\theta_{14} = 2\left(\tan\frac{-B \pm \sqrt{(B^2-4AC)}}{2A}\right)^{-1}$$

After obtaining θ_{14}, geometric relations of mechanism can be used to find θ_{13}. From geometrical point of view, inverse of tangent of θ_{13} is suitable in this process, which requires x and y components of coupler:

$$x_{13} = -a_2\cos\theta_{12} + a_1 + a_4\cos\theta_{14}$$

$$y_{13} = -a_2\sin\theta_{12} + a_4\sin\theta_{14}$$

$$\theta_{13} = atan2(y_{13}, x_{13})$$

This process completes the definition process and position analysis of four-bar mechanism. Path following can be done after mechanism is completely defined by using geometric relations once again.

Point C is the point, on which position analysis is implemented in order to detect its position in different orientations of mechanism. Since a four-bar may be placed in a different coordinate system, position of point C shall be reflected to the world coordinate system at the end of the procedure.

Position of point C is written as:

$$C_{Xr} = a_2 \cos\theta_{12} + a_{cx} \cos\theta_{13} - a_{cy} \sin\theta_{13} \quad (13)$$

$$C_{Yr} = a_2 \sin\theta_{12} + a_{cx} \sin\theta_{13} - a_{cy} \cos\theta_{13} \quad (14)$$

Equations (13) and (14) are positions of point C in $X_r O_2 Y_r$ coordinate system. In world coordinate system, position of point C becomes:

$$C_x = C_{Xr} \cos\theta_0 - C_{Yr} \sin\theta_0 + x_0 \quad (15)$$

$$C_y = C_{Xr} \sin\theta_0 + C_{Yr} \cos\theta_0 + y_0 \quad (16)$$

3.2.3. Straight-Line Mechanisms

There are some popular straight-line mechanisms in literature [42]. The path seen in the Figure 3.6 is a representative straight-line mechanism, which is more specifically called Hoeken's straight-line mechanism:

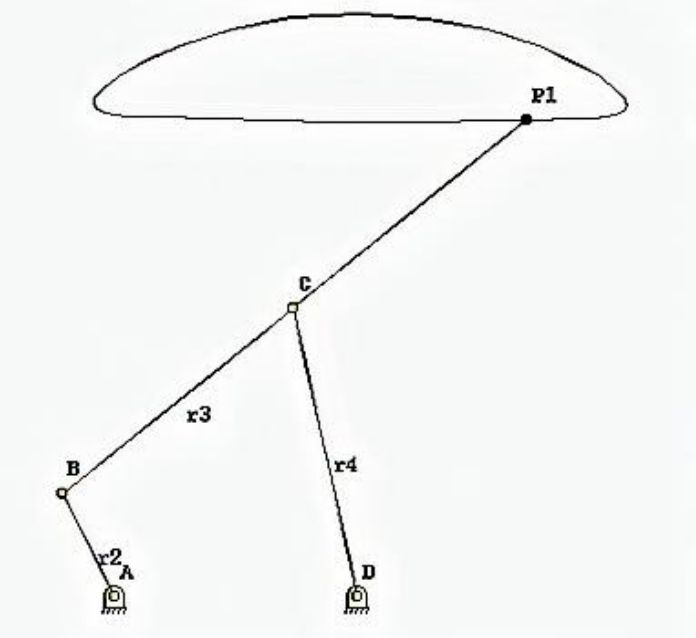

Figure 3.6. Hoeken's Straight-Line Mechanism [42]

As seen from Figure 3.6, point P1 follows a path whose lower half is a complete horizontal line. There are other famous straight-line examples in literature such as Watt's straight-line mechanism (Figure 3.7) or Chebyshev Linkage (Figure 3.8).

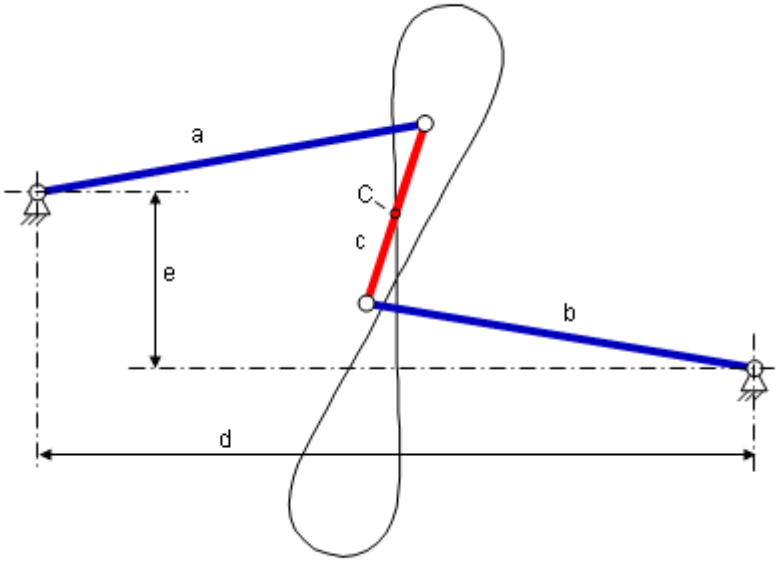

Figure 3.7. Watt's Straight-Line Mechanism [42]

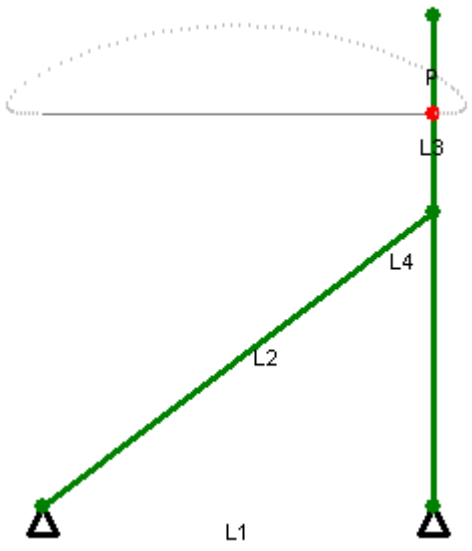

Figure 3.8. Chebyshev Linkage [42]

3.2.4. Grashof Condition

Grashof condition defines the rotation behavior of a four-bar linkage according to link lengths. Links are classified as:

S: length of shortest link

L: length of longest link

P: length of one remaining link

Q: length of other remaining link

For a four-bar to be called Grashof Linkage, $S + L \leq P + Q$ condition should be satisfied. By meeting this condition, at least one link is able to complete a full rotation with respect to ground plane. There are 3 types of Grashof Linkage:

1) Crank-rocker: The shortest link is adjacent to the ground link.
2) Double-crank: Ground link is the shortest link.
3) Double-rocker: Coupler completes a full rotation if ground link is opposite to the shortest link.

If a four-bar does not meet the Grashof condition, none of links can complete a full rotation.

3.2.5. Transmission Angle

Transmission angle is the angle between coupler and output element of mechanism shown in Figure 3.9:

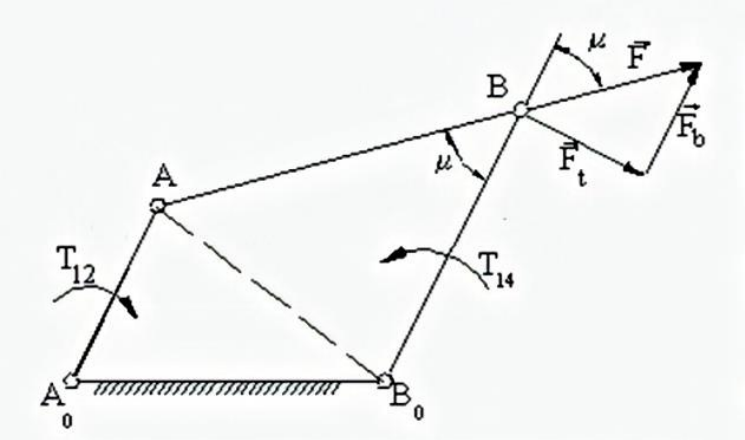

Figure 3.9. Transmission Angle Illustration [42]

Figure 3.9 shows that joint B is exposed to a certain force, whose components are normal and tangential forces, during the motion of mechanism. The ideal scenario is annihilation of normal force component, which only occurs when transmission angle is 90 degrees throughout the rotation. Since this condition is impossible, the minimum deviation from 90 degrees is required to minimize the force on the joint B.

3.2.6. Optimization Tools

Throughout this thesis, Matlab optimization toolbox is utilized. It is an extensive toolbox including sophisticated functions designed to get parameters, which maximize or minimize certain objectives while satisfying problem constraints. Optimization problems are defined with matrices or separate functions.

Rest of this section is allocated to give short details about lsqnonlin, fmincon and genetic algorithm tools that are fundamental optimization algorithms lie behind this thesis.

3.2.6.1. Lsqnonlin

This tool is a nonlinear squares solver. The algorithm implicitly calculates the square sum of components of the related function. Problems are meant to be solved through this algorithm are:

$$\min_x \|f(x)\|_2^2 = \min_x (f_1(x)^2 + f_2(x)^2 + \cdots + f_n(x)^2)$$

Algorithm starts with an initial condition x_0 and continuous to run until obtaining a solution satisfying the desired tolerance value. It is possible for users to define upper or lower bounds for the results of parameters of defined function. The function inputted to lsqnonlin algorithm should return a vector of values rather than a sum of squares.

The downside of this algorithm is that it does not let users to define either a nonlinear constraint or linear constraint function, which restricts users from narrowing down the results they intend to obtain. In other words, it is not possible to eliminate unnecessary results that can be filtered by defining constraints to parameters.

3.2.6.2. Fmincon

Different than lsqnonlin, fmincon expects a scalar,vector or matrix input to run its minimization function. It has a structure such that:

$$\min_x f(x) \begin{cases} c(x) \leq 0 \\ ceq = 0 \\ A.x \leq b \\ Aeq.x = 0 \\ lx \leq x \leq ub \end{cases}$$

In the aforementioned function c(x) refers to nonlinear inequality, ceq refers to nonlinear equality, A.x ≤ b refers to linear inequality and Aeq.x = 0 refers to linear equality. All these aspects render fmincon algorithm more preferable than lsqnonlin, because fmincon allows users to study between a certain range of solutions.

3.2.6.3. Ga and Gamultiobj

Genetic algorithm is an optimization method used in solving process of constrained and unconstrained optimization problems. It can be classified as an evolutionary method [43].

A fitness function is defined to genetic algorithm. The goal of the algorithm is to find the minimum value of the fitness function. A fitness function can be applied to each individual. For instance, if we have a function to minimize such that $f(x_1, x_2) = {x_1}^2 + {x_2}^2$ and an individual such that (2,3), the score of individual becomes 13. Every single individual reaches a score after applying the fitness function. Individuals come together and constitute an array, which is called population. For example, if a fitness function consists of 2 variables and population has 50 individuals, population is represented by a 50x2 matrix. Individuals are also called genes and there are different ways to represent them. In one way each gene is represented by a binary code, in another way they are in the form of real numbers [44].

Figure 3.10 shows a flow chart indicating each step of genetic algorithm. Algorithm initiates itself from an either a random or preselected population within a range either determined by algorithm default or chosen by user. Then, it calculates the fitness of each individual of population. If individuals do not meet the termination criterion of algorithm, it starts to select parents for crossover. The purpose of crossover is obtaining individuals with better fitness. Sometimes creating new generations is realized by mutation according to the crossover fraction determined by user. Mutation changes individuals in population arbitrarily. Crossover and mutation continue until the population satisfies termination criterion.

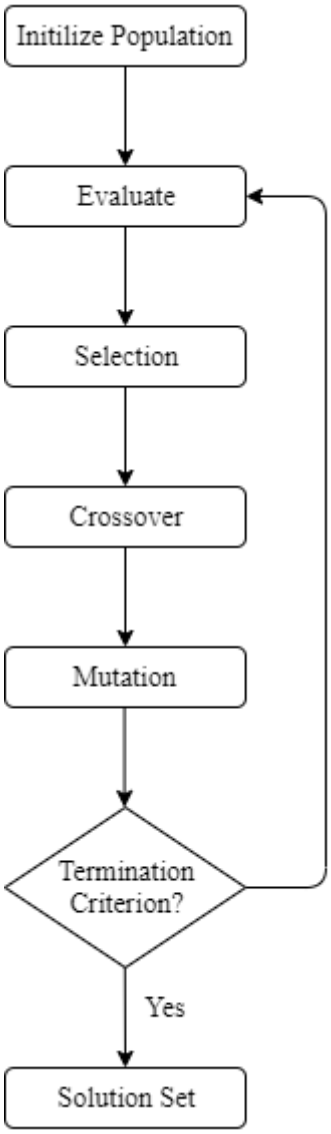

Figure 3.10. Genetic Algorithm Flow Chart [44]

In problems with one fitness function, ga function of Matlab is mainly utilized. If there exists more than one fitness function, then gamultiobj function comes into the picture. The main difference between ga and gamultiobj is that ga has one objective function to satisfy, so it gives only one individual having the best fitness score as a result at the end of the algorithm whereas gamultiobj has to build a balance between objective functions. In case of two objective functions, gamultiobj tool results in a

curve telling a trade-off between objective functions. An example plot is given in Figure 8 illustrating the process.

Figure 3.11 states that if the user wants to use the minimum value of objective function 1, it needs to sacrifice objective function 2 until a specific value and vica versa. Therefore, users have to be aware of which objective function is more important.

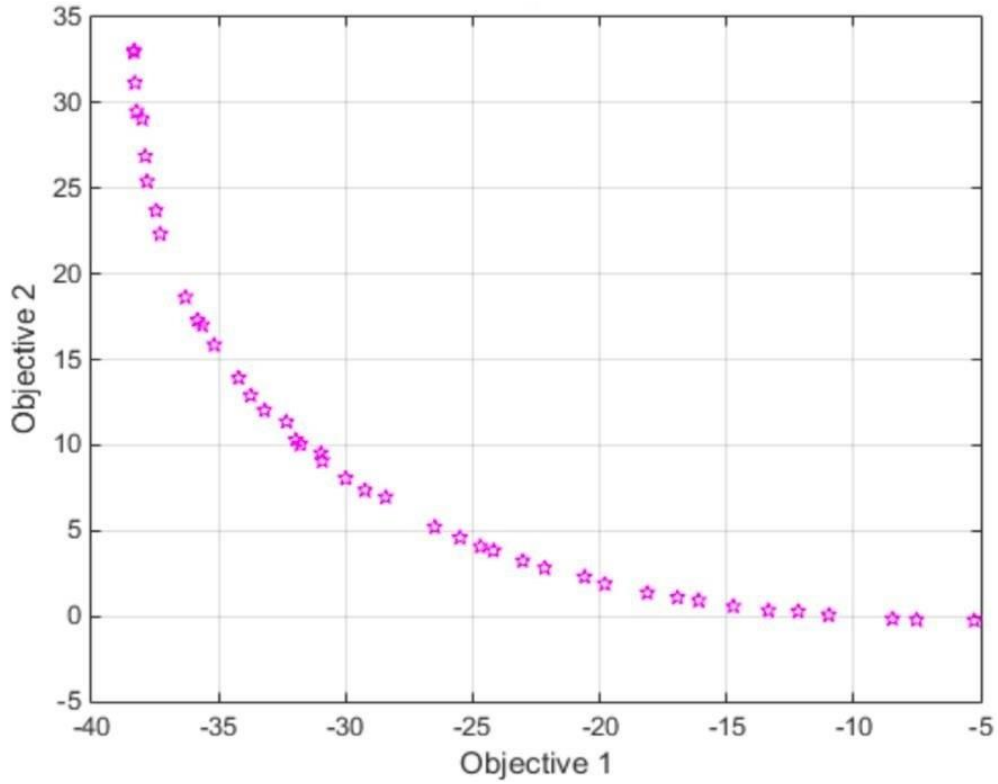

Figure 3.11. Objective function comparison regarding to gamultiobj results [44]

3.2.7. Gait Algorithms

Animal locomotion is based on lifting their legs and placing them at different positions. Legs shall be synchronized with respect to energy efficiency and stability. Leg coordination while moving is called gait. Gait characterization is done according to the lifting and placing sequence of legs. Figure 3.12 indicates some different gait

types. Crawl gait is a walking gait type. As animals change their speeds, they shift to a different gait.

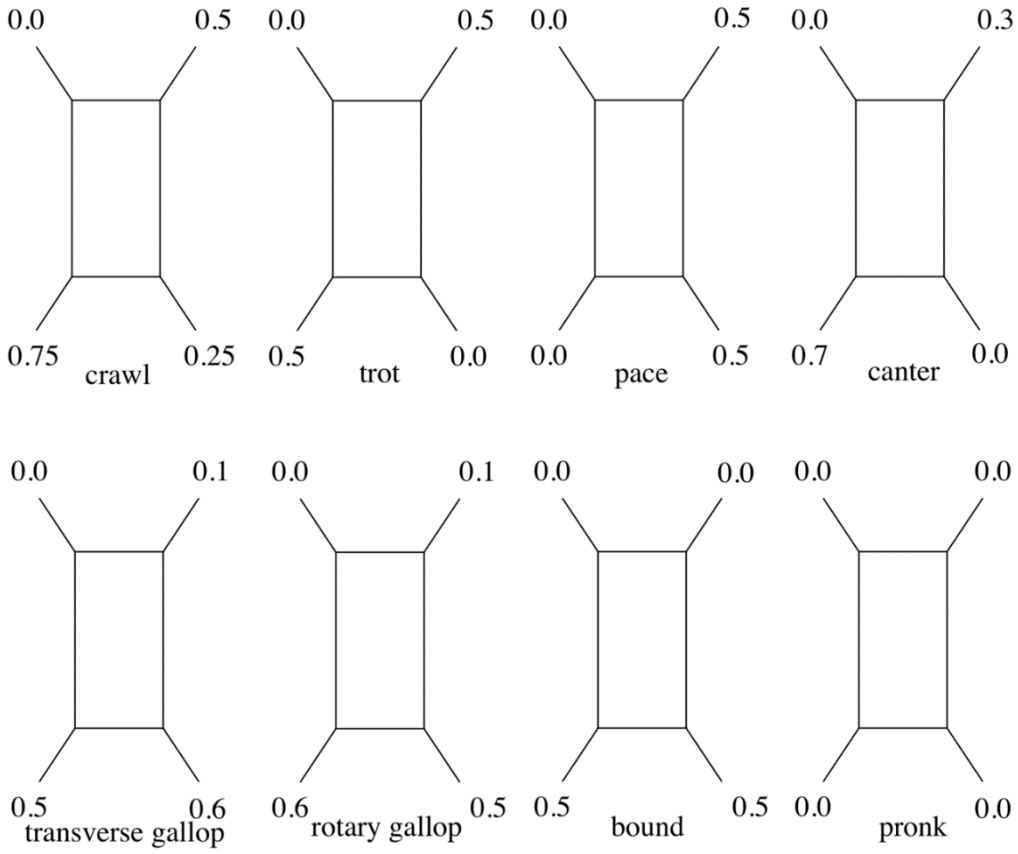

Figure 3.12. Different Gait Types [45]

Numbers written next to each leg represent relative phases according to an arbitrarily selected reference leg. The relative phase of leg i is the time elapsed from the setting down of an arbitrarily chosen reference foot until the foot of leg i is set down. Therefore, reference foot's relative phase is equal to zero. The relative phases of remaining legs are calculated as:

$$\varphi_i = \frac{\Delta t_i}{T}, 0 \leq \varphi_i \leq 1$$

where Δt_i is the time elapsed since the reference foot was set down, and T is the complete cycle time. [45]

3.3. Design Method

Leg design of a robot plays a critical role. A four bar mechanism is used as the leg, because it is relatively easy to design. In leg design processes of robots, if leg is decided to be driven by mechanisms, path generation task of mechanisms comes into the picture. The main objective of the designed leg is to stabilize robot motion and provide energy efficiency. Therefore, the leg should provide a restricted motion in the vertical axis. In order not to move in the vertical axis, tip of the leg connected to the coupler of mechanism should remain in a minimum range in the very same axis of robot. Straight-line like mechanism is a good candidate to achieve that purpose.

These mechanisms can be reorganized with appropriate parameters and integrated into robot as its leg mechanism. However, straight-line path is not only constraint that the mechanism shall provide, but it also should have a reasonable transmission angle range during crank's complete rotation. Present straight-line mechanisms can be modified from their scalable parameters such as link lengths, but transmission angle is not a scalable parameter. In other words, under circumstances that transmission angle does not meet the required force conditions, it is impossible to change it without disarrange the entire mechanism.

3.3.1. Method I: Four Position Synthesis and Optimization

Position synthesis is a useful tool in path following applications of mechanisms. The reason why four position synthesis is done rather than five position synthesis is that four position synthesis gives plenty of mechanism options, whereas five position synthesis recommends a single solution. To obtain a motion close to straight-line, four positions are determined as shown in Figure 3.13:

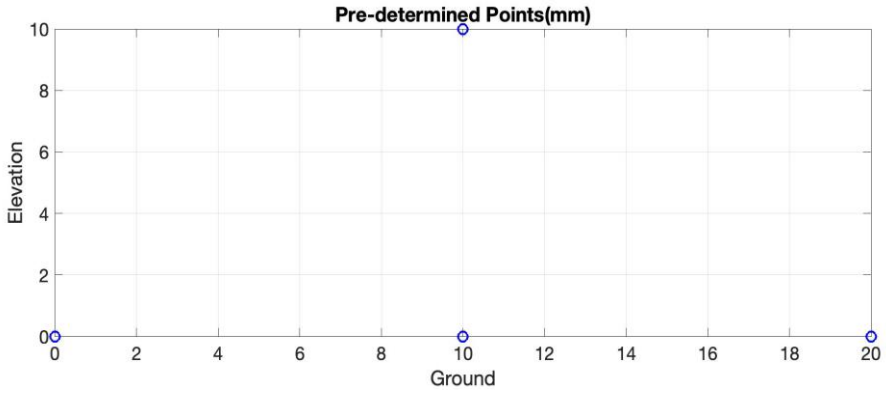

Figure 3.13. Pre-determined precision points.

Pre-scribed four points are: (0,0), (10,0), (20,0) and (10,10). 3 points are determined to be on a straight line. These are user-determined points, so they can be arranged according to the problem.

After positions are determined, four position synthesis code is run to obtain Burmester Curve in Matlab. Coupler angles are given according to foreseen motion of mechanism. A sample Burmester curve is given in Figure 3.14.

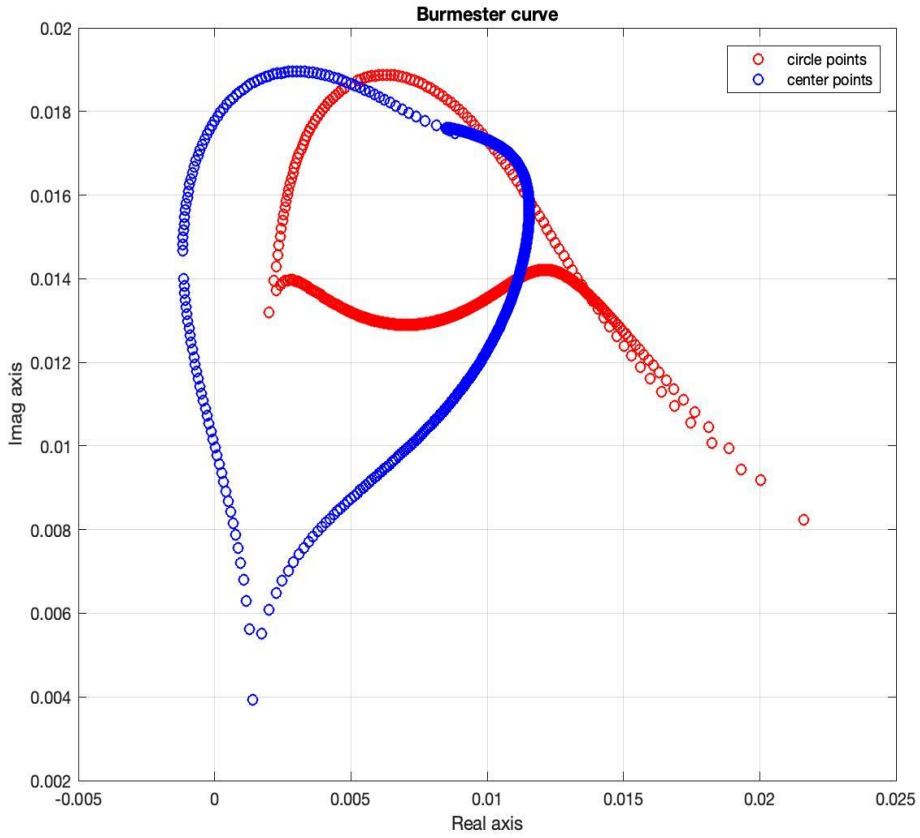

Figure 3.14. Sample Burmester Curve

In Figure 3.14, there are two types of points represented by two different colors. Circles with blue color are center points. Center points represent candidate coordinates for fixed joints of four bar mechanism. By selecting two of center points, one can determine the fixed joints of desired mechanism. Circles with red color are circle points. Circle points represent candidate coordinates for mobile joints of four bar mechanism. By selecting two of circle points, one can determine the mobile joints of desired mechanism.

Even if it is possible to control the points tip of leg passes, it is not possible to control the rest of the motion of leg. At this point, motion obtained with four position synthesis needs to be optimized. For optimization, Matlab's fmincon function is preferred because of its nonlinear constraint availability. Nonlinear constraints allow

user to restrict the solution to any region that can be described in terms of smooth function.

Nonlinear constraints are necessary to obtain a working mechanism at the end of the algorithm. Therefore, Grashof condition allowing the full crank rotation is defined into algorithm. To obtain a crank-rocker mechanism the sum of lengths of shortest and longest links should be smaller than the sum of lengths of remaining links.

Algorithm requires initial conditions and a desired path from the user to result in a solution. Pre-defined mechanism parameters constitute initial conditions. These initial conditions are defined randomly by using Matlab's random number creators in order to automatize the process. What algorithm does is to minimize the difference between the coordinates of desired path and resulting path. Desired path is given in Figure 3.15:

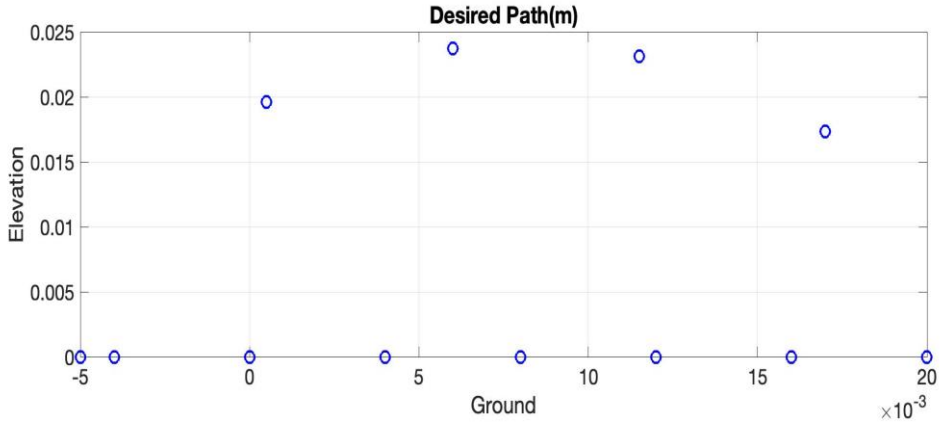

Figure 3.15. Desired Path Coded in Optimization Algorithm

The algorithm is prepared and several trials are done with it. The algorithm does not seem to be efficient to synthesize a mechanism by optimization of kinematic synthesis because of two reasons. First of all, it takes plenty of time for algorithm to give a single mechanism result. Even with a computer with a 7. generation Core i7, it takes approximately 42 hours to result in single mechanism. Secondly, the method does not guarantee that the resulting mechanism will not be affected from branch and order defects whose details are given in chapter 3.2.1. Therefore, to obtain a working

mechanism without any branch or order defect, an unknown number of trials should be done by using that algorithm. Together with the fact that algorithm takes hours to find a single mechanism, this method is eliminated.

3.3.2. Method II: Position Analysis and Optimization

Compared to position synthesis, position analysis has simpler equations. Therefore, it takes less computational time to come up with a solution. In its optimization process, genetic algorithm is preferred rather than fmincon. After one run, it is possible to try different mechanisms, which accelerates to find a mechanism showing no branch or order defects. Besides, genetic algorithm does not require any initial condition to start the algorithm.

Similar to fmincon, genetic algorithm also accepts nonlinear constraints. Therefore, Grashof condition is plugged in algorithm once again.

In the second prototype, an additional objective function is defined to genetic algorithm controlling vertical displacement of center of mass of robot.

Once the parameters of the algorithm have been determined, the path that the genetic algorithm should follow is determined as shown in Figure 3.16.

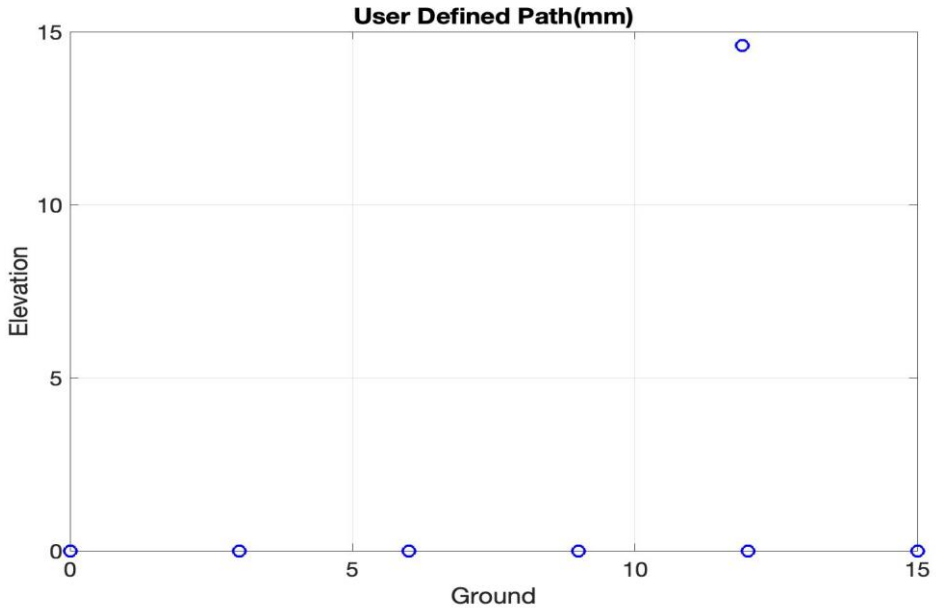

Figure 3.16. User Defined Path for Genetic Algorithm

It takes approximately two minutes to come up with 80 different mechanism solutions for genetic algorithm. The most useful mechanism is selected among them.

CHAPTER 4

DESIGN OF BULUBOT

4.1. Introduction

The robot is named Bulubot. Bulubot consists of two prototypes. Throughout the chapter, properties of both prototypes are told in detail and they are compared with each other.

4.2. Design of First Prototype

Methods expressed in previous chapter are used to design an applicable leg for Bulubot. Leg design takes a key role in determining the size of whole body. Details of design of first prototype are underlined in the rest of the chapter.

4.2.1. Leg Design of First Prototype

In the first prototype, two objective functions are defined for the genetic algorithm. One is to follow a path as close as to the user-defined path. Second is to keep transmission angle deviation from ninety degrees as less as possible. Objective functions are:

$$f(x) = (x_{desired} - x_{obtained})^2 + (y_{desired} - y_{obtained})^2$$

$$f(y) = \left| \frac{\pi}{2} - \frac{\cos^{-1}({a_3}^2 + {a_4}^2 - {a_1}^2 - {a_2}^2 + 2a_1 a_2 \cos\theta_{12}}{2a_3 a_4} \right|$$

By combining position analysis of four-bar with Matlab genetic algorithm optimization tool, a four-bar mechanism is designed as the first prototype. Genetic algorithm optimization results in 80 different solution. Some of these results is given in Table 4.1:

Table 4.1. Genetic Algorithm Results

Total path deviation	**Total transmission angle deviation**
357.8342	0
4.2503	1.1637
8.6911	0.9999
11.7	0.7694
253.4442	0.0768
10.5825	0.863
4.5581	1.149
7.5555	1.0884
7.8296	1.0873
8.7012	0.997

There is a trade-off between path deviation and transmission angle. In other words, the more path is accurate, the more transmission angle deviates. Therefore, coefficients are given to each objective function regarding to their importance and a decisive value is obtained.

$$Decisive\ value = (total\ path\ dev.) + 2(total\ transmission\ angle\ dev.)$$

Starting from the mechanism giving minimum decisive value, trials are done until a mechanism without branch or order defects is obtained.

A mechanism is designed with 10 mm crank, 15 mm rocker and 57.1 mm coupler. Leg simulation is done in Matlab as shown in Figure 4.1.

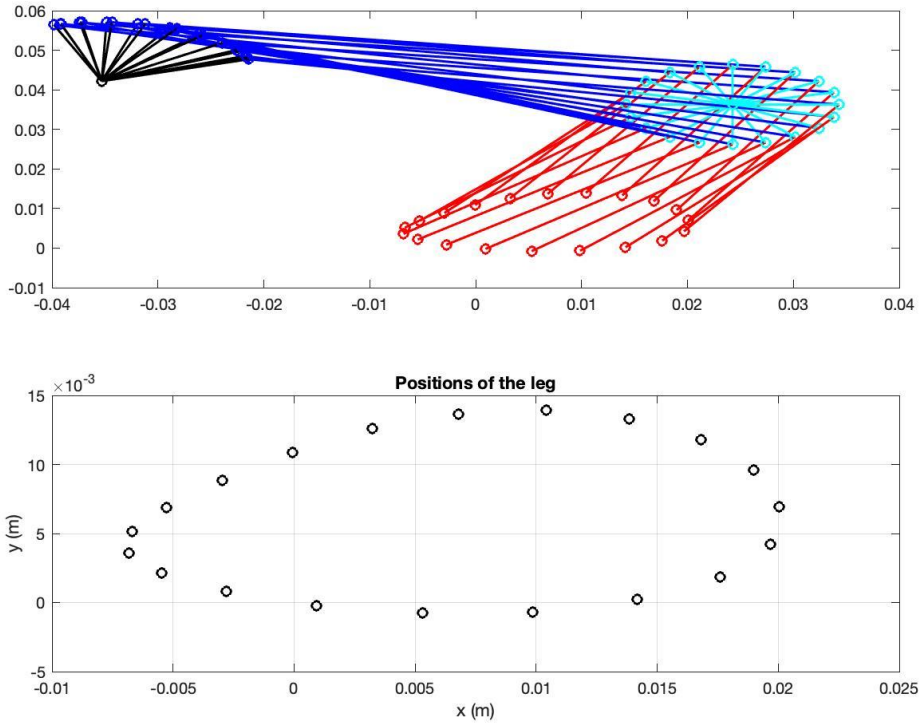

Figure 4.1. Leg Motion Simulation of First Prototype

As can be seen from Figure 4.1, the leg shows a motion close to the desired path meeting the first objective function constraint.

As for second objective function, transmission angle deviation is hold lower than 45 degrees. Minimum transmission angle observed is 70 degrees and maximum transmission angle observed is 125.3 degrees.

Even if displacement of center of mass in vertical axis does not appear as one of the objective functions, by the very nature of straight-line motion, the center of mass displacement turns out to be less than 13 mm of first prototype, which is a quite good value, compared to literature. Figure 4.2 displays the path of center of mass throughout the rotation of crank.

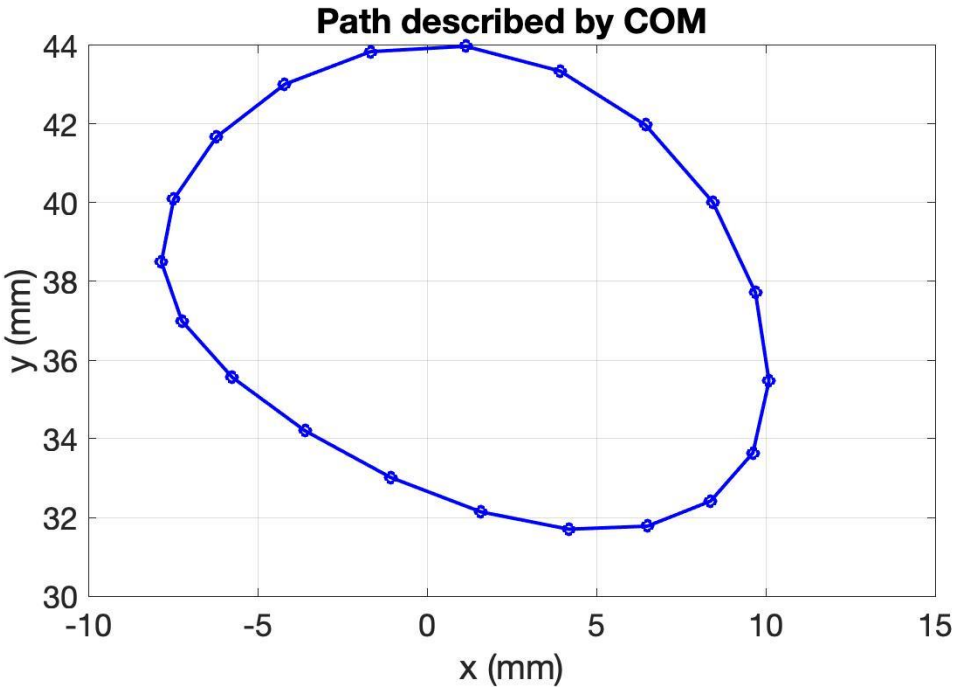

Figure 4.2. Center of Mass Displacement of First Prototype

4.2.2. Mechanical Design of First Prototype

The leg design of robot is the most important part of the mechanical design part, because it has a decisive role in dimensions of Bulubot. The designed leg is transformed into a solid model in Solidworks platform.

The leg assembly of robot consists of three parts. It has a slot shape crank making 360 degrees rotation. It has a slot shape rocker absorbing the energy coming from crank. Finally, it has a triangular coupler hosting the foot of Bulubot. In Figure 4.3, the general view of assembled leg is shown.

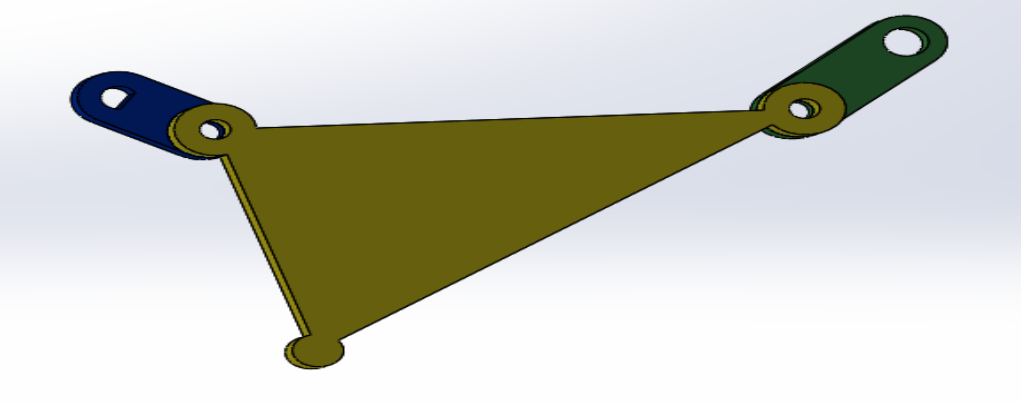

Figure 4.3. Leg Assembly of First Prototype

Coupler is connected to rocker and crank by two shafts. As shafts, M2.5 countersunk screws are used to eliminate any potential interference of shaft with the rest of the body. The countersunk screw detail of crank is illustrated in Figure 4.4. The one on the rocker is the same as the detail on crank.

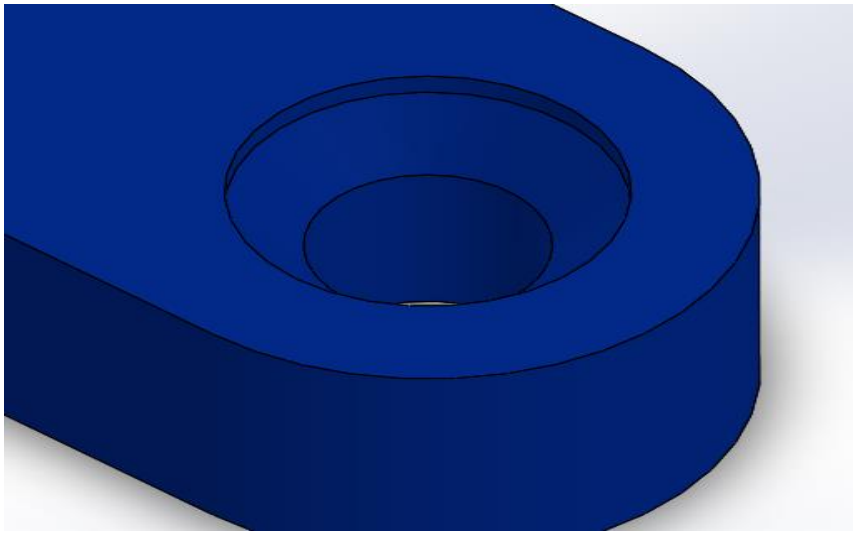

Figure 4.4. Countersunk Screw Detail

Crank of Bulubot is connected to the DC Motor whose details are given in the next section. Since the DC motor has a D-Cut interface, crank has a D-Cut detail on it as can be seen in Figure 4.5.

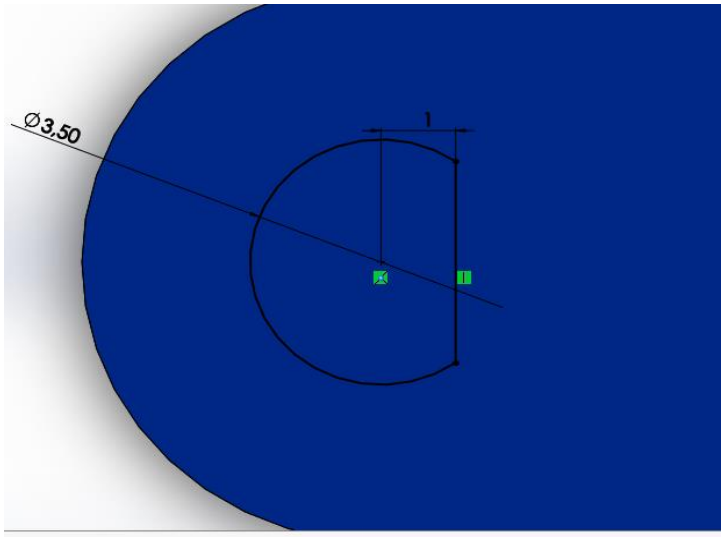

Figure 4.5. D-Cut for Motor Interference

Rocker of Bulubot is connected to a shaft integrated to the main body of Bulubot. It is a through hole interface shown in Figure 4.6.

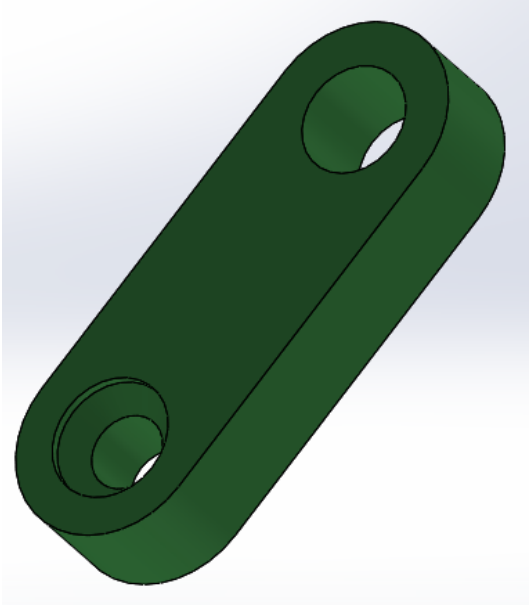

Figure 4.6. Body Interference of Rocker of First Prototype

The main body of Bulubot is designed as a rectangular prism having details regarding to the subparts assembled to it. Figure 4.7 shows a general view of main body.

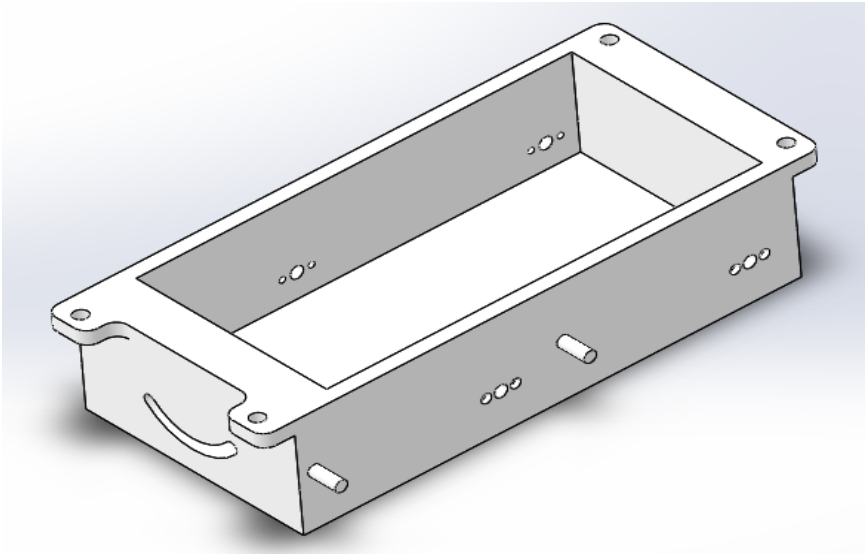

Figure 4.7. Main Body of Bulubot

Three holes, two of which are drilled with shoulders, are drilled for DC motor assembly. DC motors are assembled with M1.6 screws. Heads of screw are hidden into body through the shoulder details not to interfere with the motion of crank. Holes in between two shouldered holes houses shafts of DC motors. Figure 4.8, 4.9 and 4.10 demonstrate these details.

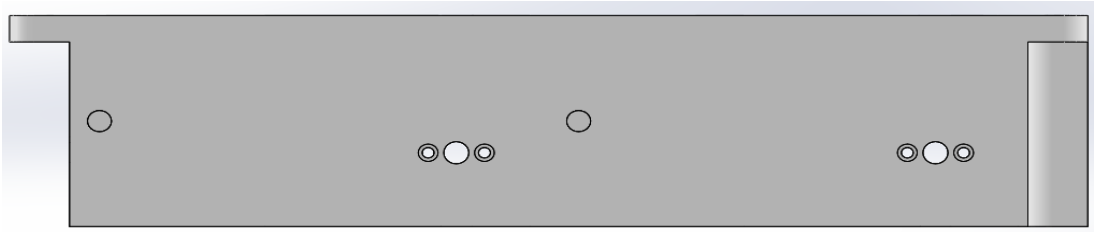

Figure 4.8. Side View of First Prototype

Figure 4.9. Motor Section Interface of Body of First Prototype

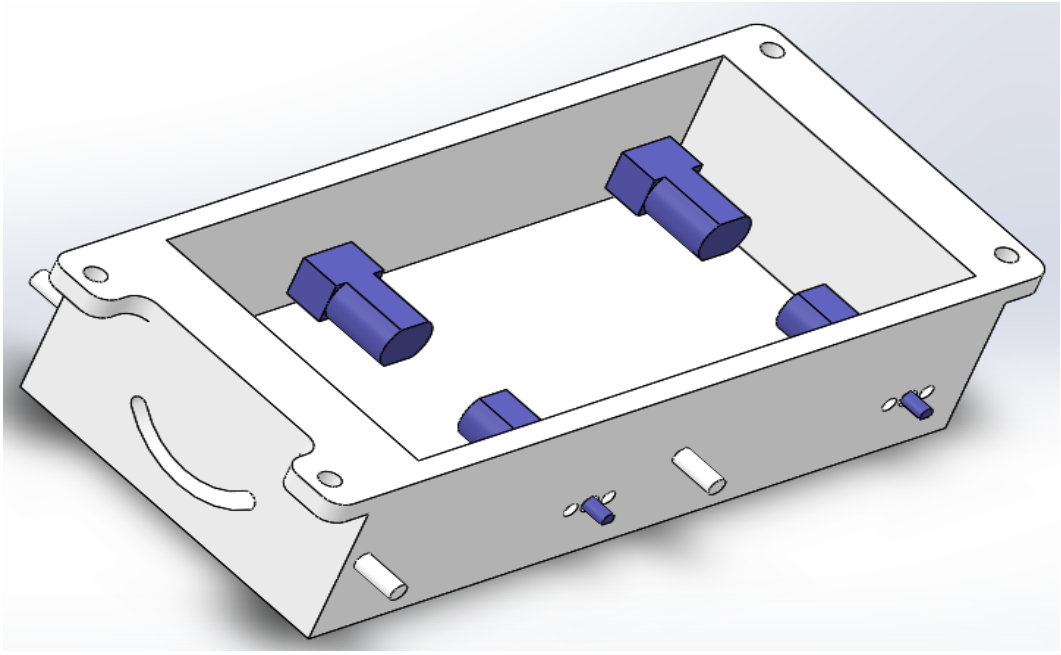

Figure 4.10. Motor Assembly Scenerio of First Prototype

The pins coming out of body are shafts of rocker. The pins and body are produced as one-piece solid part. Cranks are assembled to motors' shafts. Rockers and cranks are mounted to their shafts by interference fit. The motion of leg mechanisms is considered and no contact between each mechanism is provided. Figure 4.11 shows the relevant assembly.

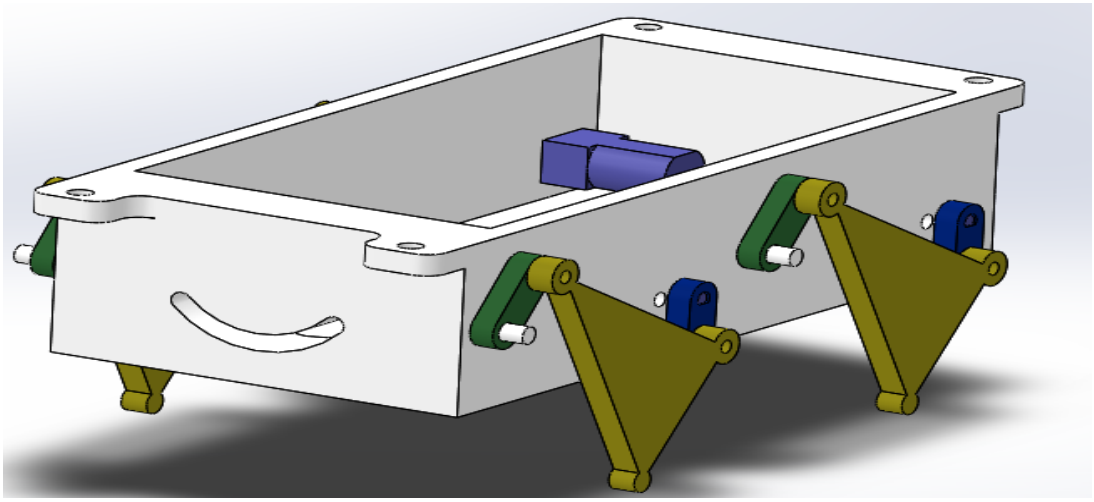

Figure 4.11. Leg-Body Assembly of First Prototype

A certain amount of space is left in the midsection of Bulubot in order to render cabling of motors and other electronic equipment possible. To provide a surface for electronics of Bulubot, a top cover is designed. It is assembled to main body from its corners with M4 screws. It has large slots for weight reduction and cabling. It is shown in Figure 4.12.

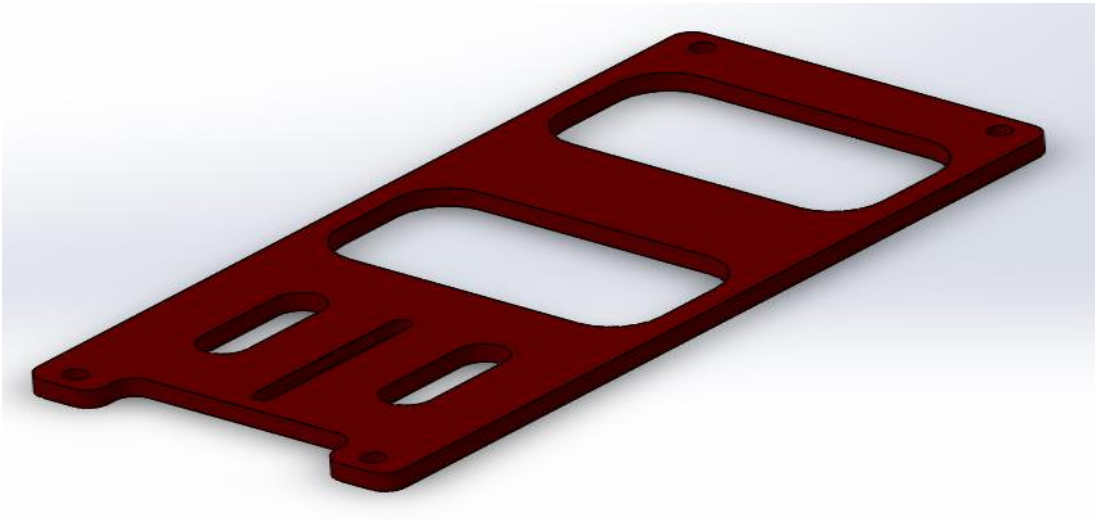

Figure 4.12. Top Cover of First Prototype

The thinnest slot locating at very front of top cover is designed to assemble sensor brackets. Sensor brackets are bought and modeled in 3D environment to simulate the assembly scenario. Figure 4.13 and 4.14 and illustrate both sensor bracket 3D model and top cover and sensor bracket assembly locations. Sensor brackets are mounted by M3 screws.

With sensor brackets, mechanical assembly is completed for Bulubot. Figure 4.15 indicates the final assembly.

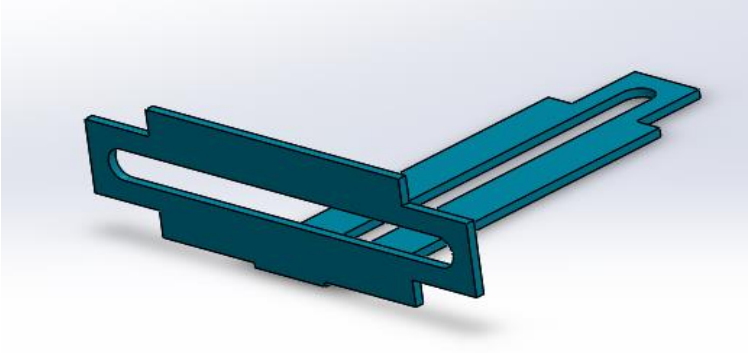

Figure 4.13. Sensor Brackets

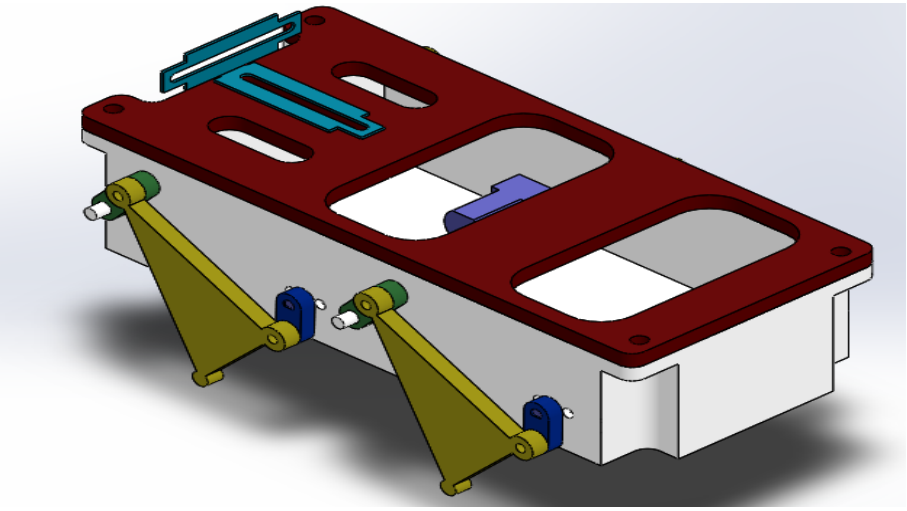

Figure 4.14. Sensor Brackets Assembly Scenerio

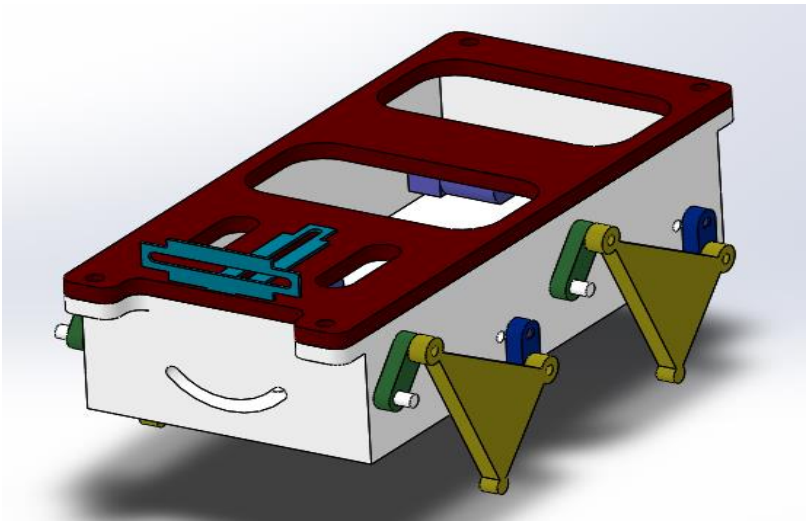

Figure 4.15. Final Assembly of First Prototype

The bill of material list of main assembly is given in the Table 4.2 for first prototype:

Table 4.2. Bill of Material of First Prototype

Item Number	Part Name	Quantity
1	Body	1
2	Top Cover	1
3	Crank	4
4	Rocker	4
5	Coupler	4
6	DC Motor	4
7	Sharp Sensor Bracket	1
8	M1.6 Screw	8
9	M4 Screw	4
10	M4 Washer	4
11	M4 Spring Washer	4
12	M4 Nut	4
13	M2.5 Countersunk Screw	8
14	M2.5 Washer	8
15	M2.5 Spring Washer	8
16	M2.5Nut	8

4.3. Design of Second Prototype

Although the first prototype has a solid structure, it has some fundamental problems. First of all, tip of its leg digs into ground causing sudden bounces during motion. Secondly, its size is large for a possible swarm algorithm. Also, it takes a lot of time to be produced because of its size.

Aforementioned problems lead to a new design. A new leg is designed to obtain a pure straight-line motion and minimum level center of mass displacement. The size of it is also considered to have a smaller robot than first prototype. Details of second prototype are told in the rest of the chapter.

4.3.1. Leg Design of Second Prototype

In the leg design process of second prototype, Matlab's genetic algorithm tool is used once again with one difference. Center of mass displacement is added to objective functions:

$$f(x) = (x_{desired} - x_{obtained})^2 + (y_{desired} - y_{obtained})^2$$

$$f(y) = \left| \frac{\pi}{2} - \frac{\cos^{-1}({a_3}^2 + {a_4}^2 - {a_1}^2 - {a_2}^2 + 2a_1 a_2 \cos\theta_{12}}{2a_3 a_4} \right|$$

$$f(z) = (x_{COMi} - x_{COMj})^2 + (y_{COMi} - y_{COMj})^2, \quad i = 1, \ j = [0,2\pi]$$

where state i represents the constant initial position of mechanism and state j represents angles from 0 to 360 degrees.

A 3D graph in Figure 4.16 is constituted from the final results.

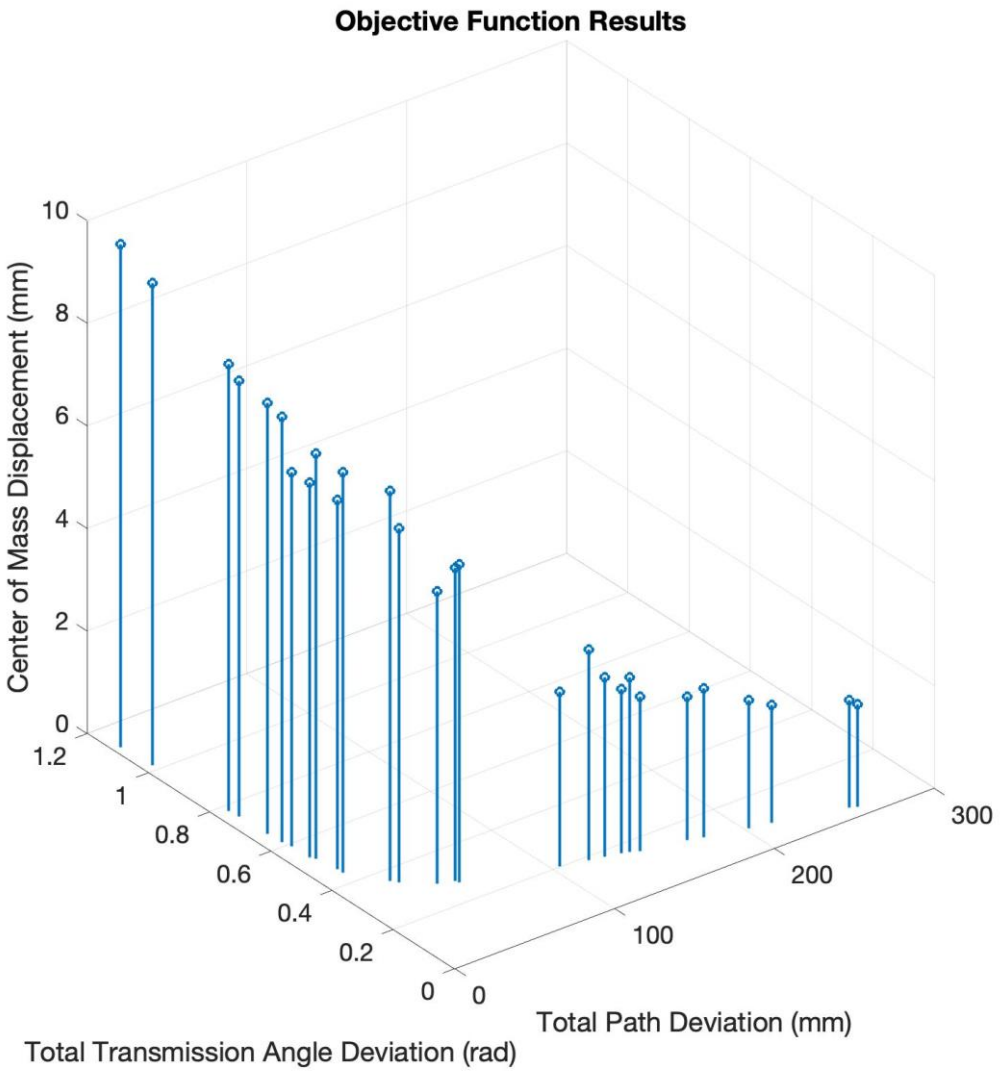

Figure 4.16. Objective Function Results of Second Prototype

The top left corner of graph is considered as the most important section of graph, because this is the region, where total path deviation and total transmission angle deviation are close to the zero. Therefore, point on the top left corner is selected as the leg mechanism.

A mechanism is designed with 7 mm crank, 12 mm rocker and 25.1 mm coupler. Leg is simulated in Matlab as shown in Figure 4.17.

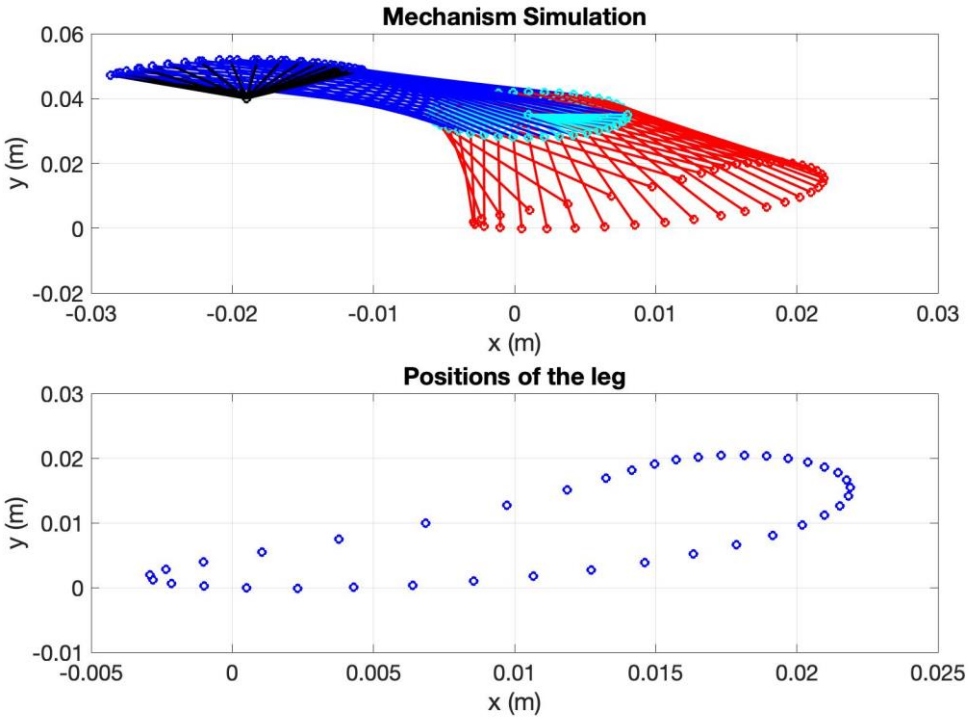

Figure 4.17. Leg Simulation of Second Prototype

Figure 4.17 shows that a straight-line mechanism is successfully obtained for second prototype meeting the first objective function.

As for transmission angle, minimum transmission angle becomes 50.7 degrees and maximum transmission angle becomes 112.2 degrees. Both angles deviate less than 45 degrees from 90 degrees meeting the second objective function.

Center of mass displacement is illustrated in Figure 4.18. Its displacement is less than 10 mm, which is a better value than the first prototype.

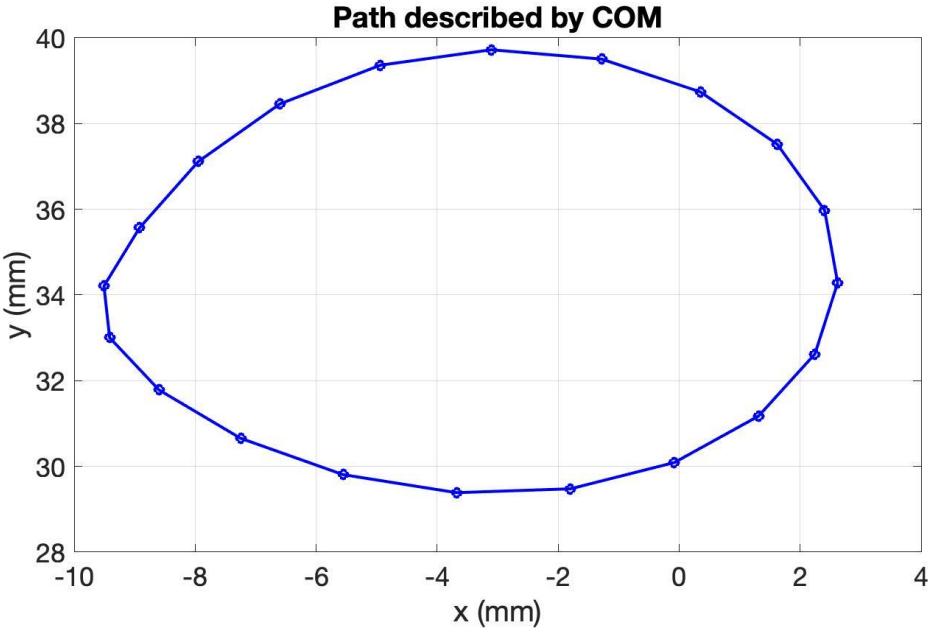

Figure 4.18. Center of Mass Displacement of Second Prototype

4.3.2. Mechanical Design of Second Prototype

In second prototype, dimensions of subparts and main assembly are different than first prototype. Assembly principle is same as first one.

Designed leg transformed into solid model with its three components. Leg assembly is shown in Figure 4.19

Figure 4.19. Leg Assembly of Second Prototype

Although crank and rocker have similar shapes to first production, design of coupler is modified. Rather than having a triangular shape, it has a curved, upright form seeming more like a leg. For mechanical balance, leg has a flat surface tip.

Since area covered by leg is diminished, the length of whole body shortens. However, because of the size of available motors, the width of body increases inevitably. Main body is lightened in new design. It has same rocker assembly interface in terms of hole dimensions, but the d-cut on crank increases in size, because a larger motor is used. Figure 4.20 illustrates new main body.

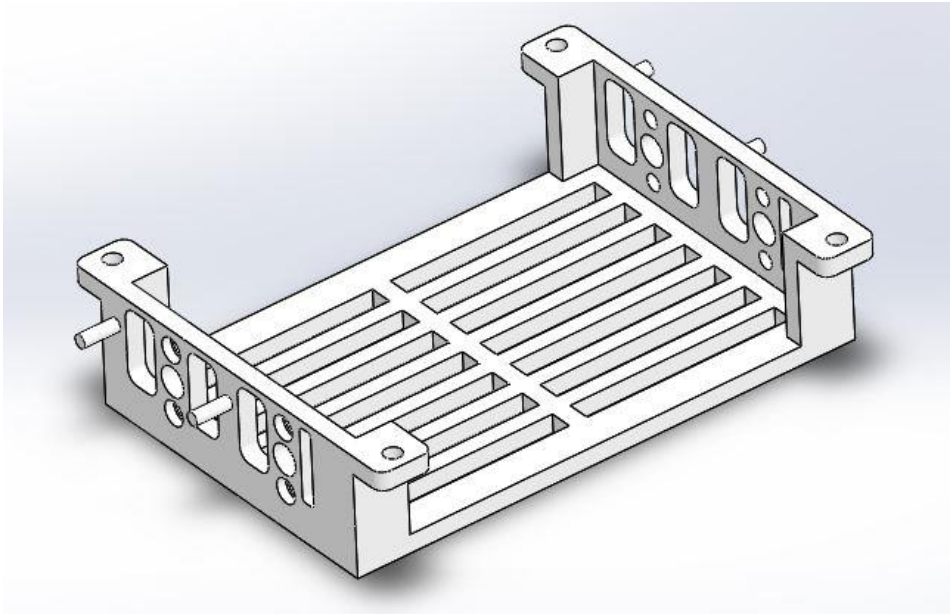

Figure 4.20. Body of Second Prototype

As it can be seen from Figure 4.21, same motor assembly principle as first prototype is embraced in second prototype. The only difference is that M3 screws are used in assembly rather than M1.6.

A top cover is designed for electronic equipment housing. Large slots are used in a pattern to ease cable in and out. Top cover is mounted with 4 M4 screws from corners of robot. Figure 4.22 shows the complete assembly.

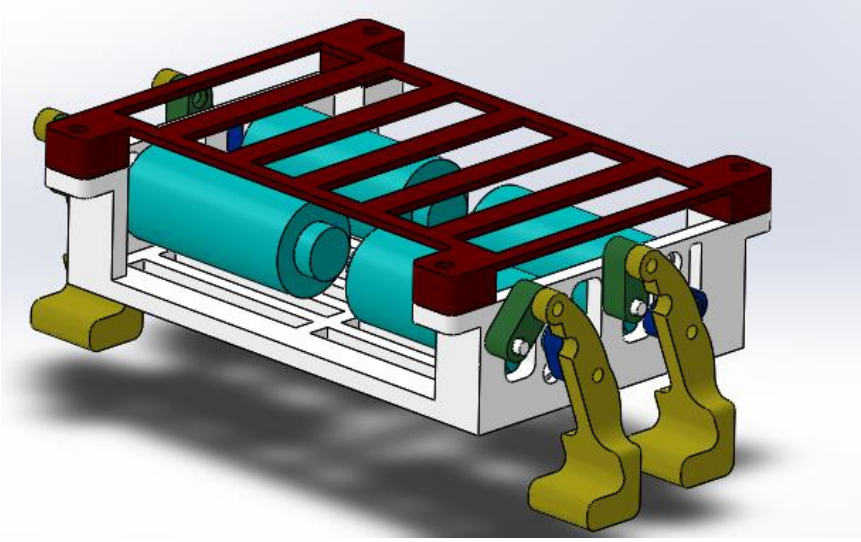

Figure 4.21. Main Assembly of Second Prototype

A sensor assembly interface is not used in second prototype, because the main objective of developing it is improving its general movement rather than its sensor responses.

The bill of material list of main assembly is given in the Table 4.3 for second prototype:

Table 4.3. Bill of Material of Second Prototype

Item Number	**Part Name**	**Quantity**
1	Body	1
2	Top Cover	1
3	Crank	4
4	Rocker	4
5	Coupler	4
6	DC Motor	4
7	M3 Screw	8
8	M4 Screw	4
9	M4 Washer	4
10	M4 Spring Washer	4

Table 4.3. Continued

Item Number	Part Name	Quantity
11	M4 Nut	4
12	M2.5 Countersunk Screw	8
13	M2.5 Washer	8
14	M2.5 Spring Washer	8
15	M2.5 Nut	8

4.4. Production of Bulubot

Bulubot is meant to be a low-cost robot meaning that it needs to be easily manufactured. Therefore, except its fasteners, all subparts of it are manufactured by 3D-printers.

Due to its availability in METU Kovan Research Labratory, Ultimaker 2+ 3D printer is utilized. It works parallel with a software belonging to same company called Cura. Figure 4.22 and 4.23 illustrates both the printer and software environment.

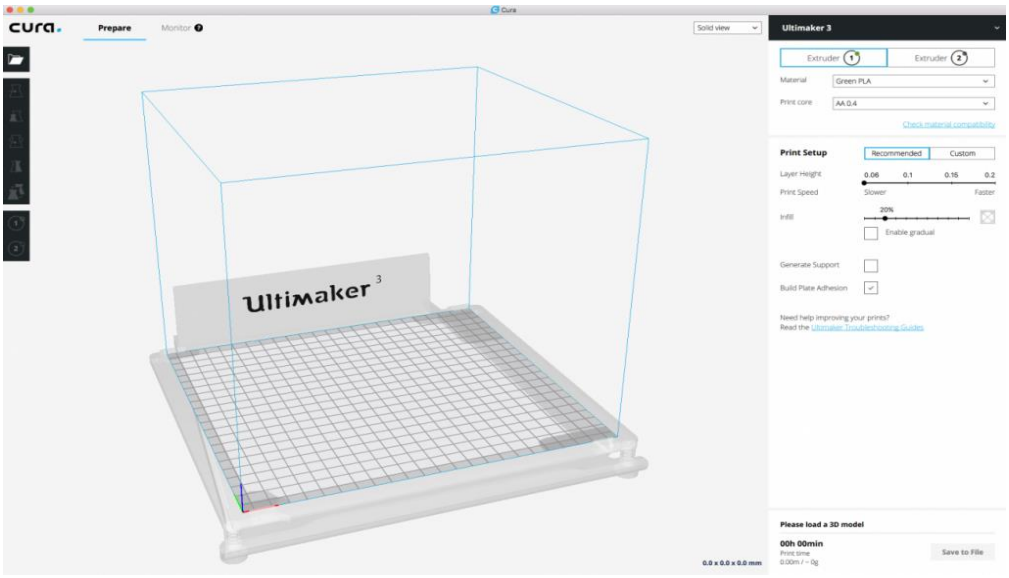

Figure 4.22. Cura Software Interface

Figure 4.23. Ultimaker 2+ 3D Printer

As 3D printer material, polylactic acid (PLA) is selected due to its tensile strength which is 14 MPa and more than the force to which Bulubot is exposed. Besides, it is easily accessible.

After completing the 3D model in Solidworks, parts are saved with .stl format with fine resolution option to get a product manufactured without defects. Then, parts are exported to Cura platform to simulate the position of part in 3D-printer and arrange final options before the printing process starts.

4.4.1. Production Time of First Prototype

Due to the lack of space on printer board, print operation is done with 2 steps. First body and top cover are printed, then the sub parts of leg are printed. Figure 4.24 and 4.25 indicate how much time it takes each step for first prototype.

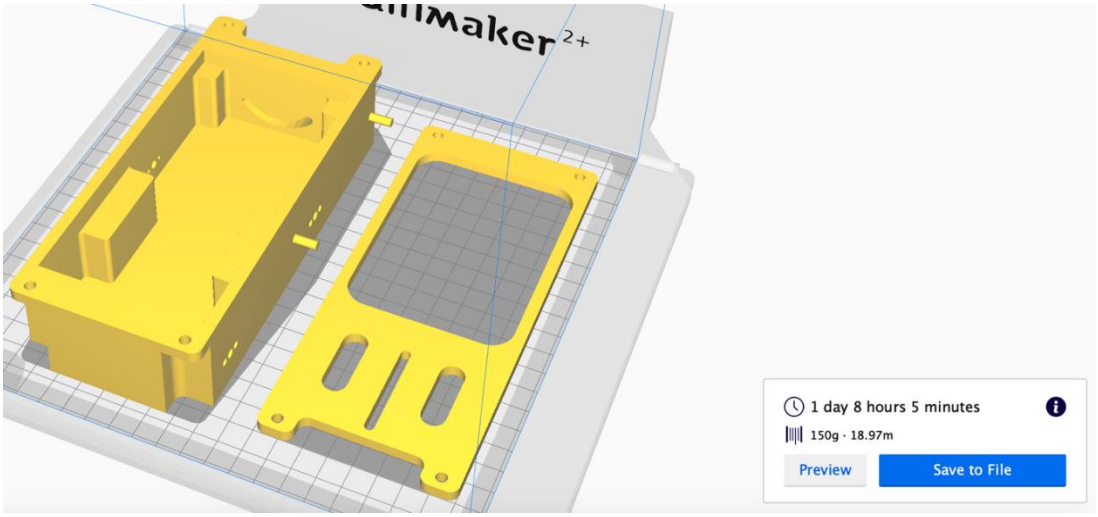

Figure 4.24. Time Elapsed for Top Cover and Body Manufacture of First Prototype

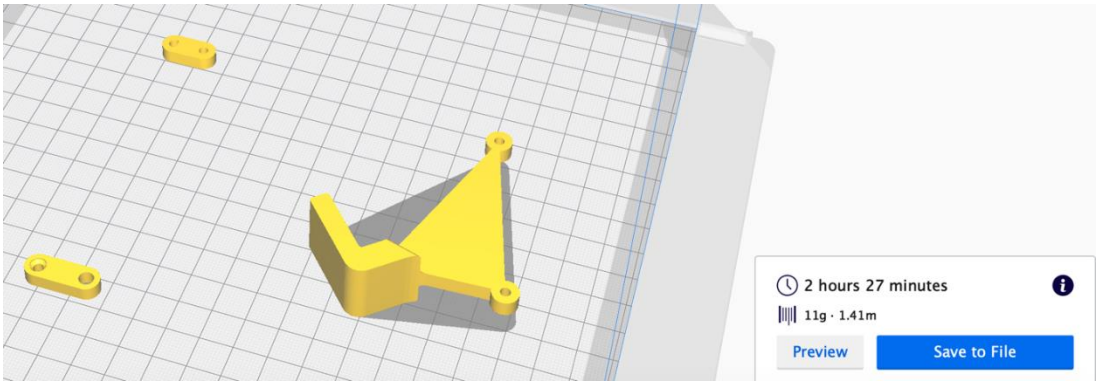

Figure 4.25. Time Elapsed for Leg Subparts Manufacture of Second Prototype

As it can be seen from figures above, it takes approximately 35 hours to produce all parts of first prototype.

4.4.2. Production Time of Second Prototype

Print operation is done in two steps as first prototype. Leg subparts are printed with body this time. Top cover is printed separately. How much time required for printing operation is illustrated in Figure 4.26 and 4.27

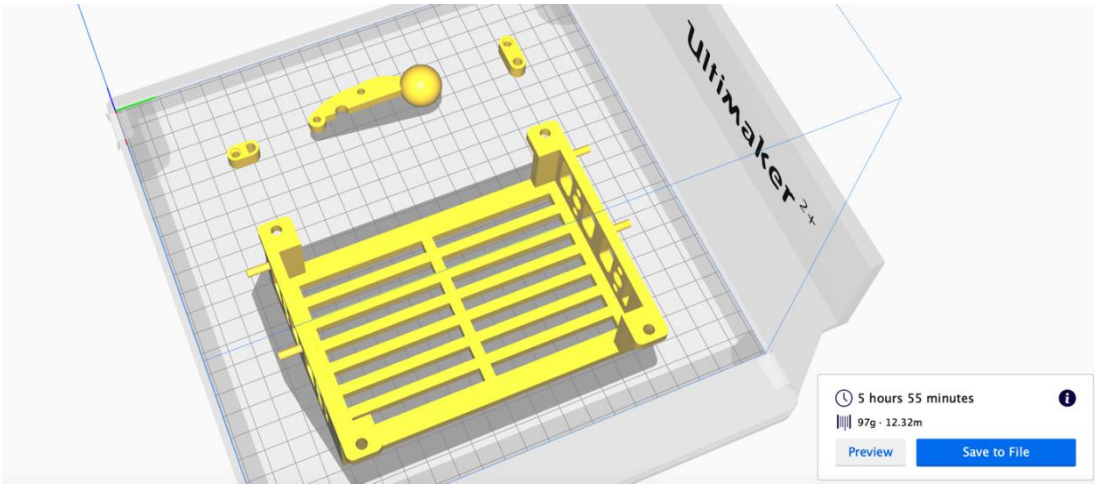

Figure 4.26. Time Elapsed for Body, Crank, Rocker and Coupler Manufacture of Second Prototype

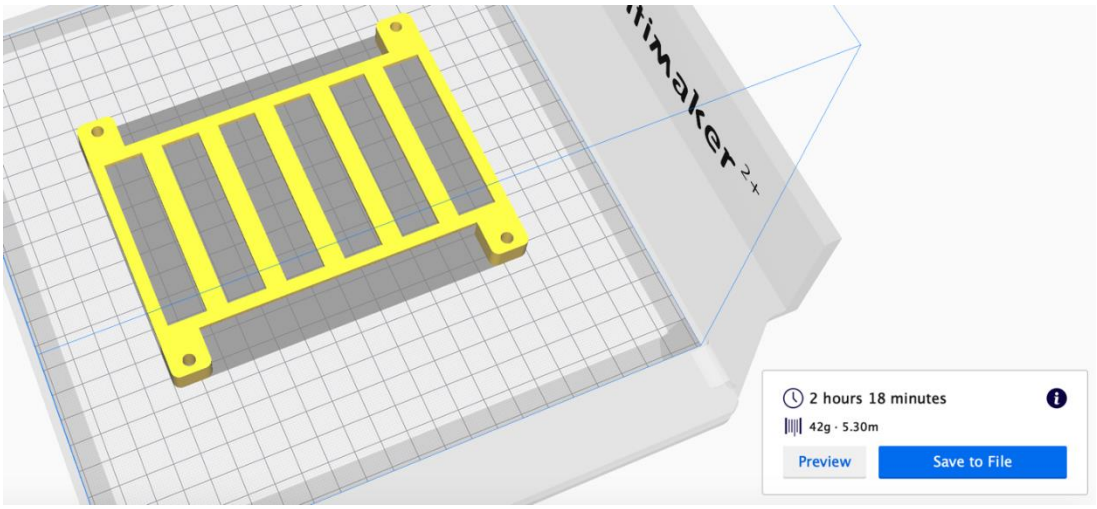

Figure 4.27. Time Elapsed for Body, Crank, Rocker and Coupler Manufacture of Second Prototype

Approximately 8 hours is necessary to produce the subparts of second prototype. It is less than one quarter of time elapsed for first prototype. A major enhancement is done in production stage of Bulubot.

4.5. Electronic Design of Bulubot

Electronic components are similar for both prototypes. Only motors are altered to have a better locomotion. Details of electronic design is given in the rest of the chapter.

4.5.1. Sensors

Two types of sensors are used for both prototypes of Bulubot. First one is Sharp GP2Y0A41SK type sensor which is an infrared proximity short range sensor. Sharp sensor is used for obstacle detection. Second one is a 10 mm light dependent resistor, which is used for light following. Figure 4.28 and 4.29 show sensors respectively.

Figure 4.28. GP2Y0A41SSK Type Short Range Sharp Infrared Proximity Sensor

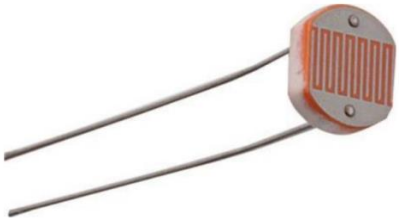

Figure 4.29. Light Dependent Resistor

4.5.2. Actuators

Actuators are mainly components driving the legs of robot. To make robot walk in a robust manner, it is desired to make it walk slowly. Therefore, in the first prototype, 4 DC motors working with 60 rpm and two TB6612FNG model DC motor drivers are utilized. Figure 4.30 and 4.31 illustrate the components respectively.

Figure 4.30. . 60 rpm DC Micro Motor

Figure 4.31. TB6612FNG DC Motor Driver

In the second prototype a different type of motor is decided to be used. An encodered motor is sought to obtain a harmonic locomotion. A deep motor research is done to find a motor in the size of the one used for first prototype. However, micro size encodered DC motors are all out of stock in Turkish basis websites. Therefore, a relatively bigger motor is compulsorily selected. A 6 Volt, 210 RPM encodered motor is used for second prototype as shown in Figure 4.32

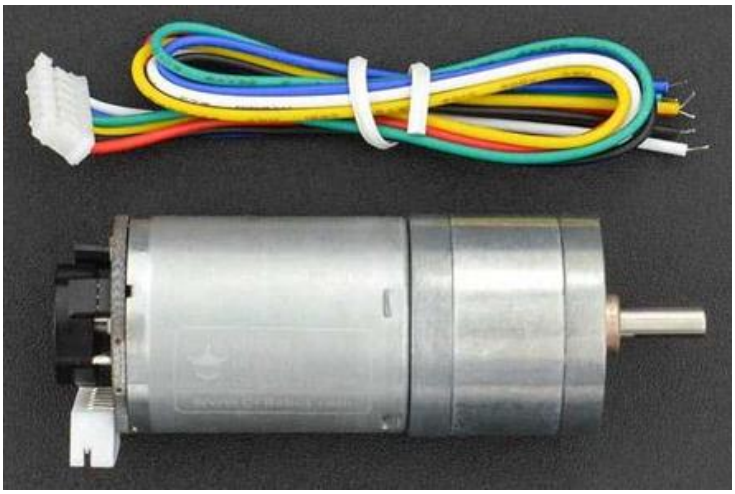

Figure 4.32. 6 Volts, 210 RPM DC Motor Driver

4.5.3. Controller

Controller is the brain of the robot. An Arduino Uno is used for both prototypes.

Figure 4.33. Arduino UNO

CHAPTER 5

EXPERIMENTAL SETUP AND RESULTS

5.1. First Prototype Experiments

After design stage is completed, production stage is started. In METU Kovan Laboratory, all parts of prototype are printed. By using necessary fasteners defined in Bill of Materials, assembly is done and walking experiments are tried. Figure 5.1 shows the complete assembly of first prototype.

Although robot is designed for rough terrains, experiments are done on a flat terrain within the bounds of possibility. Therefore, in order to increase friction between bottom of feet and surface, bottom of each foot is coated with emery. Secondly, because of the cable abundance on robot, top cover and sensor brackets could not be used on manufactured robot.

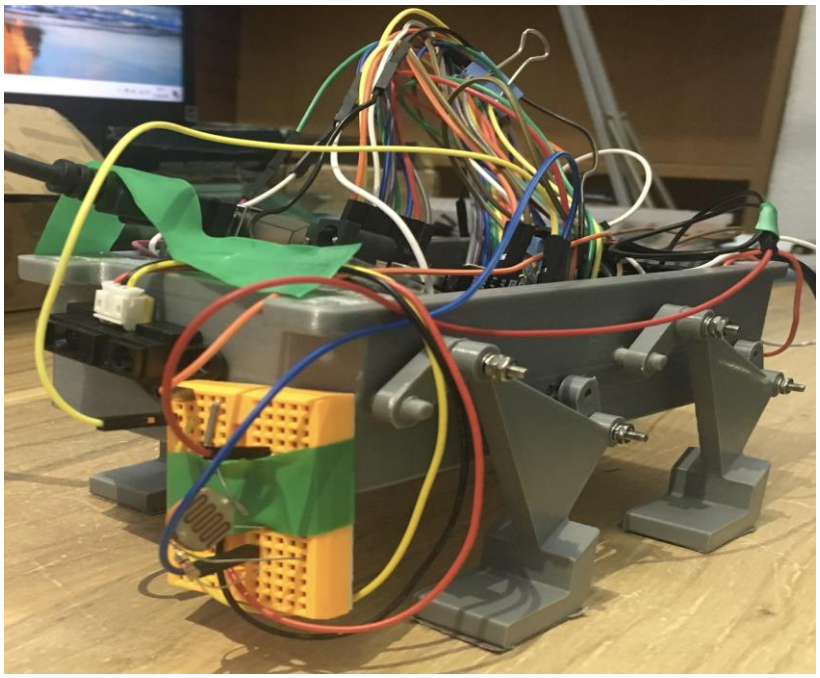

Figure 5.1. First Prototype of Bulubot

First prototype does not have an encoder, so it is not possible to read leg positions. Walking gait is controlled by recalibrating leg positions at the beginning of each trial. The first prototype has a gait similar to crawl gait, however, due to the lack of position knowledge, phase differences between legs could not be calculated. Therefore, each leg starts its rotation after the other leg completes its rotation.

5.2. First Prototype Experiment Results

First prototype of Bulubot succeeds to walk, avoid obstacles and follow a light source. First of all, because leg digs in ground, it disturbs Bulubot in vertical axis and Bulubot bounces suddenly during walking. Secondly, because a rigid gait algorithm could not be integrated into Bulubot, it realizes its straight motion by tottering. In other words, first it goes to left, then it goes to right to preserve its straight position.

5.3. Second Prototype Experiments

Second prototype is also manufactured in METU Kovan Research Laboratory. After parts are printed, assembly is done with the parts given in bill of materials. The complete assembly is indicated in Figure 5.2.

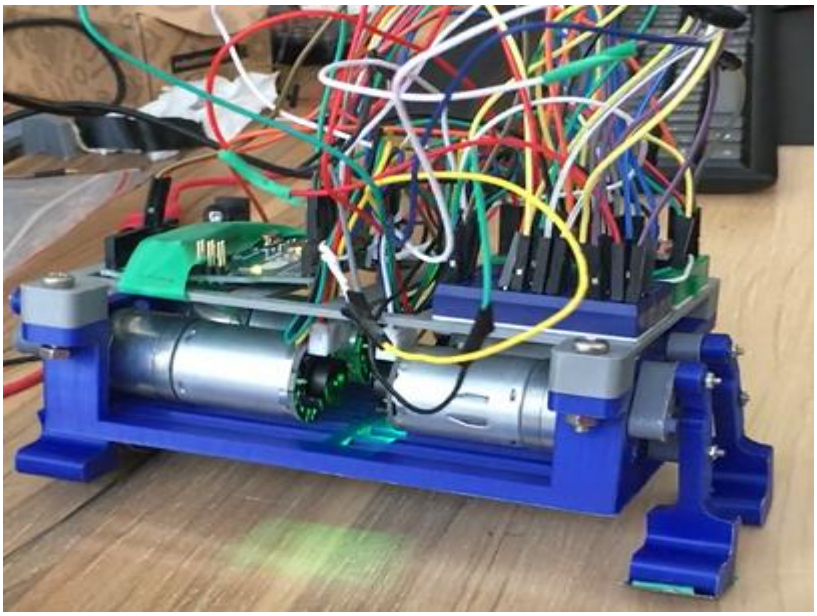

Figure 5.2. Second Prototype of Bulubot

Encodered motors are used in second prototype, so position data of each leg is accessible. Different than first prototype, gait algorithms are used in second prototype using position data coming from encoders. Robot is designed for walking, so trot gait is selected as the straight gait algorithm and crawl gait is selected for rotation. Gaits are explained under Background Information chapter. Left front leg is selected as the reference leg and rest of the legs are arranged with the necessary phase differences.

5.4. Second Prototype Experiment Results

Second prototype of Bulubot walks on a straight line with trot gait and rotates with crawl gait successfully. No sudden bounces are observed. Flat shaped foot hold the balance of robot during the motion. Second prototype also manages to avoid obstacles and follow a light source.

CHAPTER 6

DISCUSSION OF EXPERIMENTAL RESULTS

Experiments are done with two different prototypes. Their motion routines are investigated in detail. The most critical behaviors can be classified as:

1) Walking on a straight line and rotation
2) Center of mass displacement of Bulubot on vertical axis.
3) Following a light source and obstacle avoidance.

6.1. Walking on a Straight Line and Rotation

A walking robot should be able to walk on a straight line with minimum oscillation during its motion. The first prototype is problematic in that sense. It does not have any device on it giving feedback about legs' position, so legs cannot move in harmony with themselves. Each leg from each side of robot completes its rotation in order resulting in an irregular motion when robot walks. Bulubot preserves its straight-line motion by going towards left and right successively.

As for second prototype, Bulubot overcomes that problem. Encodered motors render gait algorithm usage possible. Legs move with harmony with correct crawl gait algorithm. As a result, Bulubot succeeds to stay on a straight line without dramatic oscillations.

6.2. Center of Mass Displacement

Center of mass displacement in vertical axis of whole body is directly related with the center of mass displacement of leg mechanisms. The leg designed for first prototype displaces less than 13 mm which is quite a good value. However; the motion of leg is not an exact straight-line motion, so it digs into ground by 5 mm. In the real time experiments of first prototype, this situation creates additional problem to its motion and Bulubot is exposed to sudden bounces after each leg touches to ground.

These bounces distort both proximity and light sensors resulting in low data collection performance from environment.

As far as second prototype is concerned, the leg mechanism of it has a better center of mass displacement than first prototype by 3 mm and the tip of has an exact straight-line motion. In real time experiments, since tip of leg does not dig into ground sudden bounces are not observed for second prototype. Therefore, its environment perception performance is better than first prototype.

6.3. Sensor Performance

Same sensors are utilized on both prototypes. Bulubot accomplishes avoiding obstacles and following a light source in all tests.

CHAPTER 7

CONCLUSION

In this thesis a walking robot, which is eligible for swarm applications is designed. As the first step, the most critical assets of a swarm robotic system are determined. Regarding to these criteria, design requirements of robot are specified.

After a deep literature research, type of robot, design methodology and manufacture process are decided. A walking robot study is conducted in this thesis in order to introduce a walking robot suitable for swarm algorithms to swarm robotics literature. For simplicity, a quadruped robot is decided.

In Chapter 2, how important sensing and signaling for swarm robotic systems is underlined. For accurate sensing, robot should be mobile as less as possible on vertical axis. This situation renders leg design so critical for walking robots. Therefore, design stage is started with designing a correct leg.

Legged robots in literature are investigated for a leg displacing minimum level on vertical axis and straight-line mechanisms turn out to be good candidates for this purpose. Famous straight-line mechanisms are scalable for different type of projects; however, their transmission angle values cannot be improved under unwanted circumstances. Therefore, a novel straight-line mechanism is decided to be designed. For simplicity, a four-bar mechanism is selected as the leg.

Four position synthesis is the first method to obtain a straight-line mechanism. The path leg follows shall be short for energy efficiency. Three points are selected on ground and one is selected somewhere above it. Although mechanism gives results close to straight line on the ground, the rest of the motion is not controllable. Resulting mechanisms follow long paths most of the time. Therefore, an optimization algorithm is required to optimize resulting path. Matlab fmincon function is utilized. Together with four position synthesis, it takes approximately 48 hours to obtain a single result

with that algorithm and it does not guarantee the resulting mechanism avoids branch or order defects.

As the second method, position analysis is utilized. With the geometrical relations in four bar mechanism, the path equation is obtained. Together with genetic algorithm optimization tool of Matlab, first protoype of leg is designed. Genetic algorithm gives at least 80 different solutions after a 2 minutes run. Therefore, the leg design method is set.

After leg design, mechanical and electronical design of Bulubot is done. Solidworks platform is used in mechanical design process. All parts are saved in .stl format to make it ready to be manufactured via 3D printer. Every single part of complete assembly is produced with polylactic acid (PLA) filaments. Acrylonitrile butadiene styrene material is not preferred, because its tensile and compressive strengths are less than PLA.First prototype tests do not give satisfying results, because of the insufficient motor and leg performances. In addition, its production process takes a lot of time. Therefore, a second prototype is designed.

In second prototype, micro DC motor used in first prototype is replaced with an encodered DC motor. A micro encodered DC motor could not found in market. New motor makes it possible to implement gait algorithms. Leg mechanism used in first prototype is replaced with a new leg mechanism obtained as the result of an upgraded genetic algorithm optimization method. Mechanical design of second prototype is also improved and the production time is decreased. Second prototype tests are completed successfully.

Production time is decreased to 8 hours approximately. However, it is possible to decrease production time more, if a micro encodered DC motor is used. In Figure 7.1 , the ideal robot design and its production time are illustrated. If a micro encodered DC motor had been found, Bulubot would have been manufactured approximately in 4 hours.

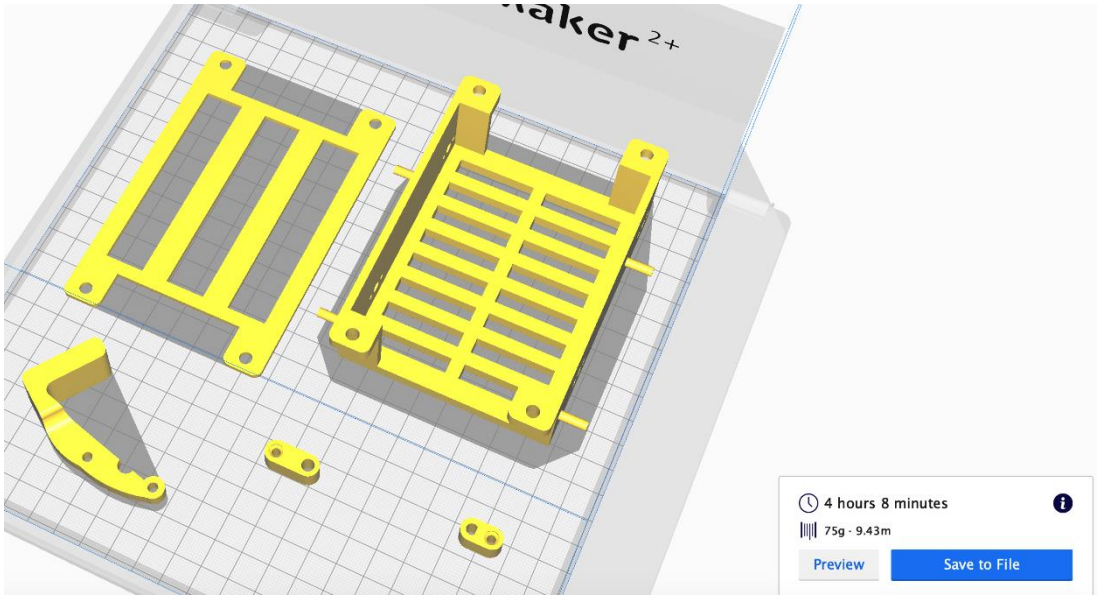

Figure 7.1. Production Time of Ideal Robot

A single robot is successfully designed, manufactured and tested in this thesis. However, there are still work to do on it. First of all, legs can be designed such that assembly process of it can be eliminated and a printed circuit board (PCB) can be replaced with breadboards. Second of all, it can be prepared for swarm applications.

Legs of Bulubot consist of three pieces and they are assembled by M2 countersunk screws. Rather than printing subparts and assembling them together, the complete assembly can be manufactured in single print. A print with soluble support material renders it possible.

Not only the cable abundance over Bulubot disturbs its motion sometimes, but it also makes electronic cabling process so difficult and complicated. A PCB design would ease hardware tests.

Finally, the main motivation of this thesis is to obtain a robot could be directly used in swarm applications. The electronic design of Bulubot should be upgraded to make it perceive environment better and communicate with its neighbors. Before real time tests, a swarm application algorithm can be prepared in a robot simulation

environment. After accurate simulation results, swarm algorithm can be test on a couple of robots.

REFERENCES

[1]. Şahin, E. *Swarm Robotics: From Sources of Inspiration to Domains of Application*. 2005. Berlin, Heidelberg: Springer Berlin Heidelberg.

[2]. Şahin, E., et al., *Swarm Robotics*, in *Swarm Intelligence: Introduction and Applications*, C. Blum and D. Merkle, Editors. 2008, Springer Berlin Heidelberg: Berlin, Heidelberg. p. 87-100.

[3]. Bayindir, L. and E. Sahin, *A review of studies in swarm robotics*. Vol. 15. 2007. 115-147.

[4]. K Parrish, J., S. Viscido, and D. Grünbaum, *Self-Organized Fish Schools: An Examination of Emergent Properties*. Vol. 202. 2002. 296-305.

[5]. Kaminka, G.A., R. Schechter-Glick, and V. Sadov, *Using Sensor Morphology for Multirobot Formations.* IEEE Transactions on Robotics, 2008. **24**(2): p. 271-282.

[6]. Reynolds, C.W., *Flocks, herds and schools: A distributed behavioral model.* SIGGRAPH Comput. Graph., 1987. **21**(4): p. 25-34.

[7]. Schultz, K.M., K.M. Passino, and T.D. Seeley, *The mechanism of flight guidance in honeybee swarms: subtle guides or streaker bees?* The Journal of Experimental Biology, 2008. **211**(20): p. 3287-3295.

[8]. Beni, G. *The concept of cellular robotic system*. in *Proceedings IEEE International Symposium on Intelligent Control 1988*. 1988.

[9]. Mondada, F., et al., *Swarm-Bot: A New Distributed Robotic Concept.* Autonomous Robots, 2004. **17**(2): p. 193-221.

[10]. Caprari, G. and R. Siegwart. *Mobile micro-robots ready to use: Alice*. in *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2005.

[11]. McLurkin, J. and J. Smith. *Distributed Algorithms for Dispersion in Indoor Environments Using a Swarm of Autonomous Mobile Robots*. 2007. Tokyo: Springer Japan.

[12]. Cianci, C.M., et al. *Communication in a Swarm of Miniature Robots: The e-Puck as an Educational Tool for Swarm Robotics*. 2007. Berlin, Heidelberg: Springer Berlin Heidelberg.

[13]. Turgut, A., et al., *Kobot: A mobile robot designed specifically for swarm robotics research*. 2007.

[14]. Wei, H., et al. *Sambot: A self-assembly modular robot for swarm robot*. in *2010 IEEE International Conference on Robotics and Automation*. 2010.

[15]. Kernbach, S. *Swarmrobot.org - Open-hardware Microrobotic Project for Large-scale Artificial Swarms*. arXiv e-prints, 2011.

[16]. Rubenstein, M., C. Ahler, and R. Nagpal. *Kilobot: A low cost scalable robot system for collective behaviors*. in *2012 IEEE International Conference on Robotics and Automation*. 2012.

[17]. McLurkin, J., et al., *A Low-Cost Multi-robot System for Research, Teaching, and Outreach*, in *Distributed Autonomous Robotic Systems: The 10th International Symposium*, A. Martinoli, et al., Editors. 2013, Springer Berlin Heidelberg: Berlin, Heidelberg. p. 597-609.

[18]. Karimi, M., et al. *WeeMiK: A low-cost omnidirectional swarm platform for outreach, research and education*. in *2016 4th International Conference on Robotics and Mechatronics (ICROM)*. 2016.

[19]. Patil, M., et al. *UB robot swarm — Design, implementation, and power management*. in *2016 12th IEEE International Conference on Control and Automation (ICCA)*. 2016.

[20]. Arvin, F., et al., *Mona: an Affordable Open-Source Mobile Robot for Education and Research.* Journal of Intelligent & Robotic Systems, 2018.

[21]. Baisch, A.T., P.S. Sreetharan, and R.J. Wood. *Biologically-inspired locomotion of a 2g hexapod robot*. in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2010.

[22]. Agheli, M., et al. *Design and fabrication of a foldable hexapod robot towards experimental swarm applications*. in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. 2014.

[23]. Kalat, S.T., et al. *TriBot: A minimally-actuated accessible holonomic hexapedal locomotion platform*. in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2015.

[24]. Adachi, H., N. Koyachi, and E. Nakano, *Mechanism and control of a quadruped walking robot.* IEEE Control Systems Magazine, 1988. **8**(5): p. 14-19.

[25]. Buehler, M., et al. *SCOUT: a simple quadruped that walks, climbs, and runs*. in *Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No.98CH36146)*. 1998.

[26]. Morrey, J.M., et al. *Highly mobile and robust small quadruped robots*. in *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No.03CH37453)*. 2003.

[27]. Kim, S., J.E. Clark, and M.R. Cutkosky, *iSprawl: Design and Tuning for High-speed Autonomous Open-loop Running.* The International Journal of Robotics Research, 2006. **25**(9): p. 903-912.

[28]. Ho, T., S. Choi, and S. Lee, *Development of a Biomimetic Quadruped Robot.* Journal of Bionic Engineering, 2007. **4**(4): p. 193-199.

[29]. Hoover, A.M., E. Steltz, and R.S. Fearing. *RoACH: An autonomous 2.4g crawling hexapod robot*. in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2008.

[30]. Birkmeyer, P., K. Peterson, and R.S. Fearing. *DASH: A dynamic 16g hexapedal robot*. in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2009.

[31]. Soltero, D.E., et al. *A lightweight modular 12-DOF print-and-fold hexapod*. in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2013.

[32]. Karakadıoğlu, C., M. Askari, and O. Özcan. *Design and operation of MinIAQ: An untethered foldable miniature quadruped with individually actuated legs*. in *2017 IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*. 2017.

[33]. Laschi, C., et al. *Design and Development of a Legged Rat Robot for Studying Animal-Robot Interaction*. in *The First IEEE/RAS-EMBS International Conference on Biomedical Robotics and Biomechatronics, 2006. BioRob 2006.* 2006.

[34]. Breckwoldt, W.A., et al. *Walking inverted on ceilings with wheel-legs and micro-structured adhesives*. in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2015.

[35]. Lohmann, S., et al., *Aracna: An Open-Source Quadruped Platform for Evolutionary Robotics*. 2012. 387-392.

[36]. Hernández, A., et al., *A Home-Made Robotic Platform based on Theo Jansen Mechanism for Teaching Robotics*. 2016.

[37]. Dupeyroux, J., et al., *Hexabot: a small 3D-printed six-legged walking robot designed for desert ant-like navigation tasks*. 2017.

[38]. Erdman, A.G., *Advanced Mechanism Design: Analysis and Synthesis*. 1984.

[39]. Akçalı, İ.D. and M.A. Arıoğlu, *Geometric design of slider-crank mechanisms for desirable slider positions and velocities.* Forschung im Ingenieurwesen, 2011. **75**(2): p. 61-71.

[40]. Mallik, A.K., A. Ghosh, and G.n. Dittrich, *Kinematic analysis and synthesis of mechanisms*. 1994, Boca Raton: CRC Press.

[41]. Söylemez, E., *Mechanisms*. 2013: Middle East Technical University.

[42]. Dijksman, E.A., *Kempe's (focal) linkage generalized, particularly in connection with Hart's second straight-line mechanism.* Mechanism and Machine Theory, 1975. **10**(6): p. 445-460.

[43]. Cabrera, J.A., A. Simon, and M. Prado, *Optimal synthesis of mechanisms with genetic algorithms.* Mechanism and Machine Theory, 2002. **37**(10): p. 1165-1177.

[44]. Dülger, L., H. Erdogan, and M. Kütük, *Matlab's GA and Optimization Toolbox: A Fourbar Mechanism Application*. Vol. 2. 2014.

[45]. McGhee, R.B. and A.A. Frank, *On the stability properties of quadruped creeping gaits.* Mathematical Biosciences, 1968. **3**: p. 331-351.

APPENDICES

A. Matlab Codes of Genetic Algorithm and Leg Simulation

```
function F = GaRobot(dv)
clc;
close all;
r1 = dv(1);
r2 = dv(2);
r3 = dv(3);
r4 = dv(4);
cx = dv(5);
cy = dv(6);
x0 = dv(7);
y0 = dv(8);
theta0 = dv(9);
theta2 =[];
G = 1;
N = 7; % Number of population
for k = 10:1:(10+N-1)
   theta2(k-9) = dv(k);
end
Ctx = [11.9 linspace(0,17,6)];
Cty = [14.6 zeros(1,6)];
for i= 1:1:7
   A = (r1^2) + (r2^2) - (r3^2) + (r4^2) - (2*r1*r2*cos(theta2(i)));
   B = (2*r1*r4) - (2*r2*r4*cos(theta2(i)));
   C = -2*r2*r4*sin(theta2(i));
   if isreal (((-C - sqrt((C^2)+(B^2)-(A^2)))/(A-B)))
      theta4 = 2*atan(((-C - sqrt((C^2)+(B^2)-(A^2)))/(A-B)));
      x13 = -r2*cos(theta2(i)) + r1 + r4*cos(theta4);
```

```
        y13 = -r2*sin(theta2(i)) + r4*sin(theta4);
        theta3 = atan2(y13,x13);
        Cxr = r2*cos(theta2(i)) + cx*cos(theta3) - cy*sin(theta3);
        Cyr = r2*sin(theta2(i)) + cx*sin(theta3) + cy*cos(theta3);
        Cx(i) = cos(theta0)*Cxr - sin(theta0)*Cyr +x0;
        Cy(i) = sin(theta0)*Cxr + cos(theta0)*Cyr +y0;
        error(i) = ((Ctx(i) - Cx(i))^2) + ((Cty(i) - Cy(i))^2);
        t_angle(i)                    =                    acos(((r3^2)+(r4^2)-(r1^2)-
(r2^2)+(2*r1*r2*cos(theta2(i))))/(2*r3*r4));
        t_dev(i) = abs((pi/2)-abs(t_angle(i)));
    else
        t_dev(i) = 1000000000;
        error(i) = 1000000000;
    end
end
F(1) = sum(error)*G;
V1 = max(t_dev);
F(2) = V1*G;
end
function [c,ceq] = GaConst(dv)
r1 = dv(1);
r2 = dv(2);
r3 = dv(3);
r4 = dv(4);
ceq=[];
N = 7; % Number of population
for k = 10:1:(10+N-1)
    theta2(k-9) = dv(k);
end
c = [r1+r2-r3-r4; theta2(1)-theta2(2); theta2(2)-theta2(3); theta2(3)-theta2(4);...
```

```
    theta2(4)-theta2(5); theta2(5)-theta2(6); theta2(6)-theta2(7)]; % Grashof's rule
end
clear all;
clc;
close all;
rng(10,'twister')
Fitness_function = @GaRobot;
nvars = 16;
%Bounds
r1 = [1 inf];
r2 = [1 inf];
r3 = [1 inf];
r4 = [1 inf];
cx = [-inf inf];
cy = [-inf inf];
x0 = [-inf inf];
y0 = [-inf inf];
theta0 = [0 2*pi];
theta2 = [0 2*pi];
LB = [r1(1), r2(1), r3(1), r4(1), cx(1), cy(1), x0(1), y0(1),...
    theta0(1), theta2(1), theta2(1), theta2(1), theta2(1), theta2(1), theta2(1), theta2(1)];
UB = [r1(2), r2(2), r3(2), r4(2), cx(2), cy(2), x0(2), y0(2), ...
    theta0(2), theta2(2), theta2(2), theta2(2), theta2(2), theta2(2), theta2(2), theta2(2)];
numbers = 0.4;
for i = 1:1:length(numbers)
    options = optimoptions('gamultiobj','ParetoFraction',
numbers(i),'CreationFcn',{'gacreationuniform'},...
        'PopulationSize',200,'CrossoverFraction',0.8,'CrossoverFcn',
@crossoverintermediate,...
        'MaxStallGenerations',3000000,'MaxGenerations',200000000);
```

```
[dv fval exitflag] =
gamultiobj(Fitness_function,nvars,[],[],[],[],LB,UB,@GaConst,options)
end
for j = 1:1:length(fval)
r1 = dv(j,1);
r2 = dv(j,2);
r3 = dv(j,3);
r4 = dv(j,4);
cx = dv(j,5);
cy = dv(j,6);
x0 = dv(j,7);
y0 = dv(j,8);
theta0 = dv(j,9);
theta2 =[];
N = 7; % Number of population
for k = 10:1:(10+N-1)
   theta2(k-9) = dv(j,k);
end
for i= 1:1:7
   A = (r1^2) + (r2^2) - (r3^2) + (r4^2) - (2*r1*r2*cos(theta2(i)));
   B = (2*r1*r4) - (2*r2*r4*cos(theta2(i)));
   C = -2*r2*r4*sin(theta2(i));
   theta4 = 2*atan(((-C - sqrt((C^2)+(B^2)-(A^2)))/(A-B)));
   x13 = -r2*cos(theta2(i)) + r1 + r4*cos(theta4);
   y13 = -r2*sin(theta2(i)) + r4*sin(theta4);
   theta3 = atan2(y13,x13);
   Cxr = r2*cos(theta2(i)) + cx*cos(theta3) - cy*sin(theta3);
   Cyr = r2*sin(theta2(i)) + cx*sin(theta3) + cy*cos(theta3);
   Cx(i) = cos(theta0)*Cxr - sin(theta0)*Cyr +x0;
   Cy(i) = sin(theta0)*Cxr + cos(theta0)*Cyr +y0;
```

```
end
figure(j)
plot(Cx,Cy,'bo')
grid on
end
clear all % clears all variables and functions
clc % clears the command window and homes the cursor
close all % closes all the open figure windows
% Lengths of Bars
AB = 12e-3; % Length of Rocker
BC = 25.05e-3; % Length of Coupler
CD = 7e-3; % Length of Crank
CE = 28.15e-3; % Length of Leg
% Position of fix points
xA = -19e-3; % x position of node A point
yA = 40e-3; % y postition of node A point
xD = 1e-3; % x position of node D
yD = 35e-3; % y position of node D
theta = 152.06*(pi/180);
for phi = 0:pi/25:2*pi
    % Position of C
    xC = CD*cos(phi) + xD;
    yC = CD*sin(phi) + yD;;
    % Calculation of Position B
    eqnB1='(xC - xBsol)^2 +(yC - yBsol)^2 = BC^2';
    eqnB2='(xBsol - xA)^2 + (yBsol - yA)^2 = AB^2';
    solB = solve(eqnB1, eqnB2, 'xBsol, yBsol');
    xBpositions = eval(solB.xBsol);
    yBpositions = eval(solB.yBsol);
    % first component of the vector xBpositions
```

```
xB1 = xBpositions(1);
% second component of the vector xCpositions
xB2 = xBpositions(2);
% first component of the vector yCpositions
yB1 = yBpositions(1);
% second component of the vector yCpositions
yB2 = yBpositions(2);
if yB1 > yA
   xB = xB1;
   yB = yB1;
else
   xB = xB2;
   yB = yB2;
end
%Calculation of position of node E
eqnE1 = '( xEsol - xC )^2 + ( yEsol - yC )^2 = CE^2';
BE = sqrt((BC^2)+(CE^2)-(2*BC*CE*cos(theta)));
eqnE2 = '( xEsol - xB )^2 + ( yEsol - yB )^2 = BE^2';
solE = solve(eqnE1, eqnE2, 'xEsol, yEsol');
xEpositions = eval(solE.xEsol);
yEpositions = eval(solE.yEsol);
xE1 = xEpositions(1);
xE2 = xEpositions(2);
yE1 = yEpositions(1);
yE2 = yEpositions(2);
if xE1 > xC
   xE = xE1;
   yE = yE1;
else
   xE = xE2;
```

```
    yE = yE2;
  end
  % Graphic of the mechanism
  subplot(2,1,1)
  plot([xA,xB],[yA,yB],'k-o','LineWidth',1.5)
  hold on % holds the current plot
  plot([xB,xC],[yB,yC],'b-o','LineWidth',1.5)
  hold on
  plot([xC,xE],[yC,yE],'r-o','LineWidth',1.5)
  hold on
  plot([xC,xD],[yC,yD],'c-o','LineWidth',1.5)
  hold on
  subplot(2,1,2)
  plot(xE,yE,'-ko','LineWidth',1.5)
  hold on
  xlabel('x (m)'), ylabel('y (m)'), grid,...
  title('Positions of the leg'),...
end
```