

**İZMİR KATİP ÇELEBİ UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE AND
ENGINEERING**

DESIGN OF WIRELESS CONTROLLED ROBOTIC HAND



M.Sc. THESIS

Faruk Sanberk KIZILTAŞ

Department of Biomedical Technologies

**Thesis Advisors: Assist. Prof. Dr. Fatih Cemal CAN
Assist. Prof. Dr. Erkin GEZGİN**

DECEMBER 2017

**İZMİR KATİP ÇELEBİ UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE AND
ENGINEERING**

DESIGN OF WIRELESS CONTROLLED ROBOTIC HAND



M.Sc. THESIS

**Faruk Sanberk KIZILTAŞ
(Y140101001)**

Department of Biomedical Technologies

**Thesis Advisors: Assist. Prof. Dr. Fatih Cemal CAN
Assist. Prof. Dr. Erkin GEZGİN**

DECEMBER 2017

İZMİR KATİP ÇELEBİ ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

KABLOSUZ KONTROL EDİLEBİLEN ROBOTİK EL TASARIMI



YÜKSEK LİSANS TEZİ

**Faruk Sanberk KIZILTAŞ
(Y140101001)**

Biyomedikal Teknolojiler Anabilim Dalı

**Tez Danışmanları: Yrd. Doç. Dr. Fatih Cemal CAN
Yrd. Doç. Dr. Erkin GEZGİN**

ARALIK 2017

Faruk Sanberk KIZILTAŞ, a M.Sc. student of İzmir Katip Çelebi University Graduate School of Science and Engineering student ID **Y140101001**, successfully defended the thesis entitled “**DESIGN OF WIRELESS CONTROLLED ROBOTIC HAND**” which he prepared after fulfilling the requirements specified in the associated legislations, before the jury whose signatures are below.

Thesis Advisor : **Assist. Prof. Dr. Fatih Cemal CAN**

İzmir Katip Çelebi University

Thesis Co - Advisor : **Assist Prof. Dr. Erkin GEZGİN**

İzmir Katip Çelebi University

Jury Members : **Assoc. Prof. Dr. Gökhan KİPER**

İzmir Institute of Technology

Assist. Prof. Dr. Özgün BAŞER

İzmir Katip Çelebi University

Assist. Prof. Dr. Savaş ŞAHİN

İzmir Katip Çelebi University

Date of Submission: 01 December 2017

Date of Defense : 22 December 2017





To my family,



FOREWORD

Firstly, I would like to thank to my supervisors Assist. Prof. Dr. Fatih Cemal CAN and Assist. Prof. Dr. Erkin GEZGİN who shared their valuable information with me. Whenever I consult them, they spared their precious time and helped me with patience. I am also grateful to Mustafa Volkan YAZICI for helping me to connect and print mechanical parts of robotic hand.

I also would like to thank to Hacer TÜRETKEN to help me for sewing flexible sensors on gloves.

I am too much grateful to Gökçenur TÜRETKEN for being too much patient with me and supporting me.

I also would like to thank my dear Uncle Orhan KIZILTAŞ and Aunt Zeynep KIZILTAŞ to opening their doors during my study.

Finally, I would like to thank to my parents who raised and supported me until today. I know when I need help you will support me.

December 2017

Faruk Sanberk KIZILTAŞ

TABLE OF CONTENTS

FOREWORD	ix
ABBREVIATIONS	xiii
LIST OF TABLES	xv
LIST OF FIGURES	xvii
SUMMARY	xix
ÖZET	xxi
1. INTRODUCTION	1
1.1 Robotic Hands	2
1.2 Prosthesis Hands	7
1.3 Wireless Communication	11
1.4 Aim of the Study	13
2. DEGREES OF FREEDOM AND ANATOMY OF HUMAN HAND	15
2.1 Forearm	15
2.2 Bones of human hand and degrees of freedom	15
3. STANDARDS AND CONFIGURATION OF ELECTRONIC COMPONENTS	19
3.1 Standard Of IEEE 802.15.4 And Zigbee Protocol	19
3.1.1 Standard of IEEE 802.15.4	19
3.1.2 Zigbee.....	20
3.2 Zigbee Device Types And Network Topologies.....	21
3.2.1 Zigbee device types.....	21
3.2.2 Zigbee network topologies.....	22
3.3 Xbees And Configuration Of Xbees By Using X-CTU.....	23
3.3.1 XBee.....	23
3.3.2 Configuration of xbees by using X-CTU	27
3.3.2.1 Channel	27
3.3.2.2 Pan adress.....	27
3.3.2.3 My adress	27
3.3.2.4 Destination adress	28
4. DESIGN AND MANUFACTURING OF ROBOTIC HAND	29
4.1 Design Criteria and Component Descriptions.....	29
4.2 Designing and Manufacturing of the Robotic Hand	29
4.2.1 Finger and hand sizes	30
4.2.2 Tendon – wire systems.....	31
4.2.3 Kinematic analysis of fingers.....	35
4.2.4 Working Area of fingers	37
4.2.5 Inverse kinematic analysis of fingers.....	43
4.2.6 Controlling of robotic hand.....	44
5. CHARACTERIZATION AND CALIBRATIONS OF SENSORS	47
6. PROGRAMMING THE ROBOTIC HAND	65
6.1 Arduino	65
6.1.1 Arduino mega.....	65
6.2 The Used Programming Software	66
7. DESIGN OF ARDUINO CIRCUIT	67

8. EXPERIMENTS	71
9. CONCLUSION	75
REFERENCES	77
APPENDICES	81
APPENDIX A	82
APPENDIX B	83
APPENDIX C	84
APPENDIX D	85
APPENDIX E.....	91
CURRICULUM VITAE	95



ABBREVIATIONS

MCP	: Metacarpal Phalanx
DIP	: Distal Interphalanx
PIP	: Proximal Interphalanx Joints
DoF	: Degrees of Freedom
FE	: Flexion / Extension
AA	: Abduction / Adduction
CMB	: Carpal Metacarpal Bones
PWM	: Pulse Width Modulation
PAN ID	: Personal Area Network Identifier
WPAN	: Wireless Personal Area Network
CH	: Channel
DL	: Destination Low Address
MY	: 16-bit source Address
AT	: Transparent Mode
API	: Application Programming Interface
BD	: Baud Rate
BT	: Bluetooth
CSMA / CA	: Carrier Sense Multiple Access / Collision Avoidance
BPSK	: Binary Phase Shift Keying
QPSK	: Quadrature Phase Shift Keying



LIST OF TABLES

Table 1.1 : Developed robotic hands.....	7
Table 1.2 : Prosthesis hands for commercial purposes (general characteristics)[19]	10
Table 1.3 : Prosthesis hands for commercial purposes (kinematic characteristics)[19]	11
Table 2.1 :Degrees of freedom of human hand	18
Table 3.1 :Comparison of LR – WPAN with other wireless networks [41]	19
Table 3.2 :IEEE 802.15.4 High Level Characteristics [41]	20
Table 3.3 :Wireless Communication Networks [42,52].....	21
Table 3.4 :Comparison of ZigBee Devices [50].	22
Table 3.5 :XBee S1 versus XBee S2 [43]	24
Table 3.6 :Pins of XBee S2[43]	26
Table 3.7 :XBee configurations.	28
Table 4.1 :Finger sizes in length and thickness.....	31
Table 4.2 : Denavit – Hartenberg parameters	35
Table 4.3 : Some finger movements of robotic hand.	42
Table 5.1 :Analog output values of flexible sensor.....	48
Table 5.2 :Formula and graphic according to flat position	49
Table 5.3 :Analog output values of flexible sensor according to graphs	50
Table 5.4 :Formula and graphic according to $y = x^2$ equation	51
Table 5.5 :Formula and graphic according to $y = x^3 + 2x^2 + 3$ equation.....	52
Table 5.6 :Measurement results of every 10 degree.....	52
Table 5.7 :Formulas and graphics for every 10 degree angle	53
Table 5.8 :Measurement results of the equation $y = x^2/25$	55
Table 5.9 : Measurement results of the equation $y = x^2$	55
Table 5.10 : Formulas and graphics according to $y = x^2/25$ and $y = x^2$ equations	56
Table 5.11 : Measurement results of the equation $y = \frac{x^2}{8}$	59
Table 5.12 : Measurement results of the equation $y = \frac{x^2}{16}$	59
Table 5.13 : Formulas and graphics according to $y = x^2/8$ and $y = x^2/16$ equations	60
Table 7.1 :Electrical components on Arduino Mega 2560 (Transmitter)..	67
Table 7.2 :Electrical components on Arduino Mega 2560 (Receiver).....	68
Table 8.1 : Some of finger movements	73



LIST OF FIGURES

	<u>Page</u>
Figure 1.1 : Designed by Leonard Aydt [3]	2
Figure 1.2 : Designed by Louis G. Caron [4].....	3
Figure 1.3 : Designed by James F. Mullen [5].....	3
Figure 1.4 : Designed by UTAH / MIT [6].....	4
Figure 1.5 : Designed by Carl F. Ruoff [7].....	5
Figure 1.6 : Designed by Gaiser et al [8]	5
Figure 1.7 : Designed by Mouri et al [9].....	6
Figure 1.8 : Designed by Yamano and Maeno [10].....	6
Figure 1.9 : Designed by Theodore Opuszenski [15]	8
Figure 1.10 : Designed by Robinson George B [16].....	8
Figure 1.11 : Designed by George T. Pinson [17]	9
Figure 2.1a :Muscles on the forearm from bottom side [45].....	15
Figure 2.1b :Muscles on the forearm from top side [46].....	15
Figure 2.2 :Metacarpal joint motions (adduction and abduction,flexion, extension) [49].....	16
Figure 2.3a :Anatomy of Human Hand Bones [47].....	17
Figure 2.3b : Anatomy of Human Hand Joints [48]	17
Figure 3.1 : Zigbee network topology [43].....	22
Figure 3.2 :XBee usb adapter.....	25
Figure 3.3 :XBee breakout board	25
Figure 3.4 :X-CTU	27
Figure 3.5 :Venn diagram of XBee network [43]	28
Figure 4.1 :Index finger	30
Figure 4.2 :Palm sizes (mm)	31
Figure 4.3a : Tendon – wire systems n-tendon system with pretension spring [2] ..	32
Figure 4.3b : Tendon – wire systems n-tendon system with two opposed tendons [2].....	32
Figure 4.3c : Tendon – wire systems 2n-tendon system [2]	32
Figure 4.3d : Tendon – wire systems n+1 tendon system [2].....	32
Figure 4.4a : First part of index finger, view from bottom side.	33
Figure 4.4b : First part of index finger, view from top side.	33
Figure 4.5 : Assembly of robotic hand.....	34
Figure 4.6 : Palm holes for wires.	34
Figure 4.7 : Kinematic models of finger in X, Y plane	35
Figure 4.8 : Drawing trajectory of fingertips.	36
Figure 4.9a : Simulation of gripping a spherical body t=1	37
Figure 4.9b : Simulation of gripping a spherical body t=4.....	37
Figure 4.9c : Simulation of gripping a spherical body t=6	37
Figure 4.9d : Simulation of gripping a spherical body t=7.....	37

Figure 4.10 : Workspace of robotic index finger (0.1 radians).....	38
Figure 4.11 : Workspace of robotic index finger (0.01 radians).....	39
Figure 4.12 : First condition.....	39
Figure 4.13 : Second condition	40
Figure 4.14 : Combined in MATLAB first situation and second situation.....	40
Figure 4.15 : Dotted area is the difference between the 3 DoF and 2 DoF index finger working area	41
Figure 4.16 : Parameter of Index Finger.	43
Figure 4.17 : Wrist to position servos	45
Figure 4.18 : Assembling of servos and wrist.....	45
Figure 5.1 :The working principle of the flexible sensors [51].....	47
Figure 5.2 :Flexible sensor.....	49
Figure 5.3 : $y = x^2$ and $y = x^3 + 2x^2 + 3$ graphs.....	50
Figure 5.4 :Experimental setup without flex sensor.....	54
Figure 5.5a :Experimental setup with flex sensor ($y = x^2$).....	54
Figure 5.5b :Experimental setup with flex sensor($y = x^2/25$).....	54
Figure 5.6 :Blue = x^2 , Yellow = $\frac{x^2}{8}$, Red= $\frac{x^2}{16}$, Green = $\frac{x^2}{25}$	57
Figure 5.7 :Experimental setup without flex sensor ($\frac{x^2}{8}, \frac{x^2}{16}$).	58
Figure 5.8a :Experimental setup with flex sensor($y = x^2/8$)	58
Figure 5.8b :Experimental setup with flex sensor($y = x^2/16$).....	58
Figure 5.9 : All graphs of measurement results. Blue = x^2 , Yellow = $\frac{x^2}{8}$, Red= $\frac{x^2}{16}$, Green = $\frac{x^2}{25}$	61
Figure 5.10 : Flexible sensors on gloves	62
Figure 5.11a : Calibration position while fingers on flat position.	63
Figure 5.11b : Calibration position bending fingers from PIP.....	63
Figure 5.11c : Calibration position bending fingers from MCP.	63
Figure 5.11d : Calibration position as a fist.....	63
Figure 6.1 : Arduino Mega.....	65
Figure 6.2 : Arduino software	66
Figure 7.1 : XBee S2 wireless antenna connection to Arduino Mega 2560	68
Figure 7.2 :Flexible sensor connection to Arduino Mega.....	69
Figure 7.3 : Flexible sensor connection board (Transmitter Arduino).....	70
Figure 7.4 : Receiver Arduino Mega connection board.	70
Figure 8.1a : Diagram of design glove with flexible sensors	72
Figure 8.1b : Diagram of design Arduino Platform for the glove (transmitter)	72
Figure 8.1c : Diagram of design Arduino Platform for the robotic hand (receiver).	72
Figure 8.1d : Diagram of design robotic hand	72
Figure 8.2 : Grabbing an object.....	74

DESIGN OF WIRELESS CONTROLLED ROBOTIC HAND

SUMMARY

Thanks to the rapid technological development, robot usage in human life increases day by day. Various kinds of robots that are designed for different tasks aim to increase the quality of human life by replacing or reducing required human power in related areas.

Being able to interact robots from a distance by using wireless communication not only provide mobility and ease of use to the user but also promote the usage of robots into various areas.

Considering these, this study aims to design a lower degrees of freedom robotic hand with respect to the natural anatomy of the hand and control it by the help of wireless communication protocols. Throughout the study, nine degrees of freedom robotic hand was designed, finger motions were modelled and simulation studies were carried out. Also, giving brief information about IEEE 802.15.4 standard, known wireless communication types were compared and the most suitable one for the task was selected as XBee S2 wireless communication platform. Control of robotic hand was provided with five flexible sensors, which were sewn on the glove on each finger. In this thesis, for the first time, characterization studies about flexible sensor was carried out and before starting to control the robotic hand, calibration was made according to the working principle of the flexible sensors. Except thumb, two independent motion for each robotic finger were provided by taking two measurements from each flexible sensor. In addition, the robotic hand fingers were kept in the same position with the fingers of the human hand by using of servo motors.

Control of the robotic hand was decided to be carried out by using Arduino platform and XBee modules. Future works related with this study was also discussed and listed in conclusion part.



KABLOSUZ KONTROL EDİLEBİLEN ROBOTİK EL TASARIMI

ÖZET

Hızla ilerleyen teknoloji sayesinde, robot kullanımı insan hayatının içinde her geçen gün daha da fazla yer almaktadır. Tasarlanan farklı türdeki robotlar belirli alanlarda gereken insan gücünün yerini alarak veya onu hafifleterek insan yaşamının kalitesini arttırmayı hedeflemektedir.

Kablosuz iletişim ile robotlarla uzaktan etkileşimde bulunulabilmesi kullanıcıya kolaylık ve hareket kabiliyeti kazandırırken, aynı zamanda çeşitli alanlarda robot kullanımını teşvik etmektedir.

Buradan yola çıkarak, ilgili çalışmada doğal el anatomisine göre kısıtlı serbestlik dereceli bir robotik el tasarlanarak, kontrolünün kablosuz haberleşme protokolleriyle yapılması hedeflenmiştir. Çalışma içerisinde dokuz serbestlik dereceli robotik bir el tasarlanmış, parmak hareketleri modellenmiş ve benzetim çalışmaları yapılmıştır. Ayrıca, IEEE 802.15.4 standardı hakkında kısaca bilgi verilerek, bilinen kablosuz iletişim türleri karşılaştırılmış ve XBee S2 kablosuz iletişim platformunun bu çalışmanın hedeflerine en uygun platform olacağı belirlenmiştir. Robotik elin kontrolü, eldiven üzerine her bir parmağa denk gelecek şekilde dikilen beş esneklik sensörleriyle yapılmıştır. Daha önce yapılmamış olan esneklik sensörünün karakteristiği ile ilgili çalışmalar yapılmış ve robotik elin kontrolü başlamadan önce esneklik sensörlerinin çalışma prensibine göre kalibrasyon yapılmıştır. Başparmak hariç, her bir esneklik sensöründen 2 veri alınarak robotik elin parmaklarında iki bağımsız hareket sağlanmıştır. Aynı zamanda servo motorlar kullanılarak robotik elin parmaklarının, insan elinin parmaklarının pozisyonuyla aynı pozisyonda kalması sağlanmıştır.

Robotik el kontrolünün Arduino platformu ve XBee modülleri yardımıyla gerçekleştirilmesine karar verilmiş ve gelecekte yapılması planlanan çalışmalar sonuç bölümünde tartışılmış ve listelenmiştir.



1. INTRODUCTION

In 21st century, while people want to obtain a healthy and high quality lifestyle, they do not want to work in dirty, dangerous, and demeaning jobs. At this point, robots will alter our daily lives at least by being assigned such as helper, servant, and assistants to surgery operations in medical area, helpers in any search or rescue operations, diggers, transporters in dangerous situations and constructors or destructive [1].

Robotic area is a multi-disciplinary scientific field that is combined some systematic disciplines such as mechanical, computer, industrial, electrical and electronics engineering disciplines [1].

Firstly, mechanical design and construction of hand are directly related to mechanical engineering. Secondly, design of electronic circuits, sensors and batteries are related to electrical and electronics engineering. Thirdly, control of robot behavior via software is related to computer engineering. As a conclusion, a mixture of mentioned engineering disciplines is used to construct and control robotic hand.

The first robotic hands were designed as a three-arm gripper. Since it is understood that the grippers weren't as effective as human hands, anthropomorphic hand researches have been started considering the ease of production, low cost and easy control.

The knowledge of robotic hands and the efforts to grasping and manipulating operation of robotic hands were based on the 1970s. Robotic hand design and control is a problem. Many research groups worked on design and controlling of hand. If the robotic hand dexterity increases, used number of joints will also increase. This makes difficult to control the robotic hand. On the other hand, if used number of actuators is reduced and the degree of freedom is increased, the robotic hand will be lack of high power density and efficient actuation. In addition, researches have been done about how many fingers should be on hand and how many joints should be on a finger. As a result of these researches, it has been decided that there should be at least 3 fingers in one hand to grasp [2].

Usage of the tendon system, the actuators place behind the wrist. system provides lower inertia and lower friction than the other systems. At the same time, flexibility and low cost are other advantages. On the other hand, movement gaps is a problem for tendon-wire systems [2].

Researches about anthropomorphic robotic hands were based on 1910s [3-4]. Many robotic and prosthesis hand have been produced so far. Some of them are used for commercial purposes; some are used for research purposes.

1.1 Robotic Hands

Leonard Aydt designed one of the first robotic hands in 1910 [3]. He designed a robotic hand with tendon-spring. Extension movements of fingers were provided through springs. Fingers consist of three phalanxes.

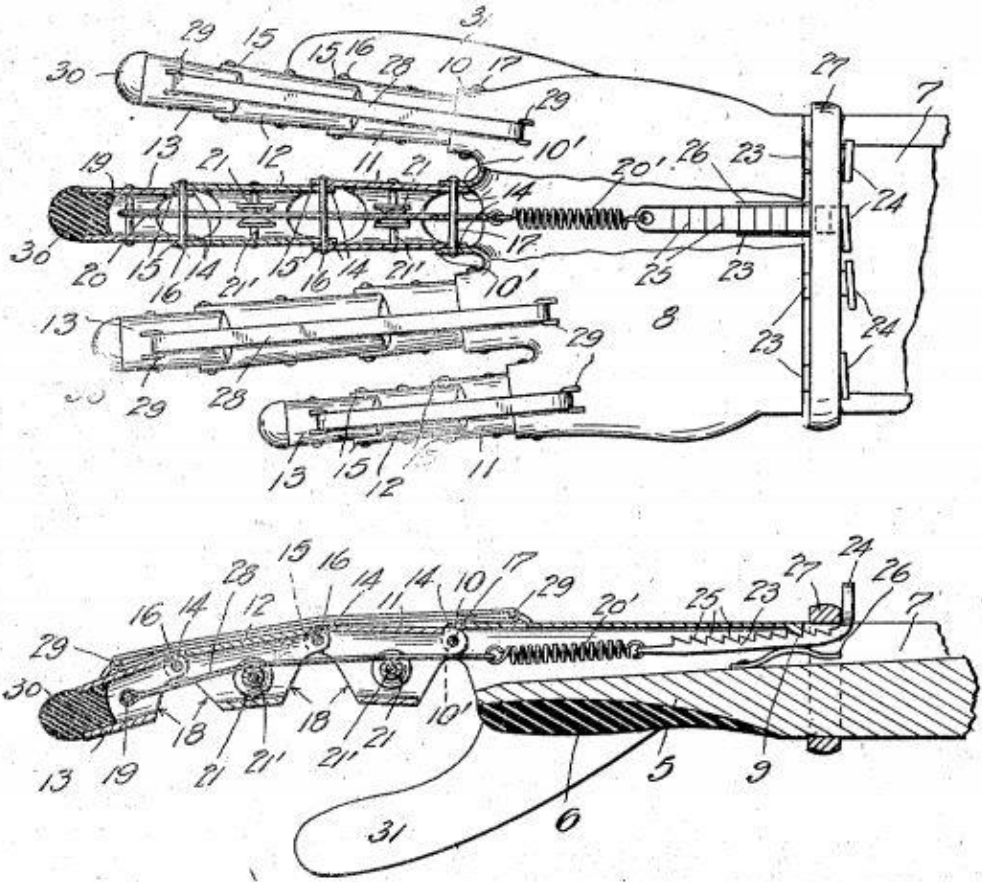


Figure 1.1 : Designed by Leonard Aydt [3].

Louis G. Caron designed another robotic hand in 1918 [4]. Caron's aim was provide flexion and extension movements to the fingers of robotic hand. Another aim was fixing robotic fingers in flexion position. Robotic fingers consist of three phalanxes. Finger movements were provided through gear wheels, worm wheels and bell-crank. In addition, springs were used. The thumb moved around more than one axis.

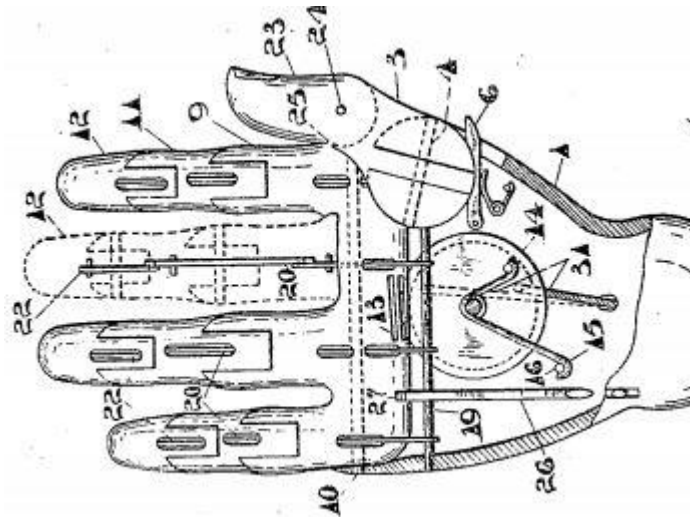


Figure 1.2 : Designed by Louis G. Caron [4].

James F. Mullen designed a robotic hand in 1970 [5]. James F. Mullen talked about lack of robotic hands, which were designed until to his time. The most important of these lacks were, fingers could not move independently from each other, end of thumb cannot touch the end of other fingers. In addition, Mullen noted that when designed robotic hand; size of robotic hand should be close to the size of human hand, thumb should move more than one axis, fingers should move independently, the distal phalanx of the thumb, should touch distal phalanx of the other fingers. Designed robotic hand by Mullen, was controlled tendon – wire system (Figure 1.3).

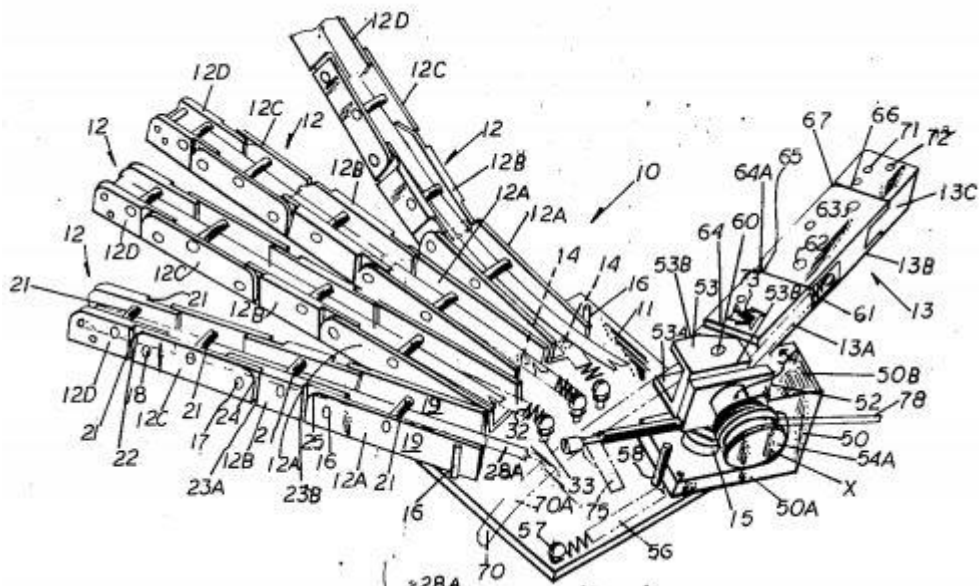


Figure 1.3 : Designed by James F. Mullen [5].

One of the robotic hand designs was emerged with a project which was started by UTAH and M.I.T. in 1982. Future works about robotic hand were presented. In project focused on marketing, applicability, persistence and stable theory and practice

approach on design. Thought that robotic hands would have a wide marketing area thanks to three reasons; firstly, robotic hands could be used in the industrial sector to variety of tasks. Secondly, robotic hands could permit remote human presence in hostile environments, distance environments or hazardous areas such as space, undersea and chemical laboratory. Thirdly, robotic hands could be used in military area. Also in this project, pointed out that what should be considered when producing robots. These are greater economy, higher performance and reduced possibility of injury to people. UTAH and M.I.T designed robotic hand with four fingers which were controlled tendon-wire system. Fingers consist of three phalanxes. Every finger has four joints [6].

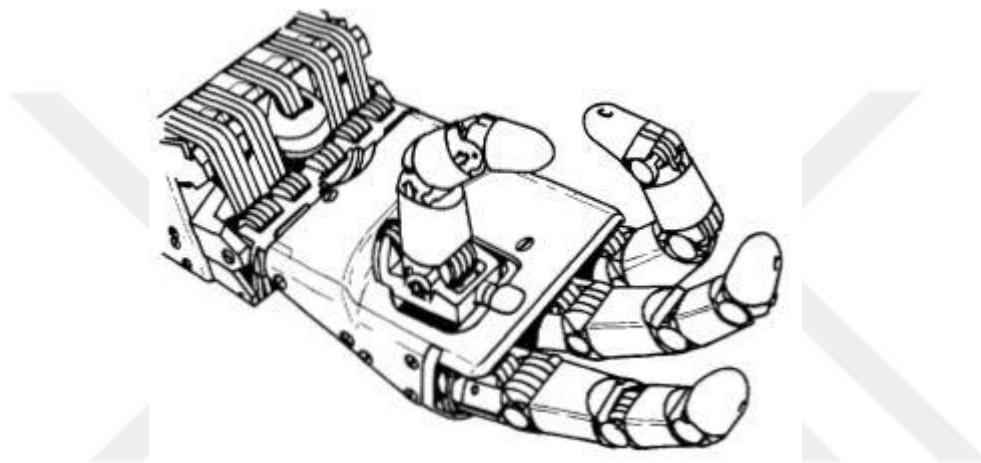


Figure 1.4 : Designed by UTAH / MIT [6].

In 1990, Carl F. Ruoff, and J. Kenneth Salisbury made a robotic hand, which has three fingers with nine DoF [7]. In this research, noted that past robotic manipulators consist of robotic arm with six DoF and an end effector. The problems that arise from these manipulators are they cannot hold objects in various shapes and the whole arm must move even in small displacements. This causes the manipulator cannot be used effectively in confined areas.

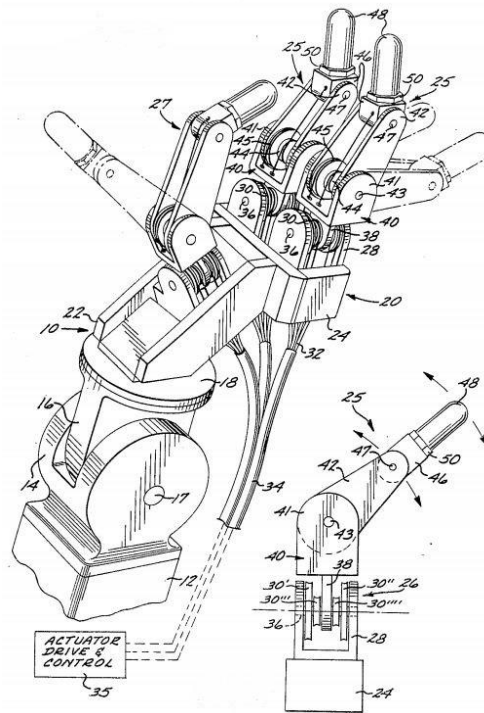


Figure 1.5 : Designed by Carl F. Ruoff and Salisbury [7].

A new robotic hand was presented by Gaiser et al [8]. One of the most important features of this design is the usage of flexible fluid actuators, which are excellent in power according to weight of robotic hand. It is pneumatically actuated. Hoses make spirals on the joints. When pressure is applied fingers bend from the joints. Sensivity is increased by using position feedback in robotic hand. Control is provided by using of 11 joints.

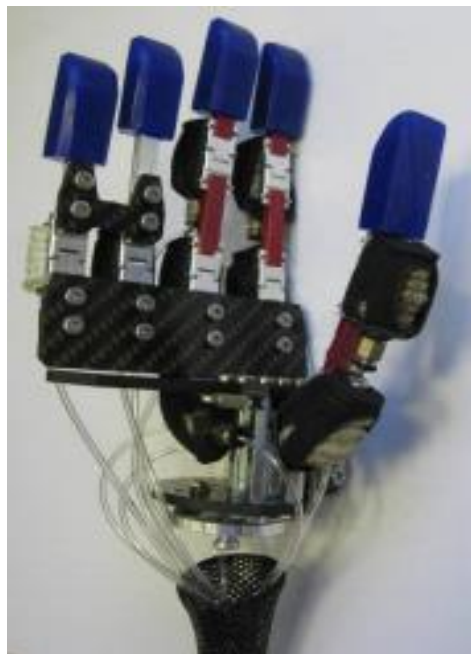


Figure 1.6 : Designed by Gaiser et al [8].

Mouri et al have renovated their old design called Gifu Hand II as Gifu Hand III [9]. In new design, the gap in power transmission is reduced and the workspace of the thumb is improved. With the innovations, the robotic hand has been able to manipulate the object like a capable human hand. The Gifu Hand III design has 20 joints and 16 degrees of freedom. The movement of some joints depends on the other joints.

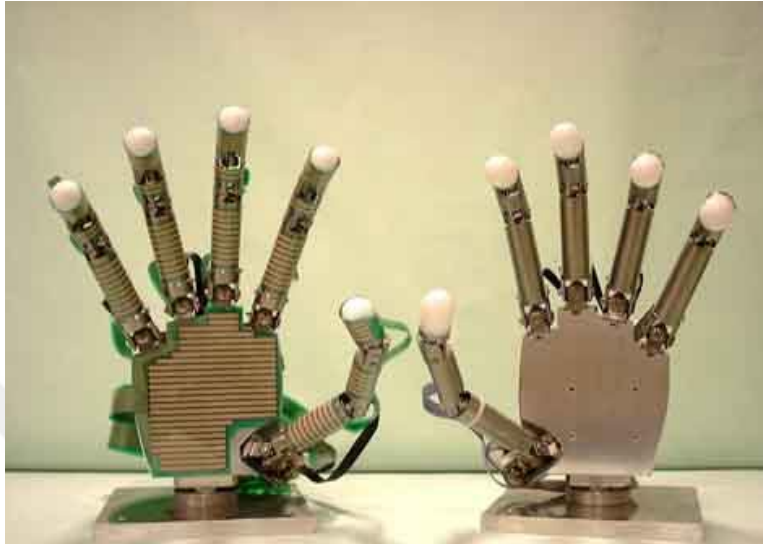


Figure 1.7 : Designed by Mouri et al [9].

Yamano and Maeno have implemented a hand design with elastic elements that can store their drive power in robotic hand working [10]. Robotic hand performs its task with a proper gripping motion without the need for a power. 20 degrees of freedom mechanism is used in this design.

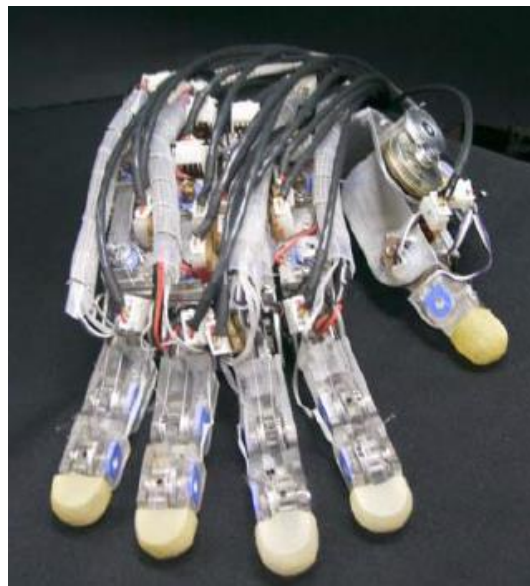


Figure 1.8 : Designed by Yamano and Maeno [10].

Table 1.1 : Developed Robotic Hands.

	Number of Fingers	Degrees of Freedom	Number of Joints	Actuation Method
Utah/MIT Hand [6]	4	16	16	Tendon – driven
COG Hand [11]	4	4	7	Tendon – driven
DLR Hand II [12]	4	13	16	DC motor – bevel gear
Gifu Hand II [13]	5	16	20	DC motor – worm gear
FRH 4 [8]	5	11	11	Flexible fluidic actuators (pneumatic)
Shadow Hand [14]	5	20	24	Tendon –driven
Yamano / Maeno [10]	5	20	20	Ultrasonic motor

1.2 Prosthesis Hands

Theodore Opuszanski designed one of the first prosthesis hands in 1951 [15]. This hand was artificial muscle actuated. His primary goal was the fact that the artificial hand should look like a real human hand and should be useful for many daily tasks. His other aims about artificial hand were; the artificial hand should be powerful, controllable, and as light as possible. This hand was designed as electrically and hydraulically operated.

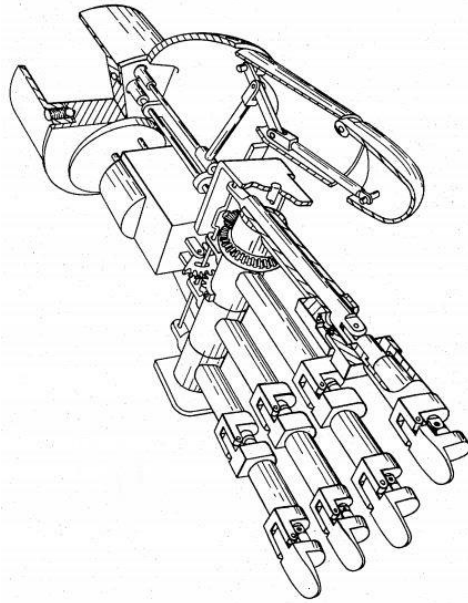


Figure 1.9 : Designed by Theodore Opuszanski [15].

Robinson George B. designed another prosthesis hand in 1952 for amputee people [16]. The main purpose of this artificial hand was to make life easier for amputee people. In this reason, the artificial hand was controlled through the cables connected to various parts of the body. The cables took electrical signals on muscle and processed them to provide artificial hand control. This robotic hand was designed as pneumatically operated.

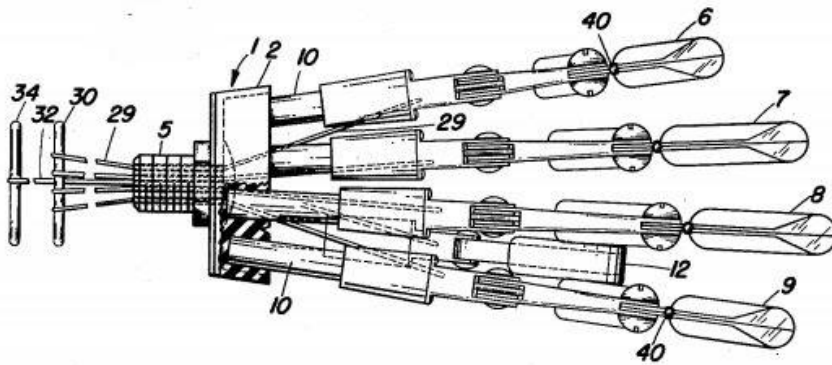


Figure 1.10 : Designed by Robinson George B [16].

George T. Pinson in his research talked about, robotic hands did not seem as a human hand and robotic hands had limited motion. Remote manipulators were used in a number of areas such as a nuclear field, in space, deep-sea research. In addition, a comprehensive robotic hand could not be produced because of high cost, the control difficulty and the design challenge. Pinson designed a robotic hand in 1970. The robotic hand was controlled with stepper motor and wires [17].

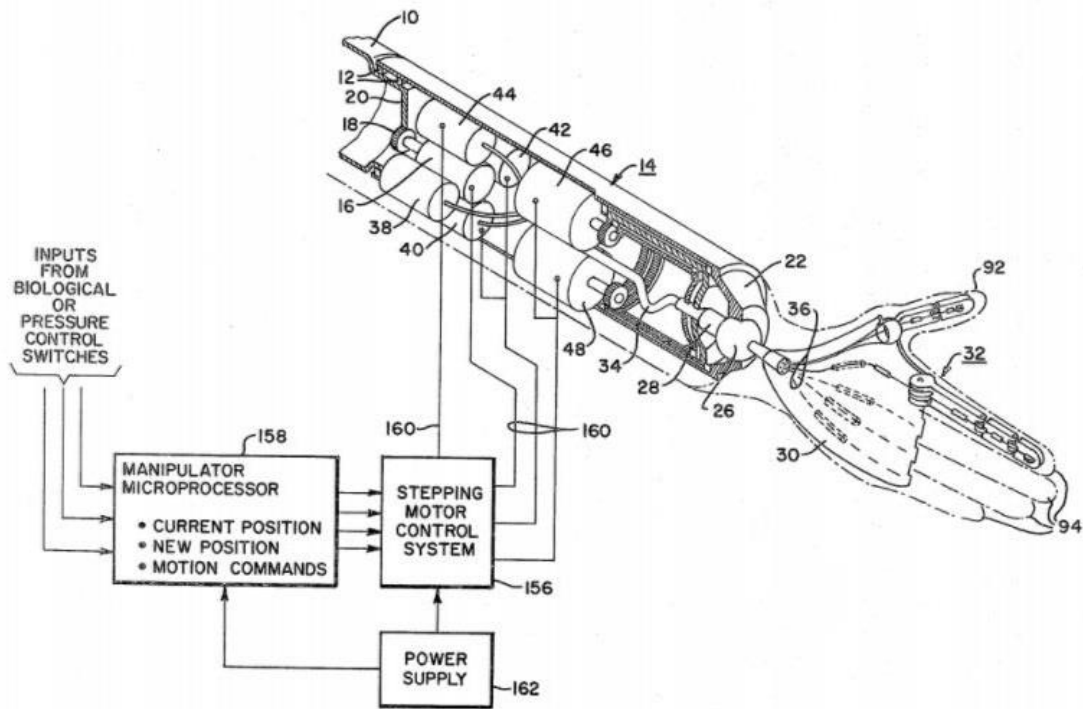


Figure 1.11 : Designed by George T. Pinson [17].

The prosthetic hand must have more than the one degree of freedom in order to be able to make multiple pattern grips in daily life. Anthropomorphic hand made by Southampton University in 2000. This robotic hand has 6 degrees of freedom. The robotic hand consists of DC motors and worm wheels. Thumb has two degrees of freedom and the other fingers have one degree of freedom [18].

Joseph T. Belter, Jacob L. Segil, Aaron M. Dollar and Richard F. Weir worked on six robotic hands with the latest technology in their work in 2013 (Table 1.2 and Table 1.3)[19].

Table 1.2 : Prosthesis hands for commercial purposes (general characteristics) [19].

	Developer	Number of Fingers	Number of Joints	Degrees of Freedom	Number of Actuators	Actuation Method
SensorHand [20]	Otto Bock	-	3	1	1	DC Motor
Vincent Hand [21]	Vincent Systems	5	11	6	6	DC Motor – Worm Gear
iLimb [22]	Touch Bionics	5	11	6	5	DC Motor – Worm Gear
iLimb Pulse [22]	Touch Bionics	5	11	6	5	DC Motor – Worm Gear
Bebionic [23]	RSL Stepper	5	11	6	5	DC Motor – Lead Screw
Bebionic v2 [23]	RSL Stepper	5	11	6	5	DC Motor – Lead Screw
Michelangelo [24]	Otto Bock	5	6	2	2	-

Table 1.3 : Prosthesis hands for commercial purposes (kinematic characteristics) [19].

	Metacarpal Phalanx (MCP) (°)	Proximal Interphalanx Joints (PIP) (°)	Distal Interphalanx Joints (DIP) (°)
SensorHand [20]	0 – 70 °	-	-
Vincent Hand [21]	0 – 90 °	0 – 100 °	-
iLimb [22]	0 – 90 °	0 – 90 °	~20 °
iLimb Pulse [22]	0 – 90 °	0 – 90 °	~20 °
Bebionic [23]	0 – 90 °	10 – 90 °	~20 °
Bebionic v2 [23]	0 – 90 °	0 – 90 °	~20 °
Michelangelo [24]	0 – 35 °	-	-

According to Table 1.3, DIP joints of all robotic hands are constant. There is 20 ° between PIP and DIP joints in iLimb and Bebionic Hands.

1.3 Wireless Communication

Surveying through the old works, one can easily notice that, one of the most important constraints of human robot interaction has been the wired communication. Wires that are used for data transfer between human and robot are both reducing the robot mobility inside its workspace and decreasing the effective interaction range. Removing these constraints become possible in today’s technology by the help of increased usage of wireless communication protocols and the researches in this field.

The first academic studies about wireless communication dates back to 1890s. The first works were about the development and origin of telegraph [25]. Later, focused on the secrecy of wireless communication [26], which types of antennas will provide better communication [27], where can be used wireless communication [28].

Recently, many surveys and researches show that, importance of wireless communication technologies are increasing day by day in our lives [29-31]. Developments of wireless communication and electronic systems have opened the way for low-cost, low-power, low complexity wireless communication technologies [30]. Also, thanks to wireless communication; we don’t need to deal with cables. These

technologies are used in diverse range of field. Robotic, medical, agriculture, home automation systems, military, habitat monitoring, climate etc.

Health status of the patients can be controlled continuously with wireless sensors, which are used in medical field [32]. Patient's disease, which cannot be identified staying in a hospital, can be monitored and traced with wireless communication sensors. For example, the hearth rhythm, breathing frequency and blood pressure can be monitored while the patient is doing sport. While working, hormonal changes in the body due to stress can be monitored instantly by the doctor. Also wireless communication technologies have been proposed for emergency medical healthcare. Duncan B., Malan D., Welsh M. from Harvard University, Gaynor M. from Boston University, Moulton S., from Boston Medical Center made "VitalDust" project for Emergency Medical Care [33-34]. Also "CodeBlue" was developed by Harvard University to provide discovery, routing, naming and security for medical sensor networks [34].

With the wireless communication sensors used in the agricultural field, the temperature and humidity values of each greenhouse can be monitored instantly on a computer. Also, the condition of plants and the rate of dehydration of the soil can be controlled from range. Weather data and geo-referenced water quality can be measured via wireless communication sensors which are used in agricultural field [35].

In smart home systems, electrical devices such as air conditioning, oven, and refrigerators can be controlled by wireless communication. They can be turned on and turned off remotely. Also for security purposes, motion detector, gas detector, fire detector can be used in smart home systems to control them from remotely [36-37].

Unmanned aerial vehicles and robots can be sent to the enemy lines and gathered information about enemy with wireless technology used in military field. In the battlefield, the conditions of the soldiers can be traced. Critical terrains, approach routes, paths and straits can be watched and soldiers can be oriented immediately [30, 37].

Habitat monitoring is a new research field. With wireless communication sensors, behaviors, position, population of animals and insects can be observed. The destructive effect of human on animal and plant has increased. Intel Research Laboratory at Berkeley initiated collaboration with the College of the Atlantic in Bar Harbor and the University of California at Berkeley to monitoring temperature humidity, barometric pressure in Great Duck Island in Maine in 2002. For this purpose, they made a wireless sensor networks in Great Duck Island [38].

1.4 Aim of the Study

In the light of these studies, aim of the thesis can be listed as follows,

- To design a lower degrees of freedom robotic hand with respect to the natural anatomy of the hand.
- To control of robotic hand by the help of wireless communication protocols.
- To perform kinematic analysis, modeling and simulating of finger movements.
- To characterize actual behavior of flexible sensors.
- To calibrate flexible sensors for the control task.
- To achieve two degree of freedom motion by using position feedback from single flexible sensor.
- To be able to grasp some solid objects by using designed robotic hands.



2. DEGREES OF FREEDOM AND ANATOMY OF HUMAN HAND

2.1 Forearm

Human hand and arm muscles are very sophisticated. For this reason, human hand has high dexterity motions. It can make many different movements to grasp something and to provide these motions; a number of different types of muscles are located on forearm (Figure 2.1a, Figure 2.1b). At least a pair of muscles is required for each joint movement [39].

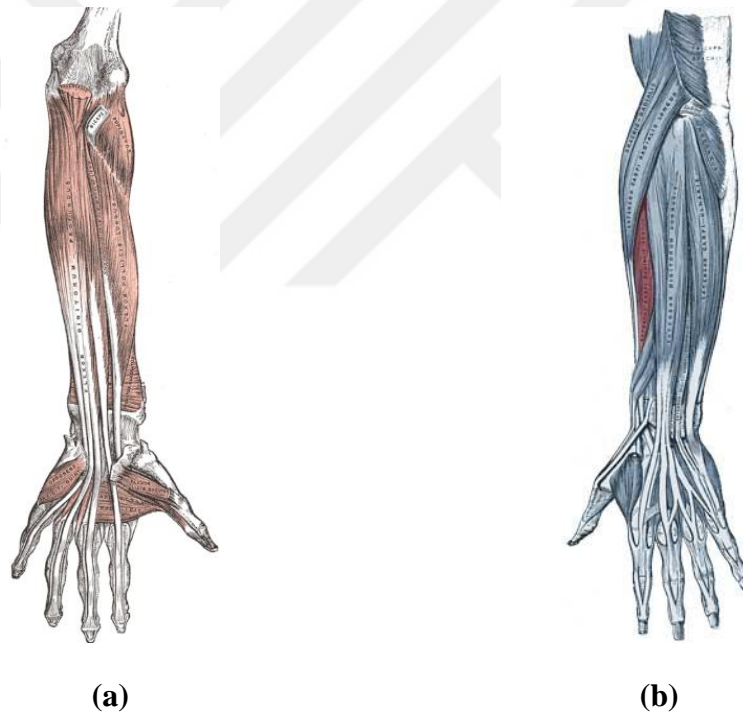


Figure 2.1 : Muscles on the forearm (a) From bottom side[45]. (b) From top side [46]

2.2. Bones of human hand and degrees of freedom

Except thumb, human fingers basically make 4 different movements. These are flexion / extension (FE), abduction / adduction (AA). The movement of the fingers towards the palm is called the flexion. Opposite of this movement is called extension.

Movement of the fingers towards the sides is called adduction. Opposite of this movement is called abduction (Figure 2.2). The thumb touches all the other fingers and this motion is called the opposition [2].

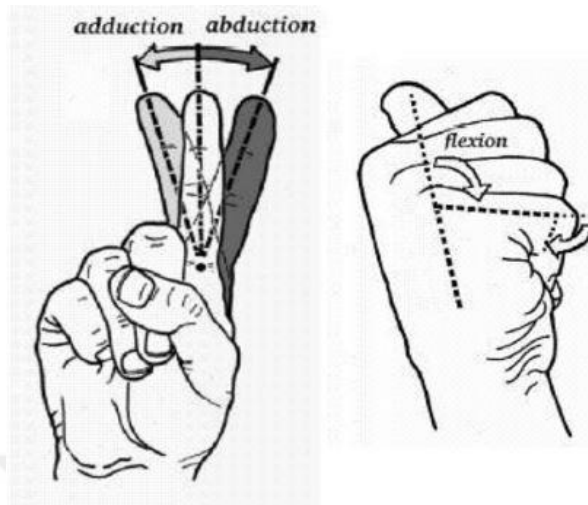


Figure 2.2 : Metacarpal joint motions (adduction and abduction, flexion, extension) [49].

Human hand has 14 phalanges (Figure 2.3a). There are five distal phalanges, four intermediate phalanges, five proximal phalanges. There is a joint between each phalanges (Figure 2.3b). Between distal phalange and intermediate phalange is called distal inter phalange (DIP). Between intermediate phalange and proximal phalange is called proximal inter phalange (PIP). Between proximal phalange and metacarpal is called metacarpal phalange (MCP). Thumb consist of two phalanges that's why there isn't intermediate phalange.

Every finger (except thumb), has 4 degrees of freedom (DoF). One DoF comes from each joint, which is between phalanges for flexion / extension (FE) movements. Totally except thumb each finger has 3 DoF for FE movements. In addition, one more DoF comes from MCP for AA movements (Table 2.1).

Thumb is the most important finger for grabbing functions. It has five DoF. Unlike other fingers, thumb can move from carpal metacarpal bones (CMC). Two DoF comes from IP and MCP for flexion and extension movements; one more DoF comes from between proximal and metacarpal phalanx for abduction and adduction movements. Also from carpal bone two DoF comes, one DoF for flexion and extension

movements, other one for abduction and adduction movements. Totally, human hand has 21 DoF.

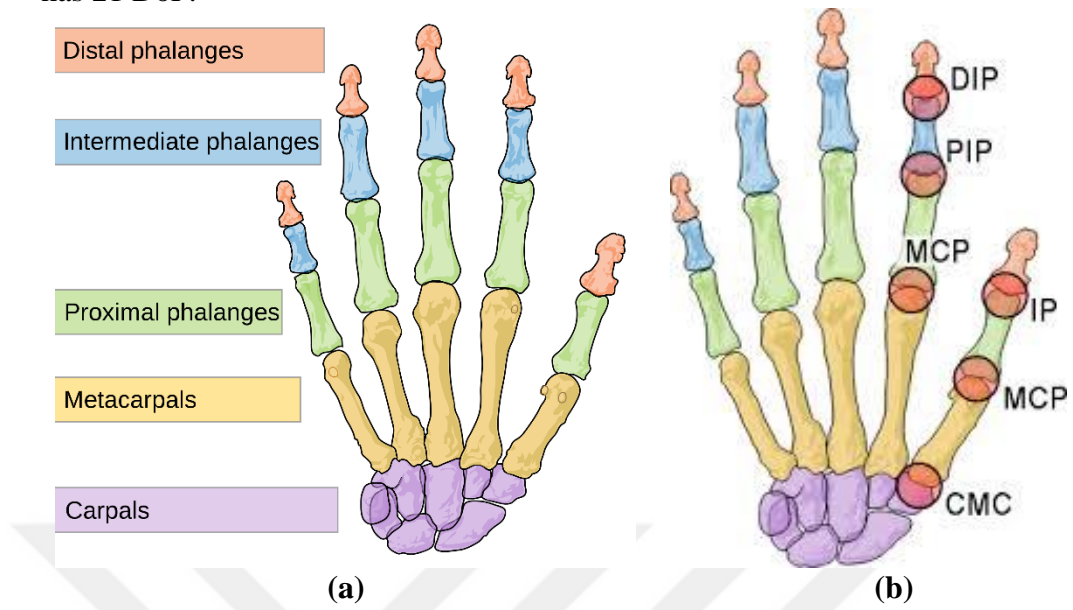


Figure 2.3 :Anatomy of Human Hand (a) Bones [47]. (b) Joints [48].

Table 2.1. : Degrees of freedom of human hand and motions.

DoF	Digit	Joint	Motion
1	Index	DIP	FE
2	Index	PIP	FE
3	Index	MCP	FE
4	Index	MCP	AA
5	Middle	DIP	FE
6	Middle	PIP	FE
7	Middle	MCP	FE
8	Middle	MCP	AA
9	Ring	DIP	FE
10	Ring	PIP	FE
11	Ring	MCP	FE
12	Ring	MCP	AA
13	Little	DIP	FE
14	Little	PIP	FE
15	Little	MCP	FE
16	Little	MCP	AA
17	Thumb	DIP	FE
18	Thumb	MCP	FE
19	Thumb	MCP	AA
20	Thumb	CMB	FE
21	Thumb	CMB	AA

3. STANDARDS AND CONFIGURATIONS OF ELECTRONIC COMPONENTS

3.1 Standard of IEEE 802.15.4 and Zigbee Protocol

Local Area Network (LAN) is a network formed by communication between devices (computer, phone, writer, e.g.) within short distance. Wireless Local Area Networks (WLAN) is the wireless communication of these devices. With the success of the WLAN, researches in this field have increased and according to these researches, developments have increased. The directions of these developments are more dependent on low-cost, low complexity and extremely low power.

Personal Area Network (PAN) is a network used for communication among personal devices in a short distance. Wireless-PAN (WPAN) is a network, communication among personal devices is provided wirelessly.

IEEE 802 Working Group 15 is specialized to work on WPAN [40]. ZigBee is a protocol based on IEEE 802.15.4 standard, which is used in many fields including the medical field.

3.1.1 Standard of IEEE 802.15.4

Due to the need for inexpensive, low power wireless sensor network applications, in December 2000 Task Group 4, under the IEEE 802 Working Group 15, was formed to begin the development of a Low-Rate WPAN (LR - WPAN). Low-Rate means low complexity, low - cost and extremely low - power. Table 3.1 shows comparison of LR – WPAN with other wireless networks [41].

Table 3.1 : Comparison of LR – WPAN with other wireless networks [41].

	WLAN	BT / WPAN	Low – Rate WPAN
Range	~ 100 m	~ 10 – 100 m	10 m
Data Throughput	11 Mb/s	1 Mb/s	< 0.25 Mb/s
Power Consumption	Medium	Low	Ultra Low

Table 3.2 :IEEE 802.15.4 High Level Characteristics [41].

Frequency Band	Two PHYs	Low-Band (BPSK)	868 MHz	1 channel - 20 kb/s
			915 MHz	10 channels - 40 kb/s
		High-Band (O-QPSK)	2.4 GHz	16 channels - 250 kb/s
Channel Access	CSMA-CA			
Range	10 - 20 m			
Addressing	Short 8 bit or 64-bit IEEE			

According to Table 3.2 IEEE 802.15.4 standard consist of two bands, Low-Band and High-Band. Low-Band is specified to operate in 868 MHz in Europe with 1 channel, 20 kb/s data rate and in 915 MHz in America with 10 channels, 40 kb/s data rate. High-Band is specified to operate in 2.4 GHz nearly for whole world with 16 channels, 250 kb/s data rate [41].

3.1.2 Zigbee

ZigBee is a protocol, which uses IEEE 802.15.4 sub-structure. On Table 3.3, well-known wireless sensor networks were compared.

Table 3.3 : Wireless Communication Networks [42,52].

	ZigBee	GPRS/GSM	Wi-Fi	Bluetooth
Focus Area	Monitoring and Control	Voice and Data Transmission	Internet Email	Instead of Cable
System Resource	4 – 32 KB	16 MB+	1 MB+	250 KB+
Battery Life (Day)	100 – 1000+	1 - 7	0,5 – 5	1 – 7
LAN Size	Unlimited	16 Mb+	32	7
Bandwidth (KB/s)	20 - 250	64 – 128+	11000	720
Range(Meter)	1 – 100+	1000+	1 – 100	1 – 10+
Successful Areas	Power Consumption and Durability	Access and Quality	Speed and Flexibility	Cost and Convenience

According to Table 3.3, the most important feature of ZigBee is the low power consumption. In addition, even the minimum number of days for battery life is considerably higher than the other networks.

3.2 Zigbee Device Types And Network Topologies

3.2.1 Zigbee device types

ZigBee networks have three different device types; coordinator, router and end device.

Coordinator, as the name implies, coordinates the network. Define, set and protect the network, and specify the communication of the addresses. In each network, there can be only one coordinator. To achieve communication, there must be one more device.

The routers are responsible for routing traffic between different nodes. Router carries information between other devices on the network. It is like a messenger. There can be more than one in each network. Other name of router is Full Functional Device (FFD).

End devices only receive and transmit data. Other name of end device is Reduced Functional Device (RFD). As the name implies they cannot communicate with other devices. They always need router or coordinator. End devices can go to sleep mode when they do not communicate. This mode saves energy. End device is the reason of long battery life of XBees.

The functions and the differences of the three devices are given at Table 3.4.

Table 3.4 : Comparison of ZigBee Devices [50].

ZigBee Network Layer Function	Coordinator	Router	End Device
Establish a ZigBee network	✓		
Permit other devices to join or leave the network	✓	✓	
Assign 16-bit network addresses	✓	✓	
Discover and record paths for efficient message delivery	✓	✓	
Discover and record list of one-hop neighbors	✓	✓	
Route network packets	✓	✓	
Receive or send network packets	✓	✓	✓
Join or leave the network	✓	✓	✓
Enter sleep mode			✓

3.2.2 Zigbee network topologies

There are four types of ZigBee network topology (Figure 3.1).

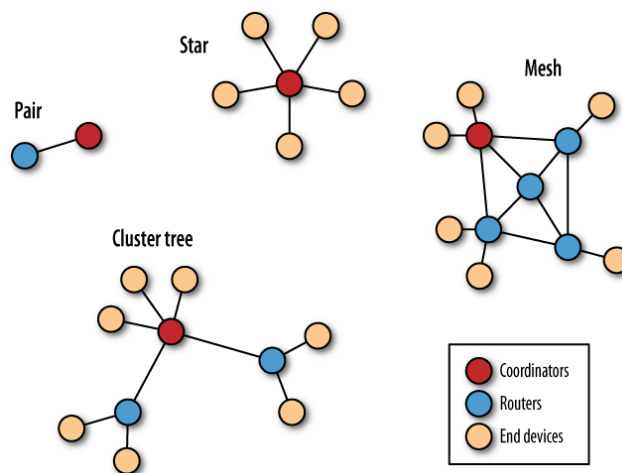


Figure 3.1 : ZigBee network topology [43].

Pair is the simplest network topology. There are two devices in this network. One of them is coordinator and the other one can be router or end device. Only these two device communicate.

In star topology, all devices are connected to the coordinator. All messages must pass through the coordinator. The coordinator distributes the message. Other devices can't communicate with each other. Generally, end devices are reason for preference in this topology because of energy saving. Usage areas of this topology is home automation systems, personal computer peripherals and toys.

In mesh topology, messages are transmitted through routers. The coordinator is responsible for the management of the network. In addition, it decide to distribute messages. Router can communicate with the router, end device and the coordinator. Mesh network can make ad – hoc, routing and self – healing. Other name of mesh network is peer-to-peer. Usage areas of mesh topology is Industrial control and imaging.

Cluster tree is similar to mesh topology. In this topology, routers cannot communicate directly.

3.3 Xbees And Configuration Of Xbees By Using X-CTU

3.3.1 XBee

XBee is a wireless antenna, which uses 802.15.4 LR-WPAN protocol for communication in 898 MHz, 915 MHz and 2.4 GHz operating frequency. There are two basic XBee radio hardware; XBee S1 and XBee S2. In addition, there are two different data transfer mode. First one is the transparent (AT) mode and second one is the Application Program Interface (API) mode.

XBee S1 is based on point-to-point communication. That's why, it is good at AT mode. AT mode is good at when sending and receiving a single data. Experiment was made for using two different data in AT mode. Sometimes second data was read as first data. In this project, nine variables are sent. In this case, data are more likely to interfere.

XBee S2 is based on mesh networking. Therefore, it is better on API mode. API mode provides to data to be sent in packets. Therefore, data interfere that occurs in AT mode does not occur in API mode. Also power consuming of XBee S2 is less than XBee S1. Comparing of XBee S1 and S2 are showed in table 3.5.

Table 3.5 : XBee S1 versus XBee S2 [43].

	Series 1	Series 2
Typical (indoor/urban) range	30 meters	40 meters
Best (line of sight) range	100 meters	120 meters
Transmit/Receive current	45/50 mA	40/40 mA
Firmware (typical)	802.15.4 point-to-point	ZB ZigBee mesh
Digital input/output pins	8	11
Analog input pins	7	4
Low power, low bandwidth, low cost, addressable, standardized, small, popular	Yes	Yes
Interoperable mesh routing, ad hoc network creation, self-healing networks	No	Yes
Point-to-point configuration	Simple	More involved
Underlying chipset	Freescale	Ember

XBee S1 uses the basis of IEEE 802.15.4. XBee S2 built on top of 802.15.4. This means, 3 features are added on XBee S2. 1- Routing, 2- ad hoc network creation, 3- self-healing networks.

Routing is the process of selecting the way in which different networks use each other to communicate with each other. Ad – Hoc Network Creation is an automated process that creates an entire radio network without human intervention. In a network, when one or more radio is missing, Self – Healing Network provides to reconfigure the network to repair any broken routes [43].

There are two basic rules to provide a communication between XBees. First, XBee's must be in the same version in order to communicate with each other. That is, XBee S1 communicate with S1, and XBee S2 communicate with S2. Second, XBee's must be configured. To configure XBee, computer connection is needed. This

communication is provided by XBee adapters (Figure 3.2). Another method to computer connection, XBee is connected to Arduino and Arduino is short-circuit. But in this method Arduino may be harmed, so it is not preferred.



Figure 3.2 : XBee usb adapter.

XBee radio has 20 connection pins (Table 3.6). There are 2 mm distance between two pins. For this reason, XBee cannot be placed on breadboard directly and XBee breakout boards are needed (Figure 3.3). In addition, XBee pin directions are shown in Table 3.6.



Figure 3.3 : XBee breakout board

Table 3.6 : Pins of XBee S2 [43].

Pin #	Name(s)	Description
1	VCC	3.3 V power supply
2	DOUT	Data Out (TX)
3	DIN	Data In (RX)
4	DIO12	Digital I/O 12
5	RESET	Module reset (asserted low by bringing pin to ground)
6	PWM0/RSSI/DIO10	Pulse-width modulation analog output 0, Received Signal Strength Indicator, Digital I/O 10
7	DIO11	Digital I/O 11
8	Reserved	Do not connect
9	DTR/SLEEP_RQ/ DIO8	Data Terminal Ready (hardware handshaking signal), Pin Sleep Control (asserted low), Digital I/O 8
10	GND	Ground
11	DIO4	Digital I/O 4
12	CTS/DIO7	Clear to Send (hardware handshaking), Digital I/O 7
13	ON/SLEEP	Sleep indicator (off when module is sleeping)
14	VREF	Not used in Series 2
15	ASSOC/DIO5	Association indicator: blinks if module is associated with a network, steady if not; Digital I/O 5
16	RTS/DIO6	Request to Send, Digital I/O 6
17	AD3/DIO3	Analog Input 3, Digital I/O 3
18	AD2/DIO2	Analog Input 2, Digital I/O 2
19	AD1/DIO1	Analog Input 1, Digital I/O 1
20	AD0/DIO0/COMMIS	Analog Input 0, Digital I/O 0, Commissioning Button

3.3.2 Configuration of XBees by using X-CTU

X-CTU is a Windows-based application (Figure 3.4). X-CTU software was used to perform change configurations of XBee.

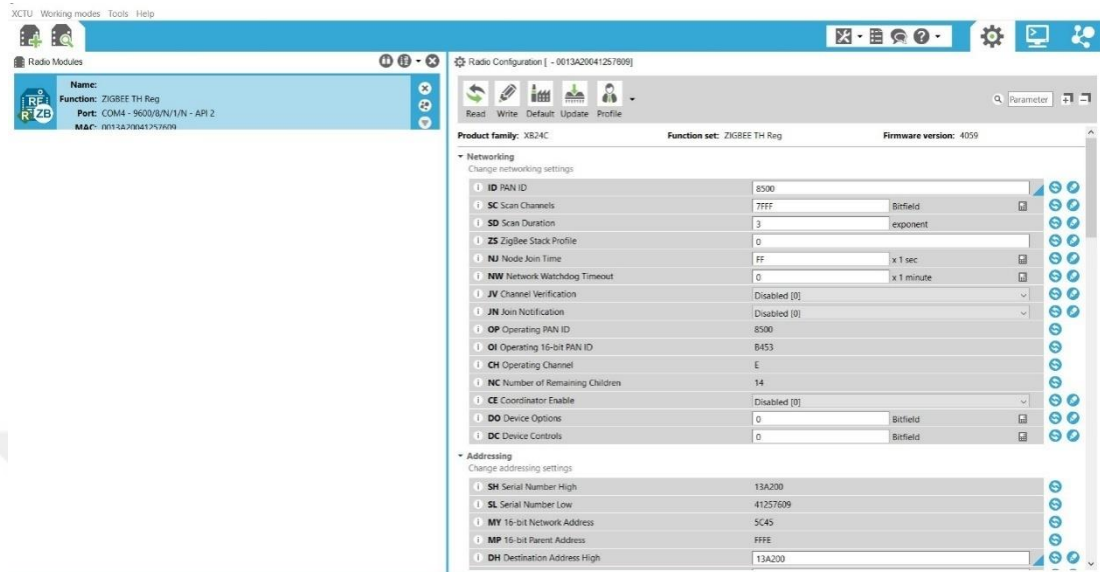


Figure 3.4 : X-CTU.

Each XBee has a unique and permanent 64-bit serial number. Serial number of our XBee's 0013A20041257609 and 0013A2004125760A. Serial numbers separate two parts. First part is Serial Number High (SH); 0013A200 is the SH. Other part is Serial Number Low (SL); the part that comes after 13A200 is forming SL. SL of our XBee's are 41257609 and 4125760A.

3.3.2.1 Channel

The channel controls the frequency band in which the XBee communicates. Most of XBees work on 2.4 GHz on 802.15.4 band. Channel operates frequency of this band.

3.3.2.2 Pan address

Pan Address is a 16-bit address specific to the network, which is set after the network is created. There are 65536 Pan ID. XBees can only communicate when they have same Pan ID in network.

3.3.2.3 My address

Each Xbee must be set with a 16 bit address on the network. Other name is Source Network.

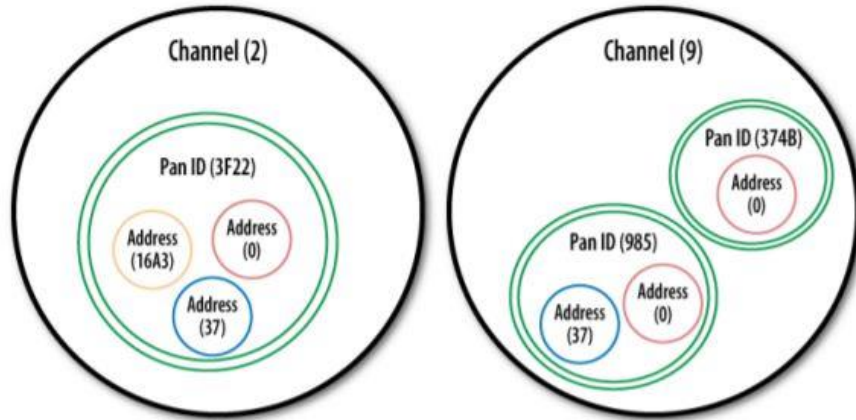


Figure 3.5 : Venn diagram of XBee network [43].

3.3.2.4 Destination address

Destination Address specifies the source address of another XBee to send data. In order to send data from one XBee to another XBee, destination address of first XBee must be the same as the source address of second XBee. There are two configurations of Destination Address. These are Destination High Address (DH) and Destination Low Address (DL).

SH and SL numbers can be set as DH and DL for two XBees to send and receive data between each other. In this thesis, since only two XBees is used, the source addresses of each other are used to communicate between each other (Table 3.7).

Table 3.7 :XBee configurations.

XBee Parameters (Arduino Mega)			
Transmitter		Receiver	
CH	E	CH	E
PAN ID	8500	PAN ID	8500
DH	13A200	DH	13A200
DL	4125760A	DL	41257609
BD (Baud Rate)	9600	BD	9600
AP (API Enable)	API enabled with escaping	AP (API Enable)	API enabled with escaping

4. DESIGN AND MANUFACTURING OF ROBOTIC HAND

4.1 Design Criteria and Component Descriptions

Design and manufacturing of the robotic hand consists of three parts. The first part is the design and production of mechanical parts. The second part is the design of Arduino circuit and programming Arduino. The third part is calibration of flexible sensor.

Some criteria and constraints were considered during the design of hand. These criteria and constraints are grabbing ability, fingers should move independently, robotic hand should be anthropomorphic, degrees of freedom of the hand, cost, developable. Design criteria and constraints of the robotic hand were ordered as follows:

- The robotic hand should be able to simulate most of the linear movements of a human hand, which are necessary for useful work.
- The robotic hand should be capable of simulating the linear optimal hand movements of an adult human hand and robotic hand is maintained substantially proportional size and shape of an adult human hand.
- Thumb of robotic hand should touch the distal phalanges of other fingers.
- All fingers of robotic hand should move independently.
- The robotic hand should be able to grab materials which have different shapes.
- Cost of robotic hand should be inexpensive.
- The robotic hand should be developable with open-source code.
- The robotic hand should act in real time with human hand.

4.2 Designing and Manufacturing of the Robotic Hand

Main aim of the design was the robotic hand has to be able to simulate most of the linear movements of a human hand, especially necessary for useful works in daily living. In addition, size of robotic hand should be close to the average size of an adult human hand. The robotic hand should be able to grab some materials in different shapes.

Designed robotic hand has 9 degrees of freedom. When designing the robotic hand, it was considered to be anthropomorphic. All the mechanical parts were designed in Autodesk Inventor.

Firstly, robotic fingers were designed. Secondly, hand was designed. All structures were proportionally sized as in an adult human hand. The 3D models of robotic hand were also inserted in the assembly drawing in order to check the compatibility of all the pieces.

After assessing the compatibility of the mechanical parts of the robotic hand, simulation were made in Motion Study in SolidWorks. The mechanical parts of the hand were produced by using 3D printer ATOM.

4.2.1 Finger and hand sizes

Index finger, middle finger, ring finger and little finger consist of three phalanges. Thumb consists of two phalanges. Length of index finger (Figure 4.1) and ring finger are same. Thickness of all fingers is the same. Size and lengths of fingers are shown in Table 4.1. In addition, hand sizes are shown on Figure 4.2. All values are in millimeter.

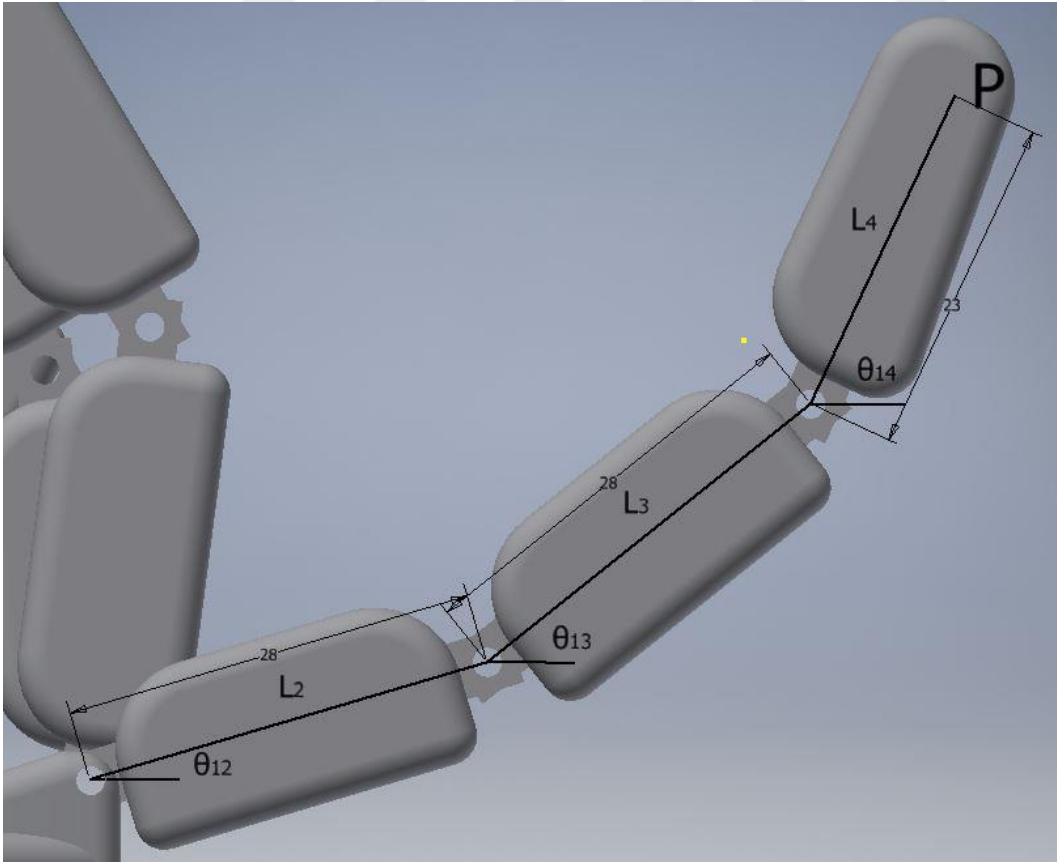


Figure 4.1 : Index finger.

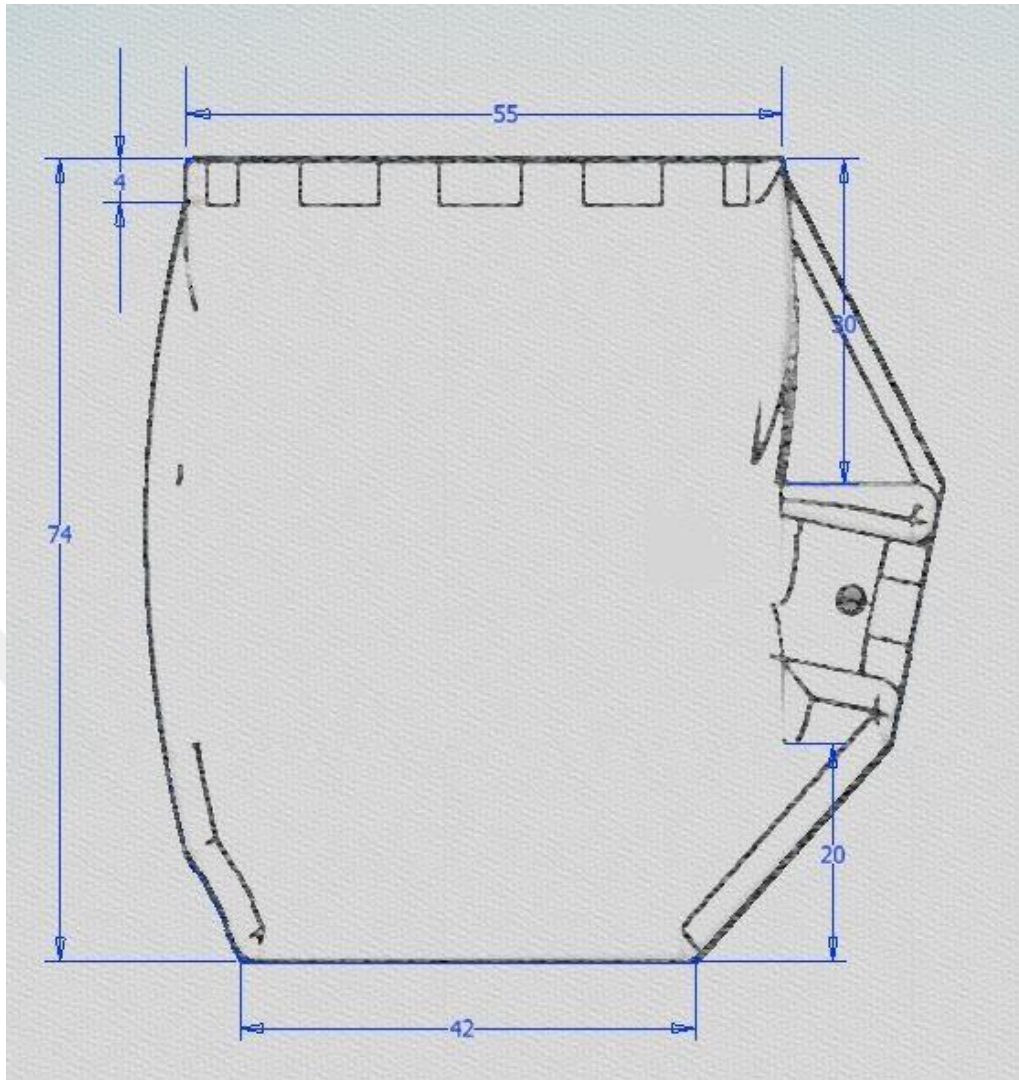


Figure 4.2 : Palm sizes (mm).

Table 4.1 : Finger sizes in length and thickness.

Length and Thickness	Thumb	Index Finger	Middle Finger	Ring Finger	Little Finger
L2	-	28 mm	31 mm	28 mm	23 mm
L3	23 mm	28 mm	31 mm	28 mm	22 mm
L4	26 mm	23 mm	23 mm	23 mm	23 mm
Thickness	12 mm	12 mm	12 mm	12 mm	12 mm

4.2.2 Tendon – wire systems

Three different tendon-wire systems are used, n , $2n$ and $n + 1$ tendon. There are two different structures in n tendon systems. The first one is the shown in Figure 4.3a.

Actuator is used for the flexion movement. When the actuator releases the wire, thanks to pretension spring fingers make extension movement. Second system is the shown in Figure 4.3b. A single actuator and two opposed tendons are used for flexion and extension movement. The advantage of n-tendon system is reduced set of required actuators, according to $2n$ system. When pretension spring is used in n tendon system, if applied force is excessive or if the system is used heavily, it will be worn faster than the two opposed tendon systems. The purpose in $2n$ tendon systems is to increase the flexibility, accuracy and precision of the movements of the robotic hand (Figure 4.3c). On the other hand, the complexity of the system increases due to the used number of actuators. In $n + 1$ tendon systems, $n + 1$ tendon and actuator are used, to control n independent joints (Figure 4.3d). This system reduce complexity according to $2n$ -tendon system. While flexion movements of fingers controlling with two actuators, extension movements is controlled with one actuator [2]. For all tendon systems, movement gaps is a problem because of the tendons.

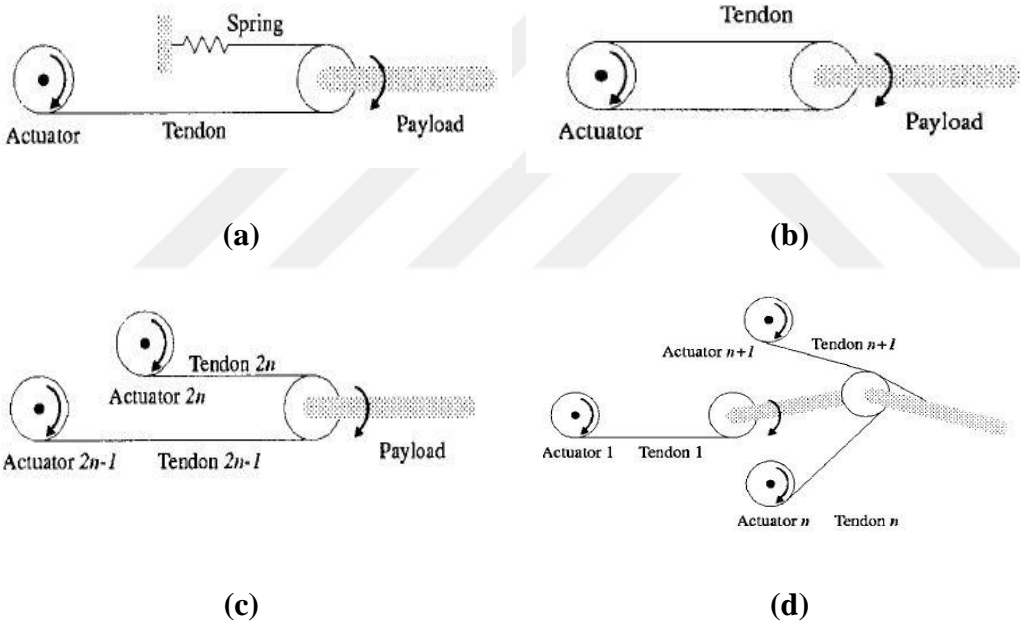


Figure 4.3 : Tendon – wire systems (a) n-tendon system with pretension spring. (b) n-tendon system with two opposed tendons. (c) $2n$ -tendon system. (d) $n+1$ tendon system [2].

Robotic hand fingers are controlled with the n tendon-wire system with two opposed tendons. Distal and proximal phalanges for index, middle, ring and little fingers have four holes (Figure 4.4a). First and second holes are inside of the distal and proximal phalanges. Third and fourth holes are the outside of these phalanges. These holes are used to connect wires to control robotic fingers. Thumb has only one wire connection

to distal phalanges. The holes on the inside of fingers are used for flexion movements; the holes on the outside of fingers are used for extension movements.

Phalanges of each robotic finger have two gaps from end to end. These gaps are for passing through wires. One gap at the bottom side, other one is at the top side (Figure 4.4a).

For each finger, there are two independent motion except thumb. Thumb has one degrees of freedom. Therefore, robotic hand has nine degrees of freedom. In addition, there are five under actuated joints. These are intermediate phalanges and for thumb proximal phalanges. Totally, the robotic hand has 14 DoF.

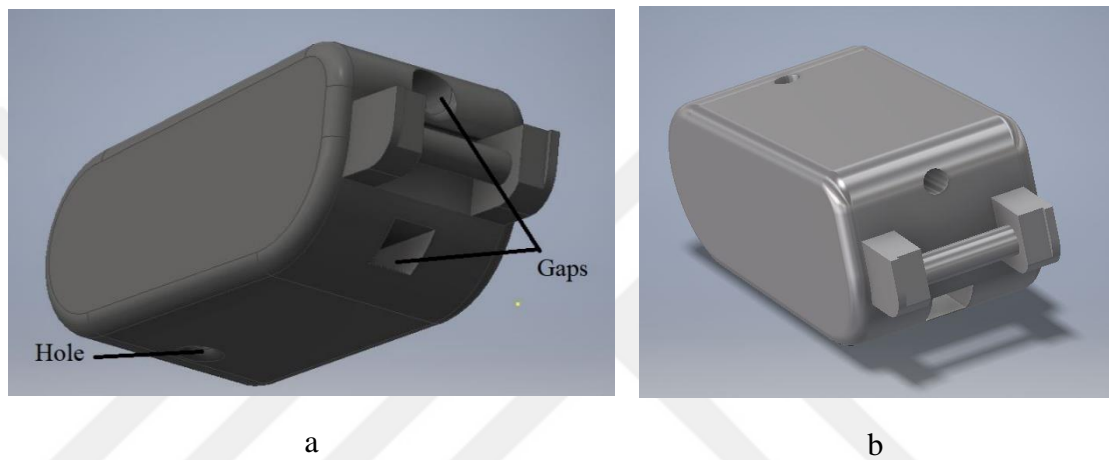


Figure 4.4 : a-) First part of index finger, view from bottom side.

b-) First part of index finger, view from top side.

After all the mechanical parts were designed, they were assembled in Inventor for checking the compatibility of parts (Figure 4.5).

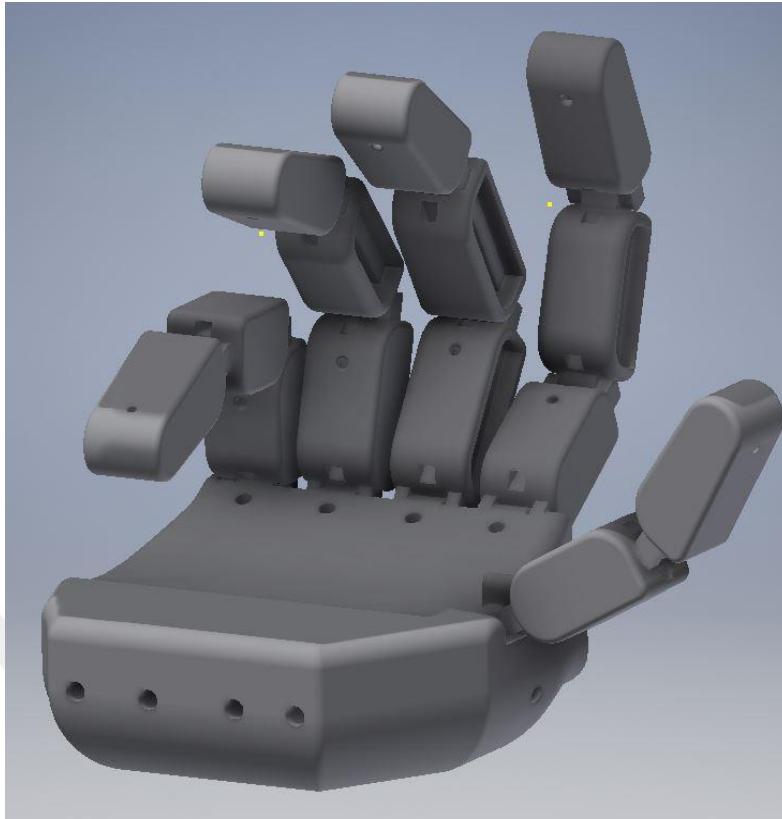


Figure 4.5 : Assembly of robotic hand.

Palm of robotic hand holes for tendon-wire system were shown in Figure 4.6.

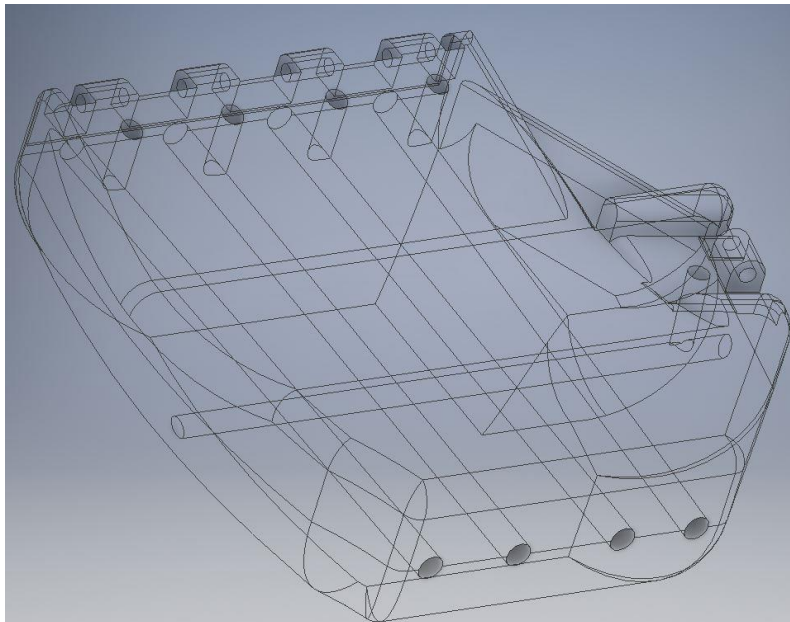


Figure 4.6 : Palm holes for wires.

4.2.3 Kinematic analysis of fingers

Drawing trajectory of fingertips and movement of fingers were observed in Motion Study in SolidWorks. The orbits shown in Figure 4.8 were created by adding actuators to the joints and using the SolidWorks. The following formulas are time - dependent kinematic analysis of finger. The rotation coefficients of the articulations were taken as $k_1 = 10, k_2 = 9$ and $k_3 = 9$.

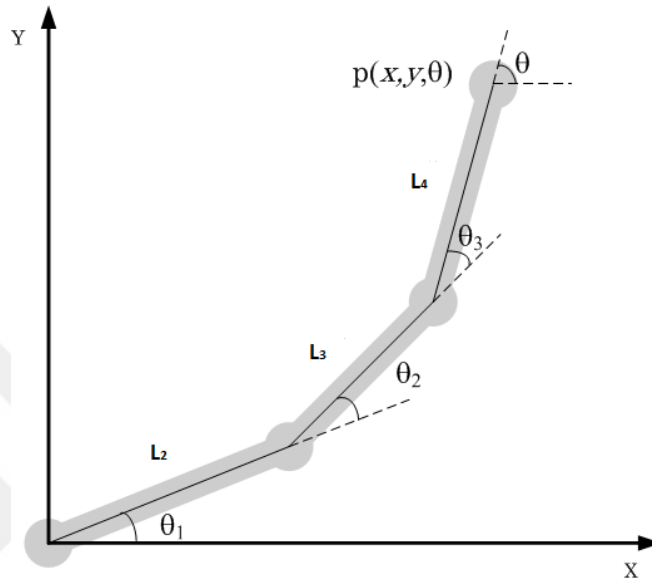


Figure 4.7 : Kinematic models of finger in X, Y plane.

Table 4.2 : Denavit – Hartenberg parameters.

Joint i	a_{i-1}	α_{i-1}	d_i	θ_i
1	0	0	0	θ_1
2	L_2	0	0	θ_2
3	L_3	0	0	θ_3
4	L_4	0	0	0

Tip point coordinates of three DoF finger are written as follows,

$$P_x = L_2 \cos(\theta_1(t)) + L_3 \cos(\theta_2(t)) + L_4 \cos(\theta_3(t)) \quad (4.1)$$

$$P_y = L_2 \sin(\theta_1(t)) + L_3 \sin(\theta_2(t)) + L_4 \sin(\theta_3(t)) \quad (4.2)$$

where $\theta_1(t) = a(t).k_1, \theta_2(t) = a(t).k_2, \theta_3(t) = a(t).k_3$ and $a(t) = \omega. r. t .$

In these formulas, $P_x(t)$ and $P_y(t)$ show the position of the last point of the fingers changes with the time. $\theta_1(t), \theta_2(t)$ and $\theta_3(t)$ show the rotation angles of distal, middle and proximal parts of robotic fingers. All rotation angle depend on movement of wire which provides the movement of robotic fingers, defined by $a(t)$, are dependent on

time. r is the radius of pulley which the rope is wrapped. ω is angular velocity of actuator.

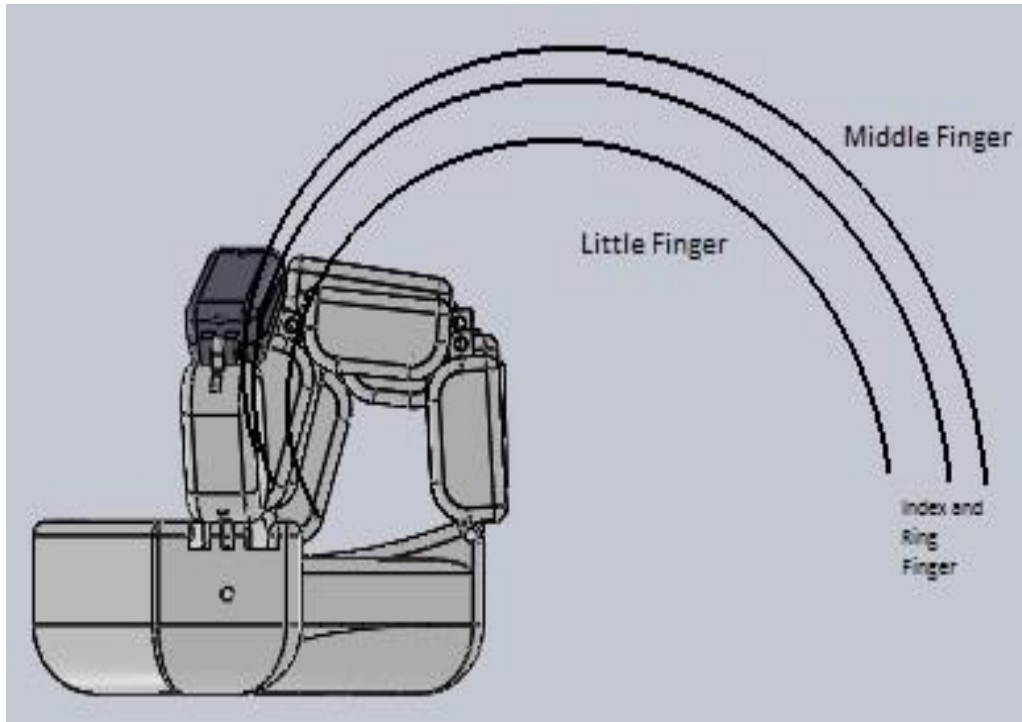


Figure 4.8: Drawing trajectory of fingertips.

In addition, a spherical body was grabbed by robotic hand in Motion Study (Figure 4.9). The simulation was realized while each finger was moving at the same time and used solid contact between the fingers and spherical body.

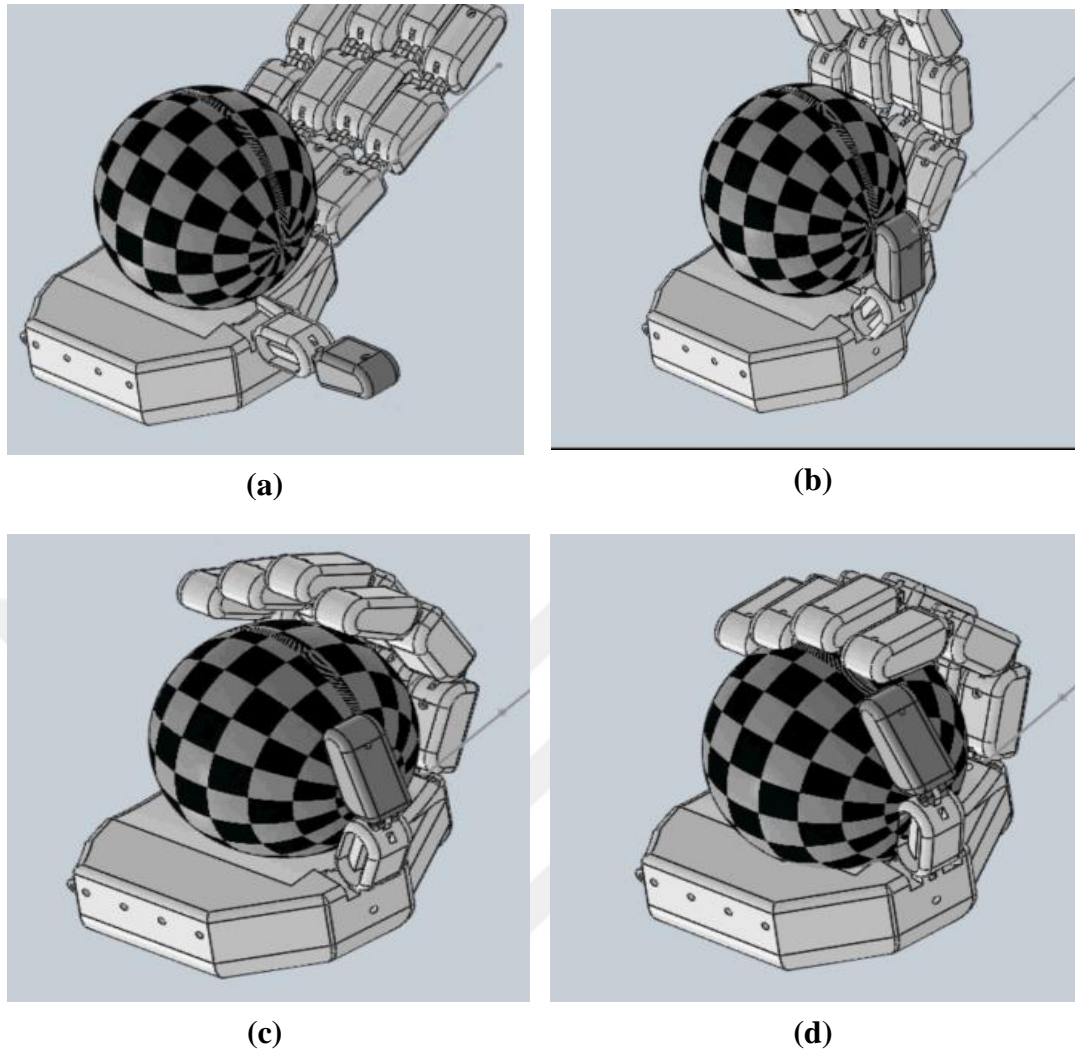


Figure 4.9 :Simulation of grabbing a spherical body. (a) $t=1s$. (b) $t=4s$. (c) $t=6s$. (d) $t=7s$.

4.2.4 Working Area of fingers

Working area of robotic fingers was created on MATLAB. If robotic fingers has three degrees of freedom, working area will be at figure 4.10 and figure 4.11. Codes of figure 4.10 and 4.11 are given in Appendix A. Designed robotic fingers are controlled with two servos, except thumb. Therefore, they have two independent motions. Thumb is controlled with one servo. First servo motor controls the distal phalange and second servo motor controls the proximal phalange. That is why, firstly, distal phalanges start to rotate. When it reaches to 90° degrees, intermediate phalanges start to rotate. In this situation, workspace of robotic fingers will be at figure 4.12 and figure 4.13. Codes, for first and second situation are given in Appendix B, C. For first working area (Figure 4.13), distal and proximal phalange were rotated from 0° to 90° , and intermediate phalange was considered as a fixed part. In Figure 4.15, the distal phalange was

considered as a vertical part of intermediate phalange. Intermediate and proximal phalange were rotated from 0° to 90° degrees.

Firstly, the working area was swept with 0.1 radians (Figure 4.10). However, the swept area was not clear enough. Therefore, the sensitivity was increased and the working area was swept with 0.01 radians (Figure 4.11).

In figure 4.14, Figure 4.13 and Figure 4.12 were combined in MATLAB. Thus, obtained working area of index finger of robotic hand. Robotic finger was accepted as index finger and measurements were given at Table 4.1.

Also to see difference between two working areas, Figure 4.15 was drawn in MATLAB. Combed area is difference between the three DoF and two DoF index finger working area. In addition, some movements of fingers were shown in Table 4.3.

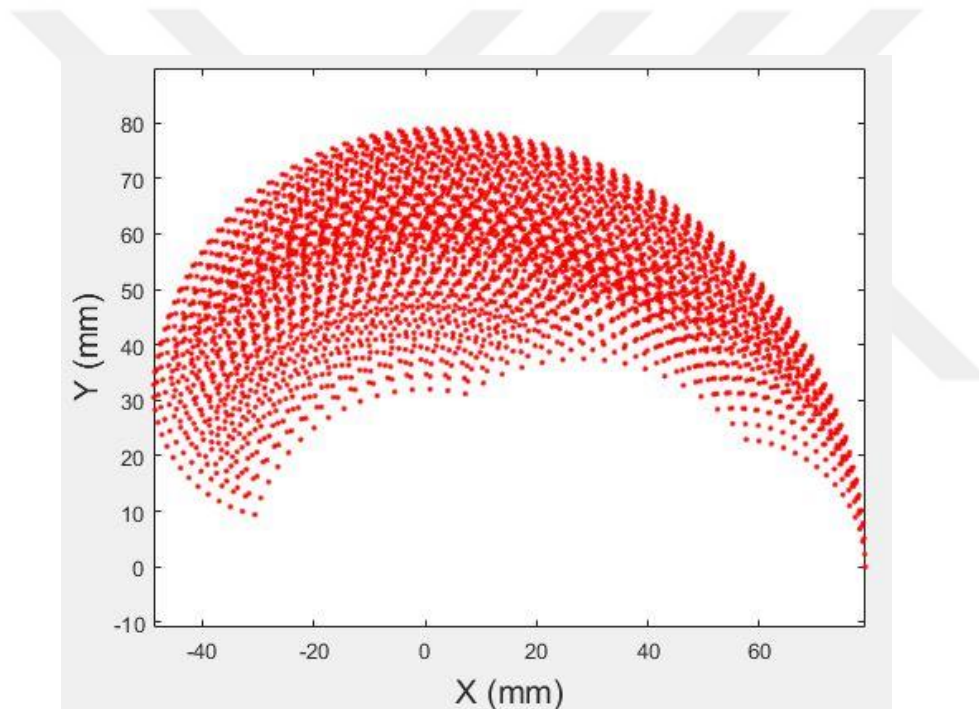


Figure 4.10 : Working area of robotic index finger (0.1 radians).

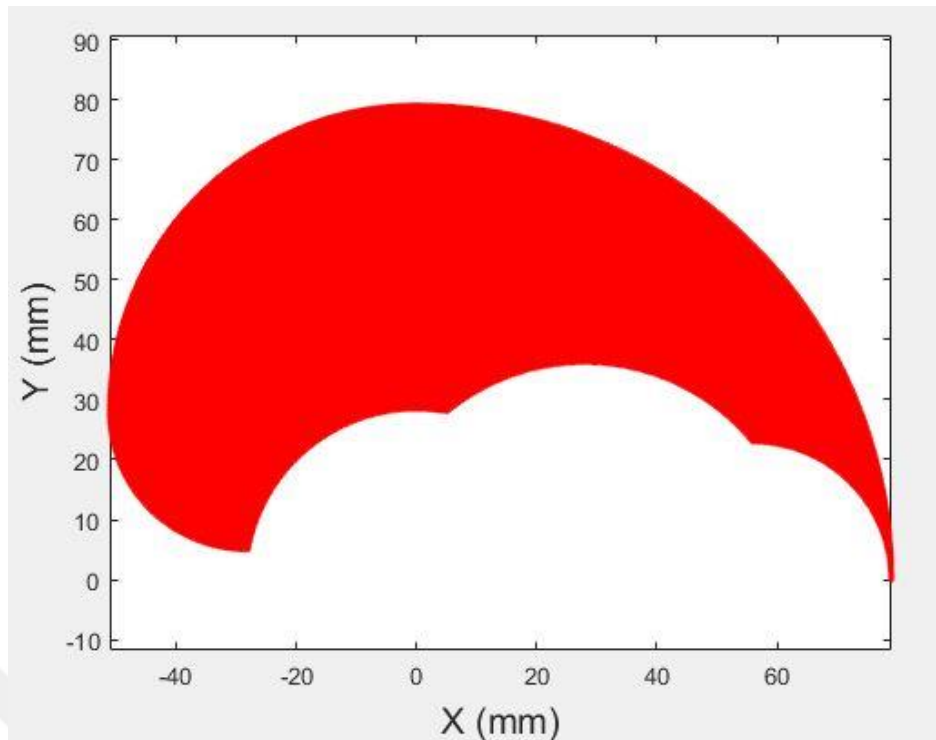


Figure 4.11 : Working area of robotic index finger (0.01radians).

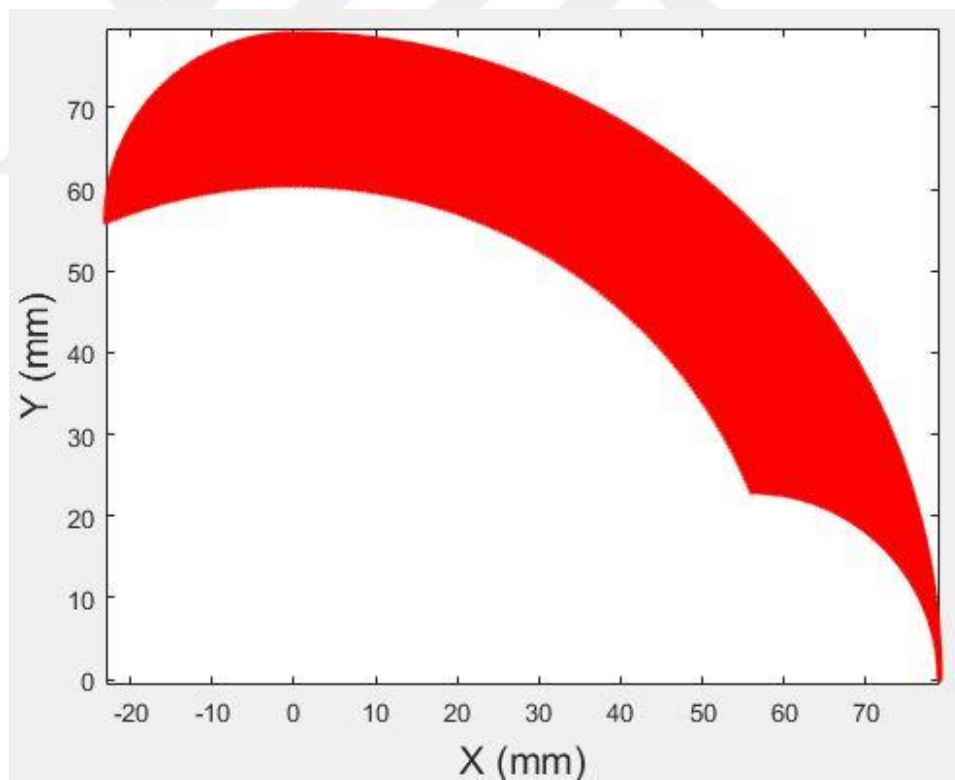


Figure 4.12 : First condition.

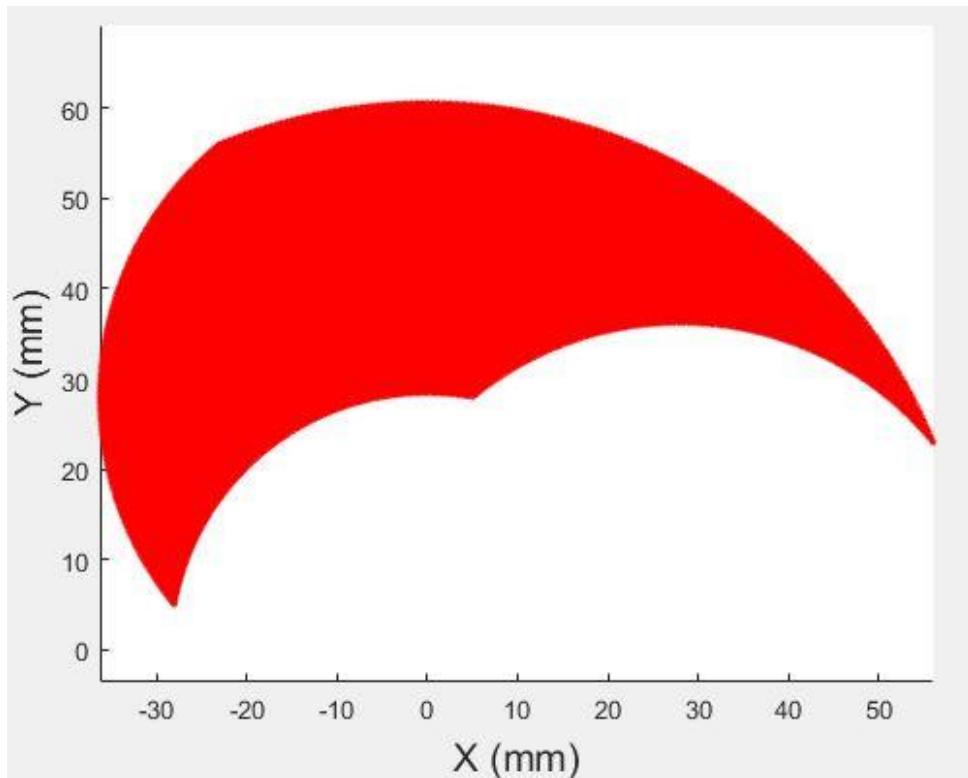


Figure 4.13 : Second condition.

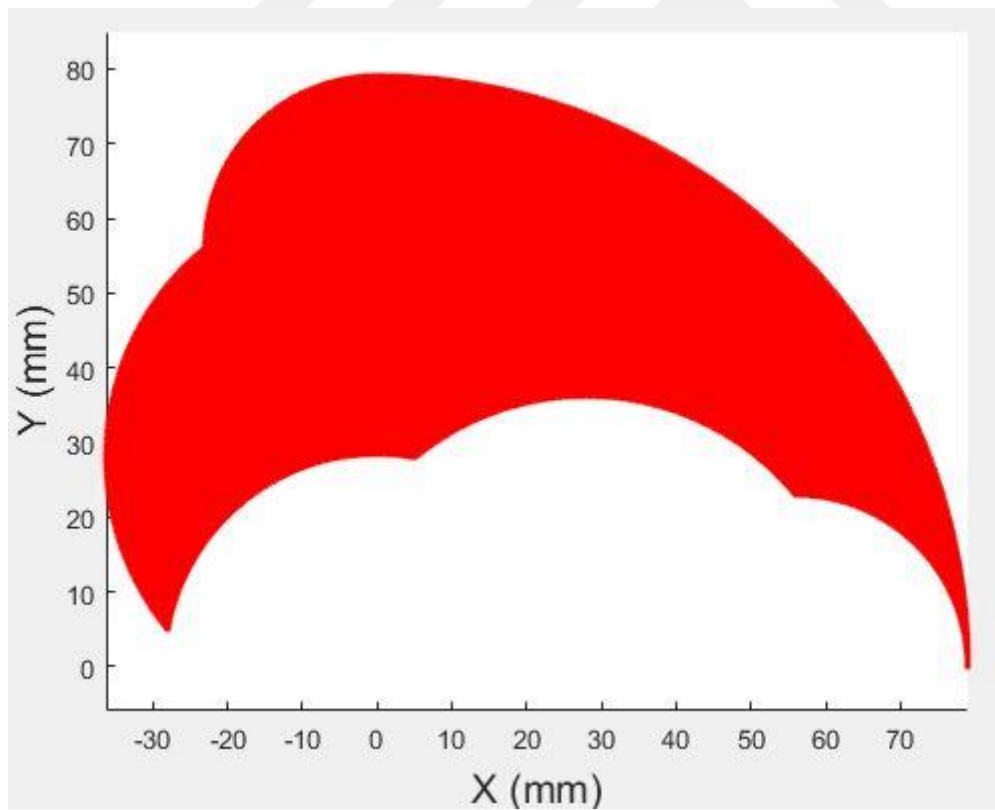


Figure 4.14 : Combination of first and second condition in MATLAB.

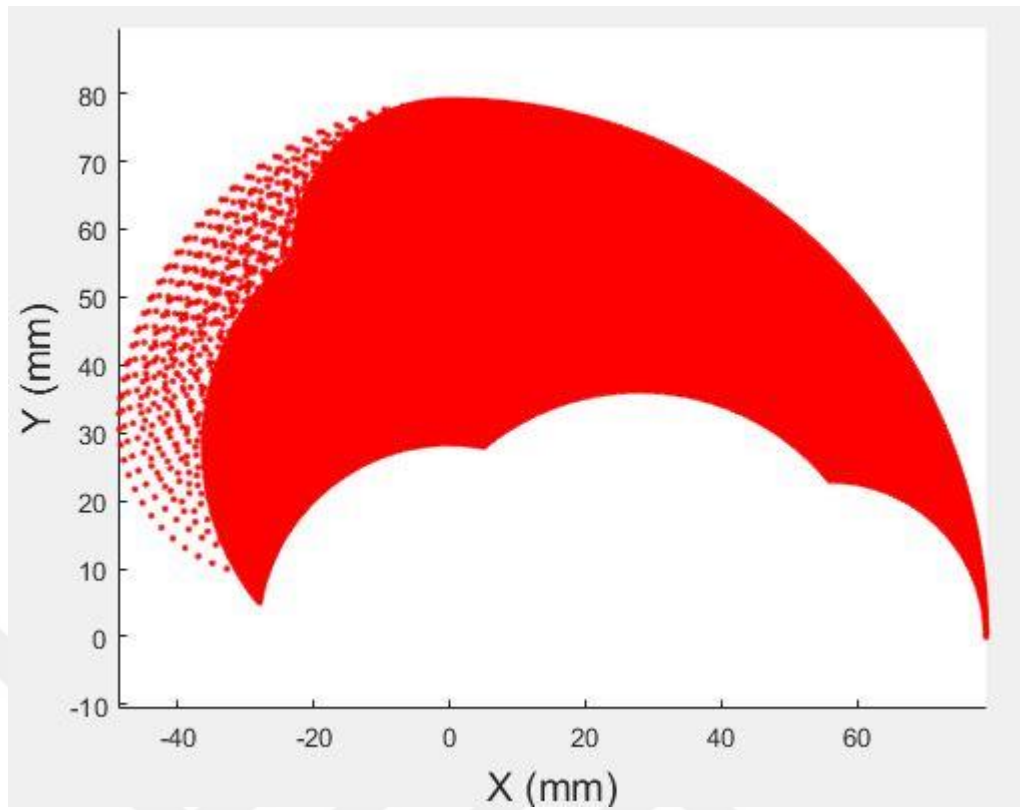
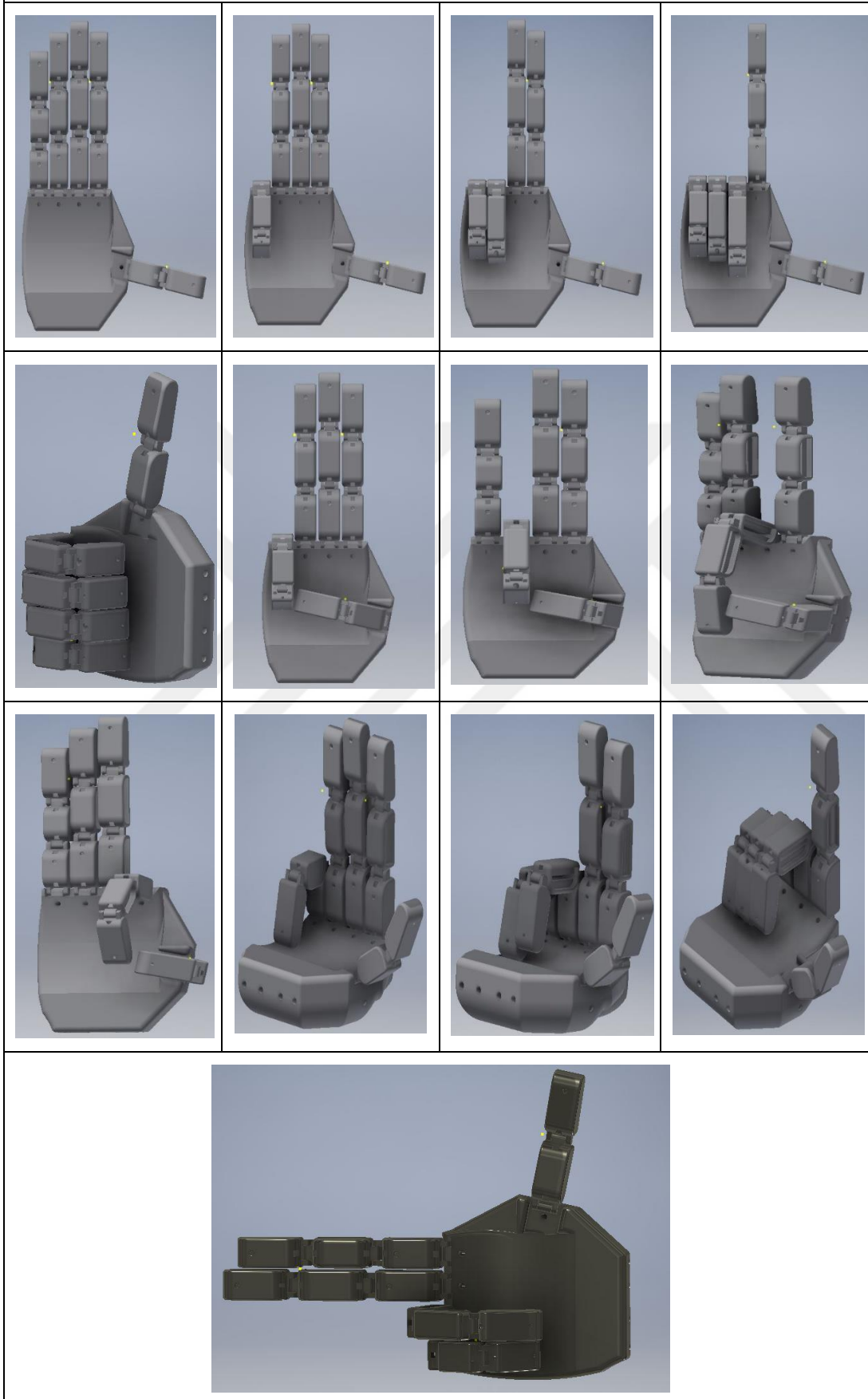


Figure 4.15 : Dotted area is the difference between the 3 DoF and 2 DoF index finger working area.

Table 4.3: Some finger movements of robotic hand.



4.2.5 Inverse kinematic analysis of fingers

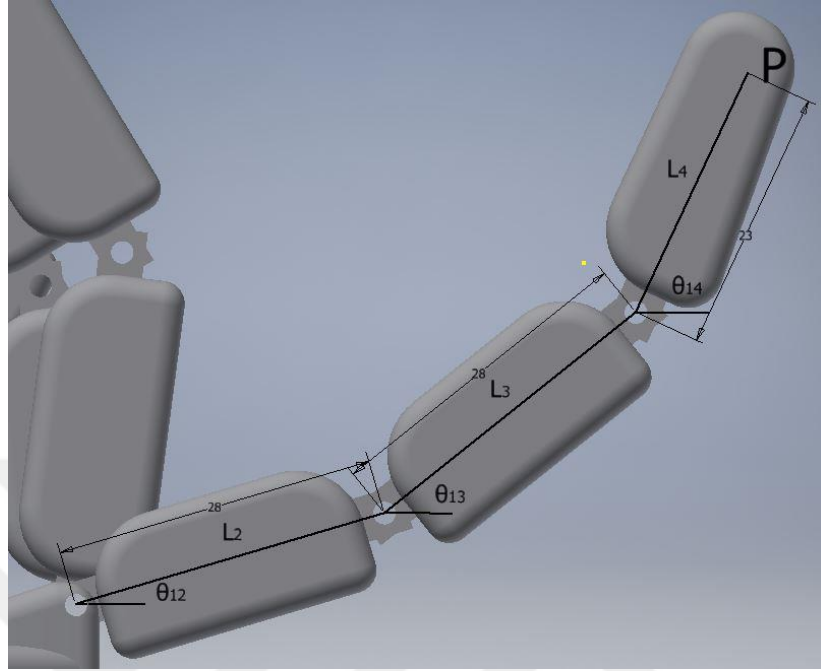


Figure 4.16 : Parameter of Index Finger.

θ_{13} begins to change after θ_{14} reaches to 90° . So;

$$\theta_{14} = \begin{cases} 0 < t < t_1, \theta_{14} \\ t \geq t_1, \theta_{14} = \theta_{13} + 90 \end{cases} \quad (4.3)$$

t_1 will be determined by experiments in experimental test part. θ_{14} changes according to motor speed. So equations (4.1) and (4.2) are used again

$$P_x - L_1 \cos(\theta_{12}(t)) = L_2 \cos(\theta_{13}(t)) - L_3 \cos(\theta_{14}(t)) \quad (4.4)$$

$$P_y - L_1 \sin(\theta_{12}(t)) = L_2 \sin(\theta_{13}(t)) + L_3 \sin(\theta_{14}(t)) \quad (4.5)$$

Square of equations (3.4) and (3.5) are added.

$$(P_x - L_1 \cos(\theta_{12}(t)))^2 + (P_y - L_1 \sin(\theta_{12}(t)))^2 = (L_2 \cos(\theta_{13}(t)) - L_3 \sin(\theta_{13}(t)))^2 + (L_2 \sin(\theta_{13}(t)) + L_3 \cos(\theta_{13}(t)))^2 \quad (4.6)$$

$$L_2^2 + L_3^2 = P_x^2 + L_1^2 + P_y^2 - 2P_x L_1 \cos \theta_{12} - 2P_y L_1 \sin \theta_{12} \quad (4.7)$$

$$\cos \theta_{12} = \frac{1 - t_2^2}{1 + t_2^2} \quad (4.8)$$

$$\sin \theta_{12} = \frac{2t_2}{1 + t_2^2} \quad (4.9)$$

$$t_2 = \tan \frac{\theta_{12}}{2} \quad (4.10)$$

$$0 = -2P_x L_1 \frac{1-t_2^2}{1+t_2^2} - 2P_y L_1 \frac{2t_2}{1+t_2^2} - L_2^2 - L_3^2 + P_x^2 + L_1^2 + P_y^2 \quad (4.11)$$

$$0 = -2P_x L_1 (1-t_2^2) - 4P_y L_1 t_2 + (1+t_2^2)(-L_2^2 - L_3^2 + P_x^2 + L_1^2 + P_y^2) \quad (4.12)$$

$$0 = t_2^2 (-L_2^2 - L_3^2 + P_x^2 + L_1^2 + P_y^2 + 2P_x L_1) - 4P_y L_1 t_2 + (-L_2^2 - L_3^2 + P_x^2 + L_1^2 + P_y^2 - 2P_x L_1) \quad (4.13)$$

$$A = (-L_2^2 - L_3^2 + P_x^2 + L_1^2 + P_y^2 + 2P_x L_1) \quad (4.14)$$

$$C = (-L_2^2 - L_3^2 + P_x^2 + L_1^2 + P_y^2 - 2P_x L_1) \quad (4.15)$$

$$t_2 = \frac{4P_y L_1 \pm \sqrt{(-4P_y L_1)^2 - 4AC}}{2A} \quad (4.16)$$

$$\theta_{12} = 2 \arctan(t_2) \quad (4.17)$$

$$\theta_{13} = \arctan 2 \left(\frac{(P_x - L_1 \cos(\theta_{12}))L_2 + L_3(P_y - L_1 \sin(\theta_{12}))}{\cos(\theta_{13})L_2^2 + L_3^2 \cos(\theta_{13})}, \frac{-(P_x - L_1 \cos(\theta_{12}))L_3 + L_2(P_y - L_1 \sin(\theta_{12}))}{-L_3^2 \sin(\theta_{13}) + L_2^2 \sin(\theta_{13})} \right), \quad (4.18)$$

4.2.6 Controlling of robotic hand

Controlling of robotic hand was provided via servo motors. Nine servo motors were used. In addition to robotic hand, another part was designed to position servo motors like a wrist (Figure 4.17).

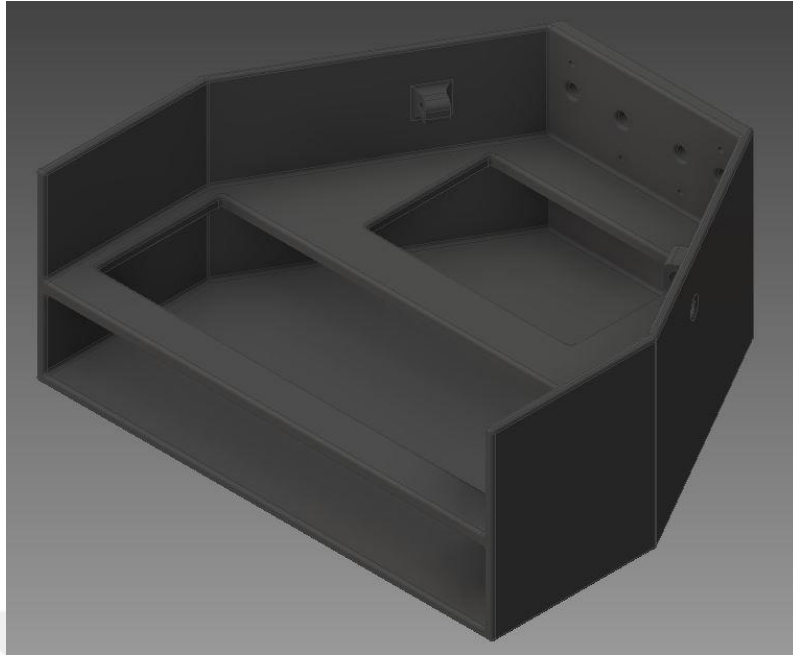


Figure 4.17 :Wrist to position servo motors.

In addition, MG 996R servo motors and wrist were assembled in Inventor for checking the compatibility of parts (Figure 4.18).



Figure 4.18 : Assembling of servo motors and wrist.



5. CHARACTERIZATION AND CALIBRATION OF SENSORS

Flexible sensor acts like a potentiometer. Its resistance changes depending on bend radius of it. Resistance value of flexible sensor is 25K ohms while it is on flat position. Resistance range of flexible sensor changes between 45K and 125K ohms depending on bend radius of it [51].

In this study, five flexible sensors were used to measure angles of real hand fingers. Flexible sensor gives an output voltage in range of 5V – 0V according to bend radius (Figure 5.1). Resistance tolerance of flexible sensors is $\pm 30\%$ [51]. This tolerance value is too much. Firstly, to avoid the tolerance value characterizing studies were done. Some researches were made and formulas were tried to find out. Finally, calibrations were made.

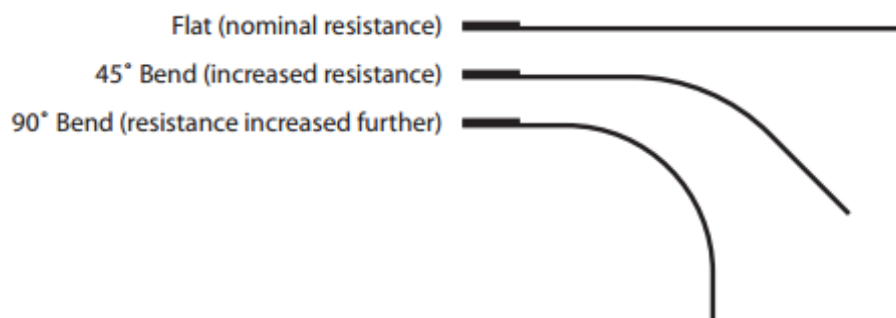


Figure 5.1 : The working principle of the flexible sensors [51].

In order to read sensor output, the output pin of the flexible sensors was attached on one of the analog input pins of Arduino Mega ADK. Arduino analog pins give a 10-bit analog output value between 0 and 1023 which changes according to 0V to 5V. Since sensor does not give directly the bend angle output, the output value of sensor should be converted to meaningful angle output.

To take a meaningful formula, bend position of flexible sensor were tried in different types of equations and graphs. The analog output values of flexible sensor were recorded for different bend positions. 10k ohm resistor was connected for each experiment. The software Mathematica was used to find a curve fit function and plot function for the recorded data.

There are white and black nodes on flexible sensor (Figure 5.2). Apart from pins which are on flexible sensor, measurements can be taken from these nodes. White nodes on flexible sensor can be considered as a resistance and these are connected as serially. First white square, which is closed to the pins have the highest resistance. Because it is far away to the ground pin.

Also flexible sensor can be thought as a rubber. If rubber is stretch it will be thin. Resistance on thin wire is more according to thick wire. That's why, if flexible sensor is bended resistance will increase (Figure 5.1).

Firstly, flexible sensor positioned on flat position. Values on Table 5.1 were created, taken from on each white square of flexible sensor while it was on flat position. There is approximately 4,2 – 4,3 mm distance between centers of each white square. Therefore, on X – Y plane beginning point was chosen zero and first white square of flexible sensor positioned to 0 point. Then measurements were taken at 4,2 – 4,3 mm intervals. Table 5.2 shows the formula and graphic for flat position measurements.

Table 5.1 :Analog output values of flexible sensor.

Flat Position Measurements on Each White Square			
Centimeter	Measurement	Centimeter	Measurement
0	636	4.75	480
0.4	627	5.2	455
0.9	615	5.6	430
1.3	605	6	390
1.75	590	6.45	358
2.15	578	6.85	315
2.6	560	7.3	257
3	545	7.75	257
3.45	527	8.3	166
3.9	508	8.6	120
4.3	495	9	67

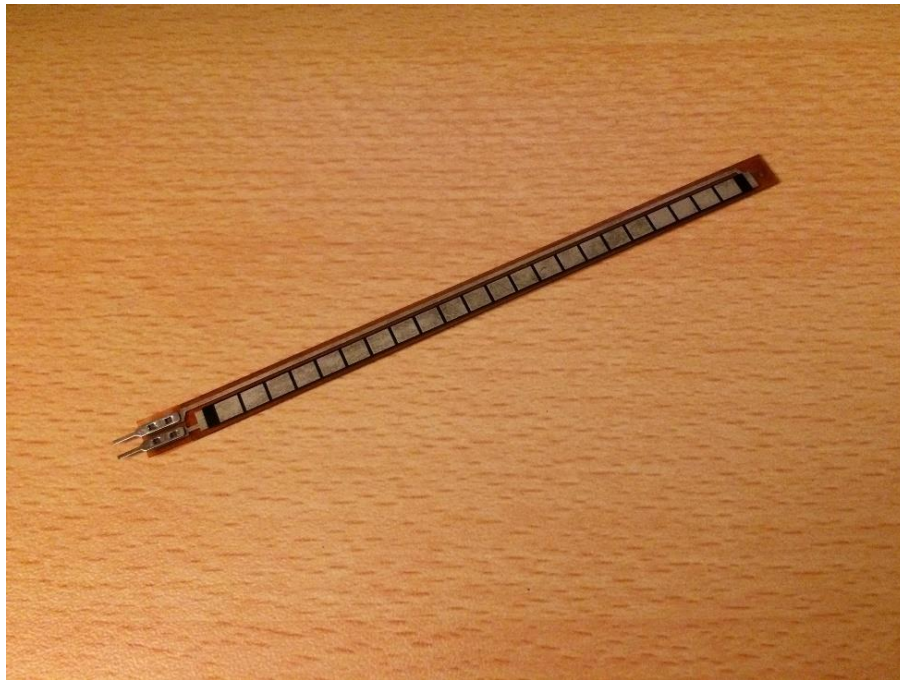


Figure 5.2 : Flexible sensor.

Table 5.2 : Formula and graphic according to flat position.

Graphic	<p>The graph displays a series of data points (blue dots) and a fitted curve (blue line). The x-axis is labeled 'cm' and ranges from 0 to 10. The y-axis is labeled 'V(scaled)' and ranges from 0 to 600. The data points are approximately as follows:</p> <table border="1"> <thead> <tr> <th>cm</th> <th>V(scaled)</th> </tr> </thead> <tbody> <tr><td>0</td><td>636</td></tr> <tr><td>0.4</td><td>627</td></tr> <tr><td>0.9</td><td>615</td></tr> <tr><td>1.3</td><td>605</td></tr> <tr><td>1.75</td><td>590</td></tr> <tr><td>2.15</td><td>578</td></tr> <tr><td>2.6</td><td>560</td></tr> <tr><td>3</td><td>545</td></tr> <tr><td>3.45</td><td>527</td></tr> <tr><td>3.9</td><td>508</td></tr> <tr><td>4.3</td><td>495</td></tr> <tr><td>4.75</td><td>480</td></tr> <tr><td>5.2</td><td>455</td></tr> <tr><td>5.6</td><td>430</td></tr> <tr><td>6</td><td>390</td></tr> <tr><td>6.45</td><td>358</td></tr> <tr><td>6.85</td><td>315</td></tr> <tr><td>7.3</td><td>257</td></tr> <tr><td>7.75</td><td>212</td></tr> <tr><td>8.3</td><td>166</td></tr> <tr><td>8.6</td><td>120</td></tr> <tr><td>9</td><td>67</td></tr> </tbody> </table>	cm	V(scaled)	0	636	0.4	627	0.9	615	1.3	605	1.75	590	2.15	578	2.6	560	3	545	3.45	527	3.9	508	4.3	495	4.75	480	5.2	455	5.6	430	6	390	6.45	358	6.85	315	7.3	257	7.75	212	8.3	166	8.6	120	9	67
cm	V(scaled)																																														
0	636																																														
0.4	627																																														
0.9	615																																														
1.3	605																																														
1.75	590																																														
2.15	578																																														
2.6	560																																														
3	545																																														
3.45	527																																														
3.9	508																																														
4.3	495																																														
4.75	480																																														
5.2	455																																														
5.6	430																																														
6	390																																														
6.45	358																																														
6.85	315																																														
7.3	257																																														
7.75	212																																														
8.3	166																																														
8.6	120																																														
9	67																																														
Codes	<pre> data = {{0, 636}, {0.4, 627}, {0.9, 615}, {1.3, 605}, {1.75, 590}, {2.15, 578}, {2.6, 560}, {3, 545}, {3.45, 527}, {3.9, 508}, {4.3, 495}, {4.75, 480}, {5.2, 455}, {5.6, 430}, {6, 390}, {6.45, 358}, {6.85, 315}, {7.3, 257}, {7.75, 212}, {8.3, 166}, {8.6, 120}, {9, 67}}; fit = FindFormula[data, x] 619.4667271781079 - 6.66061954665892x² Show[ListPlot[data], Plot[fit, {x, -20, 60}, PlotRange -> All]] </pre>																																														

Values on Table 5.3, were taken from $y = x^2$ and $y = x^3 + 2x^2 + 3$ graphics (Figure 5.3). Flexible sensor was positioned on pertinax on top of the real lengths of the graphics. From specific points as $x = -2, x = -1, x = 0, x = 1$ and $x=2$, measurements were taken and created Table 5.3. Table 5.4 and table 5.5. show graphics and formulas of taken measurements. Flexible sensor couldn't reach to $x=2$ point. That is why, from $x=2$ point, measurement couldn't taken for $y = x^3 + 2x^2 + 3$ graph.

Table 5.3 : Analog output values of flexible sensor according to graphs.

According to $y = x^2$ graphic		According to $y = x^3 + 2x^2 + 3$ graphic	
X	Y	X	Y
-2	968	-2	943
-1	805	-1	814
0	650	0	652
1	558	1	515
2	380		

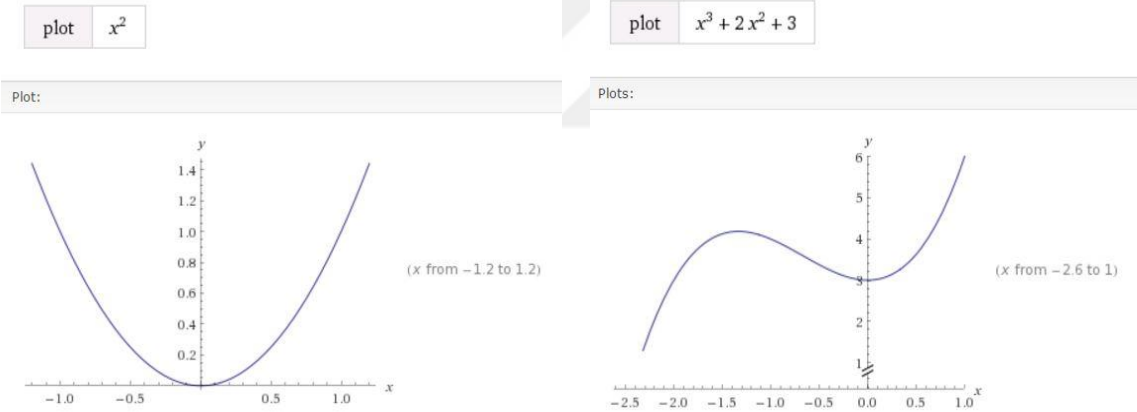


Figure 5.3 : $y = x^2$ and $y = x^3 + 2x^2 + 3$ graphs.

Some formulas and graphs were created according to Table 5.3 in Mathematica (Table 5.4, Table 5.5).

Table 5.4 : Formula and graphic according to $y = x^2$ equation.

Graphic	
Codes	<pre> data = {{-2, 968}, {-1, 805}, {0, 650}, {1, 558}, {2, 380}}; fit = FindFormula[data, x] 650.0000000000002 - 115.66666666666663x + 39.99999999999936x² - 7.833333333333342x³ - 8.49999999999986x⁴ Show[ListPlot[data], Plot[fit, {x, -20, 60}, PlotRange -> All]] </pre>

Table 5.5 : Formula and graphic according to $y = x^3 + 2x^2 + 3$ equation.

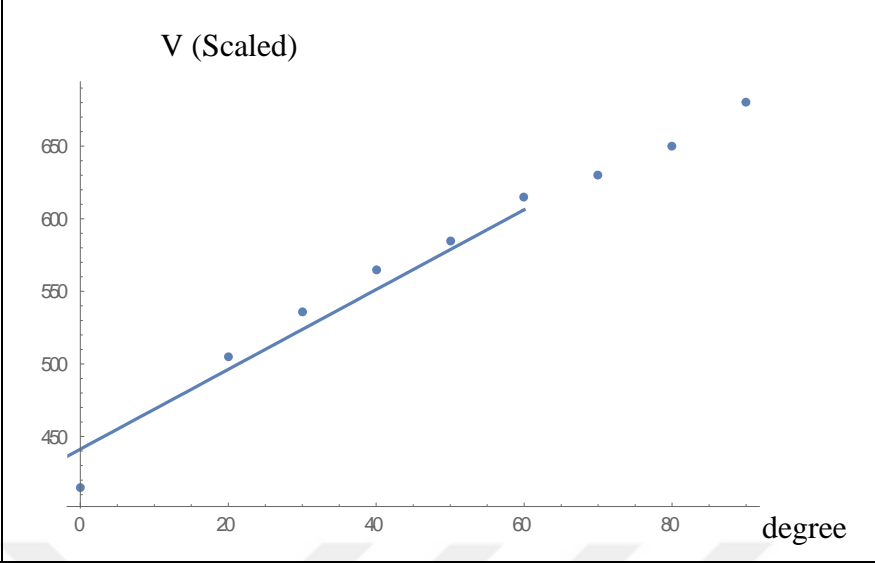
Graphic	
Codes	<pre> data = {{1, 515}, {0, 652}, {-1, 814}, {-2, 943}}; fit = FindFormula[data, x] 652.00000000000001 - 159.16666666666657x + 12.499999999999915x² + 9.666666666666613x³ Show[ListPlot[data], Plot[fit, {x, -20, 60}, PlotRange -> All]] </pre>

In another experiment, flexible sensor was fixed on the pertinax. It was bended from starting of the flexible sensor. Measurements were taken from this point for every 10 degree. Values on Table 5.6, shows the value which, changes for every 10 degrees. Table 5.7 shows the formula and graphic of this experiment.

Table 5.6 : Measurement results of every 10 degree

Degrees	Measurements	Degrees	Measurements
0	415	60	615
20	505	70	630
30	536	80	650
40	565	90	680
50	585		

Table 5.7 : Formulas and graphics for every 10 degree angle.

Graphic	<p style="text-align: center;">V (Scaled)</p> 
Codes	<pre>data = {{0, 415}, {20, 505}, {30, 536}, {40, 565}, {50, 585}, {60, 615}, {70, 630}, {80, 650}, {90, 680}}; fit = FindFormula[data, x] 441.2774193548385 + 2.748870967741939x Show[ListPlot[data], Plot[fit, {x, -20, 60}, PlotRange -> All]]</pre>

According to table 5.7 flexible sensors act like a linear.

In another experiment, flexible sensor was placed on x^2 and $\frac{x^2}{25}$ graphics (Figure 5.5a and Figure 5.5b). Instead of taking the values at the specific distance of x axis, some nodes were chosen on flexible sensor (Table 5.8, table 5.9). Measurements were taken from same nodes on flexible sensor for both graphics and X-axis trace points were found.

There are 22 white nodes on flexible sensor (Figure 5.2). Between each white nodes there are little black nodes. As “node 5, 6, 7” values were taken from white nodes, half values as “node 1.5, 2.5, 3.5” were taken from black nodes.

Table 5.8 and 5.9 consist of measurements some points on flexible sensor.

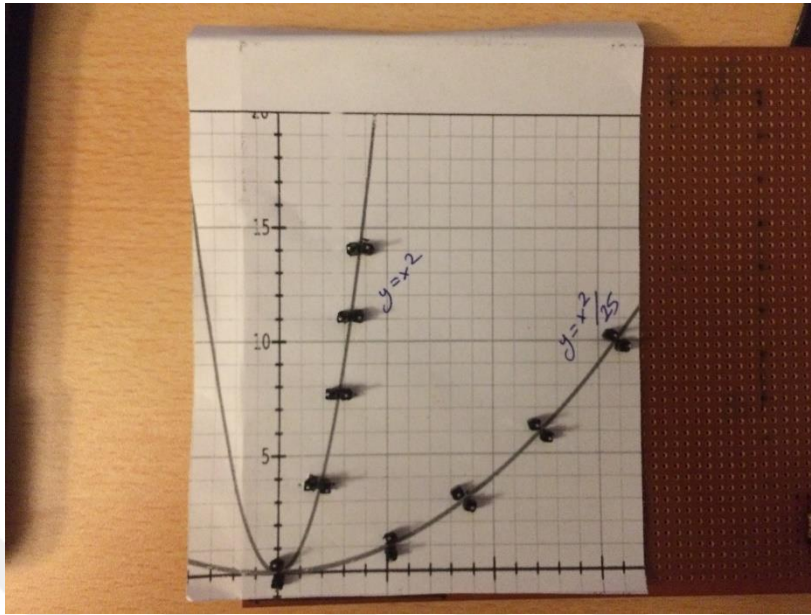
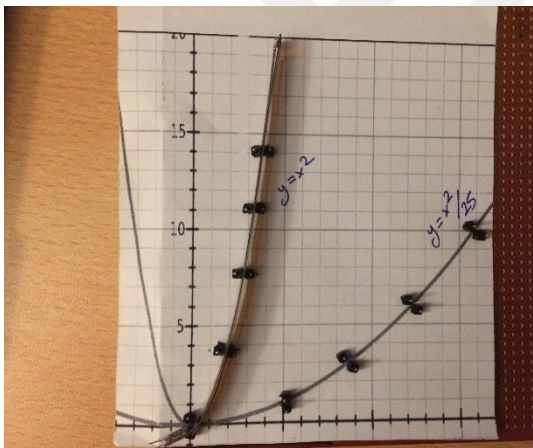
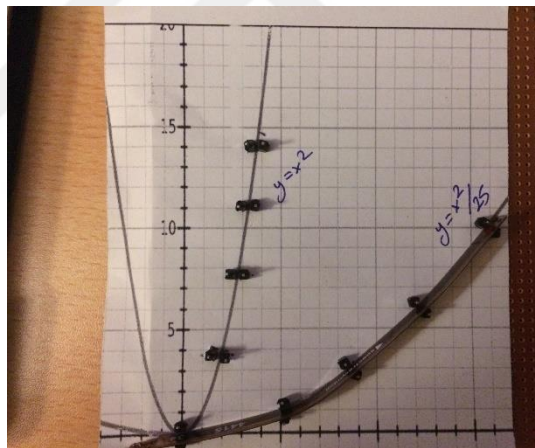


Figure 5.4 : Experimental setup without flex sensor.



(a)



(b)

Figure 5.5 : Experimental setup with flex sensor (a) ($y = x^2$). (b) ($y = x^2/25$).

Table 5.8 : Measurement results of the equation $y = x^2/25$.

Distance	Node	Measurement
0,5 cm	1,5	483
0,9 cm	2,5	464
1,4 cm	3,5	451
2 cm	5	437
2,3 cm	6	425
2,7 cm	7	410
3,4 cm	8,5	381
3,7 cm	9,5	366
4,3 cm	11	351
4,7 cm	12,5	310
5,25 cm	14	287
5,75 cm	15,5	241
6,4 cm	17,5	188
6,8 cm	19	158
7,2 cm	21	93

Table 5.9 : Measurement results of the equation $y = x^2$.

Distance	Node	Measurement
0,4 cm	1,5	495
0,6 cm	2,5	480
0,8 cm	3,5	467
0,9 cm	5	457
1 cm	6	443
1,1 cm	7	427
1,2 cm	8,5	398
1,3 cm	9,5	380
1,4 cm	11	360
1,5 cm	12,5	320
1,6 cm	14	300
1,7 cm	15,5	250
1,8 cm	17,5	196
1,9 cm	19	166
2 cm	21	94

According to these measurements, formulas were created (Table 5.10). dat1 was written according to table 5.8 $y = x^2/25$, dat2 was written according to table 5.9 $y = x^2$. Graphics were similar to x^2 and $\frac{x^2}{25}$ graphics. Reason of the inverse to x-axis, flexible sensors values continue to decrease from first point to last point. After this result, formulas were fitted to x^2 . Yellow line for x^2 , blue line for $\frac{x^2}{25}$.

Table 5.10 : Formulas and graphics according to $y = x^2/25$ and $y = x^2$ equations.

Graphic	
Codes	<pre> dat1 = {{7.2, 93}, {6.8, 158}, {6.4, 188}, {5.75, 241}, {5.25, 287}, {4.7, 310}, {4.3, 351}, {3.7, 366}, {3.4, 381}, {2.7, 410}, {2.3, 425}, {2.437}, {1.4, 451}, {0.9, 464}, {0.5, 483}} dat2 = {{2.94}, {1.9, 166}, {1.8, 196}, {1.7, 250}, {1.6, 300}, {1.5, 320}, {1.4, 360}, {1.3, 380}, {1.2, 398}, {1.1, 427}, {1, 443}, {0.9, 457}, {0.8, 467}, {0.6, 480}, {0.4, 495}} ListPlot[{dat1, dat2}, PlotJoined -> True] fit = FindFormula[dat1,x] Fit[dat1,{-x^2},x] 468.1627065387376 - 6.936076038096299x^2 6.954619033537551x^2 fit = FindFormula[dat2,x] Fit[dat2,{-x^2},x] 491.86324229549484 - 49.20058341206586x^3 105.93460840367904x^2 </pre>

After this equations, different multiples of x^2 . $\frac{x^2}{8}$ and $\frac{x^2}{16}$ were tried and other formulas were discovered (Figure 5.6).

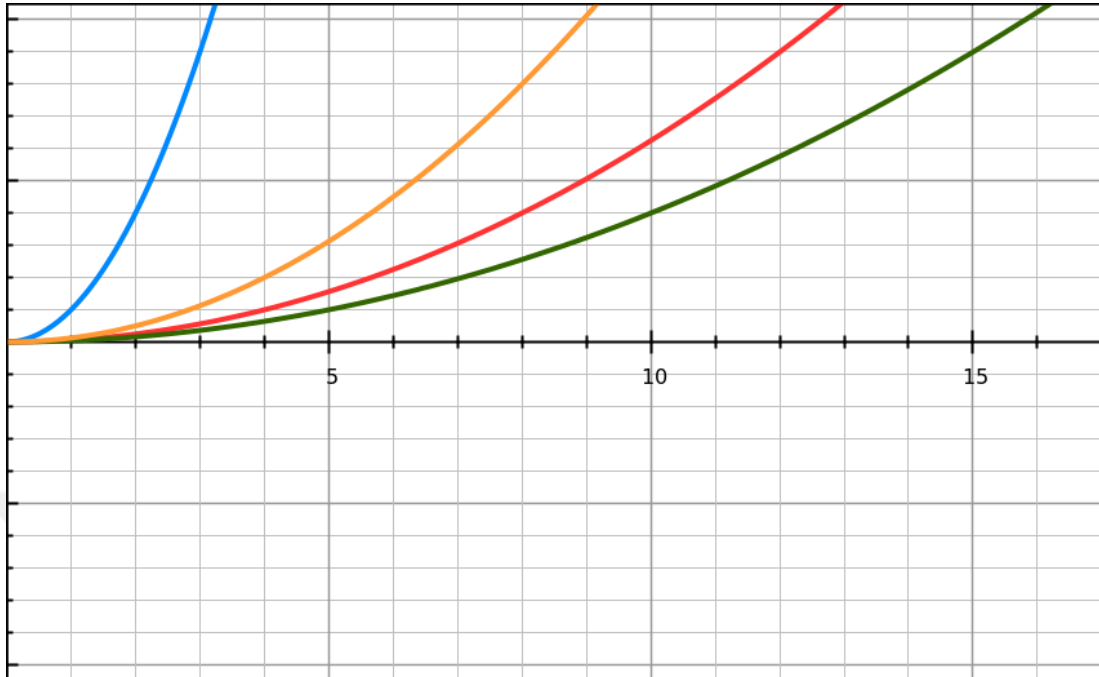


Figure 5.6 :Blue = x^2 , Yellow = $\frac{x^2}{8}$, Red= $\frac{x^2}{16}$, Green = $\frac{x^2}{25}$.

Flexible sensor was placed on $\frac{x^2}{8}$ and $\frac{x^2}{16}$ graphics (Figure 5.8a and 5.8b). Measurements were taken from same nodes on flexible sensor for four graphics. Table 5.11 and 5.12 were created according to measurements.

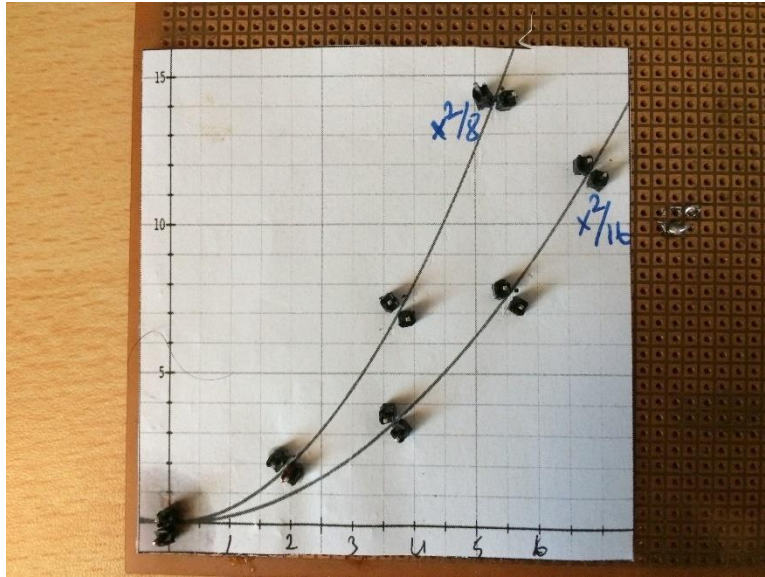
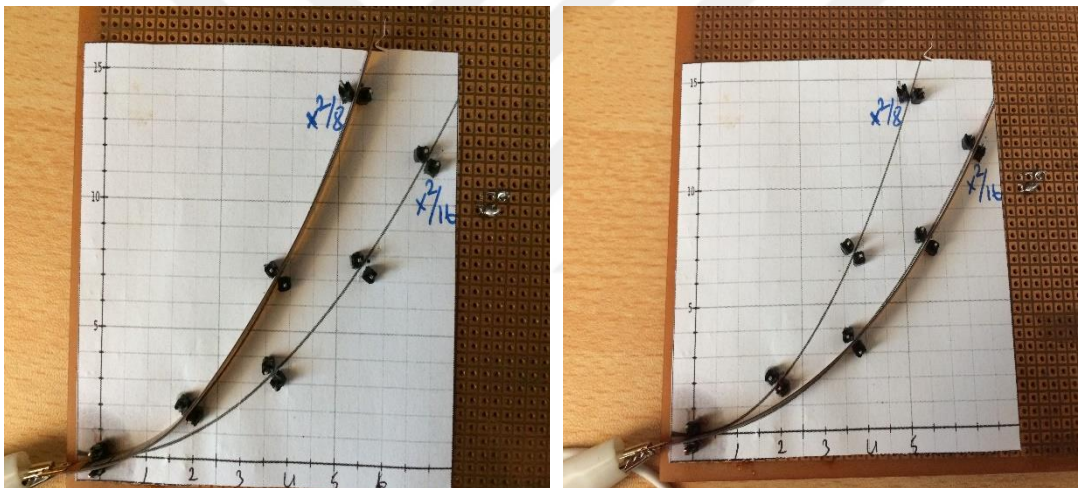


Figure 5.7 : Experimental setup without flex sensor ($\frac{x^2}{8}, \frac{x^2}{16}$).



(a)

(b)

Figure 5.8: Experimental setup with flex sensor. (a) $y = x^2/8$. (b) $y = x^2/16$.

Table 5.11 : Measurement results of the equation $y = \frac{x^2}{8}$.

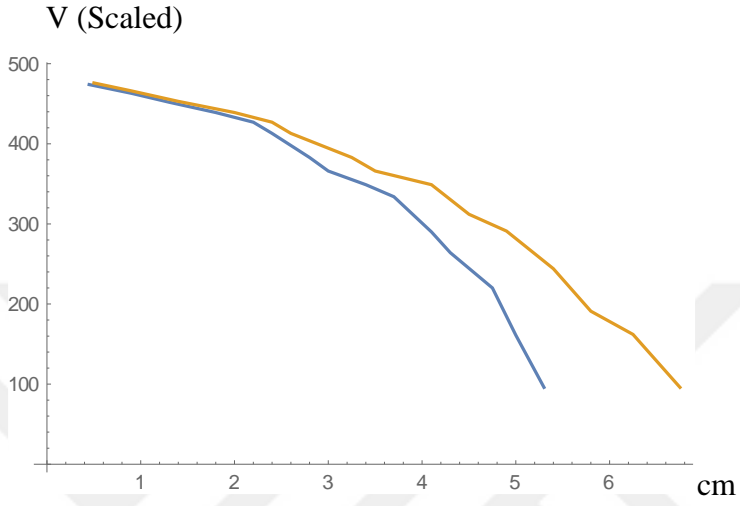
Distance	Node	Measurement
0,45 cm	1,5	474
0,9 cm	2,5	463
1,3 cm	3,5	452
1,8 cm	5	439
2,2 cm	6	427
2,4 cm	7	413
2,8 cm	8,5	383
3 cm	9,5	366
3,4 cm	11	349
3,7 cm	12,5	334
4,1 cm	14	290
4,3 cm	15,5	264
4,75 cm	17,5	220
5 cm	19	161
5,3 cm	21	96

Table 5.12 : Measurement results of the equation $y = \frac{x^2}{16}$.

Distance	Node	Measurement
0,5 cm	1,5	476
0,9 cm	2,5	466
1,4 cm	3,5	453
2 cm	5	439
2,4 cm	6	427
2,6 cm	7	413
3,25 cm	8,5	383
3,5 cm	9,5	366
4,1 cm	11	349
4,5 cm	12,5	312
4,9 cm	14	291
5,4 cm	15,5	244
5,8 cm	17,5	191
6,25 cm	19	162
6,75 cm	21	96

According to tables 2 different formulas were created and fitted to x^2 (Table 5.13). dat1 is measurement of $\frac{x^2}{8}$ and dat2 is measurement of $\frac{x^2}{16}$. According to these formulas, graphics were drawn. These graphics are similar to $\frac{x^2}{8}$ and $\frac{x^2}{16}$.

Table 5.13 : Formulas and graphics according to $y = x^2/8$ and $y = x^2/16$ equations.

Graphic	
Codes	<pre> dat1 = {{5.3,96}, {5, 161}, {4.75, 220}, {4.3, 264}, {4.1, 290}, {3.7, 334}, {3.4, 349}, {3, 366}, {2.8, 383}, {2.4, 413}, {2.2, 427}, {1.8, 439}, {1.3, 452}, {0.9, 463}, {0.45, 474}} fit = FindFormula[dat1,x] Fit[dat1,{-x^2},x] 462.2544858647029 - 4.574319352010623x^{2.6} 14.098556277871614x² dat2 = {{6.75,96}, {6.25, 162}, {5.8, 191}, {5.4, 244}, {4.9, 291}, {4.5, 312}, {4.1, 349}, {3.5, 366}, {3.25, 383}, {2.6, 413}, {2.4, 427}, {2, 439}, {1.4, 453}, {0.9, 466}, {0.5, 476}} fit = FindFormula[dat2,x] Fit[dat2,{-x^2},x] 468.9942903049116 - 6.677200728827877x^{2.1} 8.36360263829941x² ListPlot[{dat1, dat2}, PlotJoined -> True] </pre>

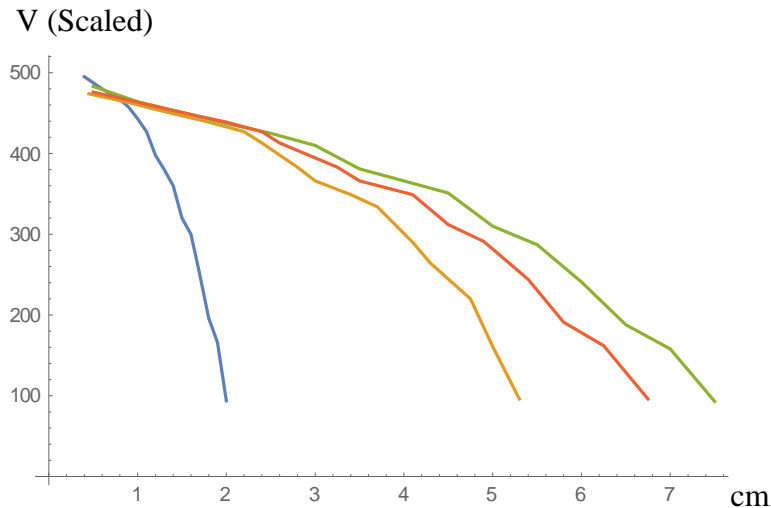


Figure 5.9 : All graphs of measurement results. Blue = x^2 , Yellow = $\frac{x^2}{8}$, Red = $\frac{x^2}{16}$, Green = $\frac{x^2}{25}$.

Figure 5.9 shows measurements of different multiples of x^2 and all of them similar to different multiples of x^2 .

After the experiments in different types of equations and shapes, some formulas were found. In addition, pointed out that each flexible sensor has own characteristic. Also, because of the long-term usage of the flexible sensor, some deformations occur on flexible sensor. So, range of measured values changes. For this reason, before the control robotic hand, for four different positions of human hand calibrations values are taken.

Firstly, flexible sensors were sewn on gloves on each finger (Figure 5.10). According to Table 5.6 and Table 5.7, flexible sensor gives a result, which is close to linear for angular experiment. That is why, according to results of the calibrations taken in four different hand shapes are mapped in Arduino.

From flexible sensors, two different values are taken from two different points. One of them is taken from proximal phalange. Other one is taken from metacarpal. Crocodile weren't connected on proximal phalanges on these figures to see calibration positions clearly. First values are taken while hand is on flat position (Figure 5.11a). Second values are taken while fingers are bended from proximal interphalangeal (Figure 5.11b). Reason of this calibration, if flexible sensor is bended from proximal phalange, taken two values from proximal and metacarpal phalange increases. Third values are taken while hand is bended from metacarpal phalange (Figure 5.11c). Reason of this, to see difference between taken values from second calibration and third calibration.

Last calibration is taken while hand is fist position (Figure 5.11d). Reason of this, to see maximum values.



Figure 5.10 : Flexible sensors on gloves.



(a)



(b)



(c)



(d)

Figure 5.11 : Calibration positions (a) while fingers on flat position. (b) fingers are bended from the PIP. (c) fingers are bended from MCP. (d) Hand position as a fist.



6. PROGRAMMING THE ROBOTIC HAND

6.1 Arduino

Arduino is a prototype platform based on open source, easy-to-use software and hardware [44]. Compatible with almost all programs (Labview, MATLAB...). Since it is compatible with a lot of components, many different projects can be made. (Servo motor, temperature sensor, humidity sensor, LED, LCD, XBee etc.)

Two Arduino Mega were used to control the designed robotic hand (Figure 6.1). One of them provides movement of fingers of robotic hand, controls the servo motors (Arduino Mega Receiver). Other one (Arduino Mega Transmitter) senses the movements of the fingers from real hand then sent them to other Arduino Mega Receiver with wireless communication platform XBee.

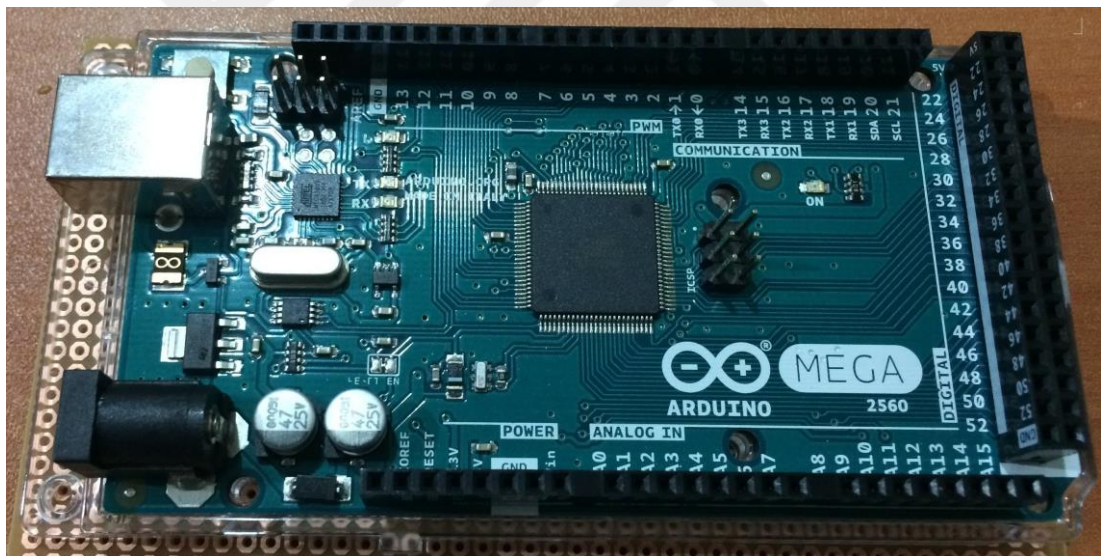


Figure 6.1 : Arduino Mega.

6.1.1 Arduino mega

The Arduino Mega is a microcontroller based on the ATmega2560. It has 54 digital input/output pins (of which 15 can be used as PWM outputs). Also it has 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button [44]. PWM is used in many applications such as communication, sound control, motor drive. 5V and 3.3V pins are used for power supply. GND is short for "Ground". AREF is short for "Analog

Reference”. It is used for set an external reference voltage (between 0 and 5 Volts) as the upper limit for the analog input pins. There are four TX and RX pins for serial communication. TX is short for “Transmit” and RX is short for “Receive”.

6.2 The Used Programming Software

Arduino Software (IDE-integrated development environment) was used to write the control code of robotic hand. The Arduino software is written in Java. Inside of software, there is a text editor to write the codes. There is a message box to show error codes and to show Arduino capacity after the codes are loaded. There is a toolbar with buttons for common functions and a series of menus. The software can connect to Arduino boards and upload the written codes to them.

Arduino software consists of two main parts, void setup and void loop. The codes that enough to be run once, such as defining the pins, opening the serial port, are written in the void setup. The codes that need to be repeated continuously are written in the loop. Commands that need to be defined, such as defining variables and importing libraries, are written before setup (Figure 6.2).



```
sketch_nov21a | Arduino 1.6.8
Dosya Düzenle Taslak Araçlar Yardım

sketch_nov21a
#include <XBee.h>
#include <Servo.h>

void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
  pinMode(2, OUTPUT);
  pinMode(3, INPUT);
}

void loop() {
  // put your main code here, to run repeatedly:
}

Derleme tamamlandı.
Çalıştırılan programın 3.414 bayt (1 %) saklama alanını kullandı. Makine Global değişkenler belleğinin 335 byte kadarını (4%) kullanıyor.

15 Arduino/Genuino Mega or Mega 2560, ATmega2560 (Mega 2560) on COM3
```

Figure 6.2 : Arduino software.

The software also allows the installation of the codes on the Arduino device. If more than one Arduino device is connected, the device on which the codes are installed should be selected. The device must be selected from the tools menu to compile the codes for the appropriate Arduino device.

7. DESIGN OF ARDUINO CIRCUIT

In design, two Arduino platform were used. One of them (transmitter) is used for measuring real hand finger degrees send them to other Arduino. Other Arduino (Receiver) is used to control servo motors according to the measured values. Electrical components and sensors on Arduino Mega as transmitter and receiver are showed on Table 7.1 and Table 7.2.

Arduino Mega (Receiver) circuit has three sub-units. These are power supply unit, servo motors and wireless communication unit. There are nine servo motors on Arduino. Stall current of a servo motor is 2.5 A (6V). For this reason, servo motors are supplied from power supply.

Arduino Mega (transmitter) has two units. These are flex sensor and wireless communication units XBee.

Table 7.1 :Electrical components on Arduino Mega 2560 (Transmitter).





Component Name	Quantity	Specification	Figures
XBee S2C Wireless Communication Module	1	XBee, 2mW Series 2 Wire Antenna, 2.4 GHz operating frequency, 120 m Communication Range.	
Flexible Sensor	5	4.5" in length. when the sensor is flexed, the resistance across the sensor increases.	

Table 7.2 : Electrical components on Arduino Mega 2560 (Receiver).

Component Name	Quantity	Specification	Figures
XBee S2C Wireless Communication Module	1	XBee, 2mW Series 2 Wire Antenna, 2.4 GHz operating frequency, 120 m Communication Range.	
Servo (MG996R)	9	Stall torque: 9.4 kgf·cm (4.8 V), 11 kgf·cm (6 V) Operating speed: 0.17 s/60° (4.8 V), 0.14 s/60° (6 V)	

XBee S2 wireless antenna was used for wireless communication between two Arduino Mega. Although XBee has 20 pins, only its four pins were used in the circuit. These pins are ground, data in (Rx), data out (Tx) and 3.3V input voltage. In the circuit, these four pins were connected (Figure 7.1) to the pins of Arduino Mega.

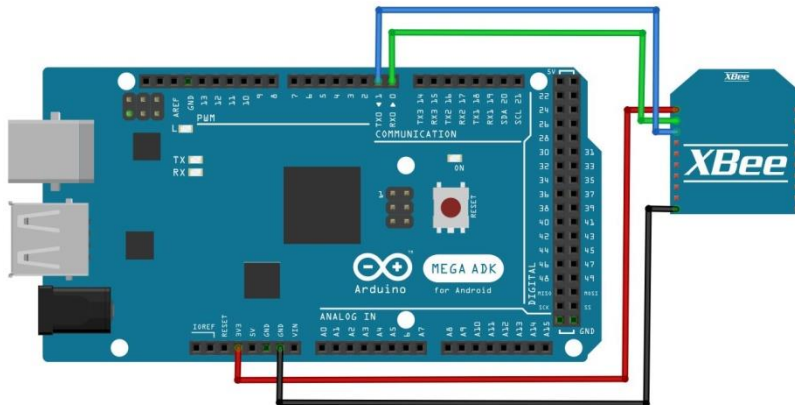


Figure 7.1 : XBee S2 wireless antenna connection to Arduino Mega 2560.

Flexible sensors are used for measuring angle of real hand fingers. Five flex sensors are used and they are placed on each finger of gloves. Two measurements are taken from each flex sensor without thumb. One of them is measured from proximal phalange. Other one is measured metacarpal. Therefore, for these fingers, there are eight DoF. For the thumb, just one measurement is taken from metacarpal. Therefore, thumb has one DoF. These data are taken from analog pins of Arduino MEGA and

they are converted to the meaningful angle values. In the other words, the output of the flexible sensor gives the orientation of the robotic hand fingers.

Measured values are sent to other Arduino Mega (Receiver) with XBee platform. There are nine servo motors on receiver Arduino Mega. According to measured values, to avoid noise because of flexible sensors, taken average values of 10 values. Then they are send to servo motors.

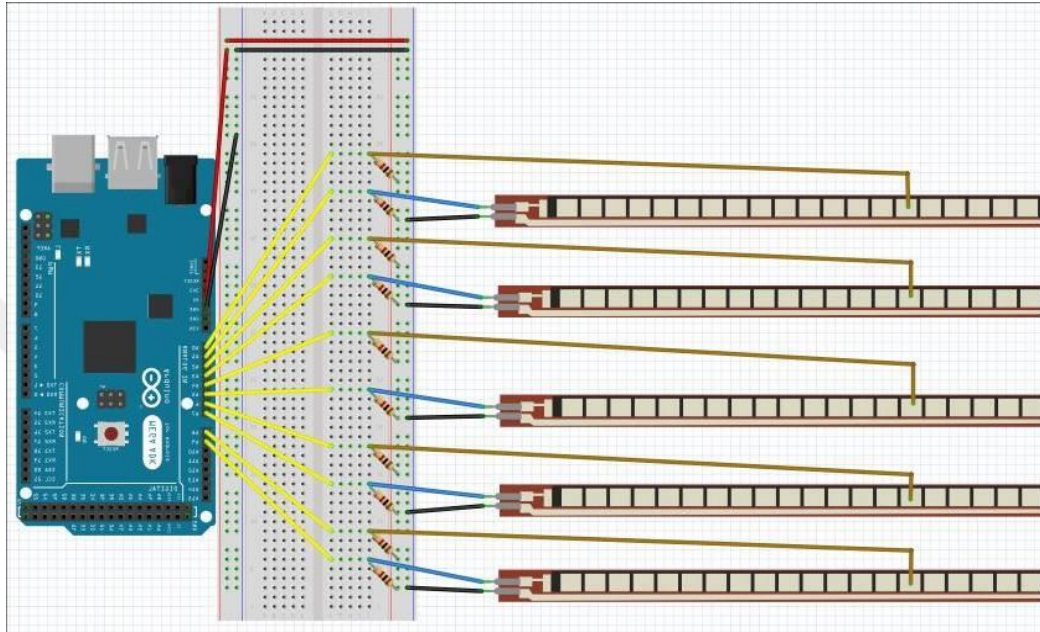


Figure 7.2 : Flexible sensor connection to Arduino Mega.

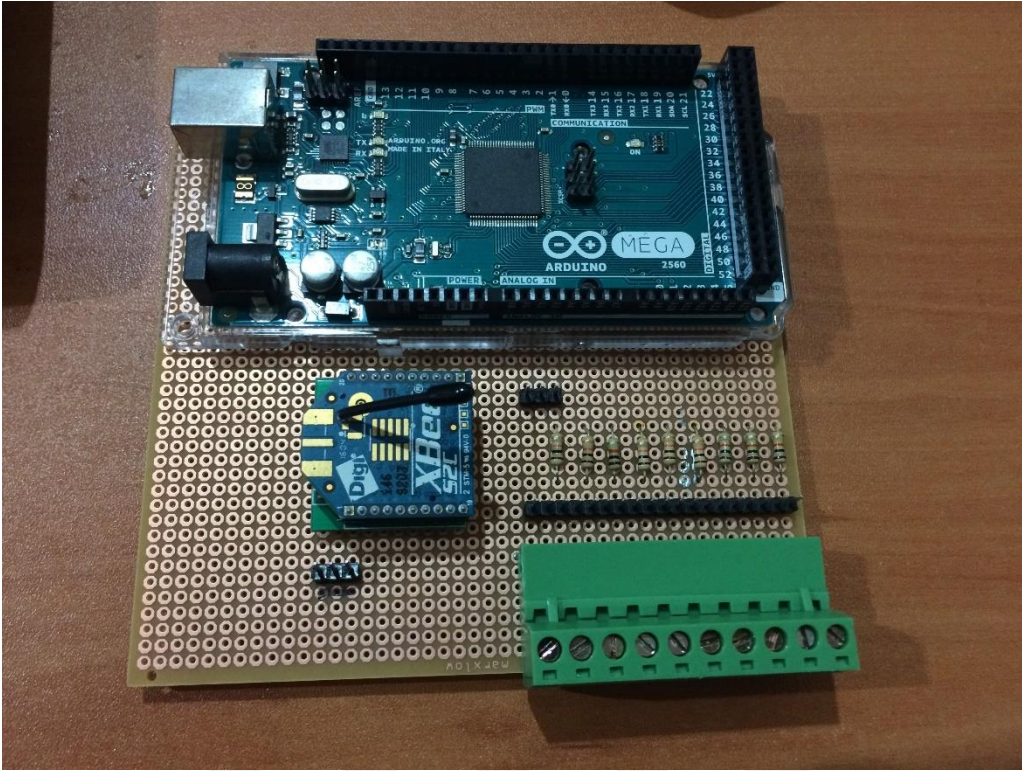


Figure 7.3 : Flexible sensor connection board (Transmitter Arduino).

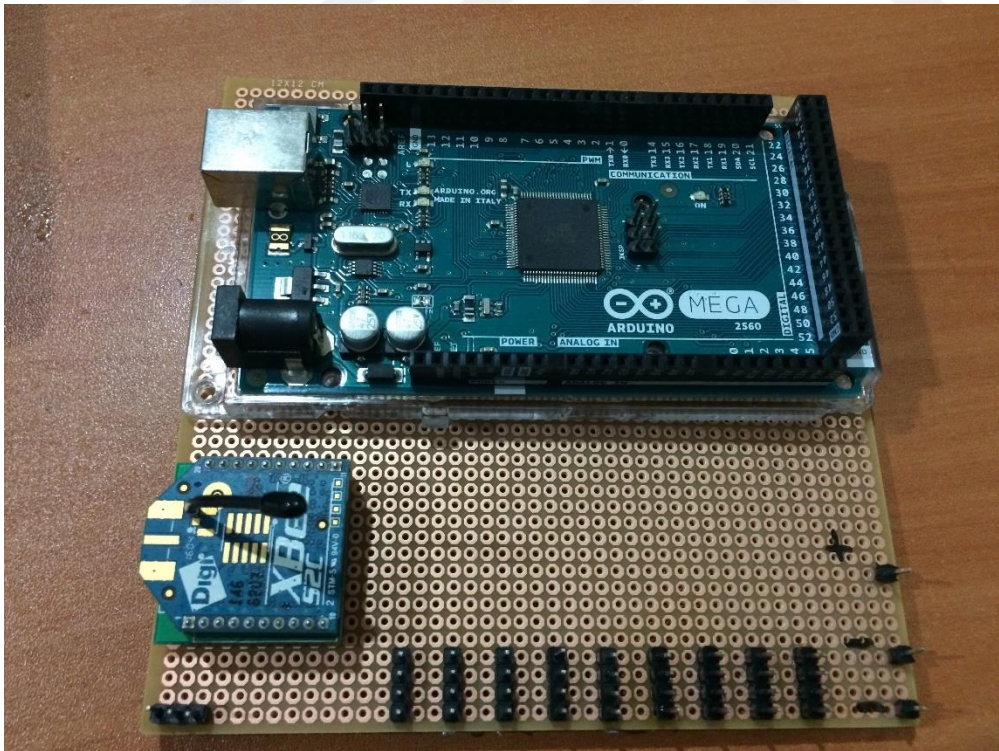


Figure 7.4 : Receiver Arduino Mega connection board.

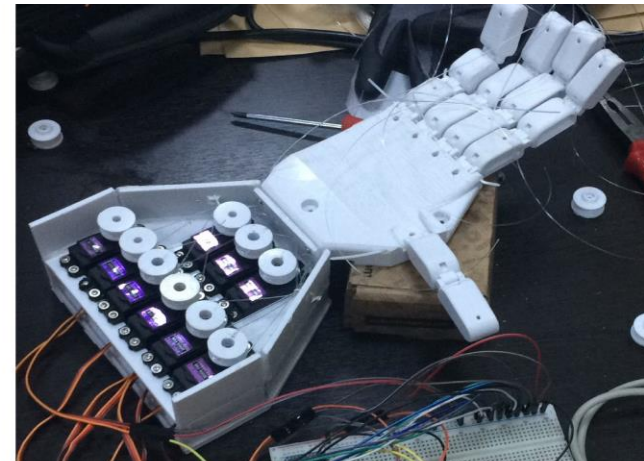
8. EXPERIMENTAL METHOD

Flexible sensors were sewed on gloves (Figure 8.1a). Except on thumb, from each flexible sensor, two measurements are taken from two different points. For four different positions, the system is calibrated. According to these calibrations, taken values are mapped in Arduino Mega (Transmitter) (Figure 8.1b) and transferred into angle values. These angle values are sent through wireless communication platform XBee to other Arduino Mega (Receiver) (Figure 8.1c). The codes in Arduino Mega Transmitter are shown in Appendix D. Arduino Mega (Receiver) controls the servo motors according to taken angle values. Since the angular values change very quickly, the averages of 10 values are taken so that they do not cause vibration on the servos. (Figure 8.1d). In addition, codes in Arduino Mega Receiver are shown in Appendix E.

After the design of system some finger movements were tried (Table 8.1). Fingers made exact motion of human hand. In addition, the robotic hand fingers were kept in the same position with the fingers of the human hand by using servo motors. In addition, an object was tried to be grabbed (Figure 8.2).



a



d



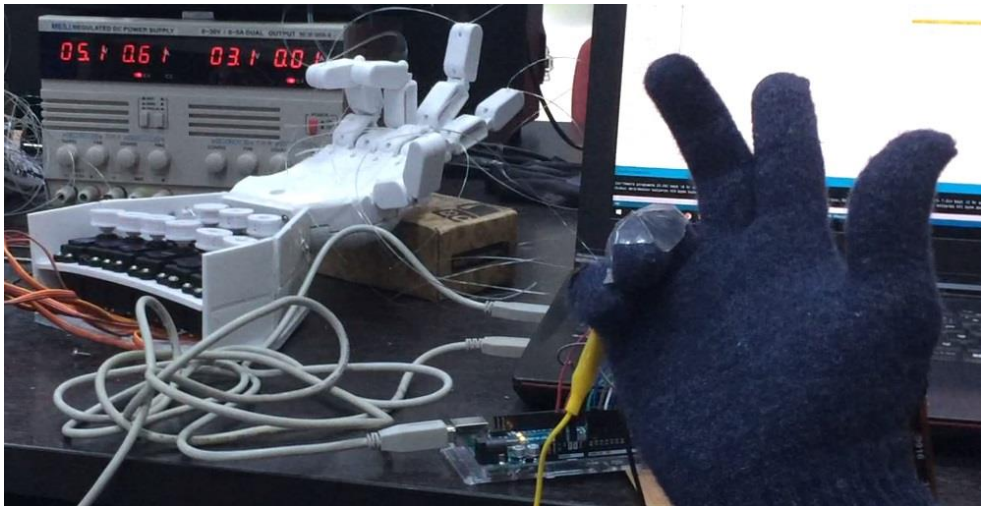
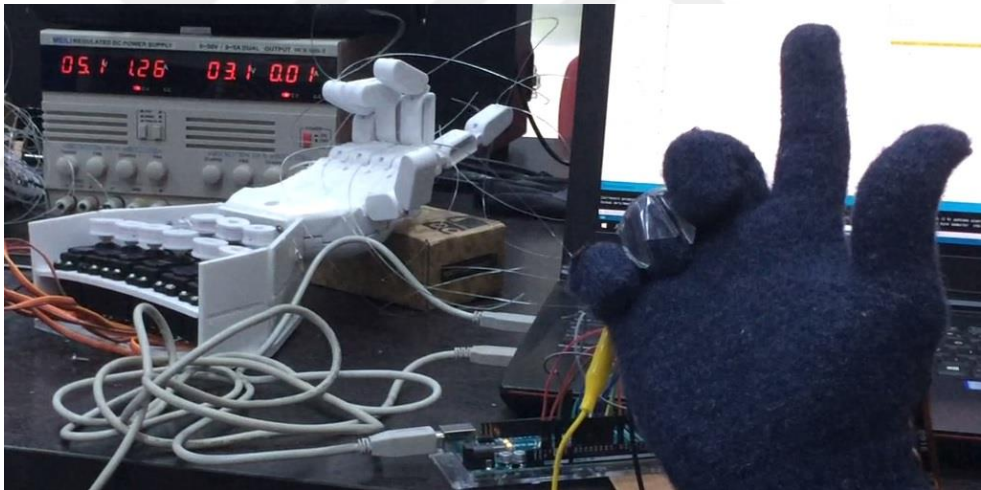
b



c

Figure 8.1 : Diagram of design (a) glove with flexible sensors, (b) Arduino Platform for the glove (transmitter), (c) Arduino Platform for the robotic hand (receiver), (d) robotic hand

Table 8.1 : Some of finger movements.



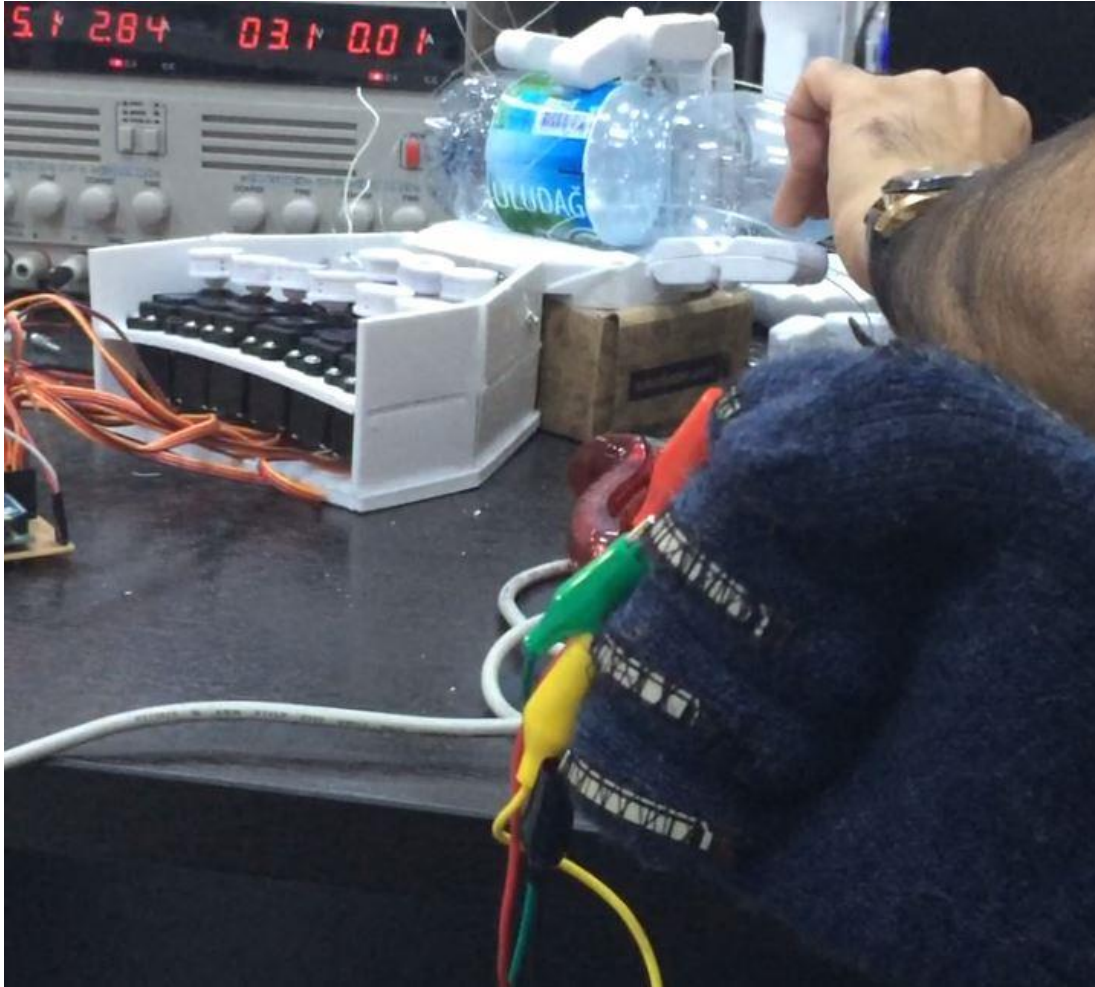


Figure 8.2 : Grabbing an object.

9. CONCLUSION

In this thesis, limited degree of freedom robotic hand was designed according to anatomy of the human hand. In order to determine the method to be used in the control of the robotic hand movement, wireless communication protocols were examined and compared. Then it was decided to use XBee S2 wireless communication platform on the Arduino platform to control the robotic hand. Manufacturing of the robotic hand was realized by 3D printer. The fingers can be manufactured as a single part even that rotational joints can move using enough clearances. Kinematic equations of the tip point of the fingers were derived. Robotic hand finger movements were modeled and gripping motion was examined and simulation works were made. Working area of the tip point was plotted in MATLAB. In order to control robotic hand, a glove was designed with flexible sensors. Characterization studies about flexible sensor was made for the first time in this study. Before start to control the robotic hand, calibration was made according to the working principle of the flexible sensors. Flex sensor calibration algorithm is also developed for the first time in this study. Except thumb, two independent motion for each robotic finger were provided by taking two measurements from each flexible sensor. In addition, the robotic hand fingers were kept in the same position with the fingers of the human hand by using servo motors. Robotic hand was tested at the end of study. We tried to grasp some objects using fingers. Furthermore, some finger movements were also tested.

In order to improve current design some considerations should be taken for future works as follows,

- Tendon materials must be selected as having minimum elastic behavior. Tendon cross-section diameter must not be too large.
- The shafts of revolute joints must be manufactured by stronger materials that have enough strength to carry the loads.
- Using enough clearances, robotic hand can be manufactured by using metal sintering method. The design will have much more strength. However, the weights of the parts must be considered.
- Characterization of the flexible sensors can be further improved. If this happens, more precise movements of the fingers can be obtained.



REFERENCES

1. Karabegović, I., & Doleček, V. (2017). *The Role of Service Robots and Robotic Systems in the Treatment of Patients in Medical Institutions*. In *Advanced Technologies, Systems, and Applications*(pp. 9-25). Springer International Publishing.
2. Pons, J. L., Ceres, R., & Pfeiffer, F. (1999). *Multifingered dextrous robotics hand design and control: a review*. *Robotica*, 17(6), 661-674.
3. Ayd, L. (1911). *U.S. Patent No. 984,179*. Washington, DC: U.S. Patent and Trademark Office.
4. Caron, L. G. (1918). *U.S. Patent No. 1,285,617*. Washington, DC: U.S. Patent and Trademark Office
5. James F. Mullen, Mechanical hand. *U.S. Patent No 3,694,021, 1972*.
6. S.C. Jacobsen, E.K. Iversen, D.F. Knutti, R.T. Johnson, K.B. Biggers, "Design of the Utah/MIT dextrous hand" In: *Robotics and Automation*. Proceedings. 1986 IEEE International Conference on. IEEE, s. 1520-1532. , 1986.
7. Ruoff, C. F., & Salisbury Jr, J. K. (1990). *U.S. Patent No. 4,921,293*. Washington, DC: U.S. Patent and Trademark Office.
8. Gaiser, I., Schulz, S., Kargov, A., Klosek, H., Bierbaum, A., Pylatiuk, C., Oberle, R., Werner, T., Asfour, T., Bretthauer, G., Dillmann, R. "A New Anthropomorphic Robotic Hand" In *Humanoids 2008-8th IEEE-RAS International Conference on Humanoid Robots IEEE.*, s. 418-422, Dec.2008.
9. Mouri, T., Kawasaki, H., Yoshikawa, K., Takai, J., & Ito, S. "Anthropomorphic robot hand: Gifu hand III," In *Proc. Int. Conf. ICCAS*, s. 1288-1293, Oct. 2002.
10. Yamano I., & Maeno, T. "Five-fingered robot hand using ultrasonic motors and elastic elements," In *Proceedings of the IEEE International Conference on Robotics and Automation IEEE* s. 2673-2678, Apr. 2005.
11. Matsuoka, Y. (1997). The mechanisms in a humanoid robot hand. *Autonomous Robots*, 4(2), 199-209.
12. Butterfaß, J., Grebenstein, M., Liu, H., & Hirzinger, G. (2001). DLR-Hand II: Next generation of a dextrous robot hand. In *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on* (Vol. 1, pp. 109-114). IEEE.
13. Kawasaki, H., Komatsu, T., & Uchiyama, K. (2002). Dexterous anthropomorphic robot hand with distributed tactile sensor: Gifu hand II. *IEEE/ASME transactions on mechatronics*, 7(3), 296-303.
14. <https://www.shadowrobot.com/products/dexterous-hand/>, access date: 17.05.2015

15. Theodore, O. (1958). *U.S. Patent No. 2,847,678*. Washington, DC: U.S. Patent and Trademark Office.
16. George B. Robinson, Pneumatically operated artificial hand. *U.S. Patent No 2,696,010, 1954*.
17. Pinson, G. T. (1981). *U.S. Patent No. 4,246,661*. Washington, DC: U.S. Patent and Trademark Office.
18. Light, C. M., & Chappell, P. H. (2000). *Development of a lightweight and adaptable multiple-axis hand prosthesis*. Medical engineering & physics, 22(10), 679-684.
19. Belter, J. T., Segil, J. L., & SM, B. (2013). *Mechanical design and performance specifications of anthropomorphic prosthetic hands: a review*. Journal of rehabilitation research and development, 50(5), 599.
- 20.. <http://www.ottobock.com/en/>, access date: 17.05.2015
21. <https://vincentsystems.de/en/>, access date: 17.05.2015
22. <http://www.touchbionics.com/>, access date: 17.05.2015
23. <http://bebionic.com/>, access date: 17.05.2015
24. <http://www.ottobockus.com/prosthetics/upper-limb-prosthetics/solution-overview/michelangelo-prosthetic-hand/>, access date: 17.05.2015
25. Marconi, Guglielmo. "*Origin and Development of Wireless Telegraphy*." The North American Review 168.510 (1899): 625-629.
26. De Forest, L. (1914). *U.S. Patent No. 1,123,119*. Washington, DC: U.S. Patent and Trademark Office.
27. Wladimir J. Polydoroff, Antenna system for wireless communication. *U.S. Patent No 2,266,262, 1941*.
28. Francis P. Meserow, Wireless communication system. *U.S. Patent No 2,883,523, 1959*.
29. Baronti, P., Pillai, P., Chook, V. W., Chessa, S., Gotta, A., & Hu, Y. F. (2007). *Wireless sensor networks: A survey on the state of the art and the 802.15. 4 and ZigBee standards*. Computer communications, 30(7), 1655-1695.
- 30.. Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., & Cayirci, E. (2002). *Wireless sensor networks: a survey*. Computer networks, 38(4), 393-422.
31. Al-Karaki, Jamal N., and Ahmed E. Kamal. "*Routing techniques in wireless sensor networks: a survey*." IEEE wireless communications 11.6 (2004): 6-28.
32. Amato, G., Chessa, S., Conforti, F., Macerata, A., & Marchesi, C. (2005). *Health care monitoring of mobile patients*. Ercim news, 60(6).
33. Welsh, M., Malan, D., Duncan, B., Fulford-Jones, T., & Moulton, S. (2004, March). *Wireless sensor networks for emergency medical care*. In *GE Global Research Conference, Boston*.
34. Malan, D., Fulford-Jones, T., Welsh, M., & Moulton, S. (2004, April). *Codeblue: An ad hoc sensor network infrastructure for emergency medical care*. In International workshop on wearable and implantable body sensor networks (Vol. 5).

35. Wang, N., Zhang, N., & Wang, M. (2006). *Wireless sensors in agriculture and food industry—Recent development and future perspective*. *Computers and electronics in agriculture*, 50(1), 1-14.
36. Sinan UĞUZ, Bayram KILIÇ, and Melike ŞİŞECİ. "Akıllı Ev Otomasyonu Sistemlerinde Zigbee Tabanlı Ağ Uygulamaları"
37. Sohraby, K., Minoli, D., & Znati, T. (2007). *Wireless Sensor Networks: Technology, protocols, and applications*. John Wiley & Sons.
38. Mainwaring, A., Culler, D., Polastre, J., Szewczyk, R., & Anderson, J. (2002, September). *Wireless sensor networks for habitat monitoring*. In Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications (pp. 88-97). Acm.
39. Kragic, D. *Anthropomorphic Hand Optimization based on a Latent Space Analysis*
40. Sean Middleton, "IEEE 802.15 WPAN Low Rate Study Group PAR". Document number IEEE P802.15-00/248r3, submitted September 2000.
41. Ivan Howitt and Jose A. Gutierrez G. "IEEE 802.15.4 Low Rate –Wireless Personal Area Network Coexistence Issues," *Wireless Communications and Networking*, WCNC 2003. IEEE, s. 1481-1486, 2003.
42. Liu, C. H., & Zhang, Y. (Eds.). (2015). *Cyber Physical Systems: Architectures, Protocols and Applications*(Vol. 22). CRC Press.
43. Faludi, R. (2010). *Building wireless sensor networks: with ZigBee, XBee, arduino, and processing*. " O'Reilly Media, Inc."
44. <https://www.arduino.cc/>, access date: 13.11.2016
45. <https://courses.lumenlearning.com/boundless-ap/chapter/muscles-of-the-upper-limb/>, access date: 02.09.2017
46. <http://thewellnessdigest.com/extensor-carpi-radialis-brevis-anatomy-origin-insertion-action/>, access date: 02.09.2017
47. <https://en.wikipedia.org/index.php?q=aHR0cHM6Ly9lbi53aWtpcGVkaWEub3JnL3dpa2kvRmluZ2Vy>, access date: 02.09.2017
48. <http://nimblevr.com/latest/doc/handModel.html>, access date: 02.09.2017
49. http://137.148.142.85/cactwiki/index.php5/Research_Assignment_2:_Prosthetics_and_Modeling_the_Human_Finger, access date: 02.09.2017
50. <http://www.cs.odu.edu/~cs752/papers/zigbee-001.pdf>, access date: 08.08.2016
51. <https://cdn-shop.adafruit.com/datasheets/SpectraFlex.pdf>, access date: 23.07.2016
52. <http://bidb.itu.edu.tr/seyirdefteri/blog/2013/09/07/zigbee>, access date: 08.08.2016



APPENDICES

APPENDIX A: Code of 3 DoF robotic finger workspace in MATLAB (0.01 radians)

APPENDIX B: Code of 2 DoF robotic finger workspace for first situation in MATLAB

APPENDIX C: Code of 2 DoF robotic finger workspace for second situation in MATLAB

APPENDIX D: Code of transmitter in Arduino

APPENDIX E: Code of receiver in Arduino

APPENDIX A

```
l1 = 28; % length of first part
```

```
l2 = 28; % length of second part
```

```
l3 = 23; % length of third part
```

```
theta1 = 0:0.01:pi/2; % all possible theta1 values
```

```
theta2 = 0:0.01:pi/2; % all possible theta2 values
```

```
theta3 = 0:0.01:pi/2; % all possible theta3 values
```

```
[THETA1, THETA2, THETA3] = meshgrid(theta1,theta2,theta3);
```

```
% generate a grid of theta1, theta2 and theta3 values
```

```
X = l1 * cos(THETA1) + l2 * cos(THETA1 + THETA2) + l3 * cos(THETA1 +  
THETA2 + THETA3); % compute x coordinates
```

```
Y = l1 * sin(THETA1) + l2 * sin(THETA1 + THETA2) + l3 * sin(THETA1 +  
THETA2 + THETA3); % compute y coordinates
```

```
data1 = [X(:) Y(:) THETA1(:)]; % create x-y-theta1 dataset
```

```
data2 = [X(:) Y(:) THETA2(:)]; % create x-y-theta2 dataset
```

```
data3 = [X(:) Y(:) THETA3(:)]; % create x-y-theta3 dataset
```

```
plot(X(:),Y(:),'r');
```

```
axis equal;
```

```
xlabel('X','fontsize',15)
```

```
ylabel('Y','fontsize',15)
```

APPENDIX B

```
l21 = 28; % length of first part
l22 = 28; % length of second part
l23 = 23; % length of third part

theta23 = 0:0.01:pi/2;

if theta23 < pi/2

    theta21 = 0:0.01:pi/2; % all possible theta21 values
    theta22 = 0;

    [THETA21,THETA22,THETA23] = meshgrid(theta21,theta22,theta23); %
    generate a grid of theta1, theta2

    X = 11 * cos(THETA21) + 12 * cos(THETA21 + THETA22) + 13 * cos(THETA21
    + THETA22 + THETA23) ; % compute x coordinates
    Y = 11 * sin(THETA21) + 12 * sin(THETA21 + THETA22) + 13 * sin(THETA21 +
    THETA22 + THETA23) ; % compute y coordinates

    data11 = [X(:) Y(:) THETA21(:)]; % create x-y-theta1 dataset
    data12 = [X(:) Y(:) THETA22(:)]; % create x-y-theta2 dataset
    data13 = [X(:) Y(:) THETA23(:)]; % create x-y-theta3 dataset

    plot(X(:),Y(:),'r.');
```

axis equal;

xlabel('X (mm)','fontsize',15)

ylabel('Y (mm)','fontsize',15)

APPENDIX C

```
l1 = 28; % length of first part
l2 = 28; % length of second part
l3 = 23; % length of third part
```

```
theta11 = 0:0.01:pi/2; % all possible theta11 values
theta12 = 0:0.01:pi/2; % all possible theta12 values
theta13 = pi/2;
[THETA11,THETA12,THETA13] = meshgrid(theta11,theta12,theta13); %
generate a grid of theta1, theta2
```

```
X = l1 * cos(THETA11) + l2 * cos(THETA11 + THETA12) + l3 * cos(THETA11
+ THETA12 + THETA13); % compute x coordinates
Y = l1 * sin(THETA11) + l2 * sin(THETA11 + THETA12) + l3 * sin(THETA11
+ THETA12 + THETA13); % compute y coordinates
```

```
data11 = [X(:) Y(:) THETA11(:)]; % create x-y-theta1 dataset
data12 = [X(:) Y(:) THETA12(:)]; % create x-y-theta2 dataset
data13 = [X(:) Y(:) THETA13(:)]; % create x-y-theta3 dataset
```

```
hold on, plot(X(:),Y(:),'r');
axis equal;
xlabel('X (mm)','fontsize',15)
ylabel('Y (mm)','fontsize',15)
```

APPENDIX D

```
#include <XBee.h>
XBee xbee = XBee();

int potpin0 = 0; // analog pin used to connect the index 1. flexsensor
int potpin1 = 1; // analog pin used to connect the index 2. flexsensor
int potpin2 = 2; // analog pin used to connect the middle 1. flexsensor
int potpin3 = 3; // analog pin used to connect the middle 2. flexsensor
int potpin4 = 4; // analog pin used to connect the ring 1. flexsensor
int potpin5 = 5; // analog pin used to connect the ring 2. flexsensor
int potpin6 = 6; // analog pin used to connect the digit 1. flexsensor
int potpin7 = 7; // analog pin used to connect the digit 2. flexsensor
int potpin8 = 8; // analog pin used to connect the thumb flexsensor

int p11d, p12d, p123, p11s, p12s, p112, p122, val11, val12, aci11, aci12, fark1; //Index
finger values
int p21d, p22d, p223, p21s, p22s, p212, p222, val21, val22, aci21, aci22, fark2;
//Middle finger values
int p31d, p32d, p323, p31s, p32s, p312, p322, val31, val32, aci31, aci32, fark3; //Ring
finger values
int p41d, p42d, p423, p41s, p42s, p412, p422, val41, val42, aci41, aci42, fark4; //Little
finger values
int p51d, p51s, val5, aci5; //Thumb values
int a=5;

void setup() {

  Serial.begin(9600);
  xbee.setSerial(Serial);
  Serial.println("keep your fingers straight");
  delay(3000);
  p11d = analogRead(potpin0);
  p12d = analogRead(potpin1);
  p21d = analogRead(potpin2);
  p22d = analogRead(potpin3);
```

```
p31d = analogRead(potpin4);
p32d = analogRead(potpin5);
p41d = analogRead(potpin6);
p42d = analogRead(potpin7);
p51d = analogRead(potpin8);
delay(500);
Serial.println("Bend your fingers from the 2nd node.");
delay(3000);
p112 = analogRead(potpin0);
p122 = analogRead(potpin1);
p212 = analogRead(potpin2);
p222 = analogRead(potpin3);
p312 = analogRead(potpin4);
p322 = analogRead(potpin5);
p412 = analogRead(potpin6);
p422 = analogRead(potpin7);
delay(500);
Serial.println("Bend your fingers from the 3rd node.");
delay(3000);
p123 = analogRead(potpin1);
p223 = analogRead(potpin3);
p323 = analogRead(potpin5);
p423 = analogRead(potpin7);
delay(500);
Serial.println("Now punch your hand.");
delay(5000);
p11s = analogRead(potpin0);
p12s = analogRead(potpin1);
p21s = analogRead(potpin2);
p22s = analogRead(potpin3);
p31s = analogRead(potpin4);
p32s = analogRead(potpin5);
p41s = analogRead(potpin6);
p42s = analogRead(potpin7);
p51s = analogRead(potpin8);
```

```

    delay(500);
    Serial.println("Now you can continue.");
}

void loop() {

    val11 = analogRead(potpin0);           // reads the value of the flexsensor (value
    between 0 and 1023)
    val12 = analogRead(potpin1);           // reads the value of the flexsensor (value
    between 0 and 1023)
    val21 = analogRead(potpin2);           // reads the value of the flexsensor (value
    between 0 and 1023)
    val22 = analogRead(potpin3);           // reads the value of the flexsensor (value
    between 0 and 1023)
    val31 = analogRead(potpin4);           // reads the value of the flexsensor (value
    between 0 and 1023)
    val32 = analogRead(potpin5);           // reads the value of the flexsensor (value
    between 0 and 1023)
    val41 = analogRead(potpin6);           // reads the value of the flexsensor (value
    between 0 and 1023)
    val42 = analogRead(potpin7);           // reads the value of the flexsensor (value
    between 0 and 1023)
    val5 = analogRead(potpin8);           // reads the value of the flexsensor (value between
    0 and 1023)

    //Index Finger
    if (val11 < (p11d + 50)) {
        aci12 = map(val12, p12d, p123, 0, 160); // scale it to use it with the servo (value
        between 0 and 160)
    }

    aci11 = map(val11, p11d, p112, 0, 160); // scale it to use it with the servo (value
    between 0 and 160)

    if (p112 < (val11 + 30)) {
        aci12 = map(val12, p122, p12s, 0, 160);
    }
}

```

```

//Middle Finger
if (val21 < (p21d + 50)) {
  aci22 = map(val22, p22d, p223, 0, 180); // scale it to use it with the servo (value
between 0 and 180)
}

aci21 = map(val21, p21d, p21s, 0, 180); // scale it to use it with the servo (value
between 0 and 180)

if (p21s < (val21 + 50)) {
  aci22 = map(val22, p222, p22s, 0, 180);
}

//Ring Finger
if (val31 < (p31d + 50)) {
  aci32 = map(val32, p32d, p323, 0, 160); // scale it to use it with the servo (value
between 0 and 160)
}

aci31 = map(val31, p31d, p31s, 0, 160); // scale it to use it with the servo (value
between 0 and 160)

if (p31s < (val31 + 50)) {
  aci32 = map(val32, p322, p32s, 0, 160);
}

//Little Finger
if (val41 < (p41d + 50)) {
  aci42 = map(val42, p42d, p423, 0, 180); // scale it to use it with the servo (value
between 0 and 180)
}

aci41 = map(val41, p41d, p41s, 0, 180); // scale it to use it with the servo (value
between 0 and 180)

if (p41s < (val41 + 50)) {

```



```

    aci42 = map(val42, p422, p42s, 0, 180);
}

//Thumb
aci5 = map(val5, p51d, p51s, 0, 180);    // scale it to use it with the servo (value
between 0 and 180)

if (aci11<0){
    aci11=0;
}
if (aci12<0){
    aci12=0;
}
if (aci21<0){
    aci21=0;
}
if (aci22<0){
    aci22=0;
}
if (aci31<0){
    aci31=0;
}
if (aci32<0){
    aci32=0;
}
if (aci41<0){
    aci41=0;
}
if (aci42<0){
    aci42=0;
}
if (aci5<0){
    aci5=0;
}
if (aci11>160){

```

```

    aci11=160;
}
if (aci12>160){
    aci12=160;
}
if (aci21>180){
    aci21=180;
}
if (aci22>180){
    aci22=180;
}
if (aci31>160){
    aci31=160;
}
if (aci32>160){
    aci32=160;
}
if (aci41>180){
    aci41=180;
}
if (aci42>180){
    aci42=180;
}
if (aci5<0){
    aci5=0;
}
uint8_t data[10] = {a, aci11, aci12, aci21, aci22, aci31, aci32, aci41, aci42, aci5};
XBeeAddress64 addr64 = XBeeAddress64(0x0013a200, 0x4125760a);
ZBTxRequest zbTx = ZBTxRequest(addr64, data, sizeof(data));
xbee.send(zbTx);
}

```

APPENDIX E

```
#include <XBee.h>
XBee xbee = XBee();
XBeeResponse response = XBeeResponse();
ZBRxResponse rx = ZBRxResponse();

#include <Servo.h>

Servo index1; // create servo object to control index 1
Servo index2; // create servo object to control index 2
Servo middle1; // create servo object to control middle 1
Servo middle2; // create servo object to control middle 2
Servo ring1; // create servo object to control ring 1
Servo ring2; // create servo object to control ring 2
Servo digit1; // create servo object to control digit 1
Servo digit2; // create servo object to control digit 2
Servo thumb; // create servo object to control thumb
int aci11, aci12, aci21, aci22, aci31, aci32, aci41, aci42, aci5; // angle of fingers
int a;
int totali1, totali2, totalm1, totalm2, totalr1, totalr2, totald1, totald2, totalt; // total
values for

void setup() {
  index1.attach(2); // attach servo of index finger 1
  index2.attach(3); // attach servo of index finger 2
  middle1.attach(4); // attach servo of middle finger 1
  middle2.attach(5); // attach servo of middle finger 2
  ring1.attach(6); // attach servo of ring finger 1
  ring2.attach(7); // attach servo of ring finger 2
  digit1.attach(8); // attach servo of digit finger 1
  digit2.attach(9); // attach servo of digit finger 2
  thumb.attach(10); // attach servo of thumb

  Serial.begin(9600);
```

```

    xbee.begin(Serial);
}

void loop() {
uint8_t sample[]={};
uint8_t data[]={};
    xbee.readPacket();
    if (xbee.getResponse().isAvailable()) {
        Serial.println(xbee.getResponse().getApiId());
        Serial.println(rx.getDataLength());
        if (xbee.getResponse().getApiId() == ZB_RX_RESPONSE) {
            xbee.getResponse().getZBRxResponse(rx);
            for (int i = 0; i < rx.getDataLength(); i++) {
                sample[i] = (int)rx.getData(i);
                data [i] = sample[i];
                delay(1);
            }
        }
    }
    }else if (xbee.getResponse().isError()) {
        Serial.println("Error reading packet. Error code: ");
        Serial.println(xbee.getResponse().getErrorCode());
    }
}

```

```

a = data[0];
aci11 = data[1];
aci12 = data[2];
aci21 = data[3];
aci22 = data[4];
aci31 = data[5];
aci32 = data[6];
aci41 = data[7];
aci42 = data[8];
aci5 = data[9];
for (int y=0; y<10; y++){

```

```
totali1 = totali1 + aci11;
totali2 = totali2 + aci12;
totalm1 = totalm1 + aci21;
totalm2 = totalm2 + aci22;
totalr1 = totalr2 + aci31;
totalr2 = totalr2 + aci32;
totald1 = totald1 + aci41;
totald2 = totald2 + aci42;
totalt = totalt + aci5;
```

```
if(y==9){
    aci11 = totali1/10;
    aci12 = totali2/10;
    aci21 = totalm1/10;
    aci22 = totalm2/10;
    aci31 = totalr1/10;
    aci32 = totalr2/10;
    aci41 = totald1/10;
    aci42 = totald2/10;
    aci5 = totalt/10;
```

```
totalr2 = 0;
totald1 = 0;
totald2 = 0;
totalt = 0;
totali1 = 0;
totali2 = 0;
totalm1 = 0;
totalm2 = 0;
totalr1 = 0;
delay(1);
```

```
}
}
index1.write(aci11);
index2.write(aci12);
```

```
middle1.write(aci21);  
middle2.write(aci22);  
ring1.write(aci31);  
ring2.write(aci32);  
digit1.write(aci41);  
digit2.write(aci42);  
thumb.write(aci5);  
}
```



CURRICULUM VITAE



Name Surname: Faruk Sanberk KIZILTAŞ

Place and Date of Birth: SİVAS 28.07.1992

Address: İstiklal Mahallesi, Ahmet Yesevi Sokak, Sefa Apt.
D:13 Serdivan/Sakarya, Türkiye

E-Mail: fsanberk@gmail.com

B.Sc.: Biomedical Engineering

List of Publications:

Kızıldaş, F. S., Can, F. C., &Gezgin, E. Design of Wireless Controlled Robotic Hand. In XX. Biyomedikal Mühendisliği Ulusal Toplantısı (BİYOMUT 2016), 2016 National Conference

Ün, M., Ün, M., & Kızıldaş, F. S. (2016, December). Analysis of fractional-order 2xn RLC networks by transmission matrices. In *Electrical, Electronics and Biomedical Engineering (ELECO), 2016 National Conference on* (pp. 423-426). IEEE

Ün, M., Ün, M., & Kızıldaş, F. S. (2016, October). Analysis of fractional-order 2xn RLC circuit network by mesh currents method. In *Medical Technologies National Congress (TIPTEKNO) 2016* (pp. 1-4). IEEE

Ün, M., Ün, M., & Kızıldaş, F. S. (2016, October). Analysis of a network of electrically coupled neurons in fractional domain. In *Medical Technologies National Congress (TIPTEKNO) 2016* (pp. 1-4). IEEE