Development and experimental study of a bio-inspired anthropomorphic robotic hand with human-hand-like biomechanical advantages and performance

A dissertation submitted to the University of Manchester for the degree of Doctor of Philosophy in the Faculty of Science and Engineering

2021

Yiming Zhu

School of Mechanical, Aerospace and Civil Engineering

Blank page

Table of Contents

List of Figures 7

List of Tables 14

Abstract 15

Declaration 16

Copyright 17

Acknowledgements 18

Chapter 1: Introduction 19

1.1 Background and significance 19

1.1.1 Background 19

1.1.2 Significance 20

1.2 Aim and objectives 21

1.3 Thesis overview 22

Chapter 2: Literature review 24

2.1 Anatomy of human hand 24

2.1.1 Morphology of bones 24

2.1.2 Structure of joints 25

2.1.3 Distribution of ligaments 26

2.1.4 Distribution of muscle-tendon system 28

2.1.5 Structure and distribution of tendon sheaths 32

2.2 Biomechanics of human hand 34

2.2.1 Coordinate system in human hand 34

2.2.2 Joint orientation and range of motion 36

2.2.3 Force transmission in human hand 38

2.3 Unique biomechanical properties embedded in human hand structures 41

2.3.1 Joint stiffness and ligamentous joints 41

2.3.2 Endpoint feasible force set (space) and extensor mechanisms 43

2.3.3 Endpoint force-velocity characteristics and flexible tendon sheaths 44

2.4 Bio-inspired design of robotic hands 45

2.5 Joint design of anthropomorphic robotic hands..47

2.5.1 Hinge joint in robotic hands..47

2.5.2 Elastic joint in robotic hands...49

2.5.3 Biomimetic joint in robotic hands..52

2.5.4 Joint in human hand joint anthroplasty..54

2.6 Actuation system design of anthropomorphic robotic hands.........................55

2.6.1 Motor-Gear linkage actuation..55

2.6.2 Motor-Flexible wire actuation 57

2.6.3 Pneumatic cylinder-Tendon actuation..59

2.6.4 Shape memory alloys actuation 60

2.7 Control methods of anthropomorphic robotic hands 62

2.7.1 Non-adaptive control 62

2.7.2 Adaptive control...67

2.7.3 EMG/EEG control 68

2.7.4 Data glove control...73

2.7.5 Ultrasound control 76

2.8 Grasping and manipulation taxonomy..78

2.9 Comparison of different robotic hands 81

2.10 Summary...84

Chapter 3: Bio-inspired design and fabrication of the robotic finger..................86

3.1 The Base Layer — Phalanges and articular cartilage 87

3.2 The Second Layer — Capsuloligamentous structures...................................89

3.3 The Third Layer — Tendons and tendon sheaths 92

3.4 Basic kinematic indexes of the robotic finger...95

3.4.1 Range of motion test...97

3.4.2 Fingertip trajectory test...99

3.4.3 Full flexion-extension speed test 100

3.5 Summary..102

Chapter 4: Theoretical analysis for three biomechanical properties of

human-finger-like structures 103
4.1 Joint stiffness associated with the ligamentous structure 103
4.1.1 Geometric analysis......103
4.1.2 Algebraic analysis 110
4.2 Feasible force space associated with the extensor mechanism......123
4.3 Force-velocity characteristics associated with the flexible tendon sheath...132
4.3.1 Static characteristics analysis......132
4.3.2 Dynamic characteristics analysis 137
4.4 Summary......141
Chapter 5: Experimental verification for the biomechanical advantages realized on the robotic finger......143
5.1 Variable joint stiffness......143
5.1.1 Experiment set-up and protocol......143
5.1.2 Results of variable joint stiffness......145
5.1.3 Discussion......146
5.2 Enlarged feasible force space......148
5.2.1 Experiment set-up and protocol......148
5.2.2 Results of enlarged feasible force space 150
5.2.3 Discussion......150
5.3Augmented force-velocity workspace 153
5.3.1 Experiment set-up and protocol......153
5.3.2 Results of force-velocity characteristics 155
5.3.3 Discussion......159
5.4 Human-finger-like grasping capability 159
5.4.1 Experiment set-up and protocol......159
5.4.2 Results of grasping test 162
5.4.3 Discussion......163
5.5 Summary......164
Chapter 6: Bio-inspired design and fabrication of the robotic hand 166

6.1 Ligamentous-skeletal structure 166
6.2 Tendon distribution 170
6.3 Artificial skin 173
6.4 Summary 175
Chapter 7: Actuation system construction and control theory of the robotic hand 177
7.1 Motor-tendon actuation system construction 177
7.2 Control methods of the tendon-driven robotic finger with flexible tendon sheaths 178
7.2.1 Control of the single-tendon-driven robotic finger 178
7.2.2 Control of the multi-tendon-driven robotic finger 185
7.2.3 Data glove-based posture control of the robotic hand 190
7.3 Summary 191
Chapter 8: Human-hand-like grasping and manipulation 193
8.1 Grasping capability test based on Cutkosky and Feix taxonomy 193
8.2 Manipulation capability test 198
8.3 Summary 201
Chapter 9: Conclusion and future work 202
9.1 Thesis overview 202
9.2 Main findings and contributions 204
9.3 Limitation and future works 207
References 210
Appendices 223
Appendix I: Grasping and manipulation video list of the robotic finger and the robotic hand 223
Appendix II: CAD model of joint capsule moulds 224
Appendix III: CAD model of tendon sheath moulds 224
Appendix IV: CAD model of actuation system connector and module 225
Appendix V: CAD model of the skeletal structure of the robotic hand with

ligaments and tendons insertion holes 225
Appendix VI: Five generations of the robotic hand..226
Appendix VII: Matlab code for ball grasping of the robotic hand....................227
Appendix VIII: Matlab code for fanning out the playing cards of the robotic hand
....286

Word Count: 39,571

List of Figures

Figure 1. Definition of the bones and joints of the human right hand 25

Figure 2. Structure of the synovial joint (modified from reference [60]) 26

Figure 3. Distribution of the ligaments in hand (modified from reference [61]).........27

Figure 4. Distribution of the ligaments in PIP joint (modified from reference [63]) ..28

Figure 5. Distribution of the tendons in the right hand (modified from reference [64]) 29

Figure 6. Detailed distribution of the tendons along the finger (modified from reference [66])...31

Figure 7. Intrinsic muscle groups of the hand (modified from reference [67]) 32

Figure 8. Distribution of each intrinsic muscle group (modified from reference [68]) 32

Figure 9. Anterior view of the palm of the hand showing the flexor synovial sheath. Cross section of a finger is also shown. (modified from reference [70]) 33

Figure 10. Detailed structure of the synovial tendon sheath (modified from reference [71])...34

Figure 11. Coordinate systems in the index finger (modified from reference [6])......35

Figure 12. Biplanar X-ray method used to measure the orientation and the motion range of the joint in 3D space. (adopted from reference [6]) 36

Figure 13. Joint orientation and axes of rotation of the thumb basal joint. (modified from reference [6]) 36

Figure 14. Direction of the force in the extensor mechanism.....................................40

Figure 15. The NAIST hand with hinge joints (adopted from reference [74])............48

Figure 16. The Southampton hand with hinge joints (adopted from reference [48]) ..49

Figure 17. The Cyberhand with hinge joints (adopted from reference [50])...............49

Figure 18. The ACT hand with hinge joints (adopted from reference [75])................49

Figure 19. The UB hand 3 with elastic joints (adopted from reference [55])..............50

Figure 20. The iHY SDM hand with elastic joints (adopted from reference [76])......50

Figure 21. The PneuNets actuator with elastic fingers (adopted from reference [79]) 51

Figure 22. The soft prosthetic hand with elastic fingers (adopted from reference [80]) 51
Figure 23. The UW robotic hand with biomimetic joints (adpted from reference [18]) 53
Figure 24. Anthropomorphic robotic hands or fingers with biomimetic joints. (a) Robotic hand from University of Cambridge. (b) Joint structure of HR-hand. (c) 3D-printed anthropomorphic soft skeleton hand. (d) 3D-printed biomimetic robotic finger. (adopted from references [19] [20] [21] [22]) ...53
Figure 25. The artificial finger joints from SBi (adopted from reference [81]) 54
Figure 26. The MRD system of the NAIST hand. (a) Overview of gear mechanism. (b) Three-axis driving gear mechanism. (c) Coupling link mechanism. (adopted from reference [74])..56
Figure 27. The Gifu hand III and its MRD system. (a) The Gifu hand III prototype. (b) The thumb structure. (c) The fingers structure. (adopted from reference [49]) 56
Figure 28. DLR/HIT Hand II and its MFD system (adopted from reference [51]).....57
Figure 29. Robonaut hand II and its MFD system (adopted from reference [52])......58
Figure 30. The highly biomimetic robotic hand of UW and its MFD system (modified from reference [18])..58
Figure 31. The Shadow hand with air muscles actuation system (adopted from reference [53])..59
Figure 32. The pneumatic cylinder-tendon actuation system in the previous version of the UW hand (adopted from reference [82])...60
Figure 33. SBC hand with SMA actuation system (adopted from reference [86])......61
Figure 34. The prosthetic hand using shape memory alloy actuation (adopted from reference [88])..61
Figure 35. A SMA-driven robotic hand prototype (adopted from reference [89]) 62
Figure 36. The position control loop of the tendon-driven robotic hand.....................64
Figure 37. The position control loop of the tendon-driven robotic hand with passive

behavior..65
Figure 38. The force control loop of the tendon-driven robotic hand.........................66
Figure 39. Basic structure of a fuzzy controller...70
Figure 40. Membership values generated by the data distribution (adopted from reference [104])..70
Figure 41. The schematic diagram of a single artificial neuron (adopted from reference [106])..71
Figure 42. The schematic diagram of an ANN model (modified from reference [108]) 71
Figure 43. Data gloves from different companies. (a) VMG 30 data glove. (b) CyberGlove data glove. (c) CyberGlove 2. (d) 5DT Data Glove 14 Ultra. (e) Humanglove. (adopted from the references [156] [158] [159] [162] [164]) .74
Figure 44. Robotic hands controlled by the data glove. (a) Shadow hand manipulation controlled by the data glove. (b) DLR/HIT Hand II grasping controlled by the data glove (adopted from the references [186] [187]) 76
Figure 45. The ultrasound image of the forearm muscles (adopted from the reference [167])..77
Figure 46. Ultrasound-controlled ‘Luke Skywalker’s bionic hand’ from Georgia Institute of Technology (adopted from the reference [175]).........................78
Figure 47. Two widely-used grasp taxonomies. (a) Cutkoskey grasp taxonomy. (b) Feix grasp taxonomy. (adopted from the reference [153] [154])..........................79
Figure 48. Bullock and Dollar manipulation taxonomy. (a) 15 types of manipulation tasks. (b) 12 types of within hand prehensile manipulation tasks. (adopted from the reference [181]) 81
Figure 49. The multi-layered structure of a human and robotic finger. 87
Figure 50. Fabrication of robotic finger bones (a) CT images of human hand bones. (b) MR images of human hand soft tissues. (c) CAD model of human finger bones. (d) 3D printed physical model of robotic finger bones with PTFE articular cartilage..88

Figure 51. Capsuloligamentous structure of the robotic finger. (a) “Crimp pattern” of the human ligament and the PET braided ribbon. (b) Designed moulds and silicone rubber joint capsule. 90
Figure 52: Design evolution of the extensor expansion...93
Figure 53: The extensor expansion covered on the dorsal side of the finger..............93
Figure 54. Designed moulds and silicone rubber tendon sheath with PTFE tubing....94
Figure 55. Distribution of the tendons along the robotic finger..................................95
Figure 56. Distribution of the tendon sheaths along the human and robotic finger.....95
Figure 57. The CAD model and the 3D printed model of the platform for one finger96
Figure 58. The constructed testbed for one finger 96
Figure 59. Components of Dynamixel motor control system.....................................96
Figure 60. Test of each joint motion range of the robotic finger 98
Figure 61. Interface of Vicon motion capture system..100
Figure 62. Space trajectory of the fingertip during flexion and extension motion....100
Figure 63. Test of the full flexion-extension speed of the robotic and human finger 101
Figure 64. Schematic diagram of the ligament’s geometry in MCP joint motion 104
Figure 65. A 3D coordinate system in MCP joint with the ulnar ligament................110
Figure 66. The length change of the three portions of the ulnar ligament with the MCP joint flexion angles..112
Figure 67. The tested strain-stress property of the PET braided ribbon 114
Figure 68. The one-ligament joint parameters change with flexion angles...............116
Figure 69. The joint torque change resulted by adding another ligament 116
Figure 70. Theoretical relation of the joint flexion stiffness and flexion angles 117
Figure 71. The ligament length change with the flexion and abduction angle in MCP joint 119
Figure 72. Theoretical relations of the joint abduction stiffness and the flexion/abduction angle..123
Figure 73. The kinematic model of the robotic index finger. 124
Figure 74. The relationships of the tendon excursion-joint motion angle and the

calculated moment arm-joint motion angle 128

Figure 75. Theoretical feasible force set and space of the robotic finger. (a) Feasible force set calculated from robotic finger's mathematical model. (b) Feasible force space generated from the feasible force set of the robotic finger's mathematical model. 131

Figure 76. Schematic diagram of the single-joint elastic tendon sheath pulley system (for quasi-static analysis). 133

Figure 77. Theoretical quasi-static force-velocity characteristics. (a) The quasi-static relationship between the endpoint normal velocity and the joint flexion angle. (b) The quasi-static relationship between the endpoint normal force and the joint flexion angle. (c) The normalized results of the theoretical data about the quasi-static maximum force and velocity that different finger models can output. 136

Figure 78. Schematic diagram of the quick-release experiment on single-joint model (for dynamic analysis)..138

Figure 79. Theoretical result of the dynamic force-velocity characteristics.............141

Figure 80. Experimental setup of the joint stiffness test. (a) Experimental setup of the joint flexion stiffness test. (b) Experimental setup of the joint abduction stiffness test...144

Figure 81. Comparison result of the theoretical and experimental relations of the joint flexion stiffness and flexion angles..147

Figure 82. Joint terminal stiffness in abduction-adduction direction. (a) Relation of the external applying torque and abduction angles from the test. (b) Comparison result of the theoretical and experimental relation of the joint abduction terminal stiffness and flexion angles..148

Figure 83. The physical models and the schematic diagrams of the robotic fingers with net and linear extensor. 148

Figure 84. Experimental setup of the fingertip force test. (a) Physical model of the fingertip force testbed. (b) Schematic diagram of the fingertip force testbed.

....149
Figure 85. Comparison result of the fingertip feasible force space in different postures. (a) The comparison results of the fingertip feasible force space in posture 1. (b) The comparison results of the fingertip feasible force space in posture 2. (c) The comparison results of the fingertip feasible force space in posture 3. (Yellow-mathematical model with net extensor; Blue-physical model with net extensor; Red- physical model with linear extensor)..................................152
Figure 86. The robotic finger model with the flexible tendon sheath and three comparison robotic finger models with rigid tendon sheaths of different thicknesses (6mm, 4mm and 2mm). 153
Figure 87. Experimental setup of the quick-release test. (a) Test rig for the quick-release experiment. (b) Schematic diagram of the quick-release test rig................155
Figure 88. Experiment result of the fingertip force-velocity characteristics with different tendon sheaths. (a) Fingertip force and velocity experiment results with the flexible tendon sheath in a 180g load. (b) Dynamic force-velocity characteristics of the four physical robotic finger models (Yellow-2mm rigid tendon sheath; Green-4mm rigid tendon sheath; Blue-6mm rigid tendon sheath; Red-flexible tendon sheath). (c) The normalized results of the theoretical and experiment data about the quasi-static maximum force and velocity that different finger models can output...158
Figure 89. Two-finger testbed and the hardware setup...161
Figure 90. Control flow chart for the robotic fingers and the robotic arm. 162
Figure 91. The robotic and human fingers grasping test with five target objects on three different surfaces...162
Figure 92. The comparison results of the successful cases of the robotic (orange line) and human (blue line) fingers. 163
Figure 93. CAD model of human metacarpal and carpal bones...............................167
Figure 94. Metacarpal and carpal bones with ligaments (modified from reference [150]) 167

Figure 95. The midcarpal joint and its two single functional units (modified from reference [151])..168
Figure 96. Skeletal model and carpal ligaments of the robotic hand. (a) Three parts of 3D printed skeletal models of the robotic hand. (b) Ligaments connection among the carpals and metacarpals..169
Figure 97. The carpal mould and the carpal cover with a band-shape ligament........170
Figure 98. The cross-section view of the human hand near the wrist........................172
Figure 99. Robotic hand wrist base model with guiding holes..................................172
Figure 100. Joint capsule and tendon sheath moulds and cured parts 173
Figure 101. Assembled robotic hand model without skin..173
Figure 102. The plaster shell layer (1st layer) and the Ecoflex-35 silicone rubber layer (2nd layer) of the artificial skin mould...175
Figure 103. Artificial skin with fingerprint and wrinkles 175
Figure 104. Assembled robotic hand model and prototype with actuation system ...178
Figure 105. Simplified model of the single-tendon-driven robotic finger. (a) Model of the distal end joint with a tendon insertion. (b) Simplified model of model a. (c) Simplified model of the three-joint finger model with FDP tendon actuation. 179
Figure 106. Single-joint model with two flexor tendons 185
Figure 107. Simplified model of the robotic finger with three tendons actuation.....188
Figure 108. Posture identification of objects grasping with data glove.....................190
Figure 109. Grasping test based on Cutkosky taxonomy 196
Figure 110. Grasping test based on Feix taxonomy...197
Figure 111. Dynamic grasping process of four objects...198
Figure 112. Manipulation test of the robotic hand...200

List of Tables

Table 1. Orientation angles of the finger joints (data from the reference [6]) 37

Table 2. Tendons and intrinsic muscles involved in each finger joint (modified from reference [6])..38

Table 3. Some comparison results of different robotic hands......................................81

Table 4. Joint motion range of human (a) and robotic finger (b).................................98

Table 5. The moment arm of each muscle tendon to each joint in the standard posture (mm)...128

Table 6. The volume of the feasible force space in different postures with net and linear extensor 152

Abstract

Human hands are versatile and capable of dealing with a mass of daily activities. Exploring their fundamental biomechanical features residing in the anatomical structures and applying them to the robotic hands has proven to be an effective approach to enhance the practical performance, however, has been a challenge all along. The obstacles exist not only in replicating the human-hand-like anatomical structures by using present technologies in fabrication and materials but also in the lack of investigation on the biomechanical principles of human hands. Thus, the overall aim of this project was to develop a novel highly biomimetic robotic hand with human-hand-like structures, biomechanical advantages as well as grasping and manipulation capabilities. The framework of this research included three stages: (1) design and fabrication of a multi-layered anthropomorphic robotic hand, (2) analysis of three underlying biomechanical advantages that exist in the human-hand-like structures by using mathematical models and experiments, (3) tests of the grasping and manipulation capabilities of the proposed robotic hand with the customized actuation system and control strategies.

The design of the robotic hand highly mimicked the human hand features in terms of the morphological structures and the material properties. The human hand bones, ligaments, joint capsules, tendon sheaths, tendons and the skin were all replicated on the robotic hand.

Subsequently, three biomechanical properties were investigated through theoretical analysis and experimental verification. Both the theoretical and experimental results show that the variable joint stiffness was obtained with the ligamentous joint structures, the feasible force space was enlarged by the reticular extensor compared with the linear extensor, and the fingertip force-velocity workspace was augmented by the flexible tendon sheaths compared with the rigid tendon sheaths.

Finally, the grasping and manipulation tests were conducted in both robotic finger and robotic hand models. The result showed the robotic finger performed a comparable grasping success rate with the human fingers in all the five target objects and three interaction surfaces. For the robotic hand, a 24 motors actuation system was built and the data glove-based position control strategy was developed. The grasping result showed that 16 grasp types in Cutkosky taxonomy and 33 types in Feix taxonomy were all accomplished. Besides, the robotic hand also successfully performed the dynamic grasping capability. In addition, six common manipulations in daily lives were tested on the prototype, which were all completed. The results showed the human-hand-like grasping and manipulation capabilities, to a certain extent, were realized on the proposed robotic hand. From another aspect, the bio-inspired rigid-flexible coupled robotic hand combined the advantages of the rigid robotic hand (good manipulability) and the soft robotic hand (good grasping capability).

Declaration

No portion of the work referred to in the Thesis has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.

Copyright

Acknowledgements

Foremost, I am extremely grateful to my family for their selfless love, continued support and heartfelt encouragement throughout my four-year PhD research process. Especially, I give my special thanks and love to my wife Boya Zhou for her thoughtful caring and accompanying. And I show my sincere appreciation to my main supervisor Dr Lei Ren for providing this research opportunity and guiding me to overcome all the difficulties to accomplish the project. His strong scientific consciousness, broad scientific vision and efficient research methods, as well as the spirit of hard work, influenced me a lot and will be a precious treasure for my future scientific research career. I would also like to thank Dr Guowu Wei for his patient and concrete guidance on the whole project, especially providing me with convenient 3D print conditions, helping me purchase some essential experiment materials and equipment and carefully revising my papers again and again. I am also thankful to Dr Zhenmin Zou for his valuable suggestion for my research. I thank all my schoolmates for their help and advice on the design of the experiments, programming and fabrication of the prototype. Lots of brilliant ideas were born from the discussion with them. Also, I would like to express my thanks to Professor Jianzhong Shang and Zirong Luo for their remote mentoring and care to my research and life. Finally, thank all the people that help me from various aspects.

Chapter 1: Introduction

1.1 Background and significance

1.1.1 Background

The human hand is a primary effector organ for our daily activities. During the interaction with objects, except the dominant control command from the brain, our hands can passively and accordingly adapt their posture, stiffness, contact force and velocity to different object shapes, dimensions, weights, softness, as well as the different environmental conditions. For example, when grasping a ball, our fingers can passively adjust the palmar orientation to make a larger contact area so that a more stable grasping can be obtained. Basing on this behavioural feature, our hands are capable of picking up a peanut, a pen, or grasping an apple, a bottle even without consciously differentiating the motion control command. And in fact, such interaction often happens in a dynamic, irregular, unstructured and uncertain environment, such as on a smooth or rough, soft or rigid surface. Moreover, our hands can also perform numerous complex manipulations, such as playing an instrument or performing surgical procedures, thanks to their intrinsic dexterity and versatility.

The excellent performance of our hands is the result of the combined action of the neural control and perception as well as the biomechanical properties of the hand itself. However, how each functional system acts and exerts an effect on the functional performance is still remaining unrevealed. Indeed, a large area of the motor cortices in the brain and the wide-distributed mechanoreceptors embedded in hand skin and subcutaneous tissues play a crucial role in the hand movements and interactions with its surroundings. For example, each finger has several specific nerves to actuate the tendons so as to move dexterously [1] [2]. And the feedback from the mechanoreceptors provides information on the object's texture, softness, weight, and motion status so that the manipulation and other functions can be performed better [3]

[4] [5].

In addition to this, the hand body itself does contribute some biomechanical fundamentals. Through the detailed mathematical and cadaveric research on the biomechanics of human hands, Chao and An [6] found plenty of unique properties of each function unit and anatomical structure, including the laxity and stiffness of the biological joints. Valero Cuevas revealed the force regulation function coming from the quantity and morphologies of the flexor and extensor tendons, especially the complex reticular extensor mechanism [7] [8] [9] [10]. We hypothesised that there must be some underlying relationships between the versatile performance of the human hand and its biological mechanisms. Take one finger as an example, the mechanical output of the fingertip is directly involved in the interaction with the external environment, which can be generated and influenced by joint properties, tendon forces distribution and transmission. Specifically, the joint stiffness could exert a direct effect on the fingertip stiffness and finger body stability. The fingertip output force in space can be potentially influenced by the force regulation of the extensor mechanisms. And the tendon sheaths, as the force transmission system, would participate in shaping the force-velocity characteristics of the fingertip. These intrinsic mechanisms play an indispensable role in the human hand outstanding performance. But how? Would it be conducive to improve its performance if we adopt these structures to the robotic hand? Can this bio-inspired structure design of robotic hand act as the mechanical intelligence to partially substitute the control algorithm so as to perform complex grasping or manipulation with a simple control strategy? This thesis aims to partially reveal this blind spot, and it is actually the very first to systematically investigate the biomechanical advantages of human hands by developing a highly biomimetic robotic hand.

1.1.2 Significance

This project provides the amputees with a more multifunctional robotic hand that can perform a wide range of daily activities like human hands. Also, it affords technical support and a biomechanics foundation for the innovative design of bio-inspired robotic hand. The biomechanical principles and natures of versatile movements of the human hand with dexterity and adaptivity are partially revealed. Meanwhile, the design methods and theories developed in this research can potentially help empower other bionic mechanisms with more biological properties. On the other hand, the proposed robotic hand prototype can be used as an effective scientific tool to investigate some theories and hypotheses of the human hand anatomy, biomechanics, and upper prostheses techniques.

1.2 Aim and objectives

The aim of the research is to design a novel bio-inspired robotic hand with human-hand-like grasping and manipulation capabilities to explore the biomechanical advantages of human hands. It is expected that the proposed robotic hand can realize wide-range grasping and complex manipulation with the simple position control and without the sensory feedback. Thus, several phased objectives are listed below:

•Make detailed research on the anatomy and biomechanics of human hands, so as to figure out the distribution, the detailed structure, the kinematic and kinetic properties of each functional component of the human hand.

•Design and fabricate a highly biomimetic robotic hand, including designing the structure and selecting appropriate materials for the artificial design, from the bones, ligaments to the tendons, tendon sheaths and skin, from the index finger to the thumb and the whole hand, which are all based on the human hand musculoskeletal biomechanics.

•Explore several unique biomechanical advantages of human hands by methods of mathematical analysis and experimental verification, including the variable joint stiffness, the enlarged feasible force space and the augmented force-velocity workspace.

•Construct a motor-tendon actuation system and develop a simple control strategy for the proposed robotic hand prototype to realize human-hand-like grasping and manipulation capabilities.

1.3 Thesis overview

The thesis is organized as follows:

The literature review is presented in Chapter 2. In the literature review, the anatomy of human hands is firstly investigated, including the bones, joints, ligaments, muscle-tendon system and tendon sheaths. Besides, the biomechanics of human hands are also presented. The joint orientation and range of motion are identified according to the coordinate system established in the human hand. Moreover, the research on the unique biomechanical properties embedded in three human hand structures is introduced. Then we reviewed the related work about the bio-inspired design, the joints, the actuation systems and control methods of the previous anthropomorphic robotic hands.

The bio-inspired design and fabrication of one robotic finger is presented in Chapter 3. The multi-layered structure is proposed and realized on the robotic finger design, including the base layer (phalanges and articular cartilage), the second layer (capsuloligamentous structures) and the third layer (tendons and tendon sheaths).

In Chapter 4, three biomechanical properties of human-finger-like structures are analyzed in mathematical methods. They are the joint stiffness associated with ligamentous structure, the feasible force space associated with extensor mechanism and the force-velocity characteristics associated with flexible tendon sheaths. Some related mathematical models are established in this section.

The three biomechanical properties analyzed in the last chapter are verified through a series of experiments in Chapter 5. And a two-finger testbed is constructed and the human-finger-like grasping capabilities are demonstrated.

Chapter 6 describes the whole bio-inspired robotic hand design and fabrication in details. Since the robotic finger design is already introduced in Chapter 3, the thumb and the carpal design are additionally presented in this section, as well as the ligamentous-skeletal structure and tendon arrangement throughout the whole robotic hand. Besides, a customized artificial skin is fabricated for the robotic hand.

In Chapter 7, a motor-tendon actuation system is constructed and the corresponding control theories are presented. Several specific theoretical models are established for controlling this novel robotic hand. To realize the function, a data glove-based posture control method is proposed.

To validate the advantages of the bio-inspired design of the robotic hand, the human-hand-like grasping and manipulation capabilities tests are presented in Chapter 8. The grasping capability is tested based on the Cutkosky and Feix taxonomy and some challenging manipulations are demonstrated in this chapter.

In the final chapter, an overview of the work and key findings in the research is provided and the potential ways of future expansion work are discussed.

Chapter 2: Literature review

This chapter mainly reviews the previous research on the anatomy and biomechanics of human hands, as well as the joint design, the actuation methods and the control strategies of robotic hands. The aim of this chapter is to get a comprehensive view of the basic knowledge of human hands and the development of robotic hands so as to obtain some inspiration of how to design a highly biomimetic robotic hand in this project and control it to realize human-hand-like functions.

2.1 Anatomy of human hand

All the anthropomorphic robotic hand designs are inspired by the human hand. Returning to the original nature of robotic hand design, it's crucial to make detailed research on the anatomy structure of human hands.

Defined in anatomy, the human hand is made up of bones, joints, ligaments, joint capsules, tendons, tendon sheaths, muscles, blood vessels, nerves and skin [57]. Though it's impractical to replicate all the features of the human hand with a complex nervous system, vascular network and self-healing skin tissues, we can still mimic some main features in designing the anthropomorphic robotic hand resorting to our current technologies 3D scanning and printing.

In the following sections, some important anatomical features of human hands are going to be identified to help us have a comprehensive and intuitive understanding of human hand components' structure and function.

2.1.1 Morphology of bones

As shown in Figure 1, the skeletal structure of one hand consists of totally 27 pieces of bones, of which 14 pieces are the phalanges, 5 pieces are the metacarpal bones, and 8 for the carpal bones. Specifically, the trapezium bone acts as the base of the thumb metacarpal bone and its unique articular surface shape plays an important role in realizing the function of the thumb. There are plenty of connection joints among these bones. For instance, the thumb metacarpal bone and the trapezium bone form the carpometacarpal (CMC) joint. And the metacarpophalangeal (MCP) joints are the connection between phalanges and metacarpal bones. Moreover, between every two phalanges, there are two kinds of interphalangeal joints which are the proximal interphalangeal (PIP) joint and the distal interphalangeal (DIP) joint. But for the thumb, there is only one interphalangeal joint which is the DIP joint.

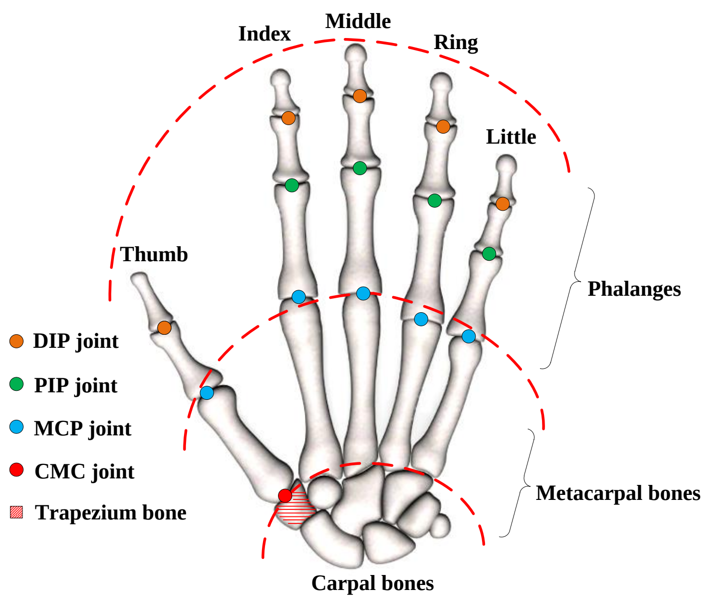


Figure 1. Definition of the bones and joints of the human right hand

2.1.2 Structure of joints

A joint is the connection part of two contiguous bones. It is a kind of capsule structure wrapping around some ligaments outside. The interphalangeal joints (DIP joint and PIP joint) can be regarded as kinds of hinge joints since they mainly providing one degree of freedom which is the flexion-extension motion, except that the DIP joint has a smaller dimension and less motion range compared with the PIP joint. However, the MCP joint allows more kinds of movements owing to its unique surface shape, including the flexion-extension, abduction-adduction and pronation-supination [58]. The CMC joint, as described in the last section, plays an indispensable role in thumb functioning. Its capsule is slightly thicker than the others', allowing sufficient motion range and simultaneously providing enough stability to the joint [59].

The finger joints can all be classified as the synovial joint which is composed of the articular capsule, the joint cavity, the articular cartilage and the ligament (details about the ligaments are in the next section). The articular capsule is a kind of layered structure with an inner synovial membrane and an outer fibrous capsule layer. In the capsule structure, there is some synovial liquid filled. And the articular cartilage covers the articular surface of the bone. Together with the synovial liquid, they provide a good lubrication environment and dexterity condition for the joint movement

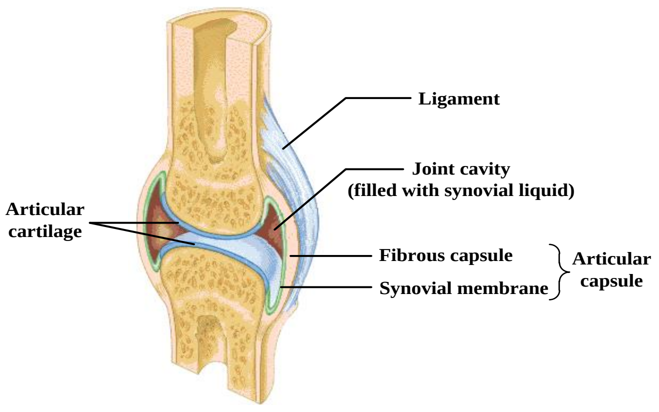


Figure 2. Structure of the synovial joint (modified from reference [60])

2.1.3 Distribution of ligaments

The ligaments are some kinds of fibrous tissues inserted on both sides of the two adjacent bones, distributed among the carpal bones, metacarpal bones and phalanges, which can restrict the range of motion at each finger joint. The overall distribution of the ligaments in the whole hand and the finger is shown in Figure 3. As we can see, there are many ligaments among the carpal bones. At the connection area between the carpal and metacarpal bones, the palmar carpometacarpal ligaments and the metacarpal ligaments bond the base of metacarpal bones and the carpal bones together, which form a solid support for fingers' movement. We can also see that, the deep transverse metacarpal ligaments exist between every two MCP joints, linking them together. Two kinds of important ligaments for every finger joints are defined as the collateral ligaments located at two sides of the joint, and the palmar ligaments (also as the volar plates) located at the palmar side of the joint. They both play a critical role in stabilizing the joint, constraining degrees of freedom, and preventing joint dislocations.

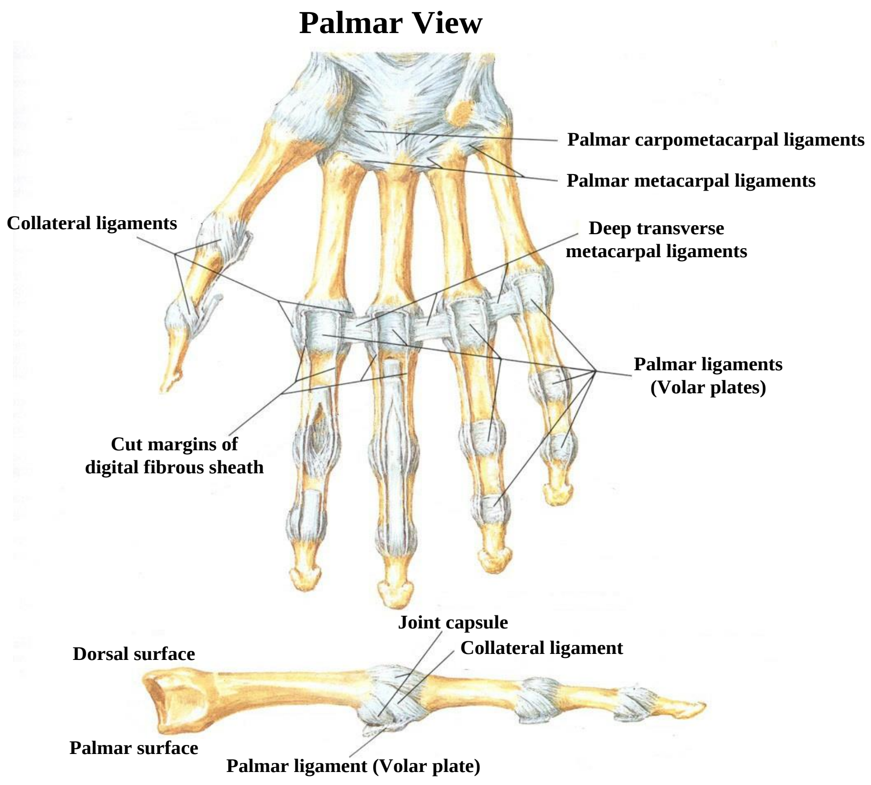

Palmar View
Palmar carpometacarpal ligaments
Palmar metacarpal ligaments
Collateral ligaments
Deep transverse metacarpal ligaments
Palmar ligaments (Volar plates)
Cut margins of digital fibrous sheath
Joint capsule
Collateral ligament
Dorsal surface
Palmar surface
Palmar ligament (Volar plate)

Figure 3. Distribution of the ligaments in hand (modified from reference [61])

Now let's further see the details of the ligaments around the finger joints. Since the ligaments of the metacarpophalangeal and interphalangeal joint are similar, we take the ligaments in the PIP joint for example. As shown in Figure 4, the three-sided ligamentous support system of the PIP joint consists of the cord and accessory collateral ligaments and the volar plate, which is anchored proximally by the checkrein ligamentous attachment. The cord collateral ligament originates on each side of the proximal phalange head's dorsal part. Then, it extends obliquely and distally to the insertion onto the tubercle at the base of the middle phalange. While, the accessory collateral ligament originates from the volar part of the phalange head, inserted onto the volar plate [62]. As a result, the joint is stable during full flexion because of the restriction of the collateral ligament, meanwhile, the volar plate prevents the joint from overextension.

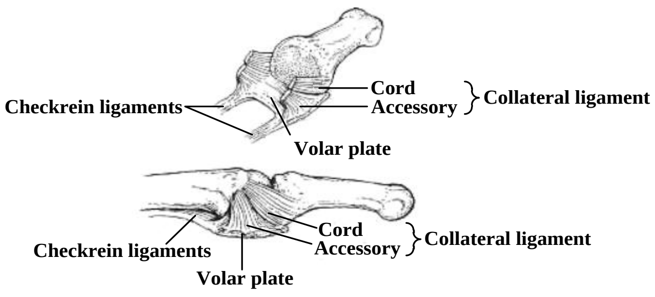


Figure 4. Distribution of the ligaments in PIP joint (modified from reference [63])

2.1.4 Distribution of muscle-tendon system

There are two sets of tendons distributing in the human hand-the flexor tendons which contract to bend the fingers and the extensor tendons which contract to straighten the fingers. They all originate from the muscle groups in the forearm and insert onto the base of the finger joints. The whole distribution structure of tendons is shown in Figure 5. From the dorsal view of the hand, the extensor tendons branch out from the extrinsic muscles of the forearm to the insertions on the dorsal side of the phalanges. As we can see, there are two extensor tendons for the thumb, called the extensor pollicis longus (EPL) and the extensor pollicis brevis (EPB). For the other fingers, the extensor digitorum extends to each distal phalanx, forming an extensor expansion structure respectively. In addition, there are two more extensor tendons: the extensor indicis (EI) for the index finger, and the extensor digiti minimi (EDM) for the little finger, contributing to their extension motion. On the other hand, from the palmar view of the hand, except the thumb, all the fingers have two flexor tendons respectively, which are the flexor digitorum profundus (FDP) and the flexor digitorum superficialis (FDS). They pass through the flexor retinaculum and spread out to each finger together, inserting to the base of the PIP joints for the FDS and the DIP joints for the FDP. The thumb only has one flexor tendon originating from the extrinsic muscle groups in the forearm, called the flexor pollicis longus (FPL), inserting to the base of the DIP joint.

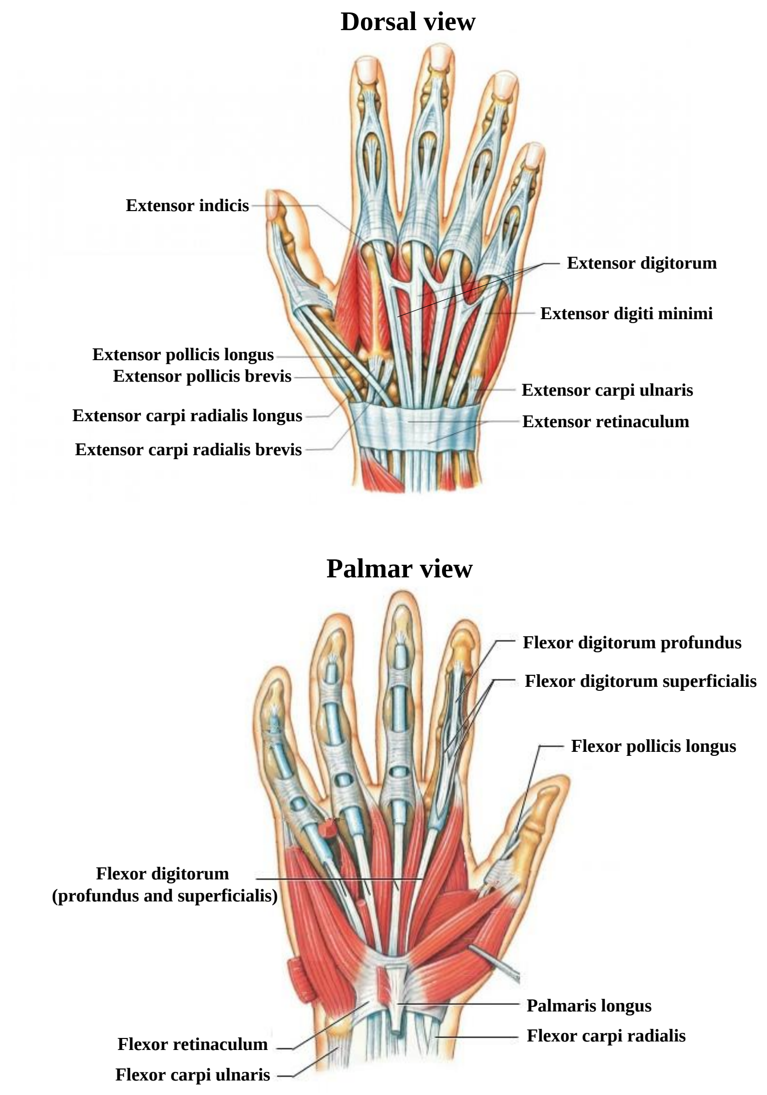


Figure 5. Distribution of the tendons in the right hand (modified from reference [64])

A More detailed structure of tendons along the finger is shown in Figure 6. We can see that, from the dorsal view, after passing through the MCP joint, the long extensor tendon grows from the extensor digitorum to the extensor expansion (hood) and then splits into three parts near the PIP joint: a central band, which is inserted into the base of the middle phalanx, and two lateral bands, which are inserted into the base of the distal phalanx. Two interosseous muscles and the lumbrical muscle connect with the proximal end of the extensor hood. The whole extensor expansion can act as a passive braking system during the flexion motion owing to its complex morphology and elastic material property. And this extensor mechanism can regulate the forces distributed along the finger which is one of the unique biomechanical advantages of human hands. Basing on the research did by Dan Hu and Lei Ren, it was found that the extensor mechanism could help reduce 22% to 61% forces in the intrinsic muscles, and 10% to 41% bone-to-bone contact force at the MCP joint, indicating its great functions on muscle force moderating and risk reduction of injury.

On the other hand, from the lateral view, we can see that the FDP tendon comes out through the FDS tendon whose insertion is at the base of the middle phalanx, inserted to the base of the distal phalanx. Besides, there are some arrows marked along the tendons during the extension and flexion motion. The red arrows represent the force direction of the long extensor tendon (LE), and the black arrows represent the force direction of the interosseous and lumbrical muscles. There is no doubt that the web structure of the extensor expansion plays an important role in both extension and flexion motion, which is a unique biomechanical advantage of the human hand.

As we can see, most of the flexor and extensor tendons originate from the extrinsic muscle groups in the forearm. There are still some muscles distributing in the palm of the hand, called the intrinsic muscle groups, shown in Figure 7, including the thumb muscle group (adductor pollicis, flexor pollicis brevis, abductor pollicis brevis and opponens pollicis), the little finger muscle group (abductor digiti minimi, flexor digiti minimi brevis and opponens digiti minimi), the interossei muscles (four dorsal interosseis and three palmar interosseis), and the lumbrical muscles [65]. They play a crucial role in precise manipulation and power grasp. Therefore, the more detailed distribution, which is really worthy of study, is shown in Figure 8.

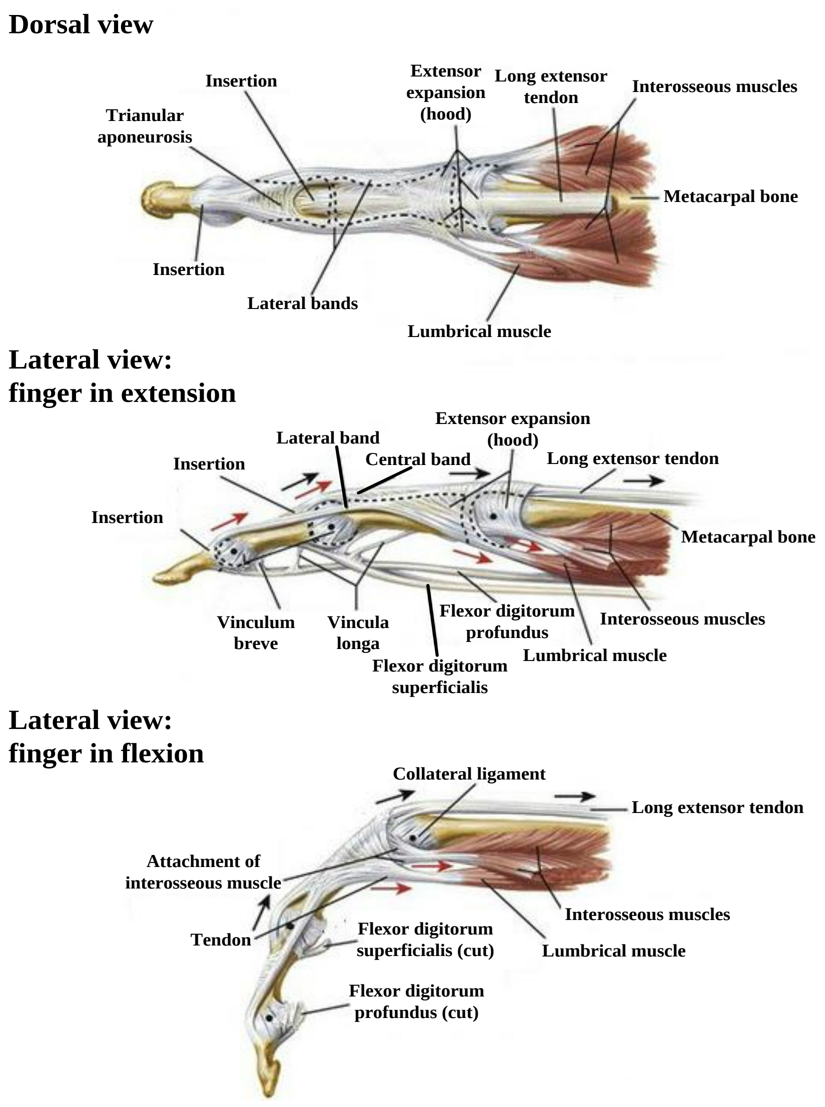


Figure 6. Detailed distribution of the tendons along the finger (modified from reference [66])

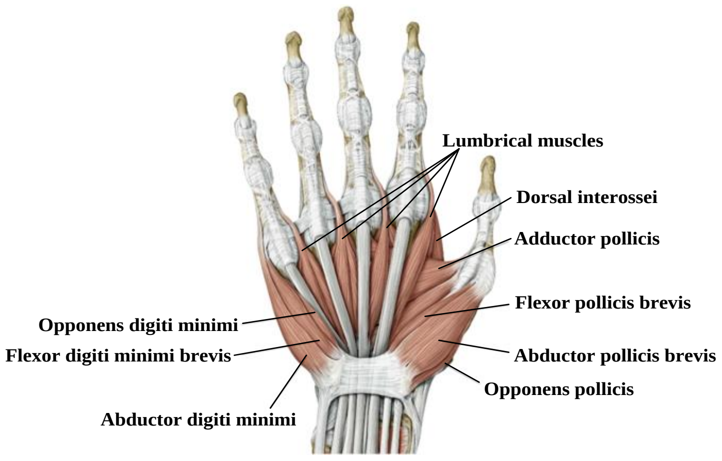


Figure 7. Intrinsic muscle groups of the hand (modified from reference [67])

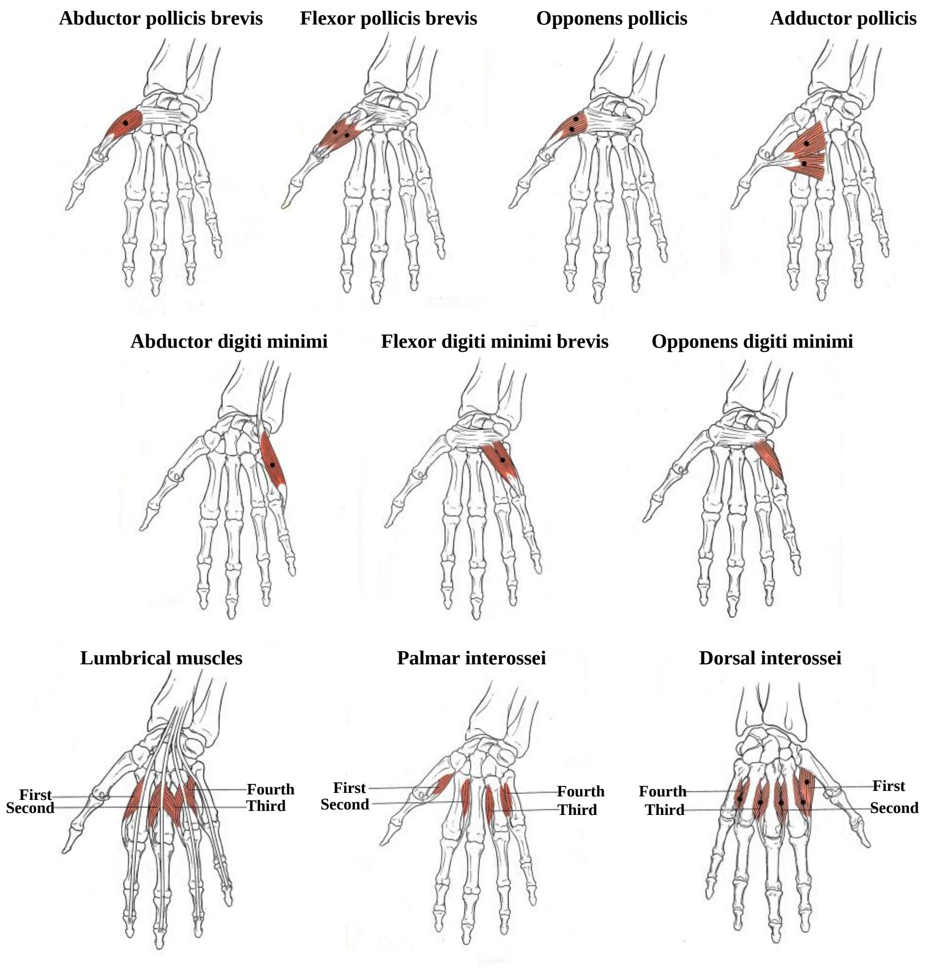


Figure 8. Distribution of each intrinsic muscle group (modified from reference [68])

2.1.5 Structure and distribution of tendon sheaths

A tendon sheath a two-layer membrane, consisting of a synovial sheath layer and a fibrous tendon sheath layer, that wraps around a tendon and has multiple insertions on the dorsal side of phalanges. It allows the tendon to stretch and not adhere to the surrounding tissues [69]. The tendon sheaths of the palm of the hand and the cross-section of the finger are shown in Figure 9.

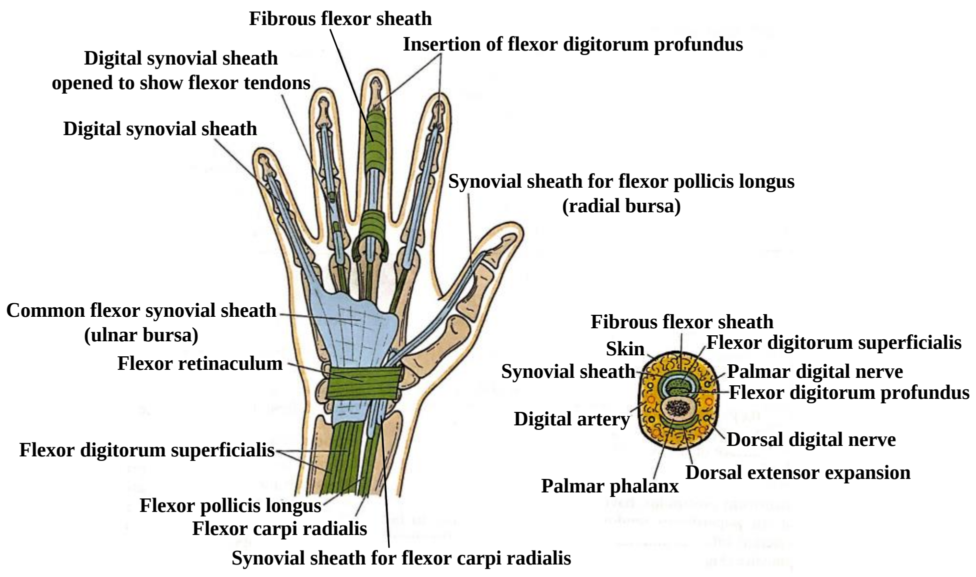

Fibrous flexor sheath
Insertion of flexor digitorum profundus
Digital synovial sheath opened to show flexor tendons
Digital synovial sheath
Synovial sheath for flexor pollicis longus (radial bursa)
Common flexor synovial sheath (ulnar bursa)
Flexor retinaculum
Flexor digitorum superficialis
Flexor pollicis longus
Flexor carpi radialis
Synovial sheath for flexor carpi radialis
Fibrous flexor sheath
Skin
Flexor digitorum superficialis
Synovial sheath
Palmar digital nerve
Flexor digitorum profundus
Digital artery
Dorsal digital nerve
Dorsal extensor expansion
Palmar phalanx

Figure 9. Anterior view of the palm of the hand showing the flexor synovial sheath. Cross section of a finger is also shown. (modified from reference [70])

We can see that a strong fibrous sheath is attached on the palmar side of each finger from the head of the metacarpal to the base of the distal phalanx. Together with the phalanges, the sheath forms a tunnel where the flexor tendons go through. The fibrous sheath is thick along the phalanges but thin and loose over the joints. On the other hand, along each finger, the FDS and FDP tendons share a common synovial sheath. These synovial sheaths permit the tendons to move smoothly inside, where the synovial liquid will reduce the friction. A more detailed structure of the synovial sheath is shown in Figure 10. The whole tendon sheath structure is a really interesting and important anatomical feature that we should pay more attention to in the design of the robotic hand.

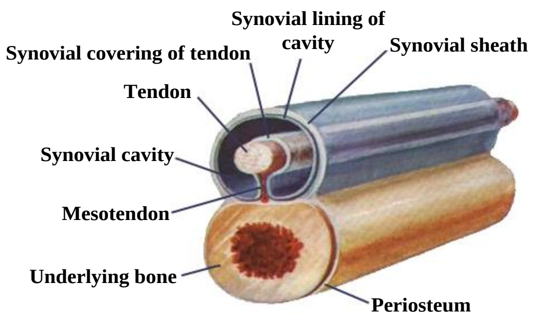


Figure 10. Detailed structure of the synovial tendon sheath (modified from reference [71])

2.2 Biomechanics of human hand

Through the section above, the complexities of the function and anatomy of the human hand have long been recognized. While the development of biomechanics provides us with another analysis perspective to accomplish new goals. From the view of biomechanics, the human hand can be considered as a linkage system. The joints between each articulate are connected by ligaments and capsules and also passed by at least one tendon. The muscles' contraction will pull the tendons to actuate the finger joints to generate a certain movement which is also constrained by the surrounding soft tissues and the articulate surface. It should be noted that a bi-joint or poly-joint mechanism normally exist in the hand owing to the fact that most of the tendons pass over one or more joints.

2.2.1 Coordinate system in human hand

Considering about these features, a three-dimensional model of the hand was established based on the cadaveric study of the human hand [6]. In the mathematical model, six Cartesian coordinate systems were established to define the orientations and locations of joints and tendons (Figure 11). Two coordinate systems are established in both the middle and proximal phalanges, and there is only one coordinate established in the distal phalanx and metacarpal. The distal coordinate systems 2, 4 and 6 are defined at the rotation centre of the middle phalanx, proximal phalanx and metacarpal heads, and the proximal coordinate systems 1, 3 and 5 are defined by translating the distal coordinate systems to the geometrical centre of the joint surface concave. The x-axis is orientated along each phalanx. The positive y-axis direction is defined from the coordinate centre to the phalanx dorsal side, and the positive z-axis is orientated from the centre to the radial side in the right hand.

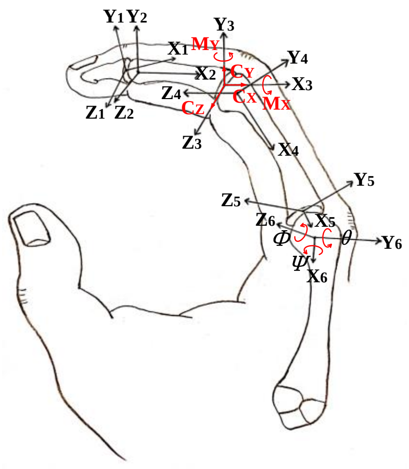


Figure 11. Coordinate systems in the index finger (modified from reference [6])

The constraint forces (C_X, C_Y, C_Z) and moments (M_X, M_Y) at the PIP joint are presented as well, which generated by the joint surface and capsuloligamentous structures. C_X is the axial compressive force, C_Y is the dorsal-volar shear force and C_Z is the radial-ulnar shear force. M_X indicates the axial twisting moment and M_Y indicates the radial-ulnar constraint moment. Besides, as shown at the MCP joint in Figure 11, the orientation angle φ is defined as the flexion-extension angle, the angle θ is defined as the abduction-adduction angle, and the rotation angle ω is defined as the pronation-supination angle.

2.2.2 Joint orientation and range of motion

To measure the orientation and the motion range of the joint in 3D space, the method of biplanar radiographic was often used with some markers attached on each finger phalanx, shown in Figure 12. By this method, the joint angles throughout the finger motion can be recorded and quantitated, e.g. the flexion-extension, abduction-adduction and pronation-supination angles in the motion of grasp or pinch. Since the special surface shape of the trapezium, the new joint coordinate system needs to be established in the thumb, which is shown in Figure 13.

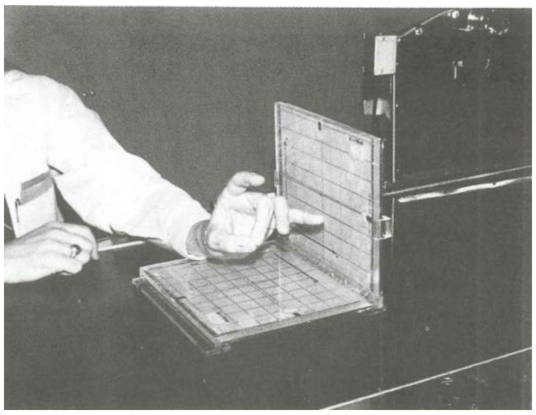


Figure 12. Biplanar X-ray method used to measure the orientation and the motion range of the joint in 3D space. (adopted from reference [6])

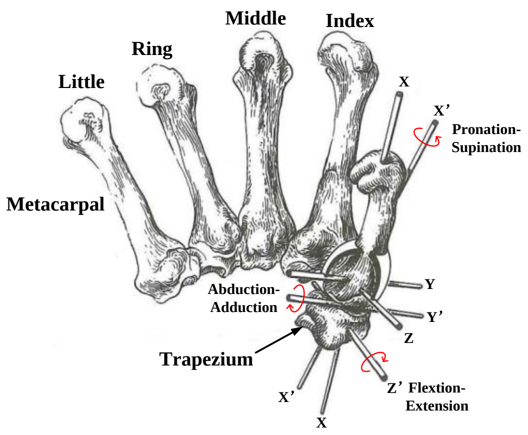


Figure 13. Joint orientation and axes of rotation of the thumb basal joint. (modified from reference [6])

According to the research did by Edmund Y, which involved sixty-three subjects (40 subjects without joint disease and 23 subjects with rheumatoid arthritis or osteoarthritis), it indicated that the measured MCP joint shows a 65~107° flexion motion range. Specifically, the MCP joint in the index finger has an 83° average motion range. And 90°, 88°, 90° are respectively for the MCP joint in the middle, ring and little finger. For the MCP joint extension motion, it has -22° for the index finger, -22° for the middle finger, -23° for the ring finger and -34° for the little finger. For the PIP joint, the flexion motion range varies from 92~125° in all fingers. And the DIP joint has averaged 77° flexion motion range and 11.45° extension range for all the fingers.

As to the abduction-adduction and pronation-supination motions, they are mainly performed by the MCP joint but do exist in the PIP and DIP joints. According to the research did by Gurbuz H, it was found that the index finger has a 41.9° abduction-adduction motion range, and the middle finger has 80.98°, the ring finger has 41.57° and the little finger has 48.53° [72].

As to the thumb joint motion, the flexion-extension motion range of the DIP joint was 100°±9°, the abduction-adduction motion range was 7.5°±10°, and the pronation-supination motion range was 8.4°±9°. For the MCP joint, the corresponding motion range was respectively 45°±16°, 8.7°±3.2°, and 12.1°±4°. And for the CMC joint, it has 53°, 42° and 17° for the flexion-extension, abduction-adduction and pronation-supination motion range [6]. The detailed data to show the tested human finger joint motion range are concluded and presented in Table 1.

Table 1. Orientation angles of the finger joints (data from the reference [6])

Finger	**Joint**	**Orientation Angles (°) (mean value)**		
		Flexion-Extension	**Abduction-Adduction**	**Rotation**

Index	DIP	73 ~ 11	\	\
	PIP	101 ~ 10	\	\
	MCP	83 ~ -22	41.9	\
Middle	DIP	80 ~ 12	\	\
	PIP	103 ~ 12	\	\
	MCP	90 ~ -22	80.98	\
Ring	DIP	75 ~ 12	\	\
	PIP	105 ~ 11	\	\
	MCP	88 ~ -23	41.57	\
Little	DIP	78 ~ 11	\	\
	PIP	103 ~ 6.5	\	\
	MCP	90 ~ -34	48.53	\
Thumb	DIP	100	7.5	8.4
	MCP	45	8.7	12.1
	CMC	53	42	17

2.2.3 Force transmission in human hand

Since the three-dimensional structure of the hand is established, we can see that the finger and thumb can be considered as a linkage system. To simplify the analysis subject, we can regard the DIP and PIP joints as hinge joints, the MCP and CMC joints as universal joints. The tension forces were assumed to be along the tendons or muscles and the external loads were assumed to be in the normal direction at the fingertip in pinch posture or at the middle point of each phalanx in grasp posture. To clarify the tendons and intrinsic muscles involved in each joint constraint, Table 2 is established.

Table 2. Tendons and intrinsic muscles involved in each finger joint (modified from

reference [6])

Finger	Joint	Tendons and intrinsic muscles involved
Index Middle Ring Little	DIP	Terminal extensor (TE)
		Flexor digitorum profundus (FDP)
	PIP	Extensor slip (ES)
		Radial band (RB)
		Ulnar band (UB)
		Flexor digitorum superficialis (FDS)
	MCP	Long extensor (LE)
		Radial interosseous (EI)
		Ulnar interosseous (UI)
		Lumbrical (LU)
Thumb	DIP	Flexor pollicis longus (FPL)
		Extensor pollicis longus (EPL)
	MCP	Abductor pollicis brevis (APB)
		Adductor pollicis (ADD)
		Flexor pollicis brevis (FPB)
		Extensor pollicis brevis (EPB)
	CMC	Opponens pollicis (OPP)
		Abductor pollicis longus (APL)

On the dorsal side of the finger, there is a complicated structure called extensor mechanism. Its anatomy and function have been studies by many researchers [73]. The transmission direction of the force in the extensor mechanism of a finger can be expressed as shown in Figure 14.

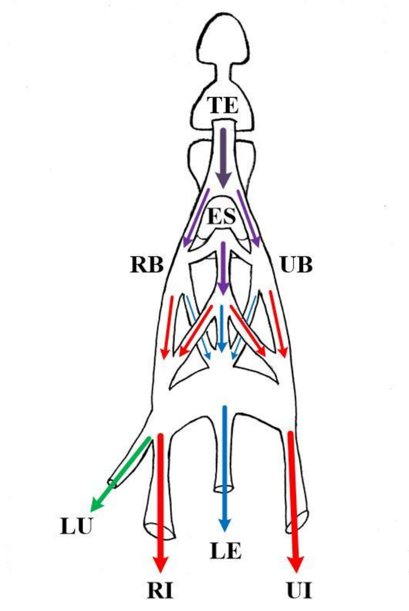


Figure 14. Direction of the force in the extensor mechanism

According to the research of [6], the relationship of force distribution among these tendons can be assumed as:

Index finger:

TE = RB + UB

RB = 0.667 LU + 0.167 LE

UB = 0.333 UI + 0.167 LE

ES = 0.333 LU + 0.167 LE + 0.333 UI + 0.333 RI

Middle finger:

TE = RB + UB

RB = 0.133 RI + 0.667 LU + 0.167 LE

UB = 0.313 UI + 0.167 LE

ES = 0.333 LU + 0.167 LE + 0.313 UI + 0.133 RI

Ring finger:

TE = RB + UB

RB = 0.333 RI + 0.667 LU + 0.167 LE

UB = 0.200 UI + 0.167 LE

ES = 0.333 LU + 0.167 LE + 0.200 UI + 0.333 RI

Little finger:

TE = RB + UB

RB = 0.317 RI + 0.667 LU + 0.167 LE

UB = 0.100 UI + 0.167 LE

ES = 0.333 LU + 0.167 LE + 0.100 UI + 0.317 RI

Likewise, the thumb has similar constraints for the force balance. To solve all the forces, the redundancy problem appears since the available equilibrium and constraint equations are less than the unknown variables. The methods based on the reduction principle and the principles of optimization can be used to solve this problem, of which the details can be found in [6].

2.3 Unique biomechanical properties embedded in human hand structures

In fact, researchers have been trying to explore the fundamental biological principles behind these excellent hand behaviours for a long time. Some unique biomechanical properties of several anatomical structures of human hands were investigated through cadaver tests or computational simulations, such as ligamentous joints, extensor mechanisms and flexible tendon sheaths.

2.3.1 Joint stiffness and ligamentous joints

Joint stiffness influences the joint flexibility and stability. In human fingers, the capsuloligamentous structures and muscle-tendon units provide the constrain forces for the joint stiffness [6]. We commonly called the joint stiffness caused by the active isometric contraction of the muscle-tendon units as the joint active stiffness and the one caused by the passive deformation of muscle-tendon units and capsuloligamentous structures as the joint passive stiffness [11] [12]. Specifically in the joint passive stiffness, the muscle-tendon units provide less than 50% contributions and the capsuloligamentous structures play the dominant role [13]. In 1989, Chao and An made a detailed research on the role of capsuloligamentous structures of the MCP joint in joint stiffness and stability, especially the collateral ligaments [6]. The ligament's length, joint laxity and joint terminal stiffness during joint motions were investigated through cadaver tests. They found the dorsal portions of both collateral ligaments provided the main restraints in joint flexion and the terminal stiffness in rotational displacement increased with the joint flexion angles increasing. Similar results were also found in Werner's research with testing more cadaver hand specimens [14] and in Lutsky's research through in vivo study [15]. To further study the deformation of the collateral ligaments in MCP joint, a three-dimensional model of MCP joint was created by Toshiyuki [16] and the change in the shape and length of each ligament portion during flexion was calculated. Actually, the joint stiffness change is mostly resulted by the deformation of the ligaments, which is the unique property of ligamentous joints and brings some biomechanical advantages. For instance, the low joint stiffness when straightening the finger can maintain good dexterity, and the high joint stiffness when full flexing the finger can substitute the intrinsic muscles to help resist the lateral force when pulling a rope or pinching a key [17]. This kind of floating ligamentous joint has hardly been precisely reproduced on robotic fingers. The robotic hand with ligaments-equipped joints designed by Zhe Xu [18] used the fishing line to partly replicate the ligamentous structure but the ligaments were designed as the linear shape not the band shape. The robotic hand designed by Chepisheva [19], the HR-hand designed by Ooga [20], the 3D-printed anthropomorphic soft skeleton hand designed by Hughes [21], and the 3D-printed biomimetic robotic finger designed by Tebyani [22] all used rubber-like materials to act as the ligaments which cannot well-perform similar properties to human ligaments since the microstructure of rubber is isotropic but that of the human ligament is anisotropic. Though Hughes has used his 3D printed skeleton hand prototype to study the passive behaviours of bionic joints, the research was still mainly focused on the joints' natural straightening state. In this article, the bionic ligamentous joint was highly replicated and the joint stiffness properties were systematically investigated by using proposed physical models.

2.3.2 Endpoint feasible force set (space) and extensor mechanisms

The concept of 'feasible force set' is proposed to mathematically describe the maximum force the fingertip can generate in every space direction, and all the possible linear combinations of these force vectors create the 'feasible force space', defining the mechanical capabilities of versatility and feasibility [23] [24] [25]. Based on this, Inouye associated the feasible force set with the hand manipulation performance [26]. The feasible force set henceforth became one of the main evaluation metrics for robotic hands or fingers [27]. It was also used to demonstrate the functional effect of the human hands' tendon routing distribution and the complex reticular structure extensor mechanism [28] [29] [31].

Valero-Cuevas [7] [8] systematically explored the influence of the extensor mechanism on the fingertip force. A computational model group was developed to simulate topology of extensor mechanism based on the model of Winslow's tendinous rhombus for exploring the tension distribution in the structure [30]. Subsequently, making use of the model, the impact on the feasible force set produced by the extensor mechanism was investigated, indicating that this network structure itself can regulate the tensions propagating to the finger joint to enable various fingertip force production capabilities [9]. But there was no comparison result to explain the biomechanical advantages of the extensor mechanism structure. To better illustrate the effect of the extensor mechanism on the fingertip force, two computational musculoskeletal models of net extensor and linear extensor were built by Synek [31], providing a result that the average forces produced by the net extensor was considerably larger. However, the research still remained on the 2D plane and computational simulation study. With the morphological advantage of the extensor mechanism gradually revealed, it was also increasingly used on the robotic hands to simplify the control architecture and achieve better biomechanical performance [18] [20] [32] [33]. They all fabricated by nylon strings, high-density polyethylene strings or rubber sheets. In spite of a number of application instances of the extensor mechanism, there was no quantitative evaluation of the effect on specific behaviours of robotic hands produced by this structure, especially the fingertip feasible force space, which was studied in detail by using physical models in this research.

2.3.3 Endpoint force-velocity characteristics and flexible tendon sheaths

The force-velocity characteristics are often used to describe the mechanical output power of actuators and end effectors of human beings [34] [35] or robotics [36] [37] [38]. For example, it has important implications for evaluating human muscle efficiency and fatigue [35]. The hyperbolic-like force-velocity relationships of human muscles were normally found by researchers [39] [40]. Haeufle and Schmitt adopted quick-release test to measure the biological muscle characteristics and obtained similar hyperbolic force-velocity relation [34]. In the robotic system, Kevin designed an elastomeric passive transmission pulley which can autonomously adjust its radius according to the tension on the actuation string so as to optimize the robotic finger's force and velocity outputs [38]. In his research, the fingertip force and velocity were tested in separate and no dynamic force-velocity characteristics were investigated like the human muscles research did. But it did provide an idea that the endpoint force-velocity characteristics can be accordingly adjusted through the design in transmission system between the actuators and the end effectors. And this transmission system exists in human hands which is the pulley-like flexible tendon sheaths structure.

We believe the flexible tendon sheaths structure potentially play a role on fingertip force-velocity characteristics adjustment. Amis and Jones [41] revealed the detailed structure of tendon sheaths and found their bulging behaviour during flexion. Lin [42] investigated the mechanical properties of the pulley-like tendon sheaths system and explained the function of constraint the tendons to prevent bow-stringing. Some researchers tried to replicate this human-hand-like structure on robotic hands. Zhe Xu [18] used laser-cut rubber sheets to act as the elastic tendon sheaths on his robotic hand and mentioned the function on adjusting the moment arm from the tendon to the joint. But the structure of tendon sheath was simplified. Chepisheva [19] used PTFE tubes to be the tendon pulleys and Ooga [20] used polyethylene tubes, which were both rigid materials. Tebyani [22] printed the tendon sheath together with the bones by using some flexible printing materials. Though researchers have used to adopt the human-hand-like tendon sheath design on robotic hands, no detailed study has been conducted on how the flexible tendon sheaths influence the fingertip force-velocity characteristics, no matter on human hands or robotic hands. In this article, this blind point was uncovered and investigated in detail.

2.4 Bio-inspired design of robotic hands

To advance our understanding of these neuroanatomical and physiological mechanisms, as well as reproduce such mechanisms in robots, quite a few physical models of bio-inspired robotic hands have been built in the last few decades [43], which however still lag far in practice comparing with the excellent performance of human hands, especially in terms of the versatility and dexterity. With the increasing in-depth research on human hands, the robotic hands have evolved from the purely mechanized rigid structures to the anthropomorphic soft-rigid hybrid structures which immensely exceed the performance of the former when taking no account of the control algorithm [44] [45].

In the early stage, the biological characteristics applied on the robotic hands mostly embodied from the appearance and the whole muscle-skeleton morphology, such as the amount of the end effectors, the degrees of freedom of the joints and the tendon-driven actuation method, such robotic hands including, to mention but a few, the Utah/MIT Hand [46], the Belgrade/USC Hand [47], the Southampton Hand [48], the Gifu hand III [49], the Cyber hand [50], the DLR/HIT Hand II [51], the Robonaut hand II [52] and the Shadow Hand [53]. Though the structural similarity to the human hands enabled these robotic hands to possess enhanced manipulation capabilities, the totally rigid components limited their adaptivity to different objects and external environments. For instance, at the moment of contact, just a minor collision could result in a fail grasping. While the truth is that the uncertain interaction environment and the limitations of the control algorithm cannot make the collision be completely avoided.

To improve robotic hand performance, soft materials were introduced to the robotic hands, such as the RBO Hand 2 with the soft body [54], the UB Hand 3 [55] and the Open Bionics Hand [56] with elastic joints, and the Awiwi Hand with the actuation system adopting elastic components [17]. With the passive behaviours supported by the soft materials, the improved grasping quality can be realized, however, accompanying by the reduced manipulation performance and load capacity.

The solution just hides in human hands. In recent years, a growing number of highly biomimetic robotic hands employing more human-like features were proposed. A highly bionic robotic hand with ligaments-equipped joints, reticular extensors and elastic tendon sheaths was designed by Zhe Xu [18], which can perform almost all the grasping types defined by human hand taxonomy only with data glove control. Other anthropomorphic robotic hands were continuously emerging, such as the robotic hand from the University of Cambridge which can simply use chopsticks [19] and the HR-hand actuated by McKibben muscles [20]. To investigate the impact of the passive behaviour of the human hands, the research group from the University of Cambridge developed a 3D-printed anthropomorphic soft skeleton hand with flexible capsular joints, largely facilitating the dynamic behaviours and interactions with the piano [21]. And recently Tebyani [22] designed a biomimetic cable-driven robotic finger and 3D printed all the components at one time. Most of the essential anatomical structures of human fingers were replicated on the model, but the overall performance still needed further improvement owing to the limitation of 3D printing materials.

Compared with the conventional ones, better performance and more functions were realized in the bio-inspired robotic hands. Even so, there is still a lot of room for improvement. Except for the technical limitations, the lack of understanding of the fundamental principles and mechanisms of biological features were the main obstacles.

2.5 Joint design of anthropomorphic robotic hands

A highly biomimetic robotic hand has been increasingly needed for many areas such as space exploration, industrial manufacture, medical treatment and personal assistant. While, some challenges must be overcome, including reaching the same degrees of freedom (DOF) of the human hand and restoring human-level dexterity. Considering this, the joint design is an inevitable and crucial part of the design of the anthropomorphic robotic hand. The joint of the human hand is almost a perfect design, with its unique surface shape which can determine the degrees of freedom, with its capsule structure and ligaments which can set the range of motion for the joint and contribute to the finger compliance, and with its cartilage and synovial fluid which can provide low-friction contact between two articulated surfaces. The researchers have been sparing no efforts to find the most effective design to mimic the joint of the human hand. In general, artificial joints can be classified into the following types.

2.5.1 Hinge joint in robotic hands

The primary orientation of the finger joints is flexion-extension. Basing on this, the hinge joint was the mainstream design at the initial stage of the development of the robotic hand, because of its simple structure and cheap manufacture costs. Even in today's world, many state of art robotic hands are still using this kind of joint. It just has only one degree of freedom which can meet the needs of numerous tasks.

The NAIST hand (Figure 15), which was designed by Nara Institute of Science and Technology, has three degrees of freedom in each finger, two of which exist in the MCP joint and another in the PIP joint. Besides, its DIP joint and PIP joint are coupled together [74]. As we can see, every joint in this robot hand is a typical hinge joint. The Southampton hand (Figure 16) studied by the University of Southampton also used the hinge joint to link each phalanx [48]. The Cyberhand which is shown in Figure 17, has 16 DOFs with a 0~90°flexion range of each joint [50]. And the ACT (Anatomically Correct Testbed) hand (Figure 18), whose finger bones' shape looks like human's, also utilized a novel hinge joint inside to realize as many degrees of freedom as possible. There are four DOFs in each finger of the ACT hand, and five for the thumb. More DOFs exist in the base of the ring and little fingers in the palm [75].

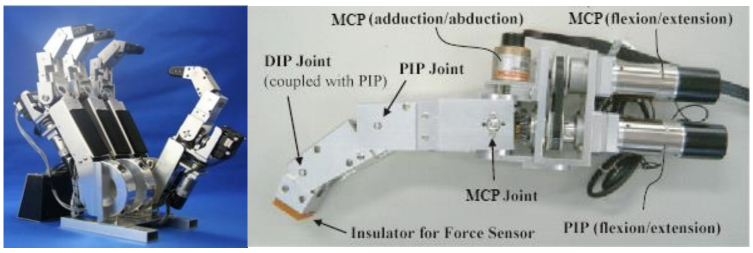


Figure 15. The NAIST hand with hinge joints (adopted from reference [74])

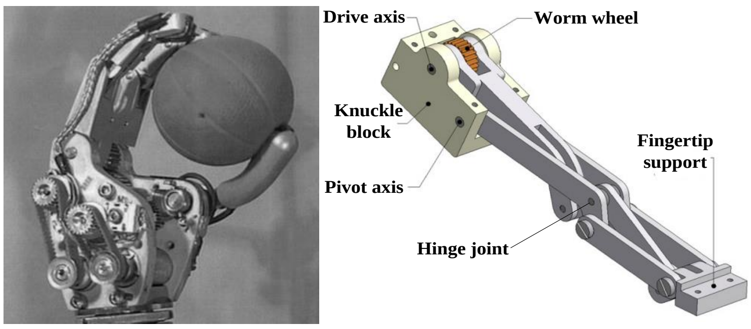


Figure 16. The Southampton hand with hinge joints (adopted from reference [48])

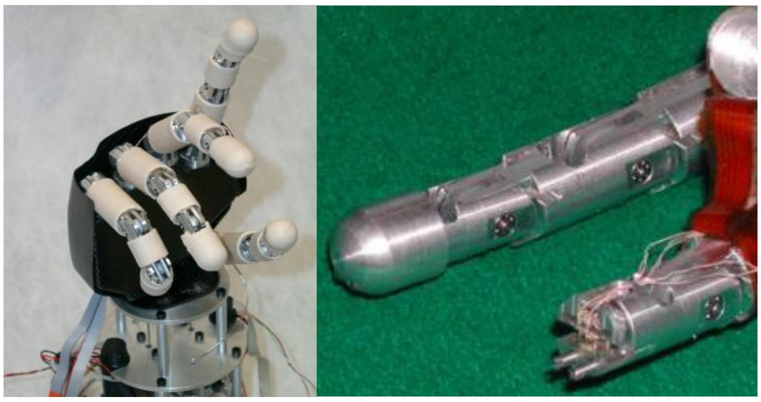


Figure 17. The Cyberhand with hinge joints (adopted from reference [50])

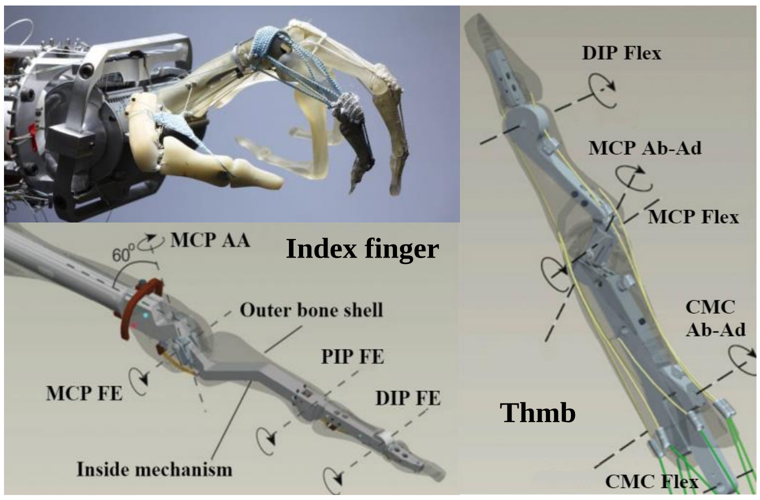


Figure 18. The ACT hand with hinge joints (adopted from reference [75])

2.5.2 Elastic joint in robotic hands

As we can see, the hinge joint has a certain degree of limitations to the joint motion, since it only has one degree of freedom which cannot match all the joints of the human fingers, especially for the MCP joints. Therefore, some researchers started to design a kind of elastic joint with some elastic materials such as the spring, rubber and other polymer synthetic materials, to mimic the high dexterity of the human finger joints.

The UB hand 3 (Figure 19) developed by the University of Bologna just used the coiled spring as the elastic joints of the hand. Each finger of the prototype has 4 degrees of freedom, resulting in a total of 20 degrees of freedom in the whole robotic hand [55]. The iRobot-Harvard-Yale (iHY) SDM hand (Figure 20) is an ingenious design that also used elastic joints to enhance its compliance and passive adaptability [76]. The hand was fabricated by using polymer-based SDM (Shape Deposition Manufacturing), which is a layered manufacturing technique. It can simultaneously fabricate the rigid links (by tough polymers) and compliant joints (by elastomeric flexures) together. , Also, some sensing and actuation components can be embedded in the structure. In this way, fewer seams and fasteners are needed so that mechanical failure can be reduced [77]. In the condition of no actuation, the joint angles are set as 25 °for the PIP joint and 45 °for the DIP joint, which was based on optimization results in previous studies [78].

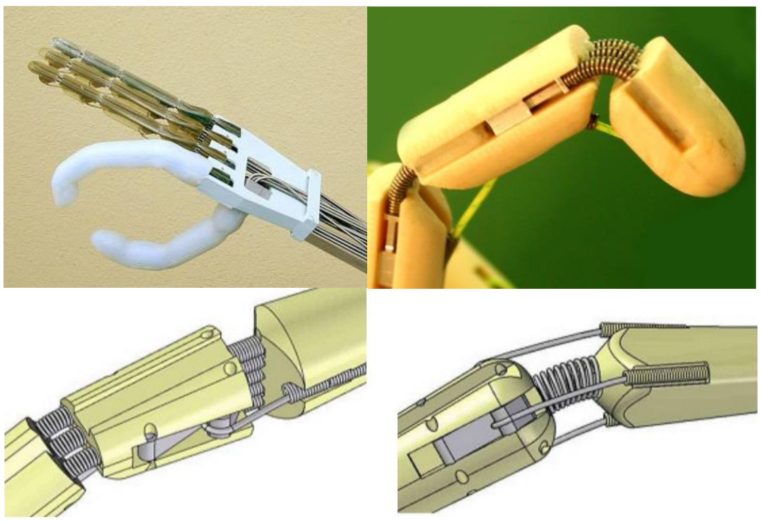


Figure 19. The UB hand 3 with elastic joints (adopted from reference [55])

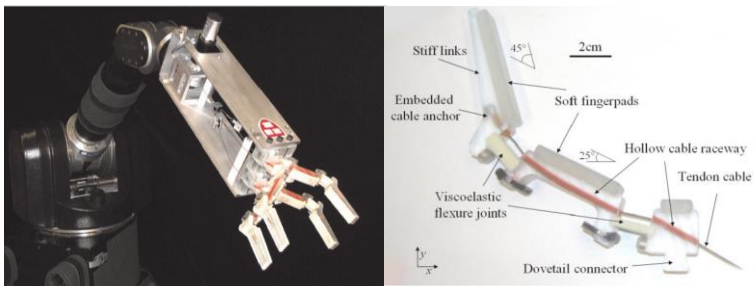


Figure 20. The iHY SDM hand with elastic joints (adopted from reference [76])

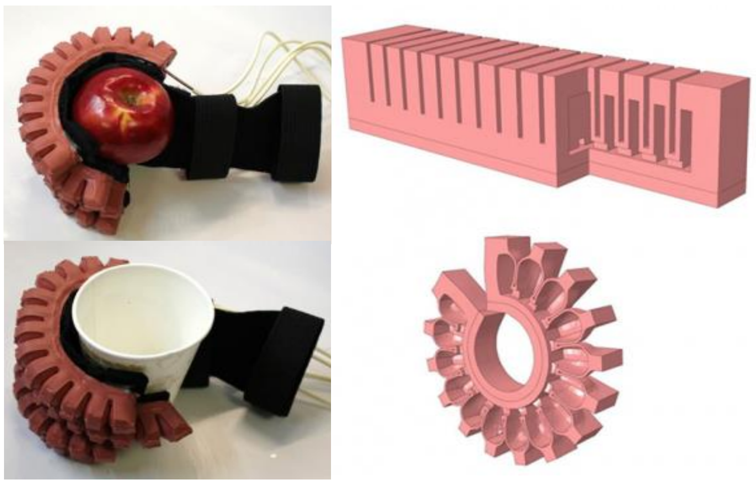


Figure 21. The PneuNets actuator with elastic fingers (adopted from reference [79])

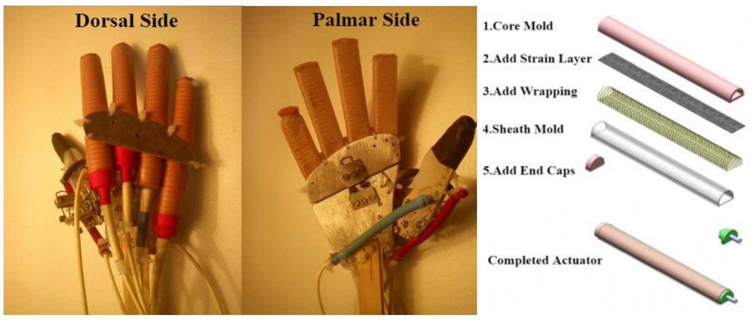


Figure 22. The soft prosthetic hand with elastic fingers (adopted from reference [80])

Another special kind of robotic hands is the soft robotic hand. The whole actuator is made soft, using PneuNets (pneumatic networks) or reinforced fibre. The PneuNets bending actuators (Figure 21) are a kind of soft actuator that was firstly designed and adopted by the Whitesides Research Group at Harvard. Inside the elastomer, some channels and chambers are fabricated, generating predesigned motions when pressurized. By modifying their material properties and geometrical shape, motion control is realized. For instance, the most expansion normally appears at the structure with the thinnest walls. Thus, the researchers can design the wall thickness throughout the whole actuator so as to obtain the desired motion [79]. While a group of students from India designed a soft prosthetic hand (Figure 22) for amputees which uses the fibre-reinforced bending actuator. Four fibre reinforced bending actuators are fabricated as the index, middle, ring and little fingers of the robotic hand. And pneumatic artificial muscles are used to actuate the thumb [80].

In sum, the cost-effective, sufficient grip strength for activities of daily living, low maintenance, and lightweight are the main concerns of this kind of soft robotic hand.

2.5.3 Biomimetic joint in robotic hands

Although the researchers have been struggling to push their ideas on the finger joints design of the robotic hand, it is still too hard to reach the performance level of the human hands. While, with the development of the 3D scanning and printing technology, the researchers start to consider utilizing these advanced techniques to copy the unique surface shape and the complex fibrous structure of the joint, simultaneously conserving the dexterity and the stability of the finger joint. A highly biomimetic robotic hand (Figure 23) made by Zhe Xu from the University of Washington was just inspired by the human finger joints structure and made it come true. They 3D printed the finger bones and equipped them with crocheted ligaments and laser-cut soft tissues. Besides, to mimic the extensor mechanism and the intrinsic muscles, a kind of resilient laser-cut rubber sheet was adopted [18]. It takes the biomimetic level of the robotic hands to a new higher level. Other anthropomorphic robotic hands (shown in Figure 24), such as the robotic hand from the University of Cambridge [19] (a), the HR-hand [20] (b), the 3D-printed anthropomorphic soft skeleton hand designed by Hughes [21] (c), and the 3D-printed biomimetic robotic finger designed by Tebyani [22] (d), all aimed to replicate the human hand structures. They used rubber-like materials to act as the ligaments in the biomimetic joints.

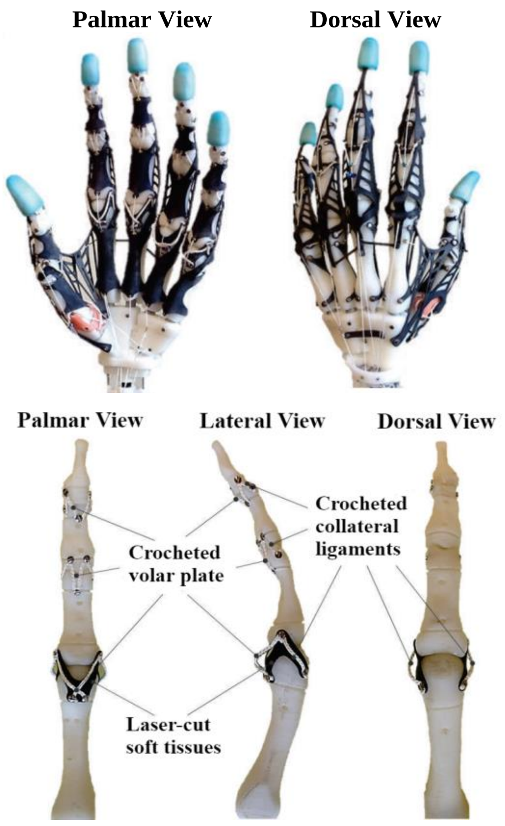


Figure 23. The UW robotic hand with biomimetic joints (adpted from reference [18])

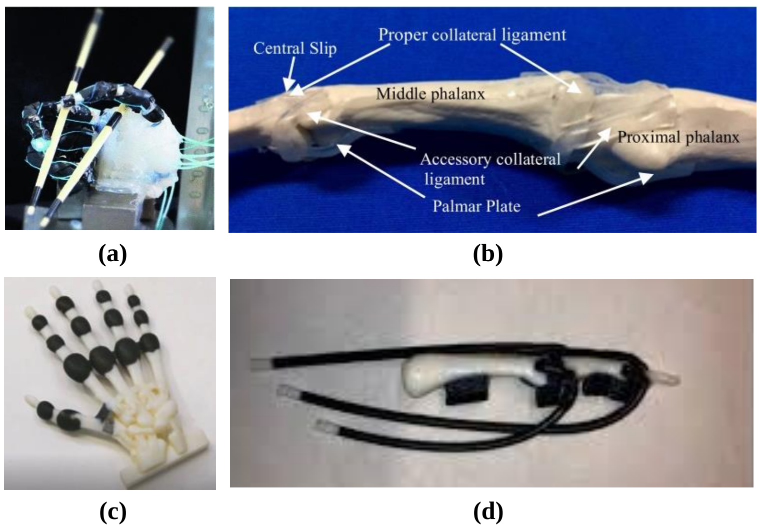


Figure 24. Anthropomorphic robotic hands or fingers with biomimetic joints. (a) Robotic hand from University of Cambridge. (b) Joint structure of HR-hand. (c) 3D-printed anthropomorphic soft skeleton hand. (d) 3D-printed biomimetic robotic finger. (adopted from references [19] [20] [21] [22])

2.5.4 Joint in human hand joint anthroplasty

In order to explore all the possible finger joint designs, we did some research about the joint for human’s finger arthroplasty, also called finger joint replacement. It is a technique that is similar to hip-or knee-replacement, which are able to replace damaged finger joints to reduce the pain of arthropath. SBi (Small Bone innovations) is one of the companies that are focused on the small bones and joints market. There are a variety of artificial finger joints produced by this company, including SRTM PIP joint, SRTM MCP joint, Silicone PIP, Silicone MCP and Preflex MCP, shown in Figure 25 [81]. The SRTM series joint implant includes two components which are an ultra-high molecular weight polyethylene (UHMWP) component and a titanium alloy stem. The Silicone series joints are fabricated by Silflex II advanced elastomer which can reduce ulnar drift and correct deformities.

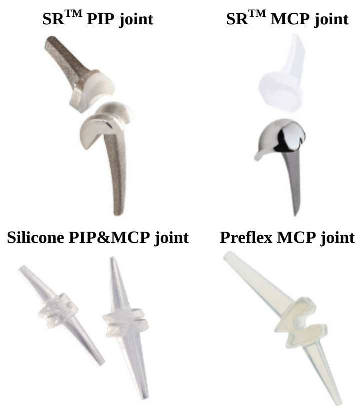


Figure 25. The artificial finger joints from SBi (adopted from reference [81])

Though we don’t need to worry about the rejection of the human body in the process of robotic hands design, it can still be inspired by the artificial finger joints for human joints replacement.

2.6 Actuation system design of anthropomorphic robotic hands

From the anatomical perspective, all the human fingers are connected with the extrinsic muscles in the forearm and the intrinsic muscles in the hand through the long or short tendons. And based on the biomechanics, the muscles act as powerful drivers, pulling the fingers to move through the high strength lines (tendons) which are well-distributed in the forearm and hand. While considering the complexity of control strategies and the structure, the researchers have still been trying to explore other drive methods to make a balance on the design. As we can see, there are many different ways to actuate the multiple finger joints, such as the motor direct-driven, the tendon driven, and the Shape Memory Alloy driven. Each drive method can bring its advantages into the robotic hand design, moving towards an increasingly mature application level.

2.6.1 Motor-Gear linkage actuation

The motor drive is a method that most widely used in the robotic hand because of its high efficiency and energy saving. With the development of electric motors and precision machining technologies, the robotic hands can use electric motors together with rigid gears and linkage system to replace the tendons, resulting in many advantages, such as high control precision and easy kinematic modelling. We call this kind of motor actuation method without flexible connection as motor rigid driven (MRD).

The NAST hand mentioned in the previous section is one of them, adopting the motor drive with gear set and linkage. As shown in Figure 26 (a), all the three motors are mounted in the palm to respectively actuate the MCP joint adduction-abduction motion, MCP joint flexion-extension motion, and PIP joint flexion-extension motion (Figure 26 (b)). Its flexion-extension motions of DIP and PIP joints are coupled by using a linkage system as shown in Figure 26 (c), so the actuation for the PIP joint can also make the

DIP joint generate flexion-extension motion [74].

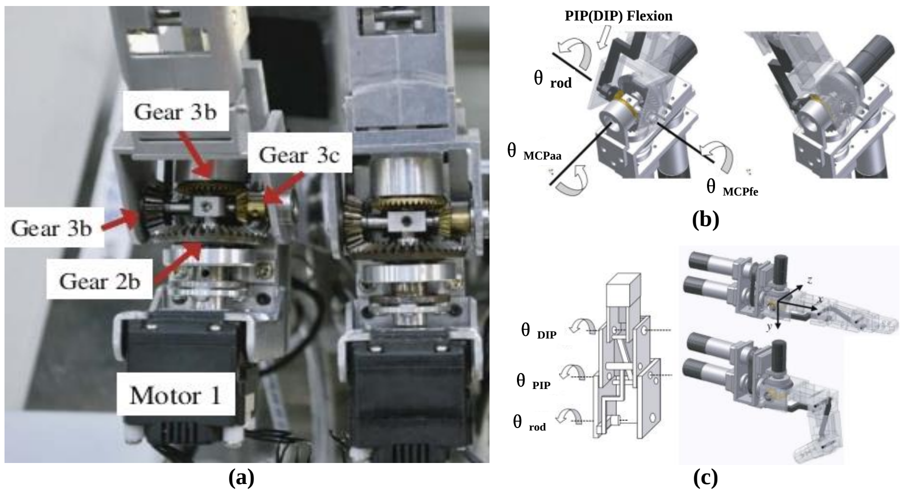


Figure 26. The MRD system of the NAIST hand. (a) Overview of gear mechanism. (b) Three-axis driving gear mechanism. (c) Coupling link mechanism. (adopted from reference [74])

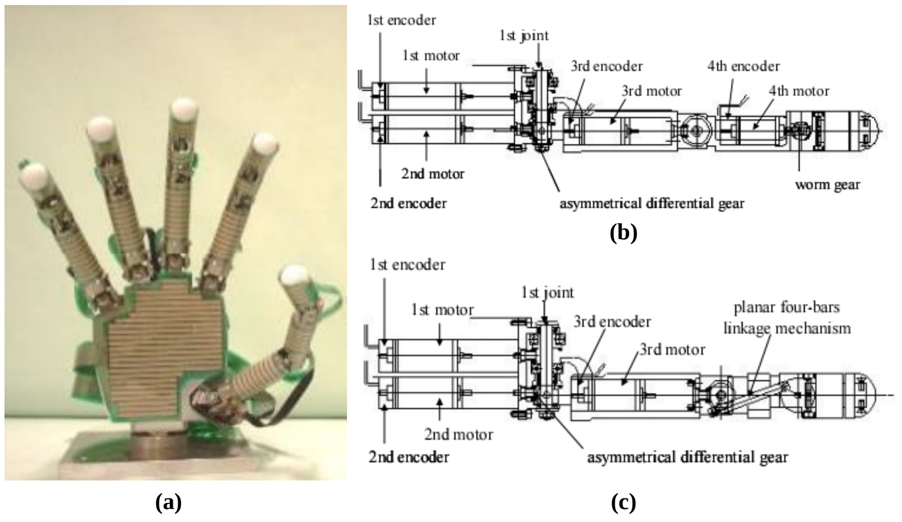


Figure 27. The Gifu hand III and its MRD system. (a) The Gifu hand III prototype. (b) The thumb structure. (c) The fingers structure. (adopted from reference [49])

The Gifu hand III (Figure 27) made by Gifu University also adopted the motor drive with a four-bar linkage mechanism. There are four degrees of freedom existing in the thumb and three degrees of freedom in each finger, allowing the abduction-adduction motion as well as the flexion-extension motion. The actuation methods of the fingers and the thumb are quite similar, except that a four-bar linkage mechanism is adopted in the fingers. Thus, the Gifu hand III has 20 joints with 16 DOF [49].

2.6.2 Motor-Flexible wire actuation

From the view of the development of robotic hands, there are some researchers still keen on using motor drive but with some flexible wires such as tendons and belts, to actuate multiple fingers, just like the human hands. We call this kind of motor actuation using flexible connection with fingers as motor flexible driven (MFD).

DLR/HIT Hand II (Figure 28) is a kind of MFD robotic hand designed by HIT (Harbin Institute of technology) and DLR Institute for Robotics and Mechatronics. Likewise, there are four joints and three DOFs in each finger, of which two joints are coupled. They used timing belts and steel wires to act as the transmission system instead of any rigid gears or linkages so that flexibility and safety performance can be obtained [51].

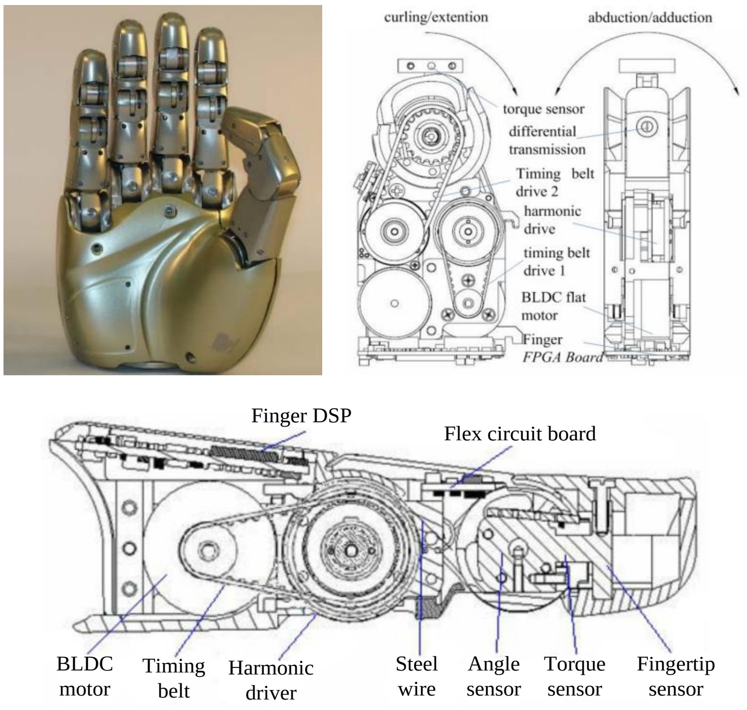
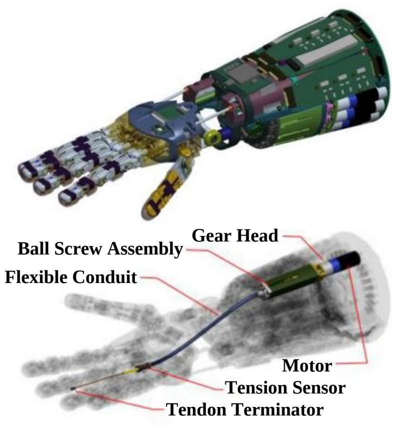


Figure 28. DLR/HIT Hand II and its MFD system (adopted from reference [51])



.

Figure 29. Robonaut hand II and its MFD system (adopted from reference [52])

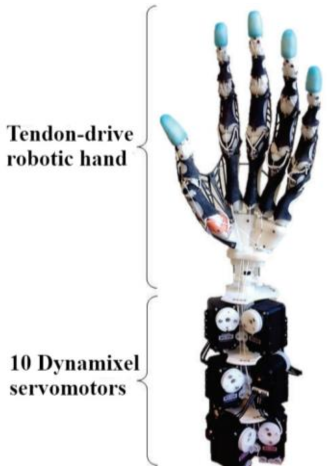


Figure 30. The highly biomimetic robotic hand of UW and its MFD system (modified from reference [18])

The second generation Robonaut hand (Figure 29) was developed by General Motors and NASA together for the purpose of more closely mimicking a human hand. Its actuation system consists of the motor, gear head, ball screw, tendon, conduit, tension sensor, and terminator, shown in the second row of Figure 29. The motor drives the ball screw and pulls the tendon through the flexible conduit [52].

The highly biomimetic robotic hand made by Zhe Xu also used the motor-tendon driven method. Ten (nine MX-12W and one AX-12A) were used to mimic the important large muscles and actuate the robotic hand, as shown in Figure 30, of which two for the flexion-extension motion of the ring and little fingers, two pairs for the independent flexion-extension motion of the index and middle fingers, one for their coupled abduction-adduction motion, one for the thumb extension and abduction, and two for the thumb flexion and adduction. Besides, there is another underactuated DOF in the palm [18].

2.6.3 Pneumatic cylinder-Tendon actuation

Some robotic hands adopt pneumatic cylinders to act as the air muscles to drive the multiple fingers through tendons.

The Shadow Dexterous Hand (Figure 31) developed by the Shadow Robot Company is one of them. The models E2P1L and E2P1R use Shadow's pneumatic "Air Muscle" actuation system [53]. In order to improve the performance of the Shadow hand air muscles drive system, the research group of UW developed their own pneumatic actuation system which was also used in the previous version of the UW hand (as shown in Figure 32) [82]. This actuation system allows the Shadow hand to generate larger motion velocity than a human hand (70 millisecond full range movement and 30 millisecond delay), sufficient forces (40 N at each finger tendon, 125N at each wrist tendon), and high compliance behaviours [83].

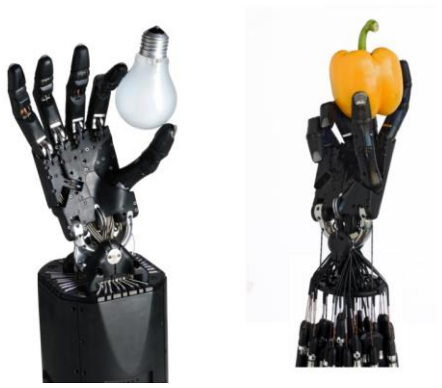


Figure 31. The Shadow hand with air muscles actuation system (adopted from

reference [53])

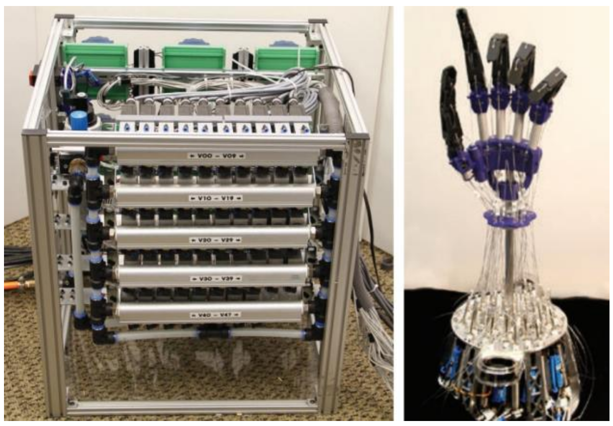


Figure 32. The pneumatic cylinder-tendon actuation system in the previous version of the UW hand (adopted from reference [82])

2.6.4 Shape memory alloys actuation

A large variety of unconventional drive methods for the robotic hands or the upper limb prostheses have already been investigated for the purpose of making up the deficiencies of the motor drive method and pneumatic drive method.. For example, the high weight and cost as well as the loud noise are the main problems for the motor drive method. And the system complexity and large space requirement of the pneumatic drive method also make troubles for the researchers. Thus, using the Shape Memory Alloys (SMAs) becomes one of the most popular and promising alternative novel actuation methods owing to their special material properties and mechanical behaviours. They can return to the predesigned shape in the condition of a certain temperature [84].

A 20-DOF robotic hand designed by DeLaurentis and Mavroidisnwas one example which was actuated by SMA wires [85]. And the five-fingered SBC hand, designed by Cho, Rosmarin and Asada, used the SMA wires to drive its 16 degrees of freedom, as shown in Figure 33. To heat the SMA wire, a voltage difference was applied on the two ends of the wire so as to generate the current on the wire. In this way, the SMA wire could deform to its originate shape [86]. Jung, Bae and Moon have proposed a lightweight SMA actuated five-fingers handwith only 6 controlled DOFs [87]. Lee, Okamoto and Matsubara installed the SMA wire inside the fingers of the robotic hand (Figure 34) [88]. A multifunctional prosthetic hand actuated by SMA was also developed by Konstantinos Andrianesis and Anthony Tzes (Figure 35). By using a direct electrical current or PWM method to heat the SMA wire, the desired actuation can be realized [89].

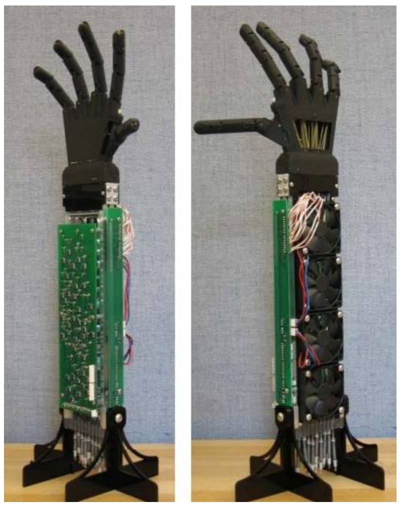


Figure 33. SBC hand with SMA actuation system (adopted from reference [86])

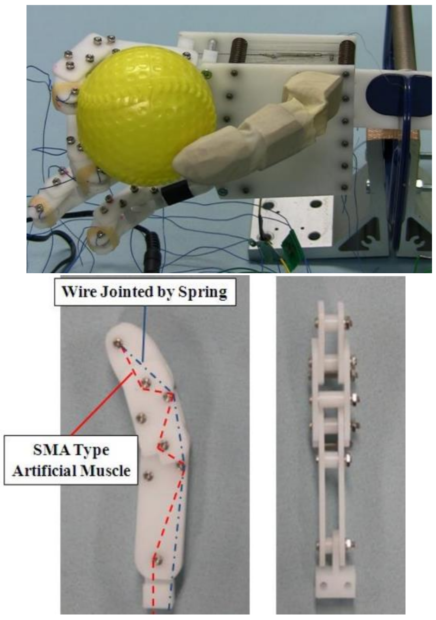


Figure 34. The prosthetic hand using shape memory alloy actuation (adopted from

reference [88])

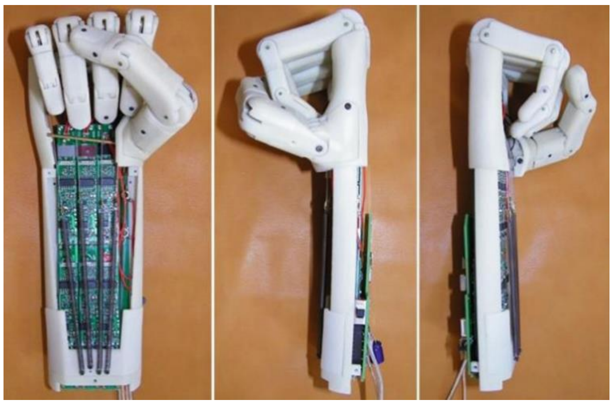


Figure 35. A SMA-driven robotic hand prototype (adopted from reference [89])

2.7 Control methods of anthropomorphic robotic hands

Human hands are capable of performing plenty of grasping and manipulation tasks, such as playing musical instruments, using chopsticks, and performing daily activities like cooking and writing. However, considering the dexterity, adaptivity and energy efficiency, none of the existing robotic hands can catch up with human hands. The reason lies in the unique biomechanical properties and the complex neuromuscular control strategies of the human hand. In fact, the biomechanics and the control strategies of the human hand are coupled together, i.e. the morphology of the bones, tendons and muscles can significantly influence the modes of the central nervous system (CNS) control [25]. It has been a long way for researchers in the control strategies exploration of the robotic hands. Though they still cannot catch up with the control level of the human hands, the development of artificial intelligence and machine learning has increasingly propelled the control of robotic hands forward. While the control strategies of robotic hands can be basically classified into the following categories.

2.7.1 Non-adaptive control

The non-adaptive control strategies can be executed through the way that the upper computer (PC) sends the control commands to the end effector. In this method, the researchers need to program in the upper computer, including the desired trajectories, postures, applied moments and joint stiffness with the error feedback from the end effector.

Position Control

The position control is to track the joints' angle variation mapping from the intended end-effector trajectory. As to the tendon-driven robotic hands, this control method is an important branch. In addition to this, there must be a forward mapping from the joints' angle variation to the tendons' elongation. Besides, there is an inevitable problem which is the actuation redundancy since one tendon's elongation can map with two or three joints' angles.

While the mapping function between the joint angles and the tendon lengths can be obtained by using Gaussian process (GP) regression and standard least squares regression (LSR) which are both kinds of machine learning technique. GPs are often used to find regression functions from sample data [90]. And it has been widely used in the field of robotics such as reinforcement learning [91] and Bayesian filtering [92]. The LSR method is normally utilized to seek the coefficients of a 3rd degree polynomial so as to minimize the L2-norm error.

The whole position control loop can be demonstrated as shown in Figure 36. The PID controller regards the error between the desired and current tendon length as the input variable, then outputs the motor force to drive the tendon to change its length and achieve the desired posture. Note that we can still utilize the motion capture system, such as VICON, to track the actual joint angles, which we can compare with the desired angles to verify the performance of the control method.

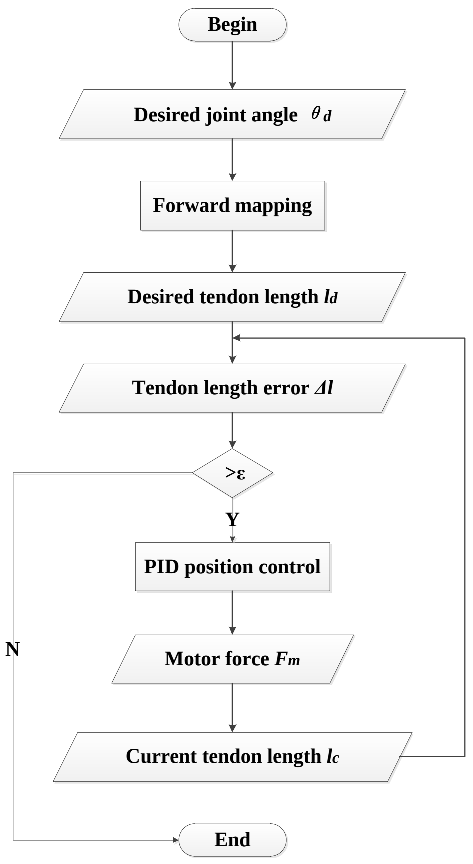


Figure 36. The position control loop of the tendon-driven robotic hand

While we know that the tendons in the human body can perform some passive behaviours owing to their unique material property and microstructure, contributing to joint stiffness. In order to mimic this passive property of the human tendons, the researchers also attempt to use some high strength fibre with a few elastic properties. As a result, sometimes we need to consider the force-length relationship of the tendon so that to make the control strategies more accurate. However, the stiffness of the tendon shows a kind of nonlinear property. The research in reference [75] provides an exponential relationship between the force and length of the tendon to demonstrate its passive behaviour. As a result, there is an additional feedback loop to the position control loop because of the passive behaviour, which is shown in Figure 37.

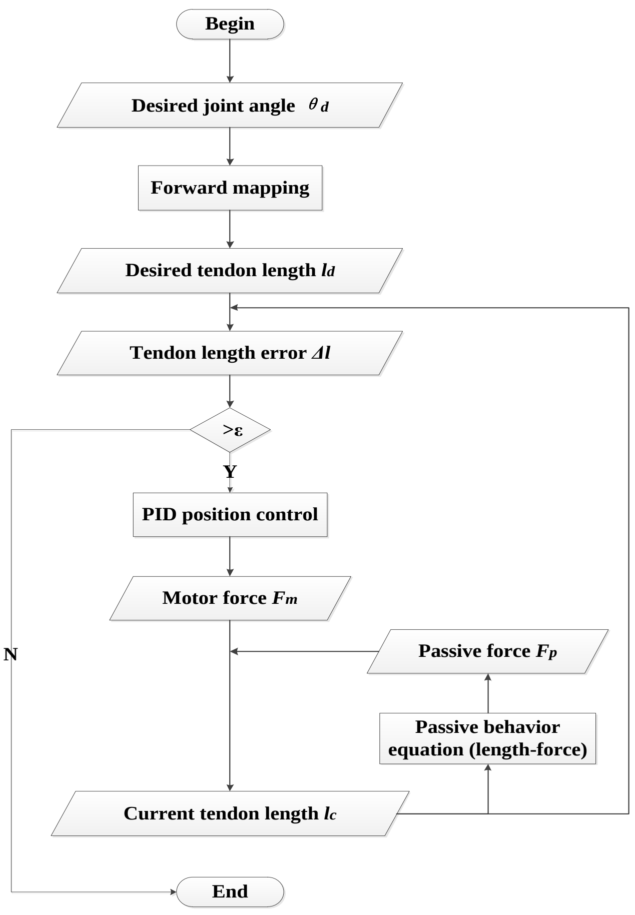


Figure 37. The position control loop of the tendon-driven robotic hand with passive behavior

Force Control

The force control is to obtain the desired fingertip force by controlling the motor’s output torque. Similarly, in terms of the tendon-driven robotic hands, there are some mapping relationships between the fingertips force and the tendons force which can be produced by the motors torque.

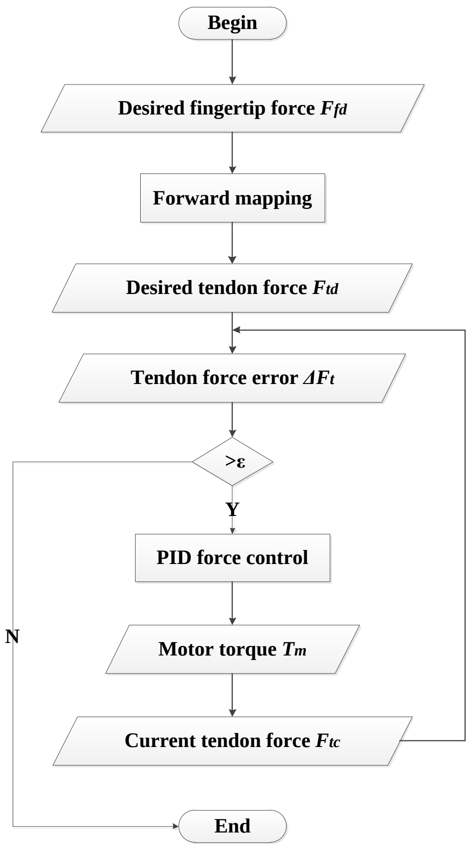


Figure 38. The force control loop of the tendon-driven robotic hand

So the force control loop (Figure 38) can be obtained easily through the analogy with the position control loop. In order to acquire the current tendon force F_{tc} as the feedback, a tensile sensor is often used. Likewise, we can also attach some tactile sensors to the surface of the fingertip to measure the actual fingertip force, which can be compared with the desired fingertip force so as to verify this force control method.

As we can see, though the two non-adaptive control methods have the advantage that there is less calculation in the control loop so that it can improve the control velocity, they still have some inevitable drawbacks, such as the error existing in the forward mappings (GPs and Least-Square mappings) which can affect the control precision, and the movement of the fingers may be not smooth. Therefore, the researchers have been exploring some other control methods with better control performance.

2.7.2 Adaptive control

As we know, human hands can manipulate objects easily and skillfully without knowing the detailed information about the objects such as the shape, kinematics or contact state. While the non-adaptive control mentioned in the previous section has assumed that the exact kinematics and Jacobian mapping from joint space to task space are pre-identified. However, in practice, this information is hardly obtained and precisely determined due to the limited knowledge about the object, the environment and the interaction process. For example, in order to successfully perform the grasping, we need to know the friction and weight of the object so as to exert proper forces which are sufficient for avoiding the slip and also not too large to damage the object. Besides, hand manipulation often leads to uncertain contact points with the object, making the control complex. Basing on this, the adaptive control comes up.

Unlike the non-adaptive control, the adaptive control can be executed only through the lower computer and the feedback from the end effector sensors. In this way, the researchers only need to load the program into the underlying driver control panel, rather than programming the control command in the upper computer. The adaptivity can be reflected in the motion self-adjustment of the end-effector with the help of tactile sensors, vision sensors and accelerometers.

Many researchers have made some progress in applying adaptive control to robotic hands. In 2000, Melchiorri used force/torque sensors to detect the translation or rotation slip motion and accordingly, to control the grasp force. However, the static friction between the robotic hand and the object needs to be known first [94]. In 2005, Ikeda *et al.* used a camera to measure the eccentricity degree of the contact region for the purpose of controlling the grasp force [95]. However, the integration of the cameras and the hand is a problem. In addition, the processing speed of the vision system can hardly catch up with the slip speed which will make the detection delay. In 2008, Taro

Takahashi *et al.* from Sony Corporation proposed an adaptive method for grasping objects with unknown properties. Based on the amount of the external force measured by the 6 axis force sensor and the tactile sensor, the three-fingered robotic hand can smoothly and quickly switch between force control and position control to perform the adaptive grasping [96]. In the same year, Daisuke Gunji *et al.* used a two-dimensional centre of pressure tactile sensors ('CoP sensors') which were thin, flexible and lightweight to detect the object slip of the robotic hand. By using the CoP sensors, the centre position of a distributed load and the total load can be measured within 1ms. The quick slip detection allows the controller to have sufficient time to process signals so that the object slipping can be prevented in time [97]. In 2012, Mike Stachowsky *et al.* proposed a tactile sensory array based grasp control strategy to control the grasping forces, making the robotic hand adaptive to the external disturbances [98].

In a word, the adaptive control is a method to make the actuator react to the complex uncertain environment and adjust the output force and motion, which is the main trend for the development of robotic hand control.

2.7.3 EMG/EEG control

The electromyography signal (EMG) and the electroencephalography signal (EEG) are often used to control the prosthesis to assist the amputees in recovering, especially the EMG signal. They just need to record the electrical activity of the muscles and brain by using some electrodes placed on the skin or the scalp and then classify these weak electric signals into different desired motions.

EMG Control

The premise of the EMG control is that there must be some relation between the EMG signals and the underlying muscle forces. However, the muscle force cannot be measured directly by using EMG. There has been some research claiming that the relationship between the muscle force and EMG is approximate to linear [99], but it is not always true. With the development of the EMG detecting and signal processing technologies [100], the EMG control is widely used in robotic hands and prostheses.

Basically, the process of the EMG control consists of data acquisition, data processing and data classification. Though the first two steps both play an important role in improving the control accuracy, the exploration of the data classification algorithms is the most interesting part and attracts many researchers to put enormous efforts into it.

There are several popular data classification algorithms for the EMG control, including binary control algorithm, variable control algorithm, fuzzy control algorithm, neural network control algorithm, etc.

The binary control algorithm means outputting ‘off’ or ‘on’ signal to the pneumatic valves, where the value is decided by the EMG signal. For example, a specified threshold value is predetermined. The EMG signal which is below this value outputs ‘off’, and which is above this value outputs ‘on’ [101]. It is easy but the deficiency is also obvious that is the finger cannot be flexed halfway or generate less force.

The variable control algorithm uses a simple proportional controller with the filtered EMG signal to realize the continuous torque control. For example, 15% and 70% of the maximum muscle contraction level can be respectively set as the minimum and maximum motor torque. And the value can be adjusted according to personal preference [102].

The fuzzy control algorithm classifies the EMG signals into more detailed levels. A fuzzy controller can be described as shown in Figure 39.

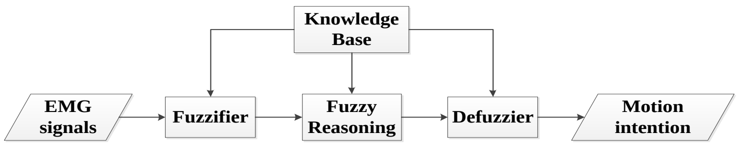


Figure 39. Basic structure of a fuzzy controller

The fuzzifier is to turn the precise voltage values of the EMG signals into the fuzzy values (membership values) and demonstrate them using corresponding fuzzy sets, which is shown in Figure 40. MBV is the membership values, and ZE means zero, PS means positive small, PM means positive middle, PB means positive big, PVB means positive very big [103] [104]. We can see that, it will account for 0.4PVB and 0.6PB when the EMG signal voltage produced by a certain muscle is 1.4V. According to this, we can put the results into the fuzzy reasoning step, which is the core of the fuzzy controller. There are some fuzzy rules basing on the fuzzy logic, often appearing in the form of 'if, then' statement. After that, we can defuzzier the outputs from the fuzzy reasoning to get the actual control signal which can produce the precise intended motion. Generally, the maximum membership principle and the weighted average method can be used in the defuzzier process.

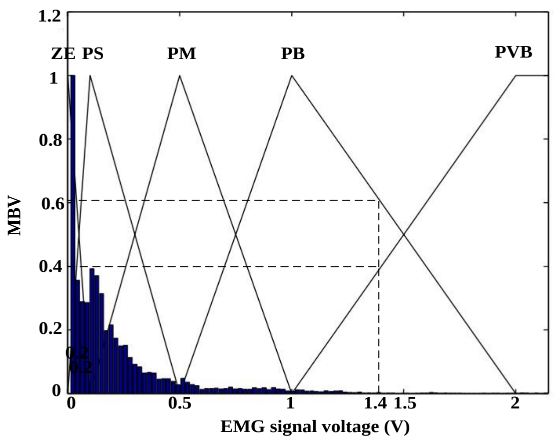


Figure 40. Membership values generated by the data distribution (adopted from reference [104])

The artificial neural network (ANN) control algorithm has certain performance characteristics in common with biological neural networks. It is often used as a mathematical tool to model the human cognition and neural system. The net structure of ANN includes many neurons which are responsible for processing information, and some connection links with associated weight which transmit the signals between the neurons [105] [106]. A simple artificial neuron is shown in Figure 41, in which, X_1, X_2 ...... X_n are the inputs, ω_1, ω_2 ...... ω_n are the input weights and b is the bias.

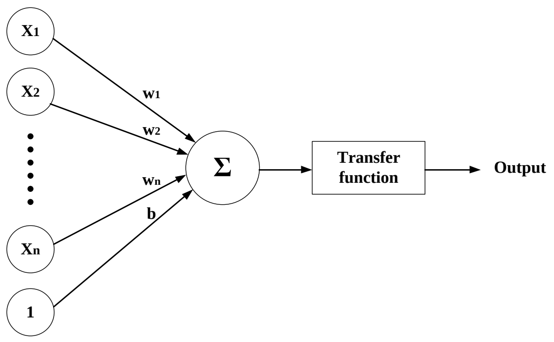


Figure 41. The schematic diagram of a single artificial neuron (adopted from reference [106])

While the neurons in ANN can process information by dynamically responding to the external inputs [107]. Normally there are three layers in ANN, which are the input layer, hidden layer and output layer. The basic ANN feed-forward model is shown in Figure 42. The number of hidden neurons must be carefully determined so that the overfitting and insufficient learning can both be prevented [108].

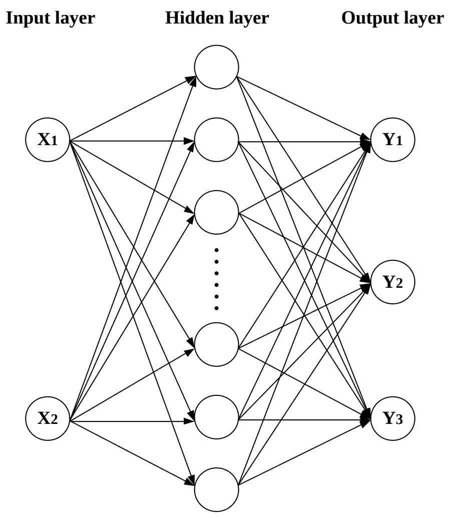


Figure 42. The schematic diagram of an ANN model (modified from reference [108])

In terms of EMG control, the input is the EMG signal and the output is the control signal of a certain intended motion. The complex relationships between the EMG signals and the intended motion or the intended grasp force can be left to the ANN hidden layer to solve [109].

In general, the ANN control algorithm is just one branch of the machine learning algorithms. Its purpose is to find out the relationship between the input and the output more precisely through learning multiple groups of actual data given by the researchers. As a result, the ANN can be widely used in many other control methods. For example, it can reduce the position error by optimizing the mapping function between the joints' angle and the tendons' length in the position control. Also, it can reduce the error of the contact force by learning the highly nonlinear relationship between the fingertip force and the joint torque. What's more, it can be used to build the model of a nonlinear relationship between the input voltage of the EAP (Electro-Active Polymers) and the output deformation [110] [111] [112] [113] [114].

EEG Control

EEG has not been used as a kind of control signal until recent years, benefiting from the development of the brain-computer interfaces (BCIs). Both the brain signals from the scalp (EEG) and the cortical surface (ECoG) can be used by BCIs. It should be noted that sufficient activity-dependent trains are still needed when using them BCIs since the stable relationships between an individual's intent and the EEG signals need to be established, just like the training in EMG control [115].

The application of EEG control still seems limited, though great progress in EEG has been made. For instance, most of the experiments stay in cursor movement [130] and spelling [117]. And the scenarios as well as the demonstration purposes are also restricted [118] [119]. In 2002, Mehrnaz Kh. Hazrati and Abbas Erfanian from Iran University of Science and Technology controlled the prosthetic hand to grasp by using single-channel single-trial EEG signals [120]. And in 2009, a new BCI system was developed by them which can control the hand to grasp and open in the virtual reality environment by online classifying the EEG signals [121]. In recent years, many robotic systems adopted the EEG control method [122] [123]. In 2015, some scientists at the Defense Advanced Research Projects Agency (DARPA) developed a prosthetic hand that could ‘feel’ things and could also be controlled by human thoughts. This is the first brain-controlled prosthetic hand in the world [124]. In 2017, an EEG-based BMI platform was developed by A. Sarasola-Sanz *et al.* to realize the control of a multi-DOF exoskeleton [125].

The process of the EEG control is similar to the EMG control, except for the data acquisition system (BCIs for EEG). However, there are still some challenges existing in applying BCI and EEG in the real world. For example, it is difficult to control the hand grasping and holding sequence by using this method. So there is still a long way for the exploration of the EEG control.

2.7.4 Data glove control

There are amounts of situations and environments that are dangerous for people going there to complete some tasks. And this needs some methods to teleoperate the robots or machines. Data glove technology is one of the popular implementation methods, especially for biomimetic robotic hands teleoperation control. When a human user wears a data glove, the sensors of the glove will detect the position of each finger. Some data gloves have force sensors on their tips so as to measure the fingertip force of the human hand.

There are many types of commercially available or research data gloves. They have different degrees of freedom and sensing methods. To record enough human hand motion data for the control, most of the data gloves have more than 10 degrees of freedom. The most widely used data glove should be the VMG 30 data glove from Virtual Motion Labs [156]. It uses bending sensors together with 6 DoF IMU to measure 28 degrees of freedom of the human hand. Other popular data gloves often use resistive bending sensors, such as the SIGMA (30 DoF measured) from Sheffield University [157], the CyberGlove data glove (18 or 22 DoF measured) [158], the CyberGlove 2 (18 or 22 DoF measured) [159] and the CyberGlove 3 (18 or 22 DoF measured) [160], from Cyberglove Systems, LLC, the SuperGlove (10 DoF measured) from Nissho Electronics [161], etc. Certainly, there are many other sensing methods that are also largely used in the data glove, such as fibre optics (5DT Data Glove 14 Ultra from Fifth Dimension Technologies Inc. [162]), magnetic (3D Imaging Data Glove from the University of East Anglia and the University of Newcastle upon Tyne [163]), Hall effect (Humanglove from Humanwave [164]), etc. Figure 43 shows some of the aforementioned data gloves. No matter which sensing technologies the data gloves use, the aim is to accurately measure the human hand posture data with as high as possible sampling frequency.

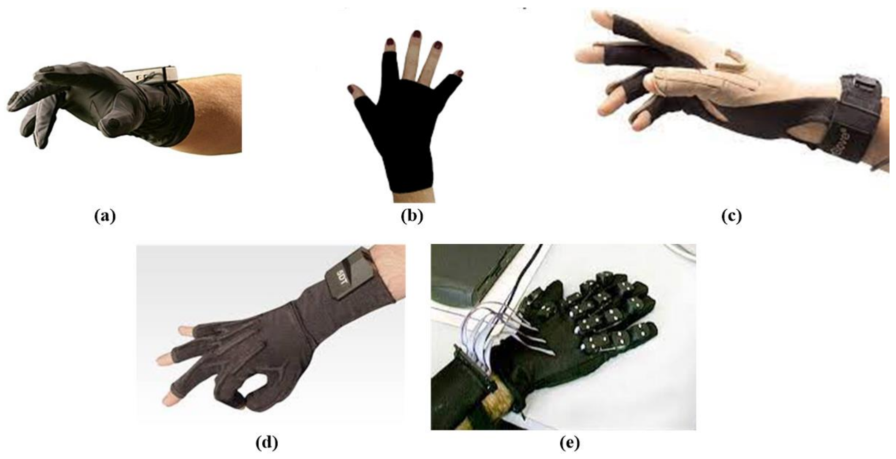


Figure 43. Data gloves from different companies. (a) VMG 30 data glove. (b) CyberGlove data glove. (c) CyberGlove 2. (d) 5DT Data Glove 14 Ultra. (e) Humanglove. (adopted from the references [156] [158] [159] [162] [164])

Many researchers choose to use the data glove to control their developed robotic hands or prostheses considering that the data glove can quickly and comprehensively track the human hand motions and record the posture data. In this way, by using these data, the robotic hands or prostheses are expected to approximately repeat the corresponding motions when the human hand grasping or manipulating objects. Normally, for the tendon driven robotic hands, the relationship between the tendon excursion and the joint angle should be identified through modelling. Since the data glove can only provide the information of joint angles, of which the data cannot be directly used to control the motors. The relative research can be seen from the reference [165] and [166]. Besides, there are some customized data gloves that can directly obtain the tendon excursion data for the tendon driven robotic hand control, for example in Zhe Xu's highly biomimetic robotic hand research [18]. Some popular robotic hands also used the data glove control method to make the grasping and manipulation demonstrations, such as the Shadow Hand [53], DLR/HIT Hand II [51], etc. Using the data glove, the Shadow hand realized many complex manipulations such as screwing off the bottle cap, opening the box and even playing the Rubik's cube. The DLR/HIT Hand II can also grasp some objects through the data glove control. The function realization of these two robotic hands relies on the data glove real-time control with the underlying help of the visual feedback from the human users. In this way, the robotic hand's postures can be relatively precisely controlled. Actually, by using the human hand data obtained from the data glove, it is possible to make the robotic hand perform some grasping and manipulation functions in an offline way. And this offline method is also what I want to try in my research since one of the main objectives is to explore how the bio-inspired structures can help the robotic hand realize functions without complex control and sensory feedback.

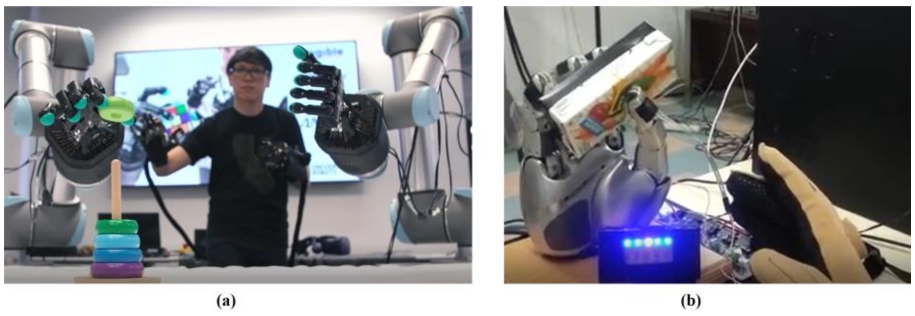


Figure 44. Robotic hands controlled by the data glove. (a) Shadow hand manipulation controlled by the data glove. (b) DLR/HIT Hand II grasping controlled by the data glove (adopted from the references [186] [187])

2.7.5 Ultrasound control

As a choice of the human-machine interface, the ultrasound imaging (US) technology is being increasingly used to control the prostheses. At early stages, the US was mostly used as a medical technique for safely inspecting the anatomical disease. Through an ultrasonic transducer or probe, the images of the muscles and tendons can be obtained and their movements can be tracked. Figure 45 shows the ultrasound images of some forearm muscles. [167] The combination of different muscles’ displacements will generate different hand motions and postures. And based on this information, the prostheses or robotic hands can be controlled by the US to realize the functions that the human users intend to. Compared with the EMG device, the US device needs to contact with the arm in a much smaller area. Besides, unlike the EMG, the US can visualize the movement of both the superficial and deep muscles. [168] And recently, the US technology has been gradually improved and shows very high accuracy on hand posture detection. [169]

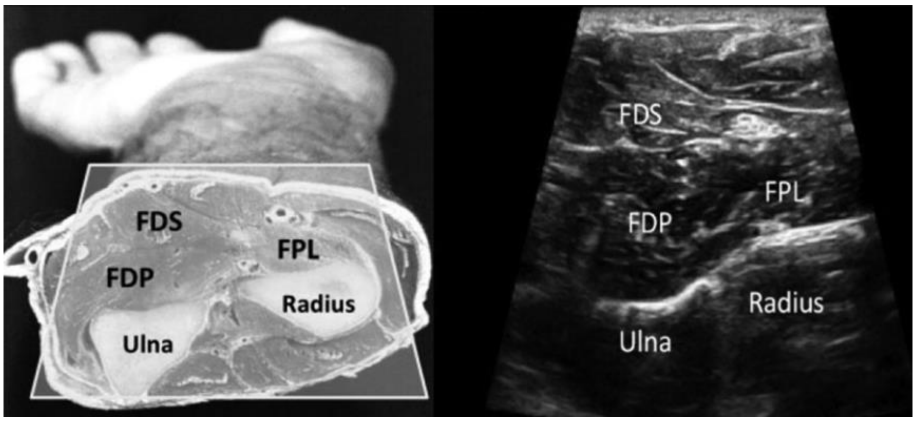


Figure 45. The ultrasound image of the forearm muscles (adopted from the reference [167])

One of the main challenges for the US is to precisely classify the hand motion modes according to different muscle synergistic activities. Ortenzi *et al.* [170] used the US to classify six hand postures and four functional grasps with three levels of force performed by the subjects. McIntosh *et al.* [171] developed a US-based hand gesture recognition algorithm that can classify 10 different hand gestures with over 98% accuracy. In the research of Akhlaghi *et al.* [172], the real-time image-based classification of a virtual hand's activities was proposed and showed around 92% accuracy on average. And the novel ultrasound imaging-based control strategy proposed by Sikdar *et al.* [173] demonstrated the individual finger movement classification with 98% accuracy, which can significantly improve the prosthesis control. These research projects enhanced the US technology and paved the way for the artificial hand control. Thus many researchers tried to use the US technology to accurately control the prostheses to realize more daily activities. Hettiarachchi *et al.* [174] developed a new wearable ultrasound muscle activity sensing system for controlling a dexterous prosthesis hand, realizing five hand gestures. And the researchers from the Georgia Institute of Technology developed a prosthesis called 'Luke Skywalker's bionic hand' which is powered by ultrasound signals. Users can even play the piano by controlling this prosthetic hand, as shown in Figure 46. [175]

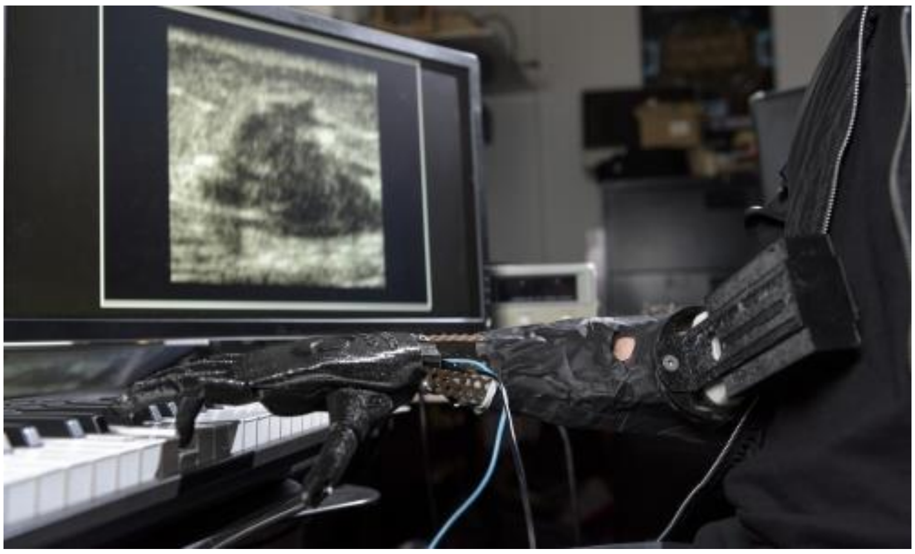


Figure 46. Ultrasound-controlled ‘Luke Skywalker’s bionic hand’ from Georgia Institute of Technology (adopted from the reference [175])

It’s true that the US is a good potential choice for the robotic hand control. However, it can only detect the movement of the forearm muscles. As to the degrees of freedom generated by the intrinsic muscles in the hand body, the US technology is powerless.

2.8 Grasping and manipulation taxonomy

To demonstrate the functional performance of the robotic hands, many researchers chose to test the grasping and manipulation capabilities of their designed robotic hands. And to explore the full potentials of the robotic hands, we need a comprehensive taxonomy of grasping or manipulation behaviours. Thus several grasping and manipulation taxonomies were proposed by researchers according to different hand postures, object states or the fingers involved. And some of them have already been widely used in the fields of robotics, prosthetics and rehabilitation.

For the grasping taxonomy, the idea of precision and power grasp was firstly identified by Napier [176]. And the taxonomies afterwards were mostly influenced by this classification. In 1989, Cutkoskey [153] constructed a tree-like taxonomy (Figure 47 (a)) including 16 grasps according to the adaptability required by small-batch tasks, i.e. power and precision, non-prehensile and prehensile, prismatic and circular, etc. And this is also one of the most prevalent taxonomies used in the robotic grasping function evaluation. Another more comprehensive grasping taxonomy was proposed by Feix *et al.* [154] in 2016, as shown in Figure 47 (b). They organized and compared the existing taxonomies in the literature and rearranged a large number of different grasps into a new taxonomy including 33 grasp types. This matrix-like taxonomy classified the grasps into two rows of thumb adducted and thumb abducted, and three main columns including power, intermediate and precision grasp. More recently, Stival *et al.* [177] firstly proposed a quantitative taxonomy of hand grasps based on muscular and kinematic data of the human hand. It consists of five groups of grasps, i.e. flat grasps, distal grasps, cylindrical grasps, spherical grasps, ring grasps, covering 20 grasp types presented in their research. Compared with the previous taxonomies, this one clarified some quantitative parameters to define different grasp types. In 2021, Mehrkish and Janabi-Sharifi [178] proposed a comprehensive grasp taxonomy particularly for continuum robots which was based on the grasp function and grasp closure, including 9 main grasp types. Though it was mainly designed for the soft robots, it can still inspire researchers to build a more comprehensive taxonomy for exploring and evaluating the grasping capability of robots.

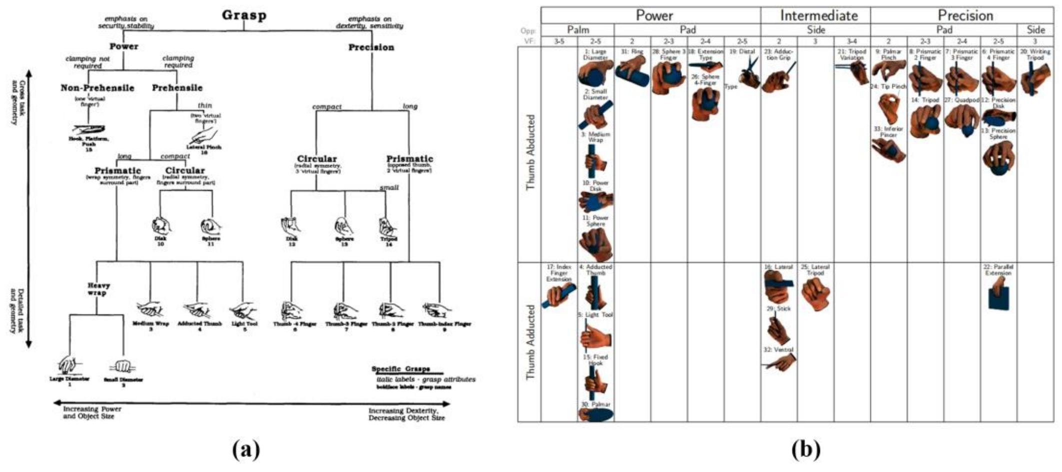
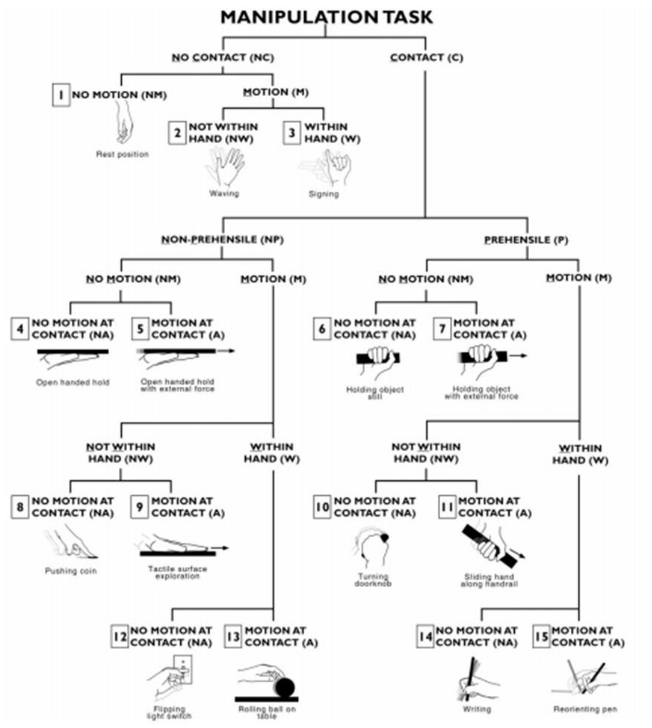
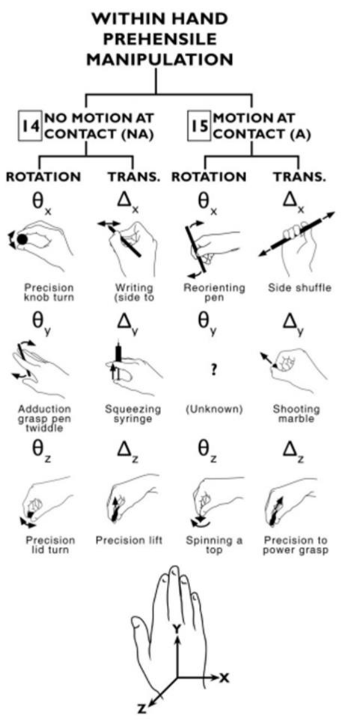


Figure 47. Two widely-used grasp taxonomies. (a) Cutkoskey grasp taxonomy. (b) Feix grasp taxonomy. (adopted from the reference [153] [154])

Different from grasping, manipulation involves the movements of hand fingers and objects. Though some classifications cannot cover all the manipulation types, they are still instructive for other researchers. Elliott and Connolly's [179] proposed an in-hand manipulation classification including three types, which are simple synergies, reciprocal synergies, and sequential patterns. They used anatomical directions to define a hand coordinate system for analysing various manipulation tasks. In 1992, for clinical application, Exner [180] classified the manipulations tasks into five types, i.e. palm-to-finger translation, finger-to-palm translation, shift, simple rotation, and complex rotation. It’s a kind of object-centric classification and not very compatible with other research. Developed from the previous grasp taxonomies and manipulation classifications, Bullock and Dollar [181] firstly systematically proposed a hand-centric and motion-centric manipulation taxonomy to identify the manipulation strategy. This tree-like taxonomy (Figure 48 (a)) presented 15 manipulation types. Specifically, based on the hand coordinate system and object motions, they classified the within hand prehensile manipulation tasks into 12 types, covering most of daily activities, shown in Figure 48 (b). However, this taxonomy hardly considered the hand configurations and the interactions with objects. For the object-centric manipulation research, Calli *et al.* [182] designed a real-life objects dataset called Yale-CMU-Berkeley Object Set, including 77 kinds of objects. It provides another approach to test the manipulation capability of robots. There are also some other manipulation taxonomies gradually being proposed mainly for some specific tasks, such as the taxonomy from the research of Paulius *et al.* [183] which is for cooking. I would say there is still no one comprehensive manipulation taxonomy being widely recognized and used for the robotic hand manipulation research.





(a) **(b)**

Figure 48. Bullock and Dollar manipulation taxonomy. (a) 15 types of manipulation tasks. (b) 12 types of within hand prehensile manipulation tasks. (adopted from the reference [181])

2.9 Comparison of different robotic hands

To clearly compare the key performance indexes of different robotic hands, I selected some related information from the literature, especially for this research project. Three types of robotic hands were investigated, i.e. the traditional robotic hands, the novel soft robotic hands and the highly biomimetic robotic hands. And some typical robotic hands of each type were chosen to present here. For this research, 10 aspects related to the design, actuation, control, mechanical and functional performance of the robotic hands were studied and summarized, as shown in Table 3. Fully understanding the previous robotic hands would inspire this project and provide a baseline to develop and evaluate this research.

Table 3. Some comparison results of different robotic hands

	Robotic hands	Actuators	Transmission system	Tendon morphology	Joint types	Joint passive stiffness
Traditional robotic hand	Southampton Hand [48]	DC Motors	Leadscrew + Interphalangeal linkages + Whiffle tree mechanism	No tendon	Hinge joint	/
	Gifu hand III [49]	Servo motors	Joint direct + Four bar linkage	No tendon	Hinge joint	/
	Cyber hand [50]	DC Motors	Tendons + Lead screw + Round wire spirals pulleys	Linear flexor + No extensor	Hinge joint	/
	DLR/HIT Hand II [51]	Super flat BLDC motors	Timing belts + Gears	No tendon	Hinge joint	/
	Robonaut hand II [52]	Brushless DC motors	Tendons + Four-bar linkage + Rigid tendon tunnels	Linear flexor + Linear extensor	Hinge joint	/
	Shadow Hand [53]	Smart Motors (PWM Maxon motor) or Air muscles	Tendons + Rigid tendon tunnels	Linear flexor + Linear extensor	Hinge joint	/
Soft robotic hand	RBO Hand 2 [54]	PneuFlex actuators (Pneumatic)	Contained chambers	No tendon	Elastic jonit (continuous)	Very low
	Festo bionichand [184]	Pneumanic airflow plate	Air chambers	No tendon	Elastic jonit (continuous)	Very low
	UB Hand 3 [55]	DC Motors	Tendons + Endoskeletal structures with path (Rigid)	Linear flexor + No extensor	Elastic joint	Low
	iRobot-Harvard-Yale (iHY) SDM hand [76]	DC Motors	Tendons + Hollow cable raceway with nylon tubes (Rigid)	Linear flexor + No extensor	Elastic joint + Hinge joint	Low and variable distal joint stiffness
	Open Bionics Hand [56]	Servo motors	Tendons + Low-friction tubes (Rigid)	Linear flexor + No extensor	Elastic joint + Hinge joint	Low
	IIT SoftHand [185]	DC Motors	Tendons + Passive anti-derailment pulleys (Rigid)	Linear flexor + No extensor	Rolling–contact joint + Hinge joint	Low
Highly biomimetic robotic hand	ACT hand [75]	DC Motors	Tendons + Guiding 'rockers' (Rigid)	Linear flexor + Net extensor	Hinge joint	/
	Robotic hand from University of Washington[18]	Servo motors	Tendons + Laser-cut rubber sheets (Flexible)	Linear flexor + Net extensor	Biomimetic joint (linear fishing line ligaments)	Potentially variable
	Cambridge Biologically inspired soft robotic hand [19]	DC Motors	Tendons + PTFE tubes with hot melt adhesive (Rigid)	Linear flexor + Linear extensor	Biomimetic joint (cutted rubber sheet ligaments and capsule)	Potentially variable
	HR-hand [20]	McKibben muscles	Tendons + Polyethylene tubes (Rigid)	Linear flexor + Net extensor	Biomimetic joint (cutted silicone ligaments)	Potentially variable
	3D-printed anthropomorphic soft skeleton hand [21]	No actuation	No transmission	No tendon	Biomimetic joint (3D print capsule)	Anisotropic/low

	Robotic hands	Maximum force performance	Maximum velocity performance	Sensors	Control	Functions
Traditional robotic hand	Southampton Hand [48]	A pinch force in excess of 45 N	Fully close in less than 1.2 s	Potentiometers on the joints	/	Selective grips
	Gifu hand III [49]	Fingertip: 3.4 N; Thumb: 3.7 N	Joint maximal speed: 7.4 Hz	Joint encoders + A 6-axes force sensor at each fingertip + A distributed tactile sensor with 859 detecting points attached to the surface of the fingers and palm.	Real-time operating system with position, velocity and force control	Selective grasps (sphere, cylinder and prism)
	Cyber hand [50]	Power grasp: About 70 N; Two-finger force: <5 N	Joint maximal speed: 45 %s	Hall effect joint angle sensors + Incremental magnetic encoders for motor position + Strain gauges cable-tension sensors; A flexible layer with contact sensors + Fingertip triaxial force sensors + Fingertip soft and compliant tactile microsensor (SCTM)system	Low level PWM and PID position control + High level feedback force control	Power grasp and low-load precision grasp (lateral pinch and cylindrical, spherical, and tripod grasp)
	DLR/HIT Hand II [51]	10 N fingertip force	/	Strain gauge joint torque sensors + Potentiometers and contactless magnetic joint angle sensors + 6 DOFs fingertip force sensors	5 levels based position control, impedance control and demo moving program (data glove control)	Selective grasps (a ball); Rough manipulations (manipulating a rectangle)
	Robonaut hand II [52]	Tip force: 2.25 kg; Payload: More than 9 kg	Fully extended and a tip speed of more than 200 mm/sec	A hall-effect sensor for absolute joint position sensing + A 6D force torque sensor for tactile load sensing + Tendon tension sensors	Joint-space torque and stiffness control with an inner position control loop	15 of the 16 Cutkosky grasps; Simple manipulations (using a drill and a portable x-ray device)
	Shadow Hand [53]	Power-grasp: Up to 5 kg	Full-range joint movement: 1.0 Hz for the Motor Hand and 0.2 Hz for the Air Muscle Hand	Hall-effect joint position sensors + Fingertip pressure sensor tactiles + Tendon force sensors for motor acutation / Solid-state pressure sensors for air muscle actuation	Joint position control + Force/pressure control, and can be integrated with a 22 sensor CyberGlove	Selective grasps (banana, apple, bulb, toy, brush, etc.); Some complex manipulations (screwing off the bottle cap, opening the box, playing the Rubik's cube，unzipping a bag, massage, doing the experiments-syringe, rotating a sphere and using a wrench)
Soft robotic hand	RBO Hand 2 [54]	0.54 kg cylinder grasping; 8 N tolerated disturbance forces	/	/	/	31/33 Feix grasps; Dynamic grasps; Elementary manipulations (clicking a pen, flipping a pen upside down, waving objects while grapsing, rotating a sphere while holding it, squeezing an orange and rolling object within the hand)
	Festo bionichand [184]	Up to 4 kg load capacity	/	Tactile force sensors + Inertial sensors for fingers and hand position + Pressure sensors for pneumatic actuation	Position and force control	Selective grasps; Simple manipulations (rotating a cube)
	UB Hand 3 [55]	6.8 N fingertip force	0.36 s fully close	Strain-gauge based joint angle and tendon tension sensors	Position and simple impedance control	Selective grasps (a bottle and a cylindrical box); Rough manipulations (fingertip manipulation of a pen)
	iRobot-Harvard-Yale (iHY) SDM hand [76]	15 N grasp force (recorded); Picking up a 22 kg weight in a cylindrical power grasp	/	Joint angle sensing (a magnetic encoder, optical flexure bending sensor and an accelerometer); Tactile sensing (an array of MEMS pressure sensors)	Open-loop and feed-forward control	Selective grasps (spherical/cylindrical power grasp, cylindrical/opposed pinch, pinch grasp with fingernails); Elementary manipulations (regrasping--opposed pinch to spherical pinch the battery, flip and pinch the key, operating a trigger while holding an item in cylindrical power grasp)
	Open Bionics Hand [56]	Less than 10 N fingertip force	/	/	/	Selective grasps (10 objectects, i.e. a mug, a soap, a magazine, a marker, a pair of sunglasses, a large rectangular box, a glass cleaner spray, a 1.5L bottle of water, a glass of water and a spoon)
	IIT SoftHand [185]	Holding force: About 20 N	/	Magnetic motor position sensors	/	Selective grasps (a total of 107 objects of different shapes); Simple manipulations (sliding and pivoting a book, rotating a test disc(+version), pouring coffee from a cup (+version))
Highly biomimetic robotic hand	ACT hand [75]	/	MCP joint abduction-adduction (0.5 rad/s); MCP joint flexion-extension (1.0 rad/s); PIP joint flexion-extension (1.0 rad/s), and DIP joint flexion-extension (1.0 rad/s)	Optical encoders of DC motors and Vicon motion capture system	Direct muscle-control (Grasping demonstration); Force-optimized joint control and motion control with passive behavior (Index finger testing)	Selective grasps (nine objects, namely, a coffee plate, key, credit card, hand towel, glasses, water bottle, cordless phone, medicine bottle, and spoon); Simple manipulation (sliding the switch, rotating the knob)
	Robotic hand from University of Washington[18]	/	/	3-axis fingertip sensors	Data glove direct tendon control	Selective grasps (31 objects, most of the Cutkoskey taxonomy); Simple manipulations (regrasping a whiteboard eraser)
	Cambridge Biologically inspired soft robotic hand [19]	Thumb tip force: Abduction 2 N; Adduction 4 N; Flexion 1.8 N	/	Vision motion capture setup	Position control	Using chopsticks
	HR-hand [20]	/	/	/	/	Selective grasps (a thread reel and an egg)
	3D-printed anthropomorphic soft skeleton hand [21]	/	/	/	/	Playing the piano

2.10 Summary

In this chapter, I analyzed the anatomical structure and biomechanics properties of the human hand, and then made some detailed investigation about the joint designs, drive methods, control strategies of robotic hands and some existing grasping and manipulation taxonomies, providing many inspirations for the research.

The design of the biomimetic robotic hand is supposed to originate from the human hand, no matter from the structure or the biomechanics properties. From the investigation, we can see that the robotic hand which adopts the biomimetic joint possesses more human-level dexterity. Therefore, this research will explore this kind of highly biomimetic design. Besides, the reticular extensor mechanism and the flexible tendon sheaths can also bring some interesting biomechanical properties as the literatures shown. Chapter 3 and 6 will show the process of the design and development of the highly biomimetic robotic hand, which is inspired by the anatomy structure of the human hand. Chapter 4 and 5 will mainly discuss the three biomechanical advantages embodied in the ligamentous joint, the reticular extensor mechanism and the flexible tendon sheath structures.

Each of the drive and control methods of the robotic hands has its own advantages according to the investigation. The motor drive is an energy-conserved and high-efficiency method. The pneumatic cylinder drive can provide more power and a smooth drive. The SMA drive is fit for the environment where light weight and low noise are needed. As to the control strategies, the non-adaptive control and adaptive control are introduced. And there are some popular implementation ways of robotic hand control, such as the EEG/EMG control, the data glove control and the ultrasound control. Chapter 7 will introduce the whole actuation and control system of the robotic hand. The motor-tendon actuation system will be constructed and the data glove-based control will be adopted.

The grasping and manipulation taxonomies investigated in this chapter give some efficient ways for evaluating the robotic hand functional performance. As discussed in Chapter 8, the grasping and manipulation capabilities of the robotic hand should be well demonstrated so as to provide strong evidence for the success of the research project.

Through the comparison of different robotic hands, we can clearly see the development of the robotic hand research. From the traditional rigid robotic hands to the novel soft robotic hands and then to the recent highly biomimetic robotic hands, each choice of the structure design, actuation and transmission systems and control strategies could bring some specific performance. Moreover, the functions realized by the previous and current robotic hands give us a baseline for evaluating the proposed robotic hand in this research.

Chapter 3: Bio-inspired design and fabrication of the robotic finger

This chapter mainly introduces the structure design and fabrication process of the robotic finger in detail. According to the anatomical structure of the human hand, three structural layers were abstracted and adopted on the robotic finger design with rapid prototyping and molding technology and unconventional soft materials. The aim of this chapter is to develop a highly biomimetic robotic finger with human-finger-like structure for the future robotic hand design.

The notion of the robotic finger design was derived from the anatomical structure and biomechanics of the human fingers. As a matter of fact, many unique advantages can be found in human fingers which extensively benefit from the soft tissues and special functional structures. Inspiring by this, we developed a multi-layered anthropomorphic robotic finger so as to reconstruct the structures of the human finger to embody the human-finger-like advantages.

Figure 49 shows a cadaveric human finger structure and the multi-layered design of an anthropomorphic robotic index finger. Referring to the figure, it can be seen that except for the skin, there are mainly six structural components, including the phalanges, the articular cartilage, the ligaments, the joint capsule, the tendons, and the tendon sheaths. These six main components in the finger have their individual features and functions which are addressed in detail in this section. Here, according to the property, we present the finger structure in three layers.

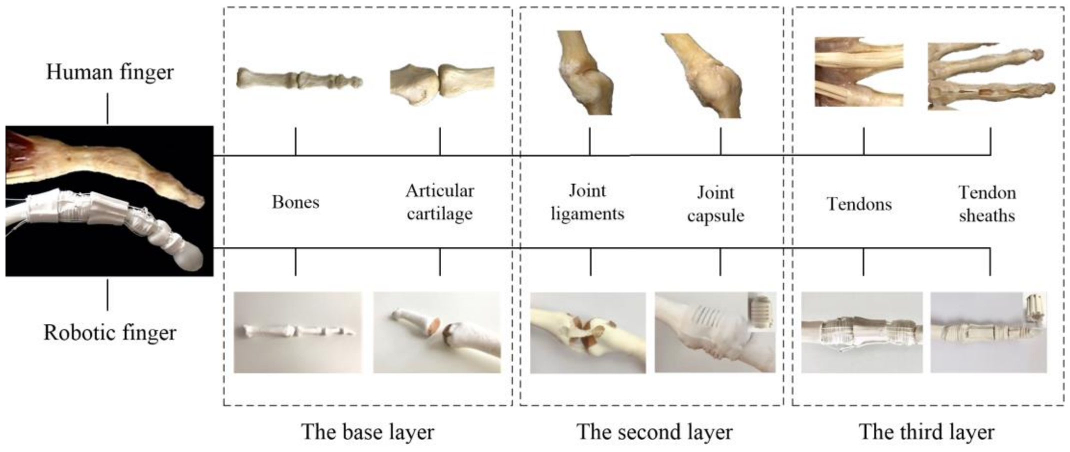


Figure 49. The multi-layered structure of a human and robotic finger.

3.1 The Base Layer — Phalanges and articular cartilage

The phalanges and the articular cartilage form the innermost part of a finger, in which the phalanges are the only rigid components in the whole finger. Three phalanges and one metacarpal bone make up three finger joints, namely, the distal interphalangeal (DIP) joint, the proximal interphalangeal (PIP) joint and the metacarpophalangeal (MCP) joint. These phalanges need to be strong and stiff enough since they play an essential role in bearing external loads. Besides, the palmar surface of each phalanx shaft is slightly longitudinally concave to accommodate the muscles or tendons without more space needed [126]. The palmar convex surface of the phalanx head can push the flexor tendons out to provide an initial flexion moment arm, avoiding actuation difficulty at the initial motion stage. Moreover, the unique shape of the biological joint articulation would exert a significant influence on the joint motion. For instance, the connections in the DIP, PIP and MCP joints are quite similar, but the surface shape of the MCP joint brings it another two degrees of freedom which provide the abduction-adduction motion and the pronation-supination motion, comparing only the flexion-extension motion existing in the DIP and PIP joints. Therefore, preserving the intact shape of the phalanges and ensuring the essential material strength are the major considerations in the design of the proposed robotic finger.

In the proposed robotic finger, the bone models come from the CT scanning data of a 23-year-old healthy male [127], the data from CT scanning was processed in the software Mimics Research 19.0® (Figure 50 (a)). After that, the computer-aided design (CAD) model (Figure 50 (c)) of the bones was generated and subsequently imported into the software SolidWorks®, where the fixture holes of the ligaments' origins and insertions were located basing on the MR images (Figure 50 (b)) and the cadaver data from previous research [6]. During this process, the intact geometry of bones was largely preserved. The modified bone models were then 3D printed with the 405nm UV white photopolymer resin by using the Anycubic Photon® S LCD-based SLA 3D printer and the printed physical model was shown in Figure 50 (d). We found that using the resin to print the model can meet the needs of form accuracy and surface flatness.

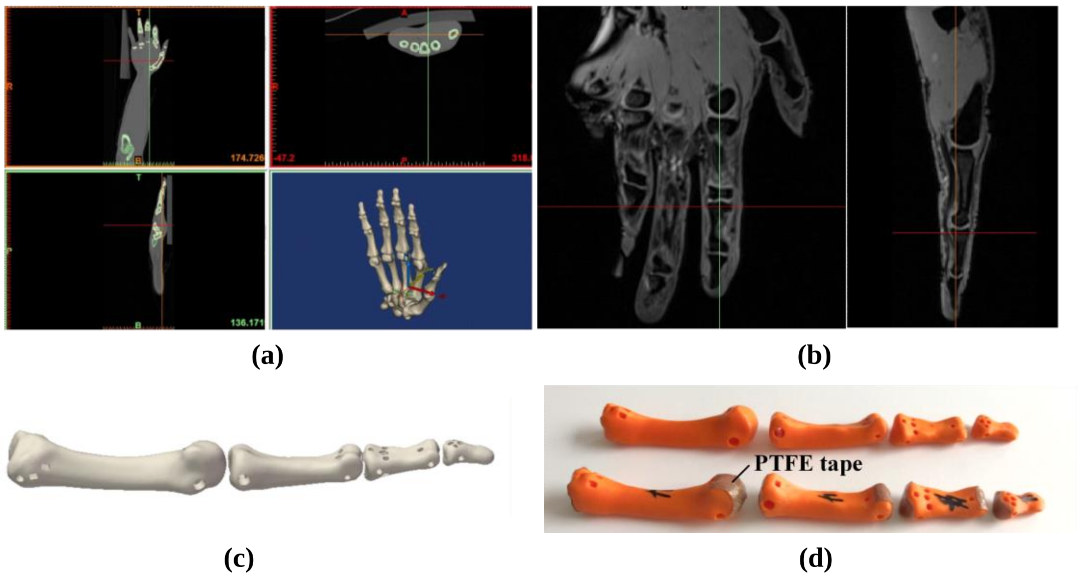


Figure 50. Fabrication of robotic finger bones (a) CT images of human hand bones. (b) MR images of human hand soft tissues. (c) CAD model of human finger bones. (d) 3D printed physical model of robotic finger bones with PTFE articular cartilage.

Covering the finger joint surface is a thin, dense, smooth, connective tissue, namely articular cartilage, which is resilient and displays viscoelastic properties, minimizing the friction and wear in the relative movement of adjacent joint surfaces to enhance the transmission efficiency of loads. Its unique compositions and morphologies allow it withstanding high cyclic loads nearly without failure and distributing the loads over the whole joint surface to reduce the stresses sustained by joints' contact [128] [129]. The water, as the most abundant component in the articular cartilage, occupies nearly 80% of the articular surface, controlling the joint lubrication. And the collagen fibrils and the proteoglycans make up the structural networks to support the internal mechanical stresses [130]. Thus, the articular cartilage can be simply regarded as a two-layer structure with around 0.5 mm thickness, including a lubrication layer and a structural base layer [131].

In the cartilage design of the proposed finger, the PVA hydrogel and the PTFE (polytetrafluoroethylene) (also known as Teflon) were at first chosen to be the alternative materials of the articular cartilage. However, the PVA hydrogel was easy to be air-dried and fall off from the joint surface. So I eventually decided to use the material PTFE adhesive tape to act as the articular cartilage. In the model, the 0.13mm thick flexible polytetrafluoroethylene (PTFE) adhesive tape is structured by PTFE coated woven fiberglass fabric, of which the PTFE layer could offer low friction and non-stick surface without leaving residues and the underneath fiberglass provides enough strength and structural stability. PTFE is often used in the cookware or the computer components because of its chemical inertness and low static and kinetic coefficient of friction (0.1) [132] [133]. Besides, it can be applied to clinical situations such as in restorative dentistry [134] [135]. In this case, the PTFE tape can be firmly stuck onto the bones' ends to act as the articular cartilage, ensuring a smooth contact of finger joint bones without damage to the joint surface shape, as shown on the physical model in Figure 50 (d).

3.2 The Second Layer — Capsuloligamentous structures

The second layer of the finger is the capsuloligamentous structure, including the joint ligaments and capsules. There are mainly three ligaments around a joint bonding two adjacent phalanges together. Taking the MCP joint as an example, two collateral ligaments respectively arise from each side of the metacarpal head and obliquely pass downwards to the insertion of the proximal phalanx base. Another thicker and stiffer ligament, called the palmar or volar plate, is located at the palmar side of the joint, preventing hyperextension [6].

The biomechanical property of the ligament is associated with its microstructure, for example, it has a characteristic sinusoidal wave pattern which is known as crimp [137]. Once a load is initially applied to a ligament, the "crimp pattern" will be straightened without causing much tension. This stage can be defined as the toe region. As the loading continues, stiffness of the ligament increases instantaneously, resulting in the elastic deformation. This region is called the elastic or linear region, followed by yielding and failing of the structure in the end [130].

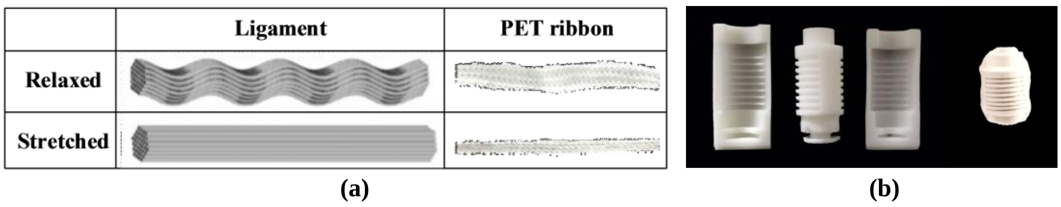


Figure 51. Capsuloligamentous structure of the robotic finger. (a) "Crimp pattern" of the human ligament and the PET braided ribbon. (b) Designed moulds and silicone rubber joint capsule.

In this proposed robotic finger design, I have tried the materials of PET, PBT and Spandex fibre to mimic the ligaments. During the test, I found the PBT was not soft enough and the Spandex fibre was too weak to be the ligaments. The material PET is soft and strong, and most importantly, can be easily braided to show the nonlinear elastic properties similar to human ligaments. Thus the polyethylene terephthalate (Polyester, PET) fibre ribbons were selected to act as artificial ligaments and were sintered onto the bones. Its braided structure closely resembles the crimp pattern of the human ligaments, providing analogous strain-stress property. Moreover, the band shape allows its different portions being stretched successively through the whole motion range to properly constrain the joint position and stiffness. Figure 51 (a) compares the human ligament and the PET ligament in different tension states.

Further, attached to the whole circumference of the articular end of each bone is a joint capsule, which consists of an outer fibrous membrane and an inner synovial membrane. Together with the ligaments, it can properly limit the range of finger joint motion and prevent excessive laxity so as to assist joint stabilization, and meanwhile, seal the synovial liquid inside which can provide a good lubrication condition [138] [139]. Additionally, similar to the joint ligaments, there are also some folds on capsules' surface allowing enough motion range when stretched and simultaneously preventing accumulation hump when compressed. Since no lubricating liquid exists in the proposed robotic finger design, only the capsule with function of improving the joint stability is considered. Several customized moulds were designed and 3D printed. The liquid silicone rubber (Polycraft GP3496-F Shore A13) was injected into the moulds and cured into a 0.5mm thick elastic capsule with folds, as shown in Figure 51 (b). The CAD model of the joint capsule moulds are shown in Appendix II.

For the presented capsule design, except for the material, the shape, dimension, number of the folds, and the wall thickness also dramatically influence its properties. The present design of the capsule has triangle-shaped folds on the dorsal-palmar sides to reduce the resistance moment and structural accumulation during flexion. The width and depth of the folds are 1.5mm and the interval distance between the folds is 0.5mm. There are no folds on the radial-ulnar sides of the capsule because of the small range of the abduction-adduction motion and the stability of the joint can be well-maintained with the proposed structure.

3.3 The Third Layer — Tendons and tendon sheaths

The tendons and tendon sheaths together constitute the finger's third layer. The muscle-tendon system is the actuation component of the human finger, of which the tendons extend from the muscles to the base of the phalanges. There are mainly two flexor tendons that are the flexor digitorum superficialis (FDS) and the flexor digitorum profundus (FDP), and one extensor mechanism covering the dorsal side of the finger whose proximal ends connect with the long extensor (LE), the ulnar interosseous (UI), the radial interosseous (RI) and the lumbrical muscle (LU), respectively. For the index finger, another tendon, called extensor indicis, spreads out from the extensor mechanism, enhancing the independent control of the index finger [140]. In fact, the tendons are quite strong with softness and compliance, and these unique biosolid characteristics allow the tendons to transmit large amounts of forces [141]. Different with the flexor tendons, the extensor mechanism is a tendinous network structure that connects the intrinsic and extrinsic muscles together to provide the function of not only an extensor but also a flexor, abductor, adductor or rotator according to the finger's targeting motion [32].

In the proposed robotic finger, the design of the flexor tendons is so much easier than that of the extensor design because of the complex anatomy structure of the extensor expansion in the human hand. As shown in Figure 52, we can see the whole design evolution of the extensor expansion, from the anatomical diagram to the schematic diagram, and the last one is the structure sketch of the design according to the abstracted Winslow's tendinous rhombus model [156]. The red points are the insertions on the base of the DIP and PIP joint, and the yellow points represent the connection points of the strings. The web structure consists of two layers: the under layer coloured in blue and the upper layer coloured in orange. The other three strings colored in black will play the role of control and restriction. Figure 53 shows the state that the extensor expansion covers on the dorsal side of the finger. We can notice that there are three ends for the web structure, the two short strings which end with the red cross means that they are connected with the intrinsic muscles, and the long one will act as the long extensor tendon.

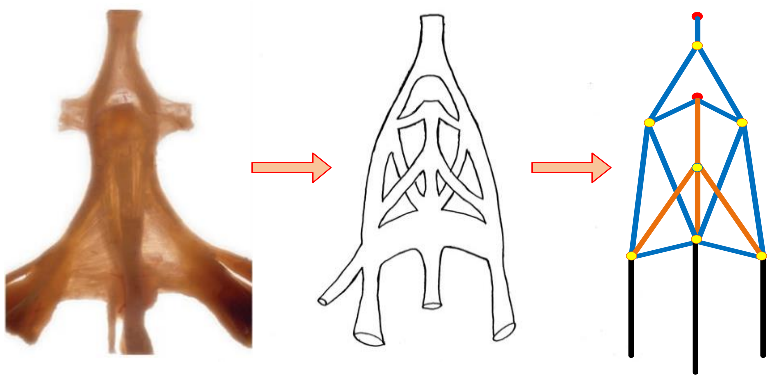


Figure 52: Design evolution of the extensor expansion

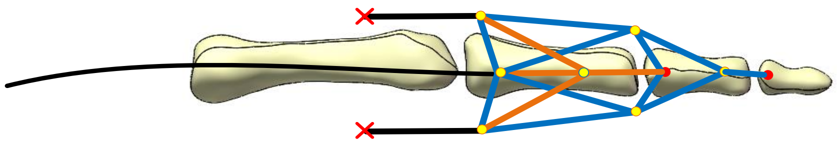


Figure 53: The extensor expansion covered on the dorsal side of the finger

Then we fabricated a tendon network by using the 30lb fishing lines with 0.43mm diameter made from 100% Polyester Dacron fibres (Troutcatchers, UK), where the connective soft tissues could be acted by the silicone rubber tendon sheaths. Through the designed holes, the tendons can be tied onto the bones firmly. Because of the similarity in the locations and functions between the lumbrical muscle and the intrinsic muscle, the LU was removed in this design to simplify the actuation and control system. Thus, the physical prototype of the tendons was shown in Figure 55.

Further, wrapping around the tendon is the tendon sheath. It is a two-layer membrane which consists of a synovial sheath layer and a fibrous tendon sheath layer, allowing the tendon to move smoothly inside. The tendon sheaths along the palmar side of the phalanges construct a pulley system that can guide the route for the flexor tendons and restrain their bowstringing behaviour as well by holding them at a relatively close distance to the joints [143]. Given that the muscle contraction can only provide limited tendon excursion, the bowstringing behaviour over the phalanges will significantly reduce the potential joint motion [42]. Figure 56 shows the schematic drawings of the components of the digital flexor tendon sheath. As can be seen, there are five annular strong pulleys (A1, A2, A3, A4, A5 in Figure 56) and three thin, pliable cruciate pulleys (C1, C2, C3 in Figure 56) [144]. Specifically, the pulley A2 and A4, which are also the main concerned parts in the physical model design, significantly influence the moment arms of the flexor tendons, altering the biomechanical properties of the finger joints [126].

Using the same fabricating method for the joint capsule, the flexible tendon sheaths were also moulded with silicone rubber and the CAD model of the moulds are shown in Appendix III. Inside the tendon sheaths, flexible PTFE tubes with 1.58mm inner diameter (Adtech Polymer Engineering Ltd, UK) were inserted to provide lubricated tunnels for the tendons. Although there is no specific tendon sheath structure for the extensor mechanism in the human finger, we designed one for the robotic finger to act as the extensor mechanism membrane embedding with channels (Figure 54). And as can be seen in Figure 56, each tendon sheath respectively covers around the middle, proximal, and metacarpal phalanges.

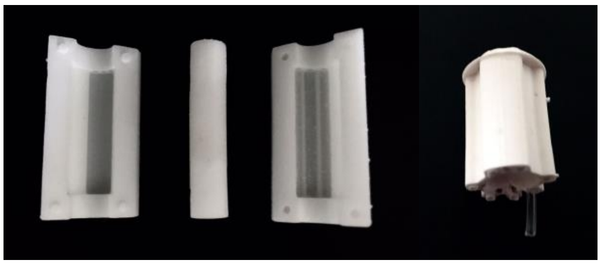


Figure 54. Designed moulds and silicone rubber tendon sheath with PTFE tubing.

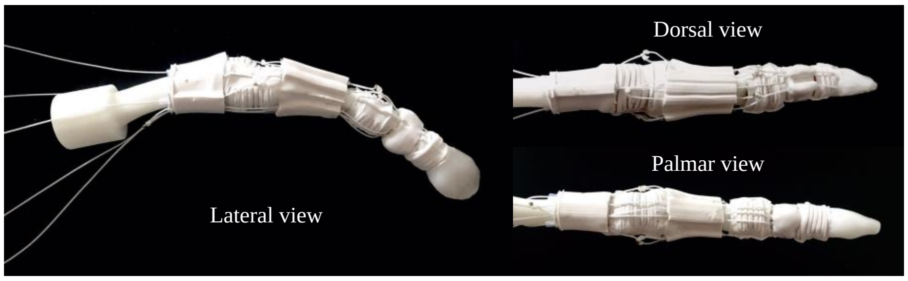


Figure 55. Distribution of the tendons along the robotic finger.

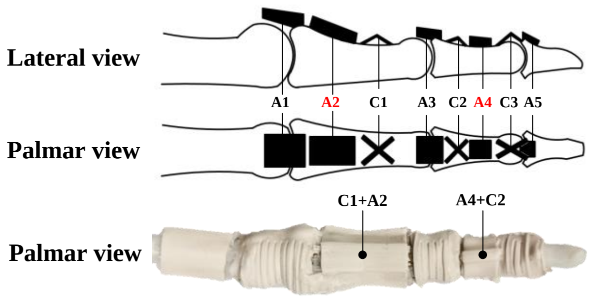


Figure 56. Distribution of the tendon sheaths along the human and robotic finger.

3.4 Basic kinematic indexes of the robotic finger

To test the basic kinematic indexes of the robotic finger, a customized testbed was designed, 3D printed and assembled, as shown in Figure 57 and . In some tests, the finger needs to be actuated, such as obtaining the fingertip trajectory or testing the finger full flexion-extension speed. Thus, five Dynamixel motors were mounted on the testbed to respectively drive the FDP, FDS, LE, UI and RI tendon in case needed, shown in Figure 58. The Dynamixel MX-12W motors were connected with PC through an USB2Dynamixel connector. An SMPS 12V 5A PS-10 (Switched-Mode Power Supply) and a 6 Port AX/MX Power Hub were used to provide the standard power for several motors at the same time. Thus, the USB2Dynamixel system with 5 motors controlled is shown in Figure 59.

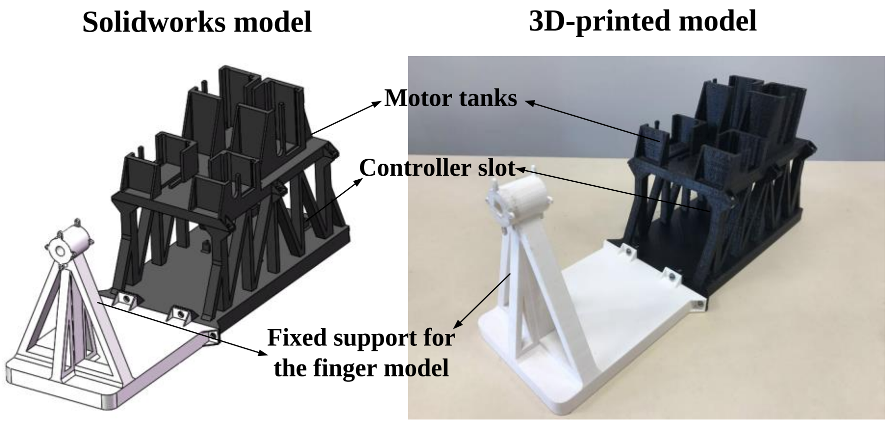


Figure 57. The CAD model and the 3D printed model of the platform for one finger

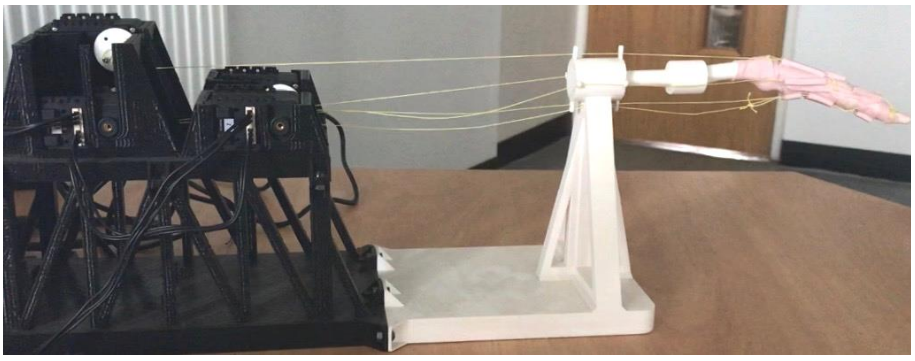


Figure 58. The constructed testbed for one finger

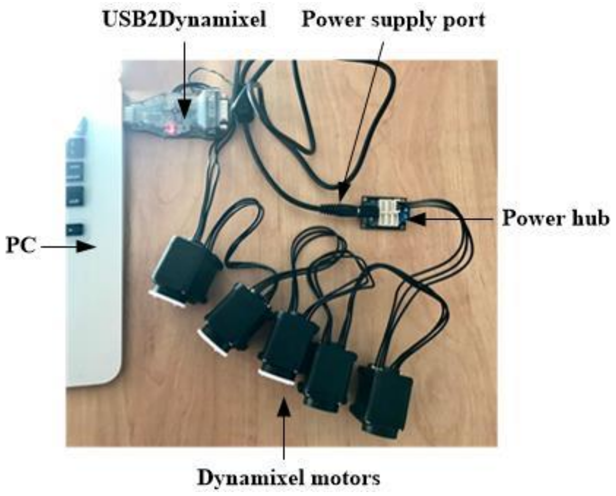
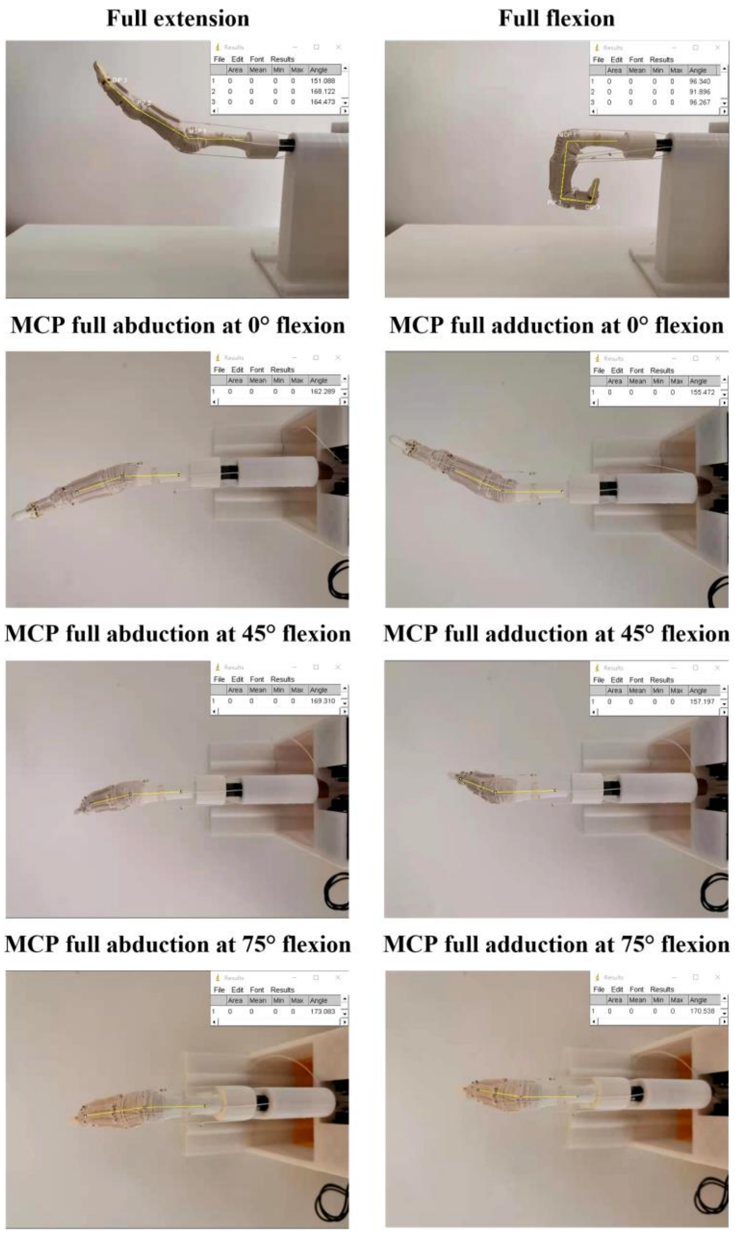


Figure 59. Components of Dynamixel motor control system

3.4.1 Range of motion test

The range of motion of each finger joint is one of the most essential and common parameters for the robotic hand or finger validation. The flexion and extension range of the DIP, PIP and MCP joint were tested. Additionally, although the abduction-adduction motion does exist in the DIP and PIP joints due to oblique orientation of the flexion-extension axis of rotation, it’s too small to be noticed. While such motion is obvious in the MCP joint. Thus, only the abduction-adduction motion range in the MCP joint was measured. Meanwhile, the eccentric shape and attachment position of the collateral ligaments in the MCP joint would cause the change of their apparent lengths and stiffness during joint rotation, leading to various ranges of MCP joint’s abduction-adduction motion. Thus, the abduction-adduction motion range in the three MCP joint flexion positions (0°, 45°, 75°) were selected. The test process is shown in Figure 60. And Table 4(a) presents the motion range of each joint. It can be considered reasonable compared with the corresponding results of the cadaver testing from Chao [6] (Table 4(b)), though most of the robotic finger joints can reach wider angle range, especially the MCP abduction-adduction motion.


Full extension
Full flexion
MCP full abduction at 0° flexion
MCP full adduction at 0° flexion
MCP full abduction at 45° flexion
MCP full adduction at 45° flexion
MCP full abduction at 75° flexion
MCP full adduction at 75° flexion
Results
File Edit Font Results

```
   Area Mean Min Max Angle
1  0    0    0   0   151.088
2  0    0    0   0   168.122
3  0    0    0   0   164.473
```

```
   Area Mean Min Max Angle
1  0    0    0   0   96.340
2  0    0    0   0   91.896
3  0    0    0   0   96.267
```

```
   Area Mean Min Max Angle
1  0    0    0   0   162.289
```

```
   Area Mean Min Max Angle
1  0    0    0   0   155.472
```

```
   Area Mean Min Max Angle
1  0    0    0   0   169.310
```

```
   Area Mean Min Max Angle
1  0    0    0   0   157.197
```

```
   Area Mean Min Max Angle
1  0    0    0   0   173.083
```

```
   Area Mean Min Max Angle
1  0    0    0   0   170.538
```

Figure 60. Test of each joint motion range of the robotic finger

Table 4. Joint motion range of human (a) and robotic finger (b)

(a)

Joint	Full Flexion Angle (°)	Full Extension Angle (°)	Full Abduction Angle (°)	Full Adduction Angle (°)
DIP	83.7	15.5		
PIP	98.5	11.9		
MCP	83.7	28.9		

0 °MCP			17.7	24.5
45 °MCP			10.7	22.8
75 °MCP			6.9	9.5

(b)

Joint	Full Flexion Angle (°)	Full Extension Angle (°)	Full Abduction Angle (°)	Full Adduction Angle (°)
DIP	73	11.45		
PIP	101	11		
MCP	83	22		
0 °MCP			10	11
45 °MCP			8	7
75 °MCP			3	4

3.4.2 Fingertip trajectory test

The repeatability of the fingertip's trajectory should be investigated, especially when the freedom of the finger is controlled in between full flexion-extension and adduction-abduction postures. Vicon motion capture system composed of 6 cameras was used in the test, shown in Figure 61. A traditional spherical reflective marker was attached to the fingertip. Besides, we attached a marker to the frame near the wrist of our robotic hand, so that we can transform the coordinate from the default world frame to the wrist frame when processing the data. The robotic finger was actuated by only FDP tendon. During the test, 5 repetitions of the full flexion-extension and adduction-abduction motions were recorded and saved as a txt file, which was imported into matlab and processed afterwards. The result in Figure 62 shows a highly repeatable space trajectory pattern, indicating that the proposed robotic finger is controllable. Besides, the trajectory obtained from the test was quite similar to the sweep motion trajectory of the human finger in the cadaver test from [145], implying that the bio-inspired robotic finger has human-finger-like kinematic properties in this respect.

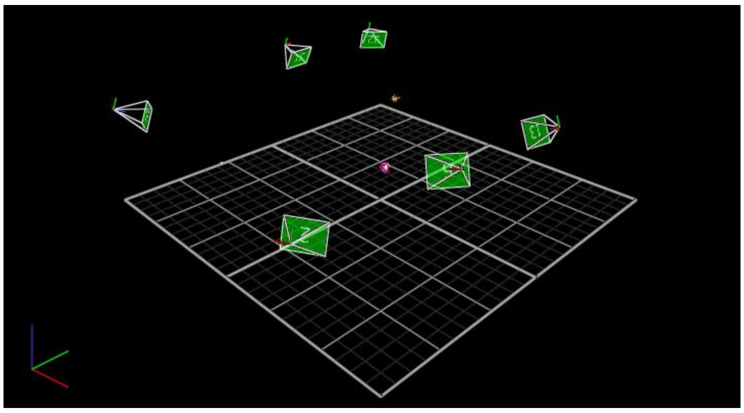


Figure 61. Interface of Vicon motion capture system

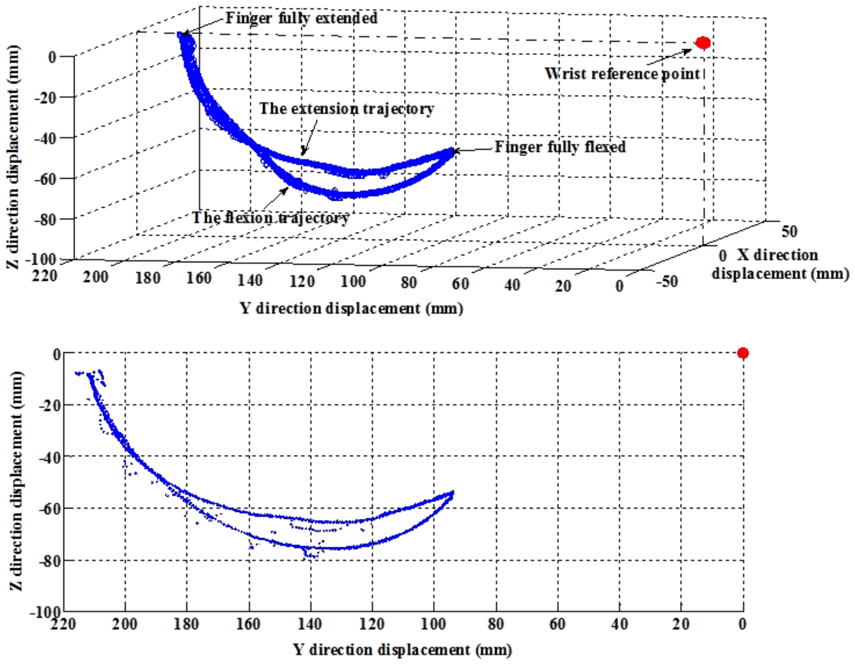


Figure 62. Space trajectory of the fingertip during flexion and extension motion

3.4.3 Full flexion-extension speed test

Using the same testbed, the full flexion-extension speed of the robotic finger was measured. Likewise, the flexion motion was actuated by only FDP tendon and the extension motion was provided by driving the LE tendon. And the actuation motor for each tendon works at full rotation speed (470rpm). The whole flexion-extension processes of the robotic and human finger were both recorded by a high-speed camera. Then the total elapsed time from the initial position to the end position of the corresponding motion was obtained from the video. The process of robotic and human finger motion was shown in Figure 63. The result shows that the full flexion motion of the robotic finger takes longer time which is 0.167s compared with the human finger which takes 0.133s. In the full extension motion, the robotic finger moves faster which totally takes 0.2s and the human finger uses about 0.25s to complete the whole motion. It indicates that the robotic finger has comparable motion speed to the human finger with the actuation of MX-12W motors, where the full extension speed of the former is even faster than that of the latter, showing good kinematic properties of the proposed robotic finger with the customized actuation system.

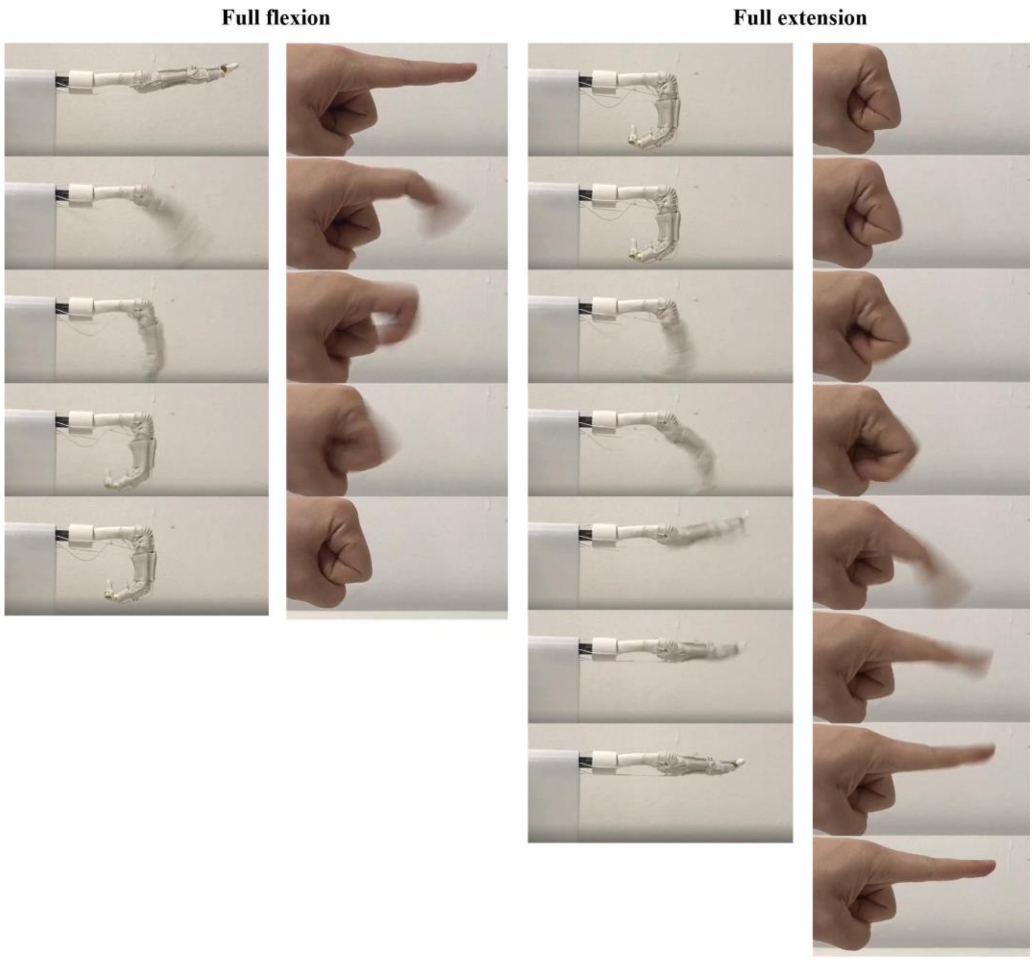


Figure 63. Test of the full flexion-extension speed of the robotic and human finger

3.5 Summary

The robotic finger, as the basic functional unit of the robotic hand, was developed first in the project. This chapter introduced the human finger anatomy and the bio-inspired multi-layered design of a robotic finger. There are totally three layers designed on the robotic finger: the support layer-phalanges and articular cartilage, the second layer-ligaments and joint capsules, and the third layer-tendons and tendon sheaths. The CT scanned and 3D printed bones model nearly restored the geometry of the human finger bones. The Teflon tape can provide the lubrication environment for joint movements as the articular cartilage does. The PET braided ribbon ligaments have similar micro-structure and properties to human ligaments. With the silicone rubber joint capsule, they can highly mimic the capsuloligamentous structure of the human finger. The tendon sheaths were also moulded with silicone rubber, into which the PTFE tubes were inserted to smoothly guide the tendons route. And the fishing line tendons were strong and flexible enough to drive the fingers freely. In the proposed robotic finger, each component shows similar morphologies and properties to its counterpart in the human finger. Afterwards, several kinematic indexes of the robotic finger were tested and a customized testbed was designed. From the results of range of motion, fingertip trajectory and motion speed, it can be seen that the robotic finger designed in this project performed similar kinematic properties to the human finger. Likewise, it is expected that similar biomechanical advantages and the resulted human-finger-like performance can be embodied through these delicately designed and fabricated functional components. To further study the biomechanical properties and performance of the proposed robotic finger, three topics, i.e., variable joint stiffness, feasible force space, and force-velocity workspace are investigated through mathematical modelling, simulation, and experimental verification in the following two chapters.

Chapter 4: Theoretical analysis for three biomechanical properties of human-finger-like structures

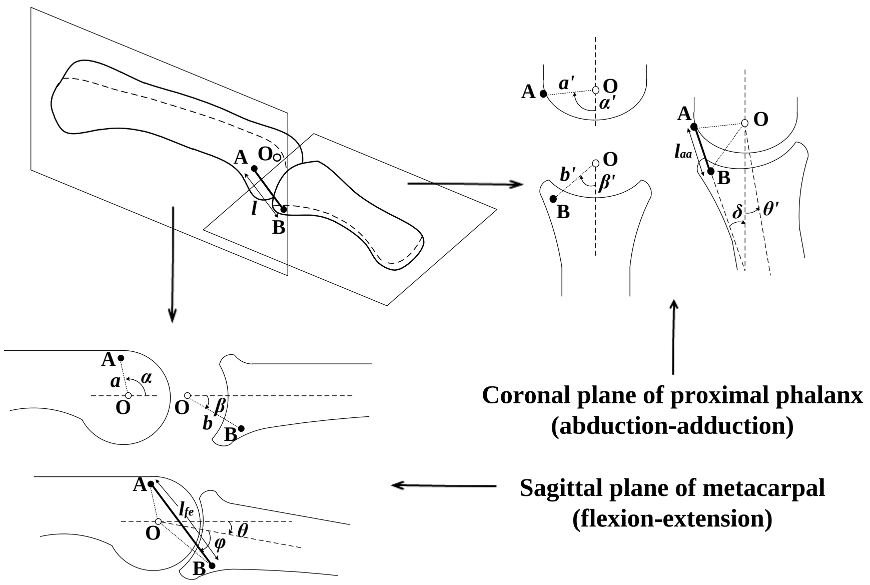
This chapter mainly analyzes three biomechanical properties of the proposed robotic finger by establishing corresponding mathematical models, including the joint stiffness, the fingertip feasible force space and the fingertip force-velocity characteristics. These biomechanical properties respectively correspond to three human-finger-like structures, i.e., the ligamentous joint, the reticular extensor mechanism and the flexible tendon sheath. The aim of this chapter is to investigate how these biological structures influence the biomechanical properties by using mathematical methods.

4.1 Joint stiffness associated with the ligamentous structure

Combined dexterity and stability of the human hands benefits a lot from the ligamentous structure of joints. To analyze its inner mechanism, a mathematical model of the ligamentous joint is established, and the joint stiffness is investigated in this chapter.

4.1.1 Geometric analysis

The joint stiffness change is mostly a result of the length change of the ligaments. In order to model and characterize such a change during the joint motion, we take the MCP joint as an example. Considering that the contact with the bone's convex may lead to a complex deformation for the ligament, the ulnar collateral ligament was chosen as the research object since it has less contact with the bone's convex than the radial collateral ligament. A schematic diagram of the MCP joint and the associated ligament is shown in Figure 64, where *O* is the intersection between the rotation axes of flexion-extension motion and abduction-adduction motion, points *A* and *B* respectively represent the origin and insertion of the ligament, and *l* is the ligament's length.



A
O
l
B
A
a′
O
α′
b′
O
β′
B
A
O
l_{aa}
B
δ
θ′
A
a
α
O
O
β
b
B
A
l_{fe}
O
θ
φ
B
Coronal plane of proximal phalanx (abduction-adduction)
Sagittal plane of metacarpal (flexion-extension)

Figure 64. Schematic diagram of the ligament's geometry in MCP joint motion

Firstly, the joint stiffness in flexion-extension motion is investigated which is normally simplified as a two-dimensional case. Referring to Figure 64 and taking the view of the sagittal plane of metacarpal (defined as the flexion-extension plane), the MCP joint is assumed to be a hinge joint during the flexion-extension motion based on the model proposed by [146]. The metacarpal bone and proximal phalanx articulate together form a hinge joint with the rotation axis passing through point *O*. As aforementioned, the ligament's origin and insertion are at points *A* and *B*, respectively. Projection on the sagittal plane, point *A* is at a distance *a* from point *O* along a line that makes an angle *α* with the long axis of the metacarpal bone, and point *B* is at a distance *b* from *O* along a line, making an angle *β* with the long axis of the proximal bone. In this definition, the parameters *a, b, α* and *β* are all constants, which depend on the anatomical structure of the MCP joint and do not change with the motion of the joint. Besides, the bones are assumed to be rigid bodies, thus no elastic compliance or surface geometry change of the bones is considered. The distance *AB*, i.e. the length of the ligament observed for flexion-extension motion, is denoted as l_{fe}. This length varies throughout the joint flexion-extension motion process. We consider an instant when the joint flexes about point *O* by an angle *θ*, in which case, the angle $\angle AOB$ is ($a+\beta+\theta$) as shown in Figure 64, and the length l_{fe} can be calculated by the cosine rule for a triangle as

$$l_{fe} = \sqrt{\left(a^2 + b^2 - 2ab\cos\left(\alpha + \beta + \theta\right)\right)} \tag{1}$$

Normally, the ligament's function is performed in the elastic deformation stage. In this case, the passive tension on the ligament can be expressed as

$$F_{fe} = k \cdot \left(l_{fe} - l_e\right) + F_e \tag{2}$$

Where F_{fe} is the passive tension on the ligament, k is the elastic coefficient, l_{fe} is the length of the ligament, l_e and F_e are the ligament's initial length and tension respectively when the elastic deformation starts.

Then the moment arm h_{fe} of the ligament to the joint on the flexion-extension plane is

$$h_{fe} = \frac{b \cdot a \cdot \sin\left(\alpha + \beta + \theta\right)}{l_{fe}} = \frac{b \cdot a \cdot \sin\left(\alpha + \beta + \theta\right)}{\sqrt{\left(a^2 + b^2 - 2ab\cos\left(\alpha + \beta + \theta\right)\right)}} \tag{3}$$

Thus, the resistance torque produced by the ligament in the flexion-extension direction is

$$\tau_{fe} = -F_{fe} \cdot h_{fe} = -kA_1 + \frac{A_2 A_1}{l_{fe}} \tag{4}$$

where k is ligament elastic stiffness coefficient, $A_1 = ab\sin(\alpha + \beta + \theta)$, $A_2 = kl_e - F_e$.

It can be seen from the above equation (4) that the torque of the ligament to the joint is the function of the joint angle θ when the ligament's biomechanical property and location are predetermined. And based on this, the joint stiffness K_{fe} in the flexion-extension direction contributed by the ligament can be derived

$$K_{fe} = \frac{\delta\tau_{fe}}{\delta\theta} = -A_1(k - \frac{A_2}{l_{fe}}) - \frac{A_2A_3}{l_{fe}^3} \tag{5}$$

where k is ligament elastic stiffness coefficient, $A_1 = ab\cos(\alpha + \beta + \theta)$, $A_2 = kl_e - F_e$ and $A_3 = a^2b^2\sin^2(\alpha + \beta + \theta)$. The result implies that the stiffness K_{fe} changes with the joint flexion angle θ.

Since the MCP joint has two degrees of freedom, including the flexion-extension motion and the adduction-abduction motion. The ligament's length change resulted from the flexion motion can also influence the joint stiffness in the adduction-abduction direction. Therefore, the coronal plane of the proximal phalanx (defined as the adduction-abduction plane), which is always perpendicular to the adduction-abduction rotation axis throughout the joint motion, needs to be added to elaborate the joint adduction-abduction stiffness as shown in Figure 64.

On this plane, point O represents the rotation axis for adduction-abduction motion, which also rotates around the flexion-extension rotation axis passing point O on the flexion-extension plane, and here points A and B are the projection of the ligament's origin and insertion on the abduction-adduction plane. On this plane, point A is at a distance a' from point O along a line that makes an angle α' with the sagittal plane of the metacarpal bone. Point B is at a distance b' from O along a line forming an angle β' with the sagittal plane of the proximal bone. The distance between points *A* and *B* is l_{aa}, representing the projection of the ligament's length on the abduction-adduction plane. In addition, the abduction-adduction rotation angle around *O* is defined as θ' which should be approximate to zero, since only in this condition the ligament does not have too much contact with the bone convex (which will greatly complex the analysis). Thus, on the abduction-adduction plane, the angle $\angle AOB$ is ($\alpha' - \beta' + \theta'$), and similarly, the length l_{aa} can be calculated as

$$l_{aa} = \sqrt{\left(a'^2 + b'^2 - 2a'b'\cos\left(\alpha' - \beta' + \theta'\right)\right)} \tag{6}$$

It should be noted that the length l_{aa} is just the projection of the actual ligament length on the adduction-abduction plane. In order to obtain the actual ligament length denoted as l, the angle δ between the ligament's projection on the abduction-adduction plane and the sagittal plane of the metacarpal bone, and the angle φ between the ligament's projection on the sagittal plane of the metacarpal bone and the adduction-abduction plane are required, as shown in Figure 64. It has

$$\delta = \pi - \alpha' - \arcsin\left(\frac{b'\sin\left(\alpha' + \theta' - \beta'\right)}{l_{aa}}\right) \tag{7}$$

$$\varphi = \pi - \alpha - \theta - \arcsin\left(\frac{b\sin\left(\alpha + \beta + \theta\right)}{l_{fe}}\right) \tag{8}$$

Where a, b, α, β, θ are the same as defined on the flexion-extension plane. From Eqs. (7) and (8), it can be seen that angle δ is related to l_{aa}, and angle φ is associated with l_{fe}.

Referring to Figure 64, the length *l* of the ligament can be implicitly represented with the flexion angle θ and the adduction-abduction angle θ' as

$$l = \sqrt{\frac{{l_{aa}}^2 \cos^2\delta}{\cos^2\varphi} + {l_{aa}}^2 \sin^2\delta} = l_{aa}\sqrt{\cos^2\delta \tan^2\varphi + 1}$$

(9)

From Eqs. (7), (8) and (9), it can be seen that the ligament stretches with the increase of not only the flexion angle θ but also the abduction angle θ'.

Given that the ligament's property is still in the linear region, then the passive tension force *F* on the ligament is:

$$F = k \cdot (l - l_e) + F_e$$

(10)

Where *k* is the elastic coefficient, *l* is the final length of the ligament, l_e and F_e are the ligament's initial length and tension respectively when the elastic deformation starts.

By projecting the ligament force onto the abduction-adduction plane, we can get one component F_{aa} of the ligament force which contributes to the resistance torque of the abduction-adduction motion.

$$F_{aa} = F \cdot \cos\eta = \left(k \cdot (l - l_e) + F_e\right) \cdot \cos\eta = k \cdot l_{aa} - \left(k \cdot l_e - F_e\right) \cdot \cos\eta$$

(11)

Where $\eta = \arccos(l_{aa} / l) = \arccos(1/\sqrt{\cos^2\delta \tan^2\varphi + 1})$, which is the angle between the ligament and the abduction-adduction plane.

The moment arm of the ligament force F_{aa} to the rotation axis O on the abduction-adduction plane is:

$$h_{aa} = \frac{b' \cdot a' \cdot \sin\left(\alpha' - \beta' + \theta'\right)}{l_{aa}} = \frac{b' \cdot a' \cdot \sin\left(\alpha' - \beta' + \theta'\right)}{\sqrt{\left(a'^2 + b'^2 - 2a'b'\cos\left(\alpha' - \beta' + \theta'\right)\right)}} \tag{12}$$

Thus, the resistance torque τ_{aa} of the ligament force F_{aa} to the joint is:

$$\tau_{aa} = -F_{aa} \cdot h_{aa} = -kA_1 + \frac{A_2 A_1 \cos\eta}{l_{aa}} \tag{13}$$

where k is ligament elastic stiffness coefficient, $A_1 = a'b'\sin(\alpha' - \beta' + \theta')$, $A_2 = kl_e - F_e$.

Considering the joint abduction stiffness K_{aa} as the function of the abduction angle θ', it can be calculated as:

$$K_{aa} = \frac{\delta\tau_{aa}}{\delta\theta'} = \frac{a'A_7\left(a'A_1^{\,2} - A_2 A_6^{\,2}\right)}{A_6^{\,3}\sqrt{A_3^{\,2}A_4^{\,2} + 1}} - \frac{a'A_3^{\,2}A_1A_4A_5A_7\left(\frac{A_2}{A_6} - \frac{a'A_1^{\,2}}{A_6^{\,3}}\right)}{\left(A_3^{\,2}A_4^{\,2} + 1\right)^{\frac{3}{2}} A_6\sqrt{\left(1 - \frac{A_1^{\,2}}{A_6^{\,2}}\right)}} - ka'A_2 \tag{14}$$

Where

$A_1 = b'\sin\left(\alpha' - \beta' + \theta'\right)$

$A_2 = b'\cos\left(\alpha' - \beta' + \theta'\right)$

$$A_3 = \tan\left[\alpha + \theta + \arcsin\left(\frac{b\sin\left(\alpha + \beta + \theta\right)}{\sqrt{a^2 - 2ab\cos\left(\alpha + \beta + \theta\right) + b^2}}\right)\right]$$

$$A_4 = \cos\left[\alpha' + \arcsin\left(\frac{b'\sin(\alpha' - \beta' + \theta')}{\sqrt{a'^2 - 2a'b'\cos(\alpha' - \beta' + \theta') + b'^2}}\right)\right]$$

$$A_5 = \sin\left[\alpha' + \arcsin\left(\frac{b'\sin(\alpha' - \beta' + \theta')}{\sqrt{a'^2 - 2a'b'\cos(\alpha' - \beta' + \theta') + b'^2}}\right)\right]$$

$$A_6 = \sqrt{a'^2 - 2a'b'\cos(\alpha' - \beta' + \theta') + b'^2}$$

$$A_7 = F_e - kl_e$$

4.1.2 Algebraic analysis

The geometrical analysis for the joint ligament presented above can provide a detailed derivation process and the specific relationship among the variables. However, to further quantify and clearly describe the ligament's length and stiffness change during the joint motion, the algebra analysis is needed and conducted as below.

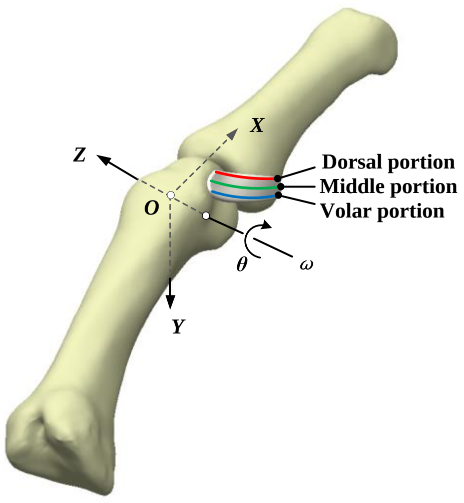


Figure 65. A 3D coordinate system in MCP joint with the ulnar ligament

To conduct the algebraic analysis, a Cartesian coordinate system was established at the MCP joint, shown in Figure 65. The origin bisects the tubercle of the metacarpal head. We take the distal, palmar and radial directions as the positive *X*-, *Y*- and *Z*-axis directions, respectively. The origins and insertions of the ulnar collateral ligament's dorsal, middle and volar portions in the coordinate system are located by referring to the cadaver data from the work of Chao [6].

In the coordinate system, the ligament length can be expressed as:

$$l = \sqrt{(x_i - x_o)^2 + (y_i - y_o)^2 + (z_i - z_o)^2} \tag{15}$$

where (x_i, y_i, z_i) and (x_o, y_o, z_o) are the coordinate of the ligament's insertion and origin.

We assumed that the joint flexion angle is θ and the position of the origin point is constant in this MCP coordinate system. The insertion point will rotate an angle θ around the z axis. Thus the coordinate of the insertion point (x_{ir}, y_{ir}, z_{ir}) after the rotation can be obtained.

$$\begin{bmatrix} x_{ir} \\ y_{ir} \\ z_{ir} \end{bmatrix} = \mathbf{R}_\theta \cdot \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix} \tag{16}$$

Where $\mathbf{R}_\theta$ is the rotation matrix

$$\mathbf{R}_\theta = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{17}$$

Then the ligament length change during the whole joint flexion motion can be calculated as follows.

In order to identify the relationship between the ligament's length in the flexion-extension direction l_{fe} and the joint flexion angle θ, we can substituted the cadaver data from the reference [6] into the equations (15), (16) and (17), and plotted out the change of l_{fe} with respect of flexion angle θ approximately within the joint range of motion as shown in Figure 66. Referring to Figure 66, the volar portion of the ligament shortened during flexion. The middle portion nearly remained the same length, but with a slight drop after 40 °flexion. While the length of the dorsal portion increased by more than 2 mm during flexion. The results show the similar variation trend of the ligament's length with Chao's work [6]. A more detailed research on the length and shape change of the MCP joint ligaments could be found in [16] with the method of microcomputed tomograms and 3D modelling. There was no significant difference in the results of this research, only with some deviation resulted by the cadaver data source.

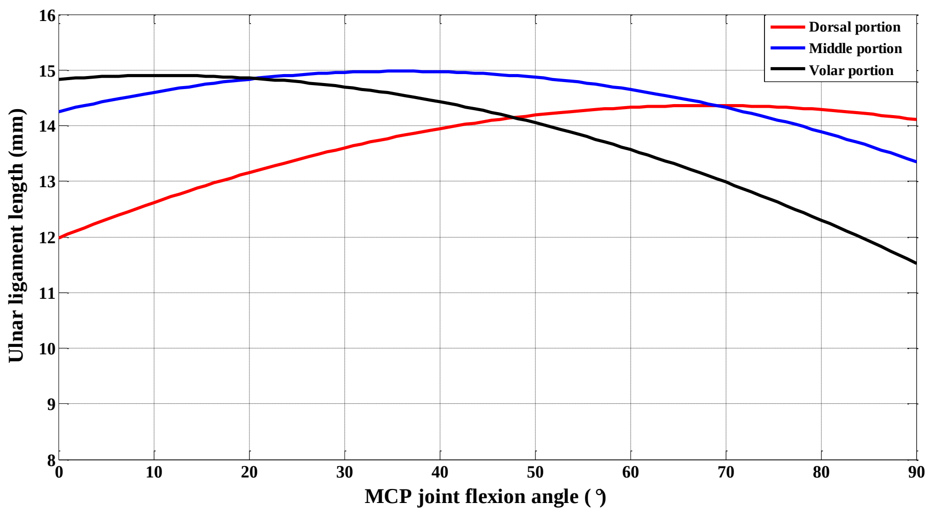


Figure 66. The length change of the three portions of the ulnar ligament with the MCP joint flexion angles

Since the flexion-extension motion is in the *X*-*Y* plane, the moment arm of the ligament to the joint is the distance from the projection of the rotation axis to the projection of the ligament in *X*-*Y* plane. In this plane, the rotation axis is projected to a point (0, 0), and the ligament is projected to a line through the point (x_o, y_o) and (x_i, y_i). Then the linear equation of the ligament's projection can be expressed as:

$$\frac{y_i - y_0}{x_i - x_0} \cdot x - y + y_0 - \frac{y_i - y_0}{x_i - x_0} \cdot x_0 = 0 \tag{18}$$

And the function of the distance from one point (x_p, y_p) to the line is:

$$d = \frac{\left| Ax_p + By_p + C \right|}{\sqrt{A^2 + B^2}} \tag{19}$$

Where $A = \frac{y_i - y_0}{x_i - x_0}, B = -1, C = y_0 - \frac{y_i - y_0}{x_i - x_0} \cdot x_0$.

Thus the moment arm *M* of the ligament to the joint in flexion-extension motion can be obtained, where (x_p, y_p) is (0, 0):

$$M = \frac{|C|}{\sqrt{A^2 + B^2}} = \frac{\left| y_0 - \frac{y_i - y_0}{x_i - x_0} \cdot x_0 \right|}{\sqrt{\left(\frac{y_i - y_0}{x_i - x_0} \right)^2 + 1}} \tag{20}$$

Assuming that the ligament is in the linear region, thus the passive tension *F* on the ligament can be expressed as:

$$F = F_0 + k\Delta l_0 = F_0 + k\left(\sqrt{(x_i - x_o)^2 + (y_i - y_o)^2 + (z_i - z_o)^2} - \sqrt{(x_e - x_o)^2 + (y_e - y_o)^2 + (z_e - z_o)^2} \right) \tag{21}$$

Where k is the elastic coefficient, F_0 is the initial tension of the linear region, Δl_0 is the ligament's length change in the linear region, and (x_e, y_e, z_e) is the location of the ligament's insertion when the ligament starts the elastic deformation.

To obtain the elastic coefficient of the artificial ligament adopted in the model, the strain-stress property of the PET ribbon was tested and shown in Figure 67. Similarly, a toe region followed by a linear region was presented in the curve, potentially providing the robotic finger joint with the similar mechanical performance of the human finger joint.

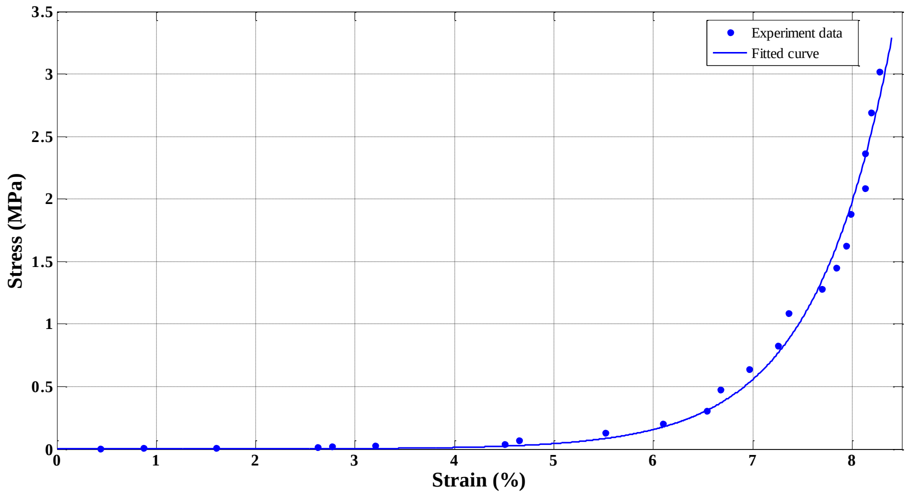


Figure 67. The tested strain-stress property of the PET braided ribbon

The resistance torque τ in the flexion-extension motion is generated by the component of the ligament's tension force on the flexion-extension plane, it has:

$$\tau = -F\cos\gamma \cdot M \tag{22}$$

Where $\gamma = \arcsin\left(\frac{|\boldsymbol{n}_l \cdot \boldsymbol{n}_{fe}|}{|\boldsymbol{n}_l| \cdot |\boldsymbol{n}_{fe}|}\right) = \arcsin\left(\frac{|z_i - z_o|}{\sqrt{(x_i - x_o)^2 + (y_i - y_o)^2 + (z_i - z_o)^2}}\right)$, which is the angle between the ligament and the flexion-extension plane. In the expression of γ, $\boldsymbol{n}_l$ is the vector of the ligament and $\boldsymbol{n}_{fe}$ is the unit normal vector of the flexion-extension plane.

The negative sign means the torque impedes the flexion motion. Then the joint stiffness contributed by the dorsal portion of the ulnar ligament can be obtained with the function:

$$K_{fe} = \frac{\delta\tau}{\delta\theta} \tag{23}$$

With the equations (20), (21) and (22), the passive tension on the ligament, the moment arm from the ligament to the joint and the resistance torque exerted by the ligament were calculated and their relations with the joint flexion angles were shown in Figure 68.

As can be seen, the tension on the ligament increased during the flexion motion, following with a slight drop after 70 °. And the moment arm keeps decreasing with the flexion angle going up, where the negative value means a positive torque which will help flex the joint, showing as the torque-flexion angle curve.

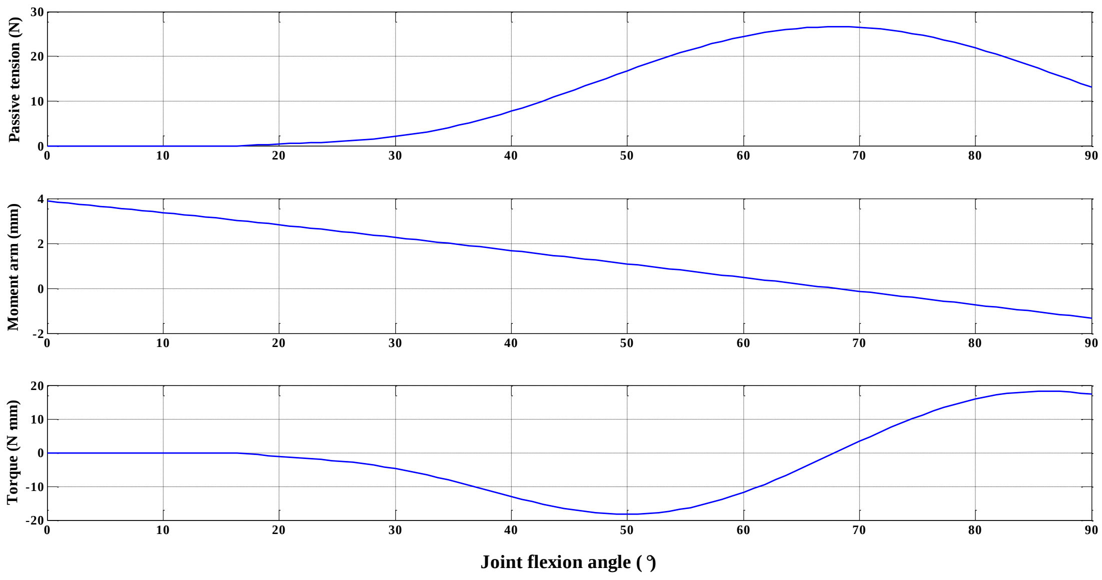


Figure 68. The one-ligament joint parameters change with flexion angles

As we can see, the torque will turn into the same direction with the flexion angle, meaning the ligament will not limit the flexion motion. This result is not in line with the actual situation where the ligament provides continuous resistance torque to the joint during the whole flexion motion. The ligament's continuous elongation can be resulted by the drift of the rotation axis caused by the joint surface shape which is not considered in the mathematical model. In addition, in the practical joint motion, there are plenty of ligament beams alternately restraining the joint during flexion. Thus, only one ligament cannot mimic the performance of the whole capsuloligamentous structure of the joint. Therefore, an additional ligament was added into the model to restore the real situation. The result was shown in Figure 69.

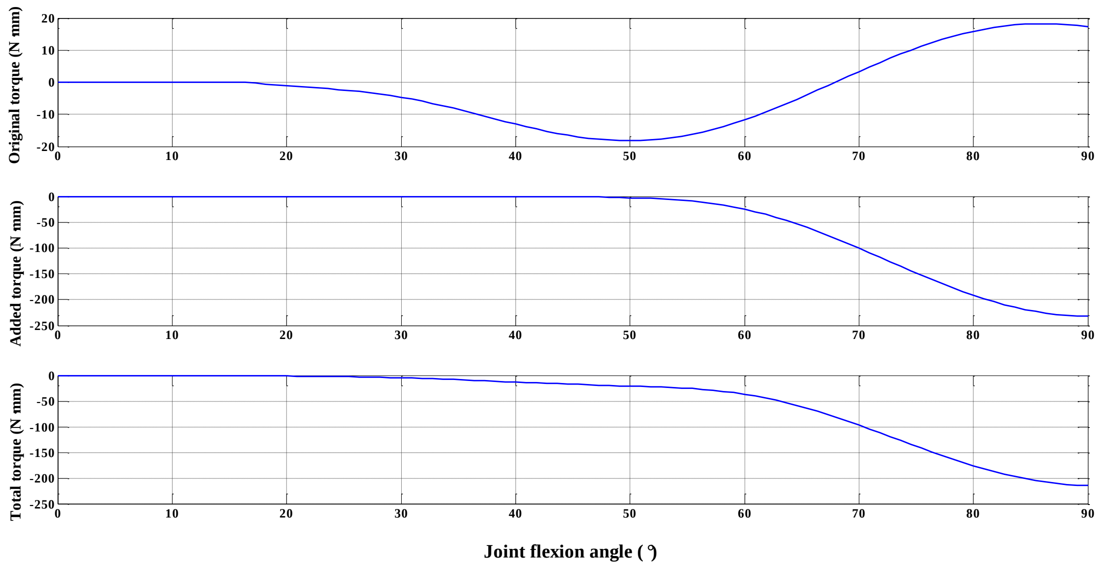


Figure 69. The joint torque change resulted by adding another ligament

As can be seen, the resistance torque generated by the ligaments remains quite low during most of the flexion motion and rapidly increases until to the 90 ° flexion angle which is slightly larger than the human MCP joint limiting flexion position. Therefore, the final result of the joint flexion stiffness was calculated with an additional ligament involved whose location depends on the specific joint design requirement. With the two ligaments identified in this research, the resultant joint flexion stiffness was shown in Figure 70. It can be seen that the flexion stiffness surges when the flexion angle θ is in the range of 70 °to 90 °.

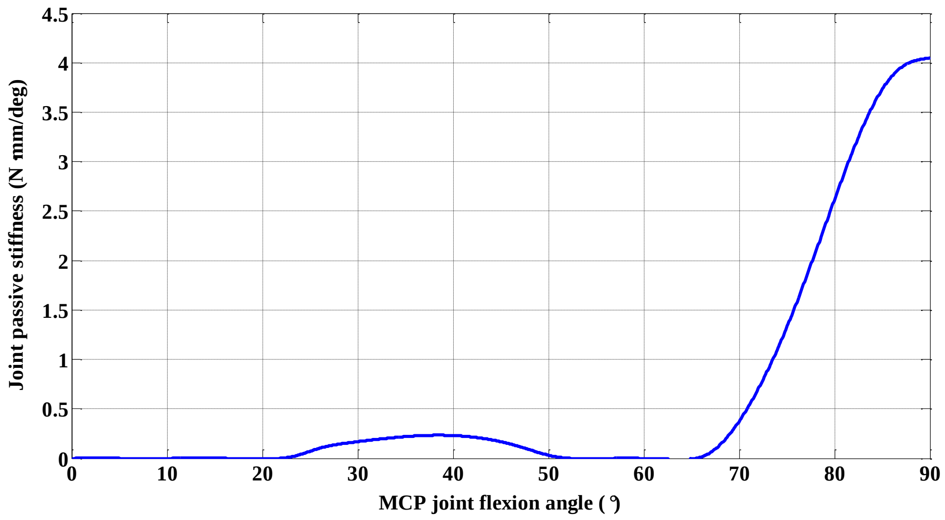


Figure 70. Theoretical relation of the joint flexion stiffness and flexion angles

Besides, as demonstrated in the geometrical analysis, the MCP joint stiffness varies with adduction-abduction motion as well. Hence, with respect to the dorsal portion of the ulnar ligament, stiffness variation in the adduction-abduction direction was investigated as follows.

To carry out the analysis, an instantaneous rotation axis of the adduction-abduction motion needs to be established in the MCP joint coordinate system. At the beginning of the flexion, this instantaneous rotation axis coincides with the *y*-axis, following with a continuous rotation around the *z*-axis during the flexion motion. Under the condition of taking the ulnar ligament as the research target, the analysis is focused on the abduction motion where the direction vector r_0 of the initial rotation axis is along the negative *y*-axis (expressed with the vector (0, 1, 0) in Figure 65.

Likewise, in the coordinate system, the ligament length can be expressed as:

$$l' = \sqrt{\left(x_i' - x_o'\right)^2 + \left(y_i' - y_o'\right)^2 + \left(z_i' - z_o'\right)^2}$$

(24)

where (x_i', y_i', z_i') and (x_o', y_o', z_o') are the coordinate of the ligament's insertion and origin. To obtain the ligament length, the positions of the insertion during the whole abduction motion are needed.

In the condition that the joint's flexion angle is θ, then the abduction rotation axis $\boldsymbol{r}$ should be:

$$\boldsymbol{r} = \mathbf{R}_z\left(\theta\right)\boldsymbol{r}_0 = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ -1 \\ 0 \end{bmatrix} = \begin{bmatrix} \sin\theta \\ -\cos\theta \\ 0 \end{bmatrix}$$

(25)

Where $\mathbf{R}_z(\theta)$ is the rotation matrix, and $\boldsymbol{r}_0$ is the initial abduction rotation axis.

Assuming that the abduction angle is θ', then the position of the insertion after the flexion and abduction motion can be expressed:

$$\begin{bmatrix} x_{i'fa} \\ y_{i'fa} \\ z_{i'fa} \end{bmatrix} = \mathbf{R}_r(\theta') \cdot \begin{bmatrix} x_{i'} \\ y_{i'} \\ z_{i'} \end{bmatrix}$$

(26)

And $\mathbf{R}_r$ is the Rodrigues rotation matrix：

$$\mathbf{R}_r = \mathbf{I} + \left(1 - \cos\left(\theta'\right)\right) \cdot \mathbf{K}^2 + \sin\left(\theta'\right) \cdot \mathbf{K}$$

(27)

Of which **I** is the standard identity matrix and **K** is the cross product matrix of the abduction rotation axis ***r***.

Therefore, the ligament length change during the whole abduction motion can be calculated and shown in Figure 71.

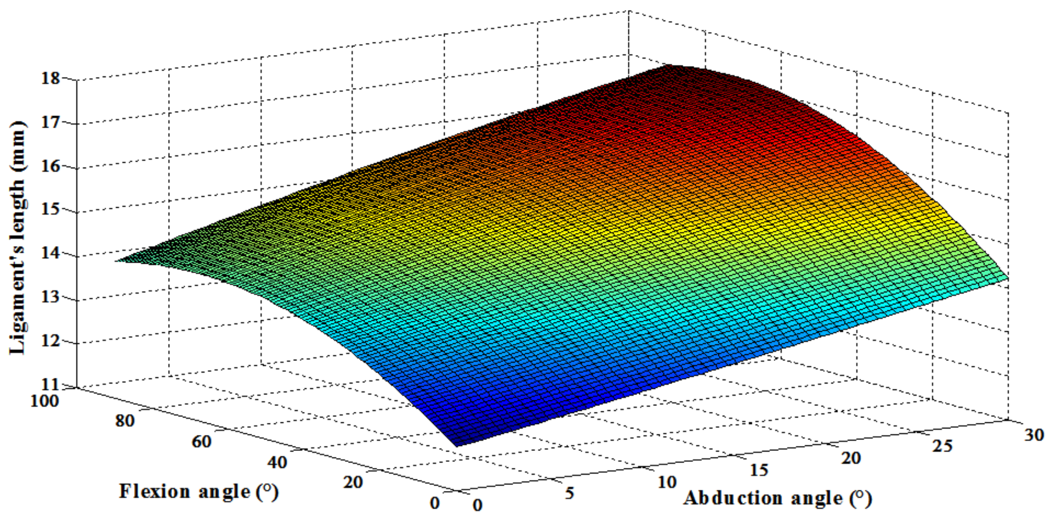


Figure 71. The ligament length change with the flexion and abduction angle in MCP joint

As can be seen, the ligament's length increases with both the flexion and abduction angles increasing.

The moment arm of ligament to the joint in the abduction-adduction motion is in coronal plane of the proximal phalanx which is the plane after a θ rotation around Z axis of the X-Z plane in the MCP joint coordinate system. Since a plane in a 3D coordinate system can be expressed as $Ax+By+Cz+D=0$, we have the abduction-adduction plane as:

$$x\tan\theta - y = 0$$

(28)

Where A = tanθ, $B = -1$, $C = D = 0$. In this plane, the projection of the rotation axis is still the point (0, 0, 0), and the ligament's origin (x_o', y_o', z_o') and insertion (x_i', y_i', z_i') are projected to the point (x_{op}', y_{op}', z_{op}') and (x_{ip}', y_{ip}', z_{ip}'). According to point-to-surface projection formula in space, we have:

$$\begin{aligned}
x_{op}' &= \frac{(B^2+C^2)x_o' - A(By_o' + Cz_o' + D)}{A^2+B^2+C^2} = \frac{x_o' + y_o'\tan\theta}{\tan^2\theta + 1} \\
y_{op}' &= \frac{(A^2+C^2)y_o' - B(Ax_o' + Cz_o' + D)}{A^2+B^2+C^2} = \frac{y_o'\tan^2\theta + x_o'\tan\theta}{\tan^2\theta + 1} \\
z_{op}' &= \frac{(A^2+B^2)z_o' - C(Ax_o' + By_o' + D)}{A^2+B^2+C^2} = z_o'
\end{aligned}$$

(29)

And the same with the projection of the point (x_{ip}', y_{ip}', z_{ip}'). Then, the moment arm M' should be the distance from the point (0, 0, 0) to the line L which is through the point (x_{op}', y_{op}', z_{op}') and (x_{ip}', y_{ip}', z_{ip}'). The line L can be expressed in a two-point form:

$$\frac{x - x_{op}'}{x_{ip}' - x_{op}'} = \frac{y - y_{op}'}{y_{ip}' - y_{op}'} = \frac{z - z_{op}'}{z_{ip}' - z_{op}'}$$

(30)

Then the vertical coordinate (x_v, y_v, z_v) from point (0, 0, 0) to the line L can be obtained:

$$\begin{aligned}
x_v &= (x_{ip}' - x_{op}')t + x_{op}' \\
y_v &= (y_{ip}' - y_{op}')t + y_{op}' \\
z_v &= (z_{ip}' - z_{op}')t + z_{op}'
\end{aligned}$$

(31)

Where $t = \dfrac{-(x_{ip}' - x_{op}')x_{op}' - (y_{ip}' - y_{op}')y_{op}' - (z_{ip}' - z_{op}')z_{op}'}{(x_{ip}' - x_{op}')^2 + (y_{ip}' - y_{op}')^2 + (z_{ip}' - z_{op}')^2}$,

Thus the moment arm *M'* should be the distance from the point (0, 0, 0) to the point (x_v, y_v, z_v), which is:

$$M' = \sqrt{x_v^{\,2} + y_v^{\,2} + z_v^{\,2}} \tag{32}$$

Likewise, the passive tension *F'* on the ligament can be expressed as:

$$F' = F_0 + k\left(\sqrt{\left(x_i' - x_o'\right)^2 + \left(y_i' - y_o'\right)^2 + \left(z_i' - z_o'\right)^2} - \sqrt{\left(x_e' - x_o'\right)^2 + \left(y_e' - y_o'\right)^2 + \left(z_e' - z_o'\right)^2}\right) \tag{33}$$

Where *k* is the elastic coefficient, F_0 is the initial tension of the linear region, Δl_0 is the ligament's length change in the linear region, and (x_e', y_e', z_e') is the location of the ligament's insertion when the ligament starts the elastic deformation.

Since the resistance torque τ' in the abduction-adduction motion is generated by the component of the ligament's tension force on the abduction-adduction plane, it has:

$$\tau' = -F' \cos\eta \cdot M' \tag{34}$$

Where

$$\cos\eta = \frac{(x_i' - x_o')(x_{ip}' - x_{op}') + (y_i' - y_o')(y_{ip}' - y_{op}') + (z_i' - z_o')(z_{ip}' - z_{op}')}{\sqrt{(x_i' - x_o')^2 + (y_i' - y_o')^2 + (z_i' - z_o')^2}\sqrt{(x_{ip}' - x_{op}')^2 + (y_{ip}' - y_{op}')^2 + (z_{ip}' - z_{op}')^2}}$$

(35)

Of which η is the angle between the ligament and the abduction-adduction plane. The negative sign in formula (B20) means the torque impedes the flexion motion. Then the joint stiffness contributed by the dorsal portion of the ulnar ligament can be obtained with the function:

$$K_{aa} = \frac{\delta\tau'}{\delta\theta'}$$

(36)

The joint abduction stiffness is shown in Figure 72. It can be seen that there is an approximate boundary formed by some black points (interpreted in Chapter 5) on the curved surface. Assuming that the surface within the boundary indicates the toe and linear deformation region, and the surface beyond this boundary represents the plastic deformation which is not in the scope of the analysis. As shown in the figure, within the boundary of the surface (marked by shadow in the range of 0 °~ 90 °flexion angles and 0 °~ 40 °abduction angles), the joint abduction stiffness increases with the joint angle increasing in both the flexion-extension and abduction-adduction direction. And if only the joint stiffness in small abduction angles is considered, which is actually the normal case, a more obvious result can be found that the joint abduction stiffness increases with the flexion angle growing. This result is also consistent with the cadaver test in reference [6].

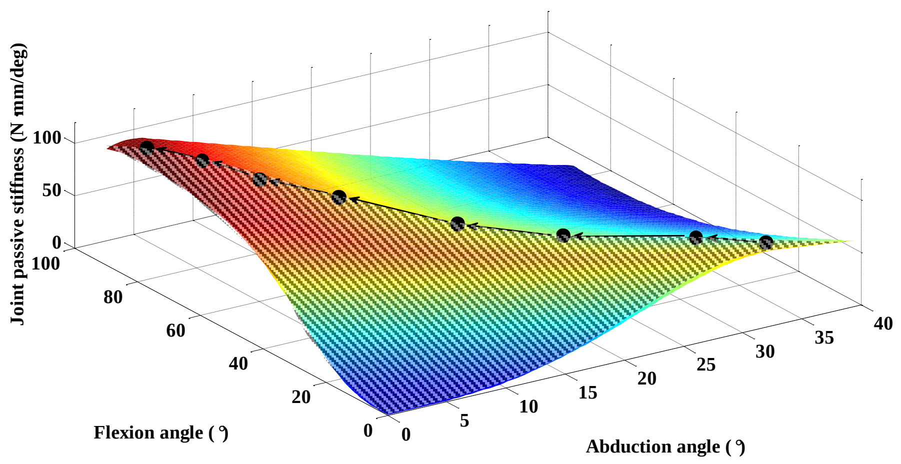


Figure 72. Theoretical relations of the joint abduction stiffness and the flexion/abduction angle

4.2 Feasible force space associated with the extensor mechanism

According to the mathematical model proposed by Valero-Cuevas [10], the kinematic model of an index finger with 4 rotational DOFs is established in Figure 73. In this analysis, without loss of generality, the DIP and PIP joints are considered as revolute joints and the MCP joint is treated a 2-DOF universal joint.

To obtain the feasible force space at the endpoint, a mapping from the muscular actions to the joints and then to the endpoint mechanical outputs needs to be defined. To formulate the mapping, an inertial reference coordinate system *O-XYZ* is established with the origin located at the geometrical centre of the metacarpal bone's head. In addition, a local coordinate frame is attached at the fingertip. In the model in Figure 73, lengths of the three links are l_1, l_2 and l_3, respectively. The joint angles are organized as a vector $\boldsymbol{q} = (q_1, q_2, q_3, q_4)$ and the endpoint posture is represented by a vector $\boldsymbol{p} = (x, y, z, \alpha)$. The rotational kinetic inputs are the four net joint torques (τ_1, τ_2, τ_3, τ_4) that produce the endpoint output force/torque as (f_x, f_y, f_z, τ_z).

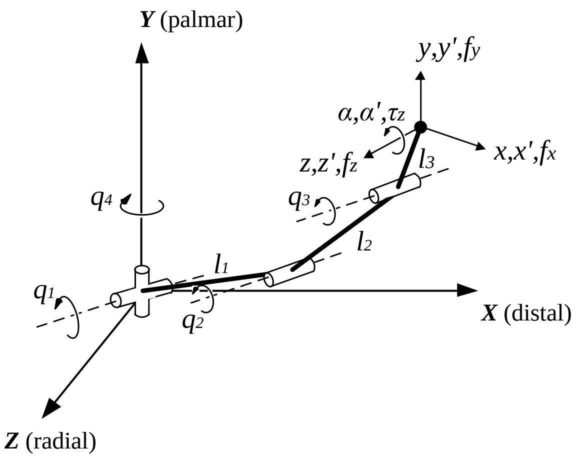


Figure 73. The kinematic model of the robotic index finger.

Based on the mathematical model in Figure 73, the forward kinematic model of the robotic finger is:

$$\begin{pmatrix} x \\ y \\ z \\ \alpha \end{pmatrix} = \begin{pmatrix} \left[l_1 \cdot \cos q_1 + l_2 \cdot \cos\left(q_1 + q_2\right) + l_3 \cdot \cos\left(q_1 + q_2 + q_3\right)\right] \cdot \cos q_4 \\ l_1 \cdot \sin q_1 + l_2 \cdot \sin\left(q_1 + q_2\right) + l_3 \cdot \sin\left(q_1 + q_2 + q_3\right) \\ \left[l_1 \cdot \cos q_1 + l_2 \cdot \cos\left(q_1 + q_2\right) + l_3 \cdot \cos\left(q_1 + q_2 + q_3\right)\right] \cdot \sin q_4 \\ q_1 + q_2 + q_3 \end{pmatrix} \tag{37}$$

And its Jacobian is shown as below with $\sin_1$ meaning $\sin q_1$.

$$J(q) = \begin{pmatrix} \left(-l_1 \sin_1 - l_2 \sin_{12} - l_3 \sin_{123}\right)\cos_4 & \left(-l_2 \sin_{12} - l_3 \sin_{123}\right)\cos_4 & \left(-l_3 \sin_{123}\right)\cos_4 & -\left(l_1 \cos_1 + l_2 \cos_{12} + l_3 \cos_{123}\right)\sin_4 \\ l_1 \cos_1 + l_2 \cos_{12} + l_3 \cos_{123} & l_2 \cos_{12} + l_3 \cos_{123} & l_3 \cos_{123} & 0 \\ \left(-l_1 \sin_1 - l_2 \sin_{12} - l_3 \sin_{123}\right)\sin_4 & \left(-l_2 \sin_{12} - l_3 \sin_{123}\right)\sin_4 & \left(-l_3 \sin_{123}\right)\sin_4 & \left(l_1 \cos_1 + l_2 \cos_{12} + l_3 \cos_{123}\right)\cos_4 \\ 1 & 1 & 1 & 0 \end{pmatrix} \tag{38}$$

Using the kinematics and based on the principle of virtual work, we can map the joint torques $\boldsymbol{\tau}$ to the static endpoint force $\boldsymbol{w}$ as

$$\boldsymbol{w} = \mathbf{J}\left(\boldsymbol{q}\right)^{-T} \boldsymbol{\tau}$$

(39)

where $\boldsymbol{\tau} = (\tau_1, \tau_2, \tau_3, \tau_4)$, $\boldsymbol{w} = (f_x, f_y, f_z, \tau_z)$, and $\mathbf{J}(\boldsymbol{q})$ is the Jacobian matrix.

In the proposed robotic finger, five motors are used to drive the four rotational DOFs, the tendon maximum forces are set to be (F_1, F_2, F_3, F_4, F_5) with the activation levels being (a_1, a_2, a_3, a_4, a_5), respectively corresponding to RI, UI, FDP, LE and FDS muscles. Based on these, the joint torques can alternatively be derived from the actuation forces and the moment arms as [10]

$$\boldsymbol{\tau} = \mathbf{R}(\boldsymbol{q})\boldsymbol{f} = \mathbf{R}(\boldsymbol{q})\mathbf{F}\boldsymbol{a} = \begin{pmatrix} r_{11} & r_{12} & r_{13} & r_{14} & r_{15} \\ r_{21} & r_{22} & r_{23} & r_{24} & r_{25} \\ r_{31} & r_{32} & r_{33} & r_{34} & r_{35} \\ r_{41} & r_{42} & r_{43} & r_{44} & r_{45} \end{pmatrix} \begin{pmatrix} F_1 & 0 & 0 & 0 & 0 \\ 0 & F_2 & 0 & 0 & 0 \\ 0 & 0 & F_3 & 0 & 0 \\ 0 & 0 & 0 & F_4 & 0 \\ 0 & 0 & 0 & 0 & F_5 \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \end{pmatrix}$$

(40)

where $0 \le a_i \le 1$, for $i = 1, 2, 3, 4$ and 5. Equation (40) presents the mapping from the tendon forces to the joint torques. In the equation, $\mathbf{R}(\boldsymbol{q})$ is a 4×5 matrix giving the moment arms, where each entry is the signed scalar moment arm value that transforms the positive tendon force into the torques at the corresponding joints.

Combining Eq. (39) and Eq. (40), the mapping from the input muscle force space to the endpoint output force space can be expressed as

$$\begin{pmatrix} \boldsymbol{f}_x \\ \boldsymbol{f}_y \\ \boldsymbol{f}_z \\ \boldsymbol{\tau}_z \end{pmatrix} = \boldsymbol{w} = \mathbf{J}(\boldsymbol{q})^{-T}\mathbf{R}(\boldsymbol{q})\mathbf{F}\boldsymbol{a} = \begin{bmatrix} \boldsymbol{w}_1 & \boldsymbol{w}_2 & \boldsymbol{w}_3 & \boldsymbol{w}_4 & \boldsymbol{w}_5 \end{bmatrix} \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \end{pmatrix}$$

(41)

Where, the 4 × 1 column vectors $\boldsymbol{w}_i$ calculated from matrix $\mathbf{J}^{-T}(\boldsymbol{q})\mathbf{R}(\boldsymbol{q})\mathbf{F}$ are the generators of the Minkowski sum [147].

Therefore, all the possible forces that the motor-tendon system can produce at the fingertip in a particular posture will be described in a zonotope called feasible wrench set. If we assume the output torque τ_z as zero, all the output will be forces, and thus the feasible wrench set can be the feasible force set of the fingertip. In addition, in the Eq. (41), $\mathbf{J}(\boldsymbol{q})^{-T}$ can be calculated from the geometry and the posture of the finger, and $\mathbf{R}(\boldsymbol{q})$ can be obtained through the experiment for testing the relationship between each joint angle change δq and each tendon excursion δs. Besides, $\mathbf{F}$ can directly refer to the maximum output torque that the motors can generate.

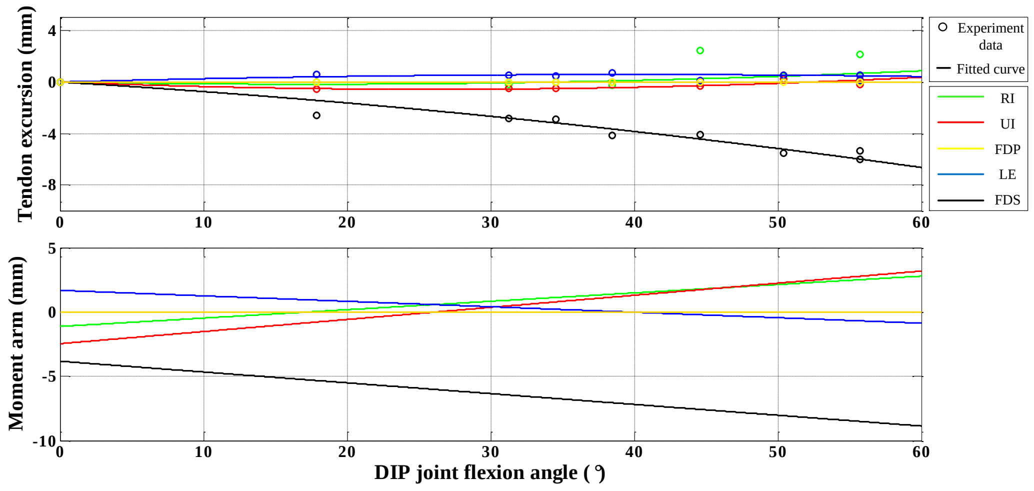
As an example, a standard finger posture (with DIP at 10°, PIP at 45°, MCP flexion at 45° and abduction at 0°) is selected to find the feasible force set [8] [23]. Herein, the structure parameters for the Jacobian matrix of the robotic finger are specified as l_1 = 0.0455 m, l_2 = 0.0258 m, l_3 = 0.0169 m, $q_1 = \pi/4$, $q_2 = \pi/4$, $q_3 = \pi/18$ and $q_4 = 0$, and the calculated result is

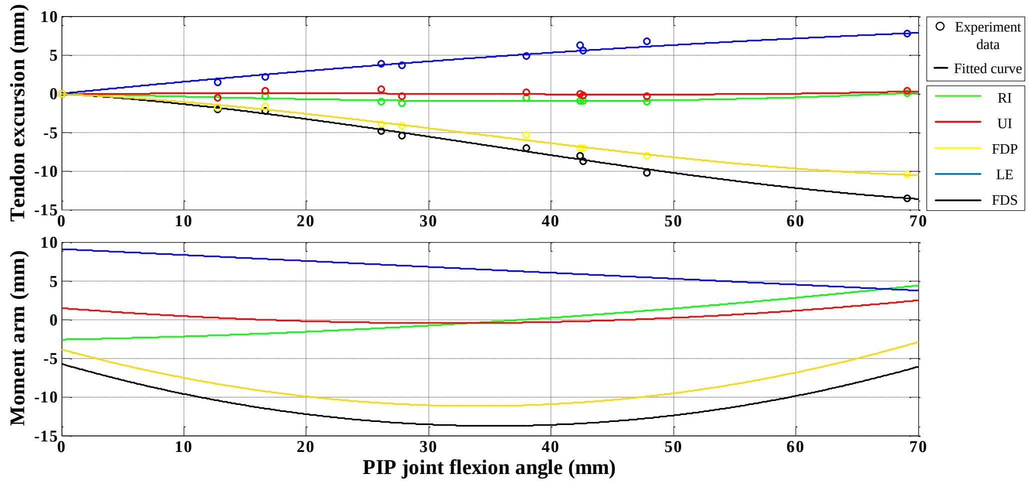
$$\mathbf{J}(\boldsymbol{q})^{-T} = \begin{pmatrix} 1.78\times10^{-15} & -38.760 & 38.760 & 0 \\ 31.153 & -70.033 & 38.880 & 0 \\ 0 & 0 & 0 & 34.247 \\ 0.090 & -0.847 & 1.756 & 0 \end{pmatrix} \tag{42}$$

Further, in order to accurately define the moment arm matrix $\mathbf{R}(\boldsymbol{q})$, we tested the relationship between the tendon excursion and the joint angle.

A testbed was constructed for obtaining the relationship between the tendon excursion and the joint angle. The robotic finger was mounted on the testbed with its flexion-extension plane perpendicular to the floor. At the end of each muscle tendon, there is a 500g load fixed through a pulley to straighten the tendon and keep it under constant tension during the whole joint motion. The test was conducted by passively move each finger joint individually with the other joints maintaining at the neutral position. At each step of the test, the joint was moved by a small angle. The joint angle and the excursion of each tendon (the descent distance of each load) were captured by a camera and obtained through the processing of the software ImageJ.

Then the experiment data of the tendon excursion was curve fitted by 3-order polynomial function and the result of the corresponding moment arm was obtained from the derivative of the above fitted curve, as presented in Figure 74.





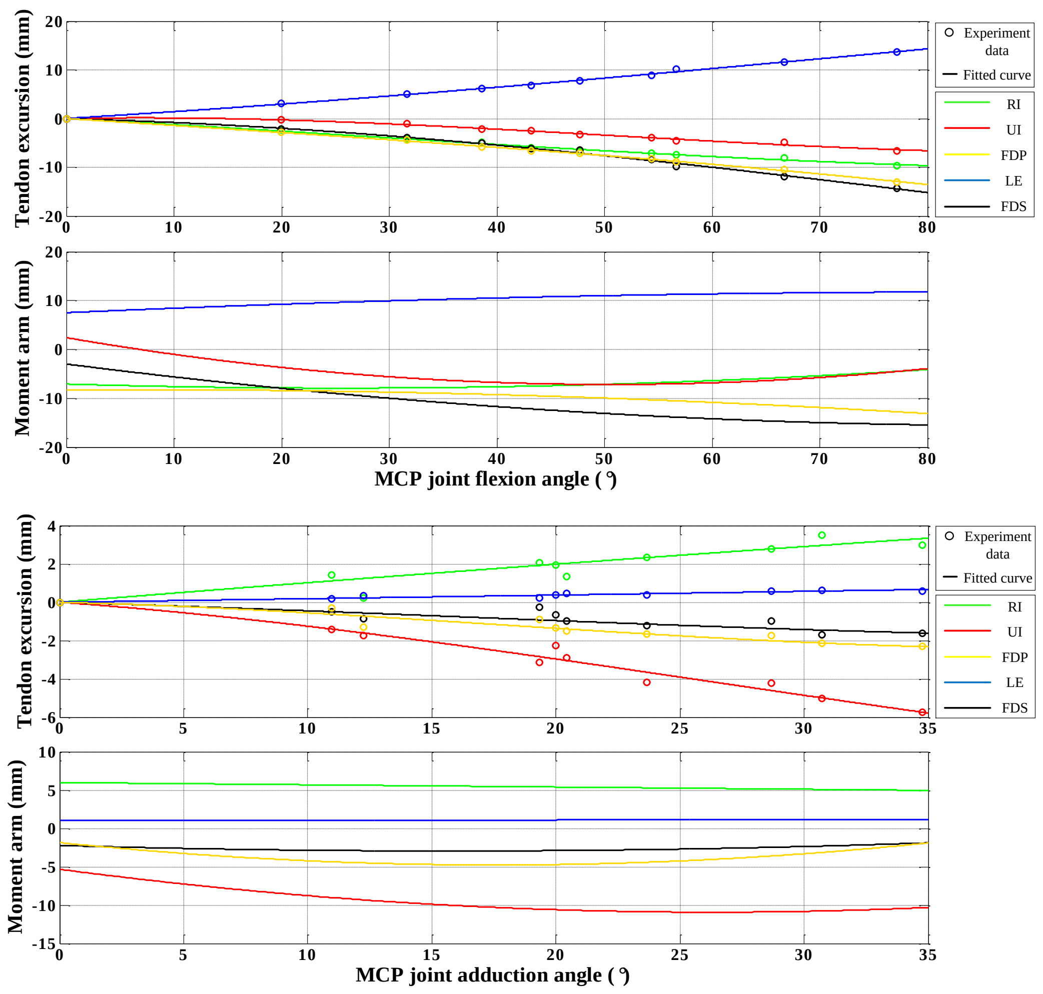


Figure 74. The relationships of the tendon excursion-joint motion angle and the calculated moment arm-joint motion angle

According to the coordinate definition, the flexion and adduction angle were set to be positive, thus the moment arm of each tendon to the finger joint was listed in Table 5, with the minus sign meaning the muscles contribute the negative torques to the joint angle increasing.

Table 5. The moment arm of each muscle tendon to each joint in the standard posture (mm)

Joint angle (°) / Tendon	DIP 10 °	PIP 45 °	MCP FL 45 °	MCP ADD 0 °

RI	0.47	-0.42	7.42	5.98
UI	1.52	0.23	7.04	-5.32
FDP	4.67	10.93	12.42	-2.22
LE	-1.26	-5.84	-10.79	1.05
FDS	0	8.79	9.57	-1.84

Since the moment arm matrix $\mathbf{R}(\boldsymbol{q})$ in this case can be expressed as:

$$\mathbf{R}(\boldsymbol{q}) = \begin{pmatrix} r_{11} & r_{12} & r_{13} & r_{14} & r_{15} \\ r_{21} & r_{22} & r_{23} & r_{24} & r_{25} \\ r_{31} & r_{32} & r_{33} & r_{34} & r_{35} \\ r_{41} & r_{42} & r_{43} & r_{44} & r_{45} \end{pmatrix} \tag{43}$$

Where r_{mn} represents the moment arm of the tendon n to the joint m. In the previouly defined coordinate system, the MCP, PIP, DIP flexion-extension motion joint and the MCP abduction-adduction motion joint were respectively defined as the joint 1, 2, 3, 4. And the muscle tendons RI, UI, FDP, LE and FDS were respectively defined as the tendon 1, 2, 3, 4, 5. Thus the moment arm matrix $\mathbf{R}(\boldsymbol{q})$ can be directly derived from Table 5 with only converting the unit from mm to m.

$$\mathbf{R}(\boldsymbol{q}) = \begin{pmatrix} 0.00742 & 0.00704 & 0.01242 & -0.01079 & 0.00957 \\ -0.00042 & 0.00023 & 0.01093 & -0.00584 & 0.00879 \\ 0.00047 & 0.00152 & 0.00467 & -0.00126 & 0 \\ 0.00598 & -0.00532 & -0.00222 & 0.00105 & -0.00184 \end{pmatrix} \tag{44}$$

Consequently, considering the maximum torque of the actual motors used in the physical prototype, we set the maximum tendon force as 9 N.

the endpoint output force can be expressed as:

$$\boldsymbol{w} = \mathbf{J}(\boldsymbol{q})^{-T}\,\mathbf{R}(\boldsymbol{q})\mathbf{F}\boldsymbol{a} \tag{45}$$

Where **F** is the muscle force matrix, and ***a*** represents the activation levels of the five muscles. It has:

$$\mathbf{F} = \begin{pmatrix} 9 & 0 & 0 & 0 & 0 \\ 0 & 9 & 0 & 0 & 0 \\ 0 & 0 & 9 & 0 & 0 \\ 0 & 0 & 0 & 9 & 0 \\ 0 & 0 & 0 & 0 & 9 \end{pmatrix}, \qquad \boldsymbol{a} = \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \end{pmatrix} \tag{46}$$

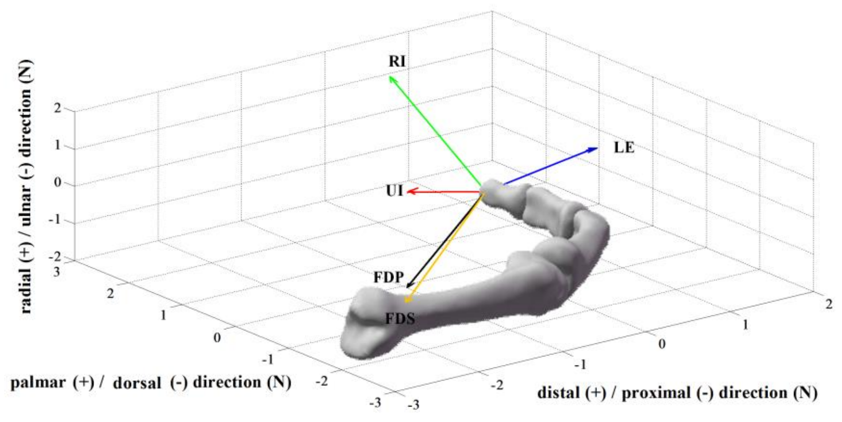
With the equations (42), (44), (45) and (46), we have:

$$\begin{pmatrix} \boldsymbol{f}_x \\ \boldsymbol{f}_y \\ \boldsymbol{f}_z \\ \boldsymbol{\tau}_z \end{pmatrix} = \begin{bmatrix} \boldsymbol{w}_1 & \boldsymbol{w}_2 & \boldsymbol{w}_3 & \boldsymbol{w}_4 & \boldsymbol{w}_5 \end{bmatrix} \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \end{pmatrix} = \begin{pmatrix} 0.3105 & 0.4500 & -2.1837 & 1.5977 & -3.0663 \\ 2.5036 & 2.3550 & -1.7669 & 0.2130 & -2.8481 \\ 1.8407 & -1.6376 & -0.6833 & 0.3232 & -0.5664 \\ 0.0167 & 0.0281 & 0.0005 & 0.0159 & -0.0594 \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \end{pmatrix} \tag{47}$$

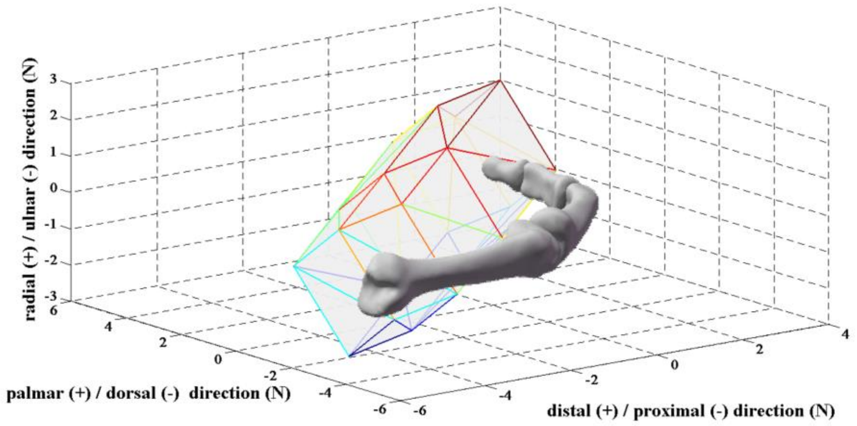
of which the column vectors $\boldsymbol{w}_i$ are the generators of the Minkowski sum for the feasible force set. And each generator $\boldsymbol{w}_i$ is the force vector that the muscle *i* produces in space. To clearly demonstrate the 3D force capabilities of the endpoint, we can enforce the constraint on the endpoint torque $\boldsymbol{\tau}_z$ to be zero. Then the new generators $\boldsymbol{w}_i'$ are:

$$w_1' = \begin{pmatrix} 0.3105 \\ 2.5036 \\ 1.8407 \end{pmatrix},\ w_2' = \begin{pmatrix} 0.4500 \\ 2.3550 \\ -1.6376 \end{pmatrix},\ w_3' = \begin{pmatrix} -2.1837 \\ -1.7669 \\ -0.6833 \end{pmatrix},\ w_4' = \begin{pmatrix} 1.5977 \\ 0.2130 \\ 0.3232 \end{pmatrix},\ w_5' = \begin{pmatrix} -3.0663 \\ -2.8481 \\ -0.5664 \end{pmatrix} \tag{48}$$

The resultant feasible force set at the specified posture is illustrated in Figure 75 (a). Further, using the same procedures, the endpoint feasible output forces were computed and simulated; the feasible force space for the index finger is illustrated in Figure 75 (b).



(a)



(b)

Figure 75. Theoretical feasible force set and space of the robotic finger. (a) Feasible force set calculated from robotic finger’s mathematical model. (b) Feasible force space generated from the feasible force set of the robotic finger’s mathematical model.

4.3 Force-velocity characteristics associated with the flexible tendon sheath

The elastic tendon sheaths can flatten down to hold the tendons close to the phalanges when the finger straightens, resulting in small initial moment arms at the joints which allows a fast bending motion but small flexion torques. Meanwhile, they can also bulge out to allow the tendons leaving a distance from the phalanges when the finger bends, increasing the moment arms at the joints to generate large flexion torques but low bending speed [41]. This flexible tendon sheaths system shows similar force-velocity self-adaptation function to the elastomeric passive transmissions in Kevin’s work [38]. We believe that this is another important advantage of the human finger. In this section, a simplified flexible tendon sheath model is established to explain its impact on the force-velocity characteristics of a finger.

4.3.1 Static characteristics analysis

Both quasi-static and dynamic cases are investigated in this section. Here, the quasi-static force and velocity characteristics describe the maximal endpoint force and velocity that the finger can achieve. For one pair of quasi-static force-velocity values, they respectively correspond to the force extremum and the velocity extremum on the entire working curve, normally obtained from the two end points of the curve. The reason we call them the quasi-static force and velocity characteristics is that the endpoint velocity can be extremely low (nearly static) when outputting the maximal force, and the endpoint force can be approximate to zero (nearly no resistance) when outputting the maximal velocity. While the dynamic force-velocity characteristics are the practical working points on the curve representing the endpoint force and velocity that the mechanism can simultaneously output at work. In fact, the working curve itself is a graphical description of the mechanism dynamic force-velocity characteristics.

Both of the quasi-static and dynamic characteristics can show the system properties from different aspects [38] [148] [149]. Therefore, both quasi-static and dynamic mathematical models for characterizing the impact of flexible tendon sheath on the force-velocity characteristics are presented as follows.

For the quasi-static analysis, considering one finger joint as shown in Figure 76, in which a tendon is held by two elastic pulleys on each phalanx. The corresponding mathematical model is established, where in the model θ is the joint flexion angle, ω is the joint flexion velocity, τ is the joint torque, l is the distance from the pulley edge to the rotation axis, d is the distance between the tendon and the phalanx, h is the distance from the tendon to the rotation axis which equals to the moment arm M, l_0 is the distance from the joint centre to the endpoint, F is the tendon force, and v is the tendon excursion velocity.

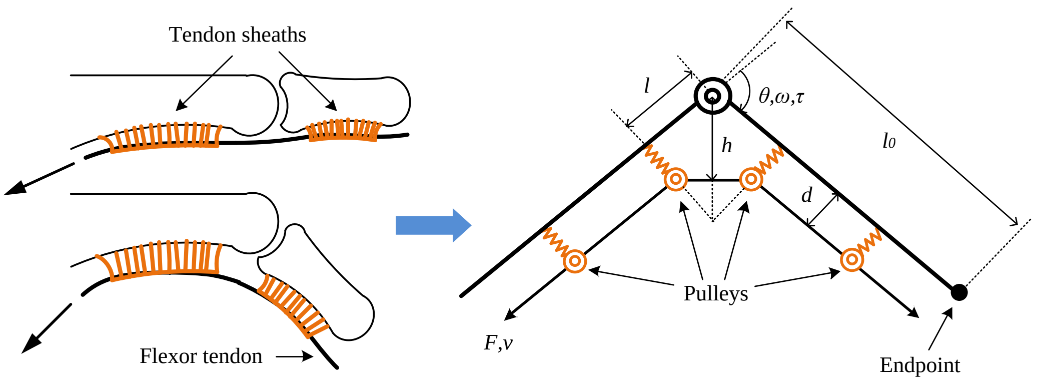


Figure 76. Schematic diagram of the single-joint elastic tendon sheath pulley system (for quasi-static analysis).

Referring to Figure 68, the moment arm can be expressed as

$$M = h = \frac{l}{\sin\left(\frac{\theta}{2}\right)} - \left(\frac{l}{\tan\left(\frac{\theta}{2}\right)} - d\right) \cdot \cos\left(\frac{\theta}{2}\right) \tag{49}$$

Given that the elastic coefficient of the pulley spring is *k*, and the initial length of the spring is d_0, the relationship between the tendon force *F* and the distance *d* can be expressed as

$$d = d_0 + \frac{F \cdot \sin\left(\frac{\theta}{2}\right)}{k} \tag{50}$$

where *d* must satisfy that $d \le \frac{l}{\tan\left(\frac{\theta}{2}\right)}$.

Then, the moment arm *M* can further be formulated as

$$\begin{aligned} M = h &= \frac{l}{\sin\left(\frac{\theta}{2}\right)} - \left(\frac{l}{\tan\left(\frac{\theta}{2}\right)} - d_0 - \frac{F \cdot \sin\left(\frac{\theta}{2}\right)}{k} \right) \cdot \cos\left(\frac{\theta}{2}\right) \\ &= l \cdot \sin\left(\frac{\theta}{2}\right) + d_0 \cdot \cos\left(\frac{\theta}{2}\right) + \frac{F}{k} \cdot \sin\left(\frac{\theta}{2}\right) \cos\left(\frac{\theta}{2}\right) \end{aligned} \tag{51}$$

where *l*, d_0 and *k* are all constants during the flexion motion, which are only related to the anatomical structure of the finger.

It can be seen from Eq. (51) that the moment arm increases as the tendon force and the joint's flexion angle increase. However, with a rigid tendon sheath, the tendon force won't change the moment arm for the reason that in this case *k* is great and thus the last term in Eq. (51) is neglectable. Besides, there is an assumption that the moment arm *M* is no less than the initial length of the spring d_0, with the minimum value d_0 occurring when the finger straightens ($\theta = \pi$).

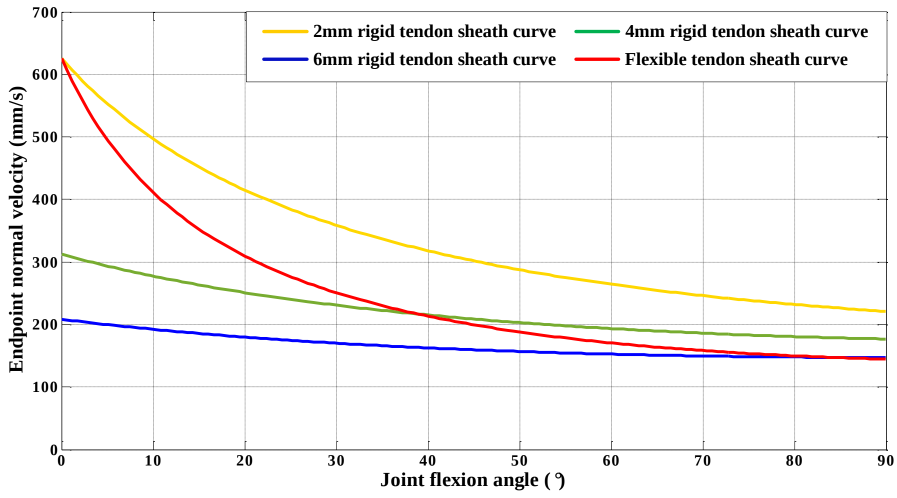
With the moment arm, the joint angular velocity ω and torque τ can be expressed as

$$\omega = \frac{v}{M} \text{ and, } \tau = FM \tag{52}$$

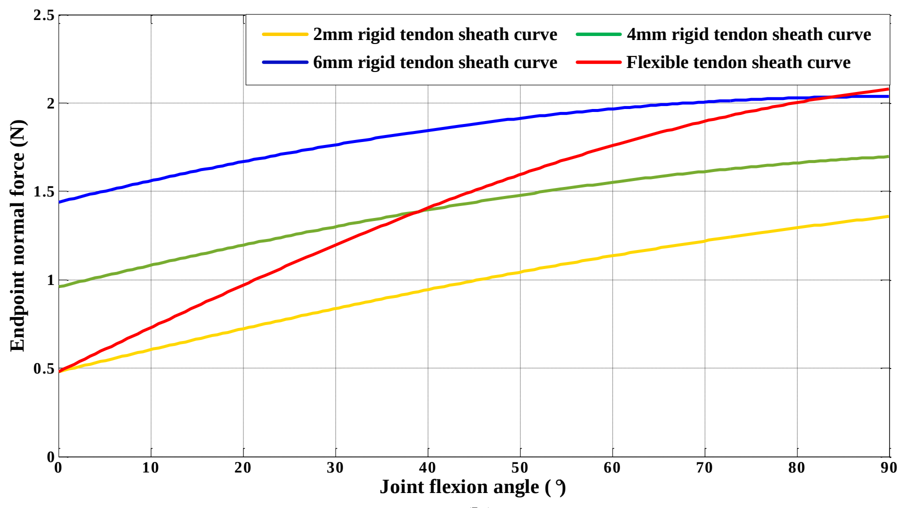
Using Eq. (52), the endpoint normal velocity v_n and force F_n can be obtained as

$$\begin{cases} v_n = \omega l_0 = \dfrac{v}{M} l_0 \\ F_n = \dfrac{\tau}{l_0} = \dfrac{FM}{l_0} \end{cases} \tag{53}$$

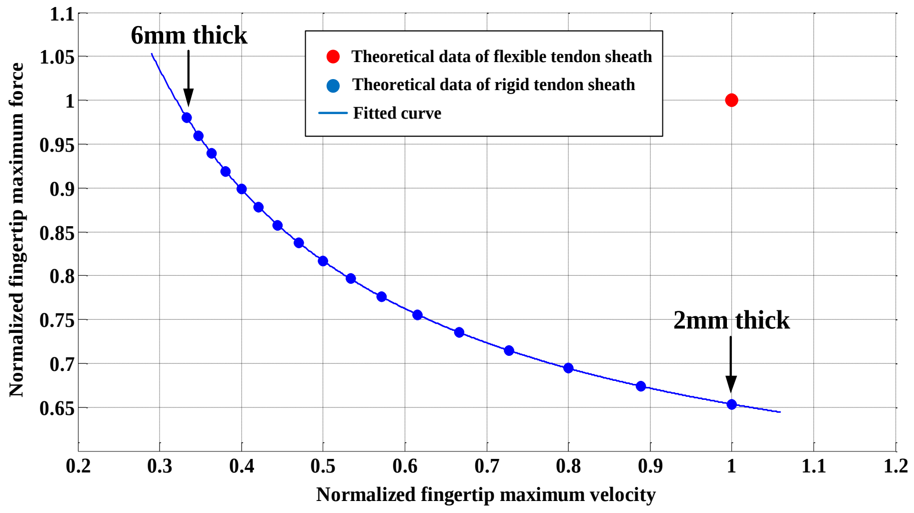
In the robotic finger with the flexible tendon sheath, the parameters are set as l=4mm, l_0 = 25 mm, d_0 = 2 mm, k = 0.8 N/mm, F = 6 N and v = 50 mm/s. With Eqs. (51) and (53), the relationships between the endpoint velocity and flexion angle as well as the endpoint force and flexion angle can be obtained. In addition, for the rigid tendon sheath, the corresponding relationships can be also derived with the same equations and parameters by only changing the thickness d_0.



(a)



(b)



(c)

Figure 77. Theoretical quasi-static force-velocity characteristics. (a) The quasi-static relationship between the endpoint normal velocity and the joint flexion angle. (b) The quasi-static relationship between the endpoint normal force and the joint flexion angle. (c) The normalized results of the theoretical data about the quasi-static maximum force and velocity that different finger models can output.

The simulation results are illustrated in Figure 77 (a) and (b). As can be seen in the figure, with the flexible tendon sheath, the finger can achieve high endpoint velocity when in the small flexion angles (Figure 77 (a)) as well as large endpoint forces near the full flexion positions (Figure 77 (b)). However, the finger with the rigid tendon sheaths can only achieve one of the situations - the thicker tendon sheath, the larger endpoint force but the lower endpoint velocity. Further, for each tendon sheath, the maximal values of the endpoint force and velocity are calculated and marked as one circle point as shown in Figure 77 (c), with the velocity as x-value and the force as y-value. The results of more rigid tendon sheaths with 2 mm to 6 mm thickness are added and the whole theoretical curve is inserted to demonstrate the tendency. To make a clear comparison, all the values in this coordinate are normalized with respect to the coordinate value of the flexible tendon sheath.

As can be seen in Figure 77 (c), a rigid thin tendon sheath allows high endpoint velocity but poor force-generating capability. Conversely, if the thickness of the tendon sheath is great, it can produce high endpoint force but low velocity since the tendon cannot be held close to the phalanx even at the initial phase of the finger flexion. A flexible tendon sheath is a potential solution for improving the force-velocity property of the finger. As the red point shown in Figure 77 (c), it can theoretically reach both high velocity and large force when needed. The result is consistent with the Kevin's [38] research, illustrating that the flexible tendon sheaths in the finger transmission system can perform similar force-velocity self-adjustment function to the elastomeric passive transmissions pulley he designed. The difference is the implementation of this function in human and the proposed robotic finger do not need extra components added in the transmission system, but it can be realized by its own structure – the flexible tendon sheaths. Besides, as mentioned in the introduction, they only studied the quasi-static force-velocity output properties.

4.3.2 Dynamic characteristics analysis

Next, the dynamic force-velocity characteristic affected by different tendon sheaths is investigated. To obtain the dynamic characteristics of the endpoint force-velocity output, a quick-release test is constructed and simulated in theoretical model in this section, the approach is commonly used to investigate the force-velocity relation of muscles contraction or bio-inspired actuators [34]. The schematic diagram of the quick-release experiment on single-joint model is shown in Figure 78. In this model, θ is the joint flexion angle, l is the distance from the pulley edge to the rotation axis, d is the distance between the tendon and the phalanx, M is the distance from the tendon to the rotation axis which is also the corresponding moment arm, l_0 is the distance from the joint centre to the endpoint, F is the tendon force, D is the distance between the tendon sheath ends on the two sides of the joint, m is the mass of the endpoint, v is the steady-state velocity of the endpoint after release, and F_m is a constant external resistance force applied on the endpoint in normal direction which equals to mg. As the system is constructed on a horizontal plane, then the influence from the gravity can be removed in this model simulation.

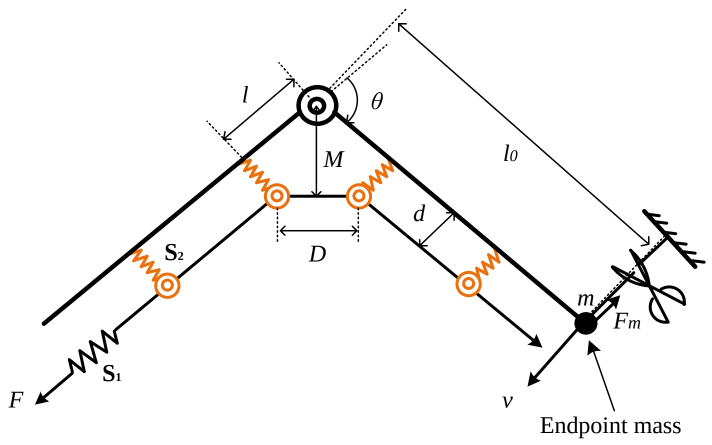


Figure 78. Schematic diagram of the quick-release experiment on single-joint model (for dynamic analysis).

To obtain the dynamic endpoint normal force and velocity output when the system reaches the steady state after release, the energy conservation approach is adopted here. According to the principle of energy conservation, the whole process of approaching steady state can be formulated as

$$\frac{1}{2}mv^2 = W_1 + 4W_2 - W_3 - 4W_4 - W_5$$

(54)

Where $W_1 = F_1^2/2k_1$, $W_2 = F_1^2\sin^2(\theta_1/2)/2k_2$, $W_3 = F_2^2/2k_1$, $W_4 = F_2^2\sin^2(\theta_2/2)/2k_2$, and $W_5 = F_m l_0(\theta_2 - \theta_1)$

With the force and geometrical conditions that

$$\begin{cases} F_2 M_2 = F_m l_0 \\ M_2 = l\sin\left(\frac{\theta_2}{2}\right) + d_0\cos\left(\frac{\theta_2}{2}\right) + \frac{F_2}{k_2}\sin\left(\frac{\theta_2}{2}\right)\cos\left(\frac{\theta_2}{2}\right) \\ \Delta x = \frac{F_1 - F_2}{k_1} = D_1 - D_2 \\ D_1 = 2l\cos\left(\frac{\theta_1}{2}\right) - 2d_0\sin\left(\frac{\theta_1}{2}\right) - 2\frac{F_1}{k_2}\sin^2\left(\frac{\theta_1}{2}\right) \\ D_2 = 2l\cos\left(\frac{\theta_2}{2}\right) - 2d_0\sin\left(\frac{\theta_2}{2}\right) - 2\frac{F_2}{k_2}\sin^2\left(\frac{\theta_2}{2}\right) \end{cases}$$

(55)

In Eqs. (54) and (55), W_1 and W_3 are the energy of the spring S_1 before and after release (steady state), W_2 and W_4 are the energy of the spring S_2 before and after release (steady state), W_5 is the work of the external resistance force F_m, k_1 and k_2 are the elastic coefficients of the spring S_1 and S_2, θ_1 and θ_2 are the joint angles before and after release (steady state), F_1 and F_2 are the tendon forces before and after release (steady state), Δx is the length change of the spring S_1 before and after release (steady state), D_1 and D_2 are the distances between the tendon sheath ends near the joint before and after release (steady state), and M_2 is the moment arm from the tendon to the joint rotation axis after release (steady state).

The numerical simulation is conducted to illustrate the dynamic impact of the flexible tendon sheath on finger force-velocity behaviour. Likewise, the parameters used are l = 4 mm, l_0 = 25 mm, d_0 = 2 mm, t_1 = 30 °, k_1 = 1 N/mm, k_2 = 0.8 N/mm and F = 13 N. The dynamic force-velocity output analysis should be conducted in various endpoint load conditions, within the range of the endpoint mass m from 50g to 180g. The external resistance force F_m would also accordingly change from 0.5N to 1.8N with the gravitational acceleration g of 10N/kg. With the Eqs. (54) and (55), the steady-state velocity output v after release can be calculated. And the corresponding steady-state force output is actually the F_m in each endpoint load condition since the force equilibrium should be satisfied in steady state. Then the theoretical dynamic force-velocity outputs in various endpoint load conditions are obtained with the result shown as the red curve in Figure 79. For the rigid tendon sheaths, the elastic coefficient k_2 in Eqs. (54) and (55) is assumed infinity, making W_2 and W_4 approximate to zero. By setting the tendon sheath thickness d_0 as 2 mm, 4 mm and 6 mm, the theoretical dynamic force-velocity characteristics of the finger with these three rigid tendon sheaths can be calculated and illustrated as shown in Figure 79 (in yellow, green and blue curves, respectively). In this case, the workspace is defined as the area surrounded by the characteristic curve and the two coordinate axes in which all the force-velocity working point can be achieved. Take the curve with a 2mm-thick rigid tendon sheath as an example, in the velocity range of 500 to 1400 mm/s, the workspace is marked as the yellow shadow area which is much smaller than that of the red curve with the flexible tendon sheath. It can be obviously seen in the figure that the dynamic force-velocity characteristic curve with the flexible tendon sheath lays above the other curves with rigid tendon sheaths, implying the flexible property of the tendon sheath can theoretically enlarge the force-velocity workspace of an finger.

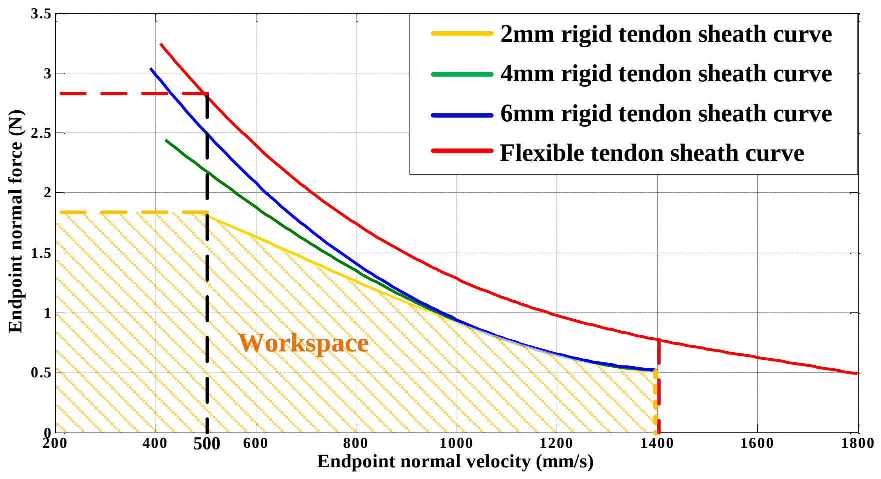


Figure 79. Theoretical result of the dynamic force-velocity characteristics

4.4 Summary

By establishing the mathematical models for the ligamentous joint, the index finger and the flexible tendon sheath, three biomechanical properties were analyzed in this chapter.

Though the model of the ligamentous joint was simplified, it was probably the very first to demonstrate the change of ligament length and joint stiffness during the finger flexion-extension and abduction-adduction motions from the perspective of mathematics, including the geometric and algebraic methods. Through the mathematical analysis, it was found that the ulnar ligament was gradually elongated with the MCP joint flexion and abduction motions. With the unique load-displacement relation of the ligament and the specific locations of the ligament’s origin and insertion, the corresponding joint flexion stiffness maintains at a low level during most of the flexion motion and increases sharply near the full flexion position. Likewise, the joint terminal abduction stiffness also increased with the joint flexion and abduction angle increasing. This variable joint stiffness is one of the biomechanical properties of the ligamentous joint.

The kinematic model of the index finger was established to analyze the feasible force space at the fingertip. Though large amounts of work on the feasible force set and space have been done by Valero-Cuevas, his research was mostly focused on the human hand and cadaver tests. We believe the feasible force space can also be one of the main evaluation indicators of the robotic hand performance and the method provided in this chapter paved the way. Besides, it was also the theoretical foundation of the comparison experiments between the net and linear extensor in the following chapter.

Taking the view of the previous research, the single-joint model with flexible tendon sheaths should be the first mathematical model to analyze the mechanical functions of the flexible tendon sheaths. Accordingly, the quasi-static and dynamic force-velocity characteristics were both investigated. The theoretical result shows that the flexible tendon sheath with 2mm initial thickness can achieve both large endpoint force and high endpoint velocity and has a larger dynamic force-velocity workspace compared with the rigid tendon sheaths with 2mm, 4mm and 6mm thickness. This should be one of the potential biomechanical advantages of the flexible tendon sheath which will be verified through experiments in the following chapter.

Chapter 5: Experimental verification for the biomechanical advantages realized on the robotic finger

This chapter uses experimental methods to verify the theoretical prediction results on the three biomechanical properties influenced by the biological structures in the last chapter. Several test rigs for three series of experiments were built and the obtained results were compared with the theoretical results. The aim of this chapter is to verify if the biomechanical advantages brought by the biological structures can be realized on the robotic finger with similar biomimetic design.

5.1 Variable joint stiffness

5.1.1 Experiment set-up and protocol

To verify the results from the mathematical analysis, the flexion and abduction stiffness for the MCP joint in the robotic finger was tested based on the prototype developed in this thesis. The top view of the flexion stiffness experimental setup is shown in Figure 80 (a). The proximal end of the metacarpal was fixed on the testbed and a motor was connected with the distal end of the proximal phalanx through a string and a pulley. In the setup, the flexion-extension plane is in the horizontal direction parallel to the floor and so is the string that is connected to the motor, such arrangement helps remove the influence of gravity. During the experiment, at each step, the motor pulled the string to flex the MCP joint by a small angle. The tension force on the string was obtained from the feedback torque of the motor, and the flexion angle as well as the moment arm which was the shortest distance from the rotation axis to the string were captured by a camera. The motion of each step was recorded by the camera, and the images (including the angle and distance measurements) were then processed and analyzed in the software ImageJ®. In data processing, the externally applied torque T was calculated as $T = F L$, where F is the tension force on the string, and L is the moment arm. Note that, in this case, the external applying torque is equivalent to the MCP joint's internal resistance torque. Using the relation that $k = d\tau / d\theta$, joint stiffness from the experiment can be calculated and the comparison result with the mathematical analysis is shown in Figure 81.

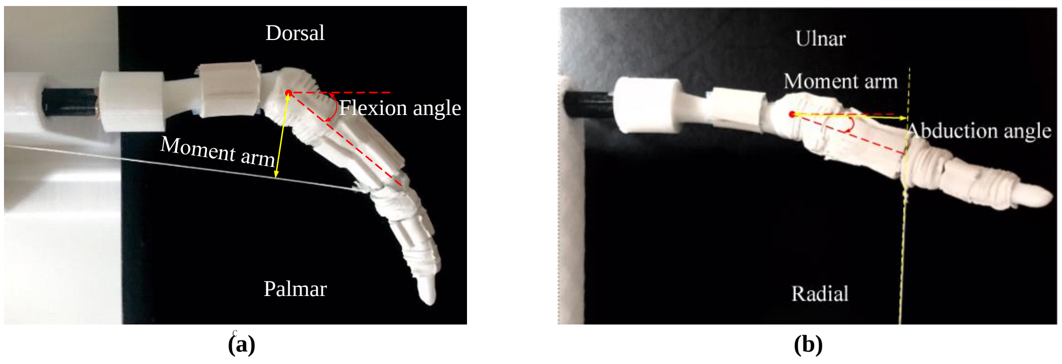


Figure 80. Experimental setup of the joint stiffness test. (a) Experimental setup of the joint flexion stiffness test. (b) Experimental setup of the joint abduction stiffness test.

Furthermore, the MCP joint abduction stiffness was also tested with respect to the various flexion angles. The side view of the experimental setup is shown in Figure 80 (b). As can be seen, the abduction-adduction plane in this setup is always perpendicular to the floor in all flexion angle test conditions. There are two motors controlling the MCP joint flexion and abduction angles, respectively. The string of the flexion-control motor inserts into the distal end of the proximal phalanx on the palmar side with its direction parallel to the floor. And the abduction-control motor is located down straight to the finger with its string vertical to the floor, inserting into the distal end of the proximal phalanx on the radial side. In the test, the MCP joint was driven to flex firstly, and at each flexion position, the joint was driven to gradually abduct. During this process, the abduction angles, the tension force on the string and the moment arm were recorded so as to obtain the passive joint abduction stiffness. The measurement and calculation methods are the same with that used in the joint flexion stiffness test.

5.1.2 Results of variable joint stiffness

The flexion stiffness of the joint is shown in Figure 81, where the blue line shows the physical experiment result and the red line implies the mathematical result. Both of the results show that the joint flexion stiffness firstly maintains at a low value in most of the flexion positions and then rapidly increases near the full flexion position where the flexion angle is at around 60 °~ 70 °. Near the end of the curves, both of the theoretical and experimental values of the joint flexion passive stiffness reach about 4N mm/deg. Through the above experiment, the relation of the joint torques and the joint abduction angles in different flexion angle conditions can be demonstrated as a group of curves shown in Figure 82 (a). And the slope at each point on the curve is defined as the joint abduction stiffness at the corresponding abduction and flexion angle. As can be seen, the larger of the flexion angle, the more rapidly the joint torques increase with the abduction angles, meaning the higher joint abduction stiffness. Here we use the concept of the terminal stiffness in reference [6] as the characteristic description, which is defined as the corresponding slope at the end of the joint load(torque)-displacement(rotation) curve. To clearly identify the end of these torque-angle curves in Figure 82 (a), a reference line (120N mm torque) is inserted into the figure, making one cross point with each curve. And the curve slopes of these cross points can be taken as the terminal joint abduction stiffness in each flexion angle condition, which are marked as the blue triangles in Figure 82 (b). In Figure 82 (b), the blue curve is fitted by these triangles to describe the uptrend of the terminal stiffness with the flexion angle increasing. In fact, each intersection point in Figure 82 (a) corresponds to a specific flexion-abduction angle pair. Using the same joint flexion and abduction angle, the terminal stiffness could also be found and marked as the black points in the mathematical result of Figure 72. Likewise, these black points are marked in Figure 82 (b) and fitted by the black curve. As can be seen, both of the results in the experiment and mathematical analysis present similar rising trend of the terminal joint abduction stiffness with the flexion angle growing, with a rapid increasing at about 50 °

~ 60 °flexion angles. The terminal stiffness values near the full flexion positions shown in both of the theoretical and experimental results can reach around 120N ·mm/deg. The results also show good consistency with that of the cadaver test in reference [6].

5.1.3 Discussion

As can be seen from the above results, no matter for the joint flexion stiffness or the joint abduction stiffness, they both maintain at a relatively low value at the beginning and then surge to a high value near the end of the flexion. However, their changing behaviours are not identical. The joint stiffness in the flexion-extension direction keeps at low values in a larger flexion angle range (0 ° ~ 65 °) compared with that in the abduction-adduction direction (0 °~ 50 °). Likewise, the full-flexion joint stiffness in the abduction-adduction direction is of higher magnitude (around 120N ·mm/deg) than that in the flexion-extension direction (around 4N ·mm/deg). These results show the anisotropic variable joint stiffness properties of the ligamentous joint. From the above research, it can be seen that the stiffness in MCP joint is not constant but changes with joint angles. And this variable stiffness behaviour exists in both flexion-extension and abduction-adduction directions, which provides the fundamental biomechanical conditions for the versatile and adaptation performance of human and robotic fingers. Especially in the abduction-adduction direction, the performance of low stiffness in a small flexion angle maintains good joint dexterity and the high stiffness near the full flexion position provides the joint good lateral force bearing capability, of which the property is directly generated from interaction postures and applied to the interaction process, such as pulling a rope with a tight grip posture.

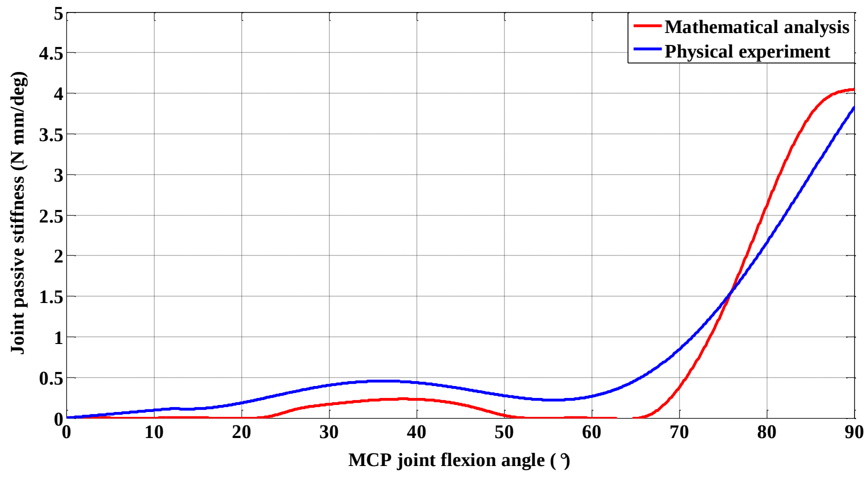
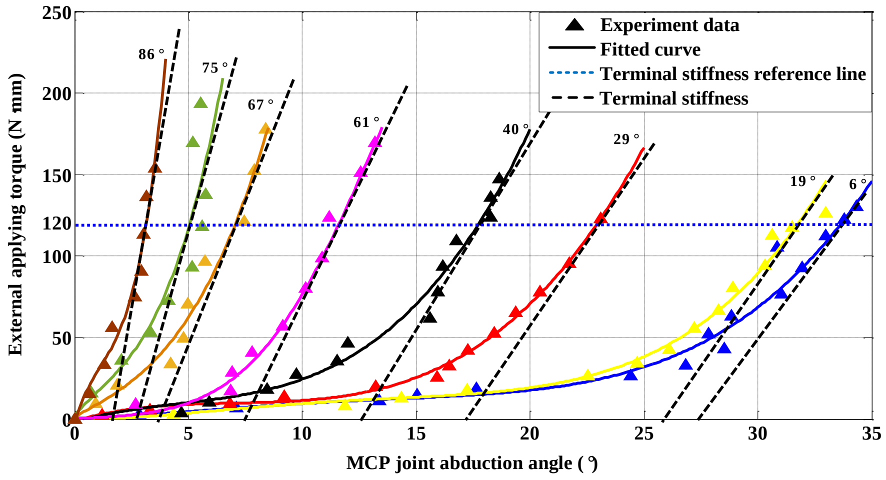
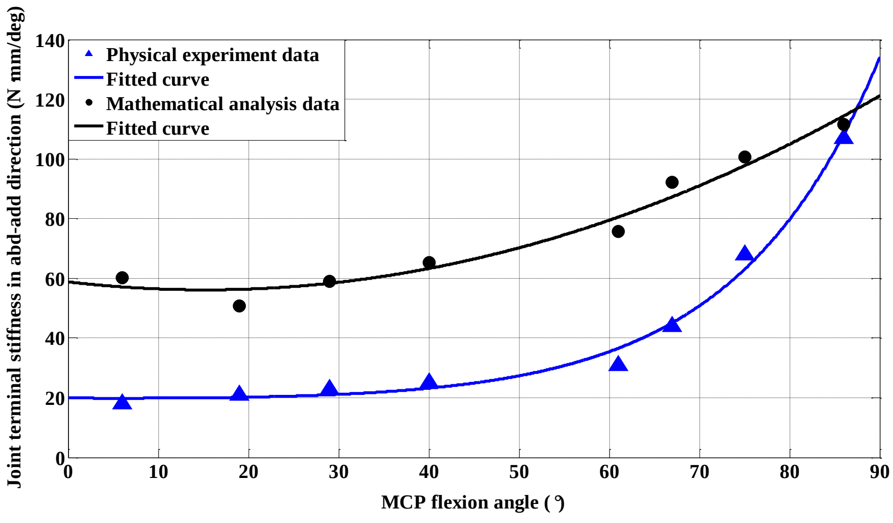


Figure 81. Comparison result of the theoretical and experimental relations of the joint flexion stiffness and flexion angles



(a)



(b)

Figure 82. Joint terminal stiffness in abduction-adduction direction. (a) Relation of the external applying torque and abduction angles from the test. (b) Comparison result of the theoretical and experimental relation of the joint abduction terminal stiffness and flexion angles.

5.2 Enlarged feasible force space

5.2.1 Experiment set-up and protocol

To investigate the effect of the network structure of extensor mechanism on the feasible force space, two robotic finger models were designed and fabricated, one with net extensor (mimicking the extensor mechanism) and the other with linear extensor, as shown in Figure 83. To better illustrate the different morphologies of the net and linear extensor, the schematic diagrams are also presented in the figure. As can be seen, the complex structure of the extensor mechanism is highly replicated on the net extensor, of which the RI, LE and UI tendons are connected with each other through the tendon branches or the elastic links. By comparison, the RI, LE and UI tendons are separated on the linear extensor, and the finger can be extended by only the LE tendon where the pulley element helps extend all the joints simultaneously.

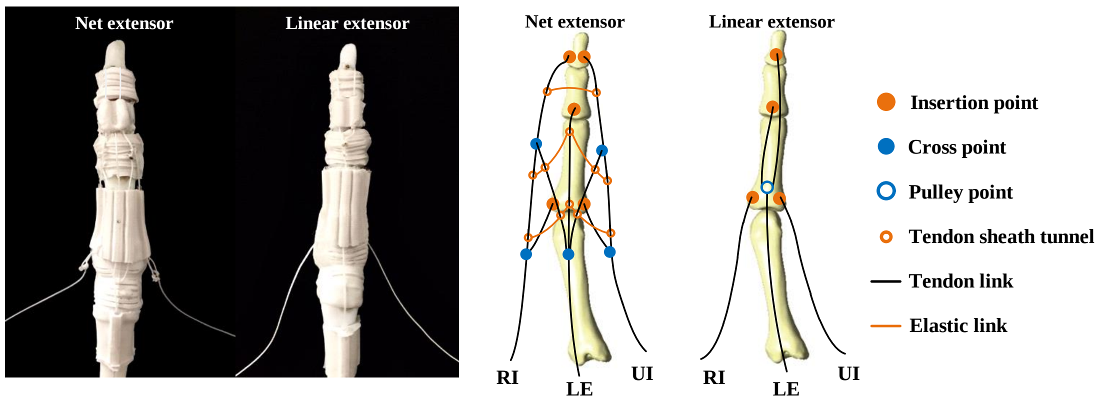
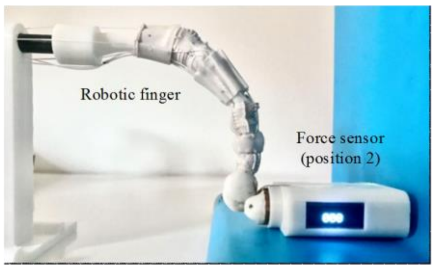
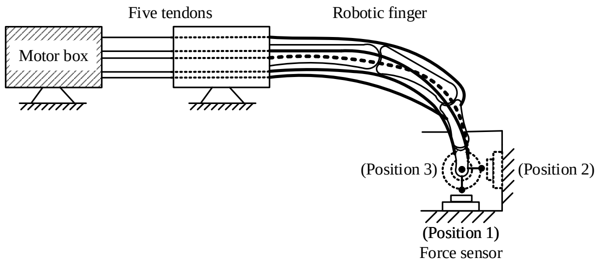


Figure 83. The physical models and the schematic diagrams of the robotic fingers with net and linear extensor.



(a)



(b)

Figure 84. Experimental setup of the fingertip force test. (a) Physical model of the fingertip force testbed. (b) Schematic diagram of the fingertip force testbed.

Experiments were then conducted to find the influence of extensor mechanism structure on feasible force space. In the experiments, three postures of the two robotic fingers were tested, of which posture 1 is given as DIP at 5 °, PIP at 25 °, MCP flexion at 40 ° and abduction at 0 °, posture 2 as DIP at 10 °, PIP at 45 °, MCP flexion at 75 ° and abduction at 0 °, and posture 3 (standard posture) as DIP at 10 °, PIP at 45 °, MCP flexion at 45 ° and abduction at 0 °. To obtain the endpoint feasible force space, a method similar with the one used in Valero-Cuevas [147] was adopted. In the experiment, as shown in Figure 84 (c) and (d), each robotic finger was actuated by five Dynamixel® MX-12W motors to pull the RI, UI, FDP, LE and FPS tendons respectively with the maximum output torque. Noting that only one tendon was pulled at a time and the other tendons were in a loose state so that the fingertip force output contributed by each tendon can be respectively recorded. In addition, for getting force information, a force-sensing resistor (FSR) together with a liquid crystal display (LCD) was placed near the fingertip, being fixed on a plate that is perpendicular to the palmar (position 1) / distal (position 2) / radial (position 3) direction (for collecting force data in the palmar, distal and radial directions). During the experiment, the force sensor was firstly placed in position 1 and one tendon was pulled with the maximum motor output torque. Then we can obtain the fingertip force from the force sensor's reading and the tendon force from the corresponding feedback torque of the motor's built-in sensor. After all the five tendons were tested, we moved the force sensor to position 2 and then position 3, until the experiments were conducted in all the three directions (palmar, distal and radial). Each tendon can generate one fingertip force composite space vector. Therefore, five fingertip force vectors were obtained from the above tests.

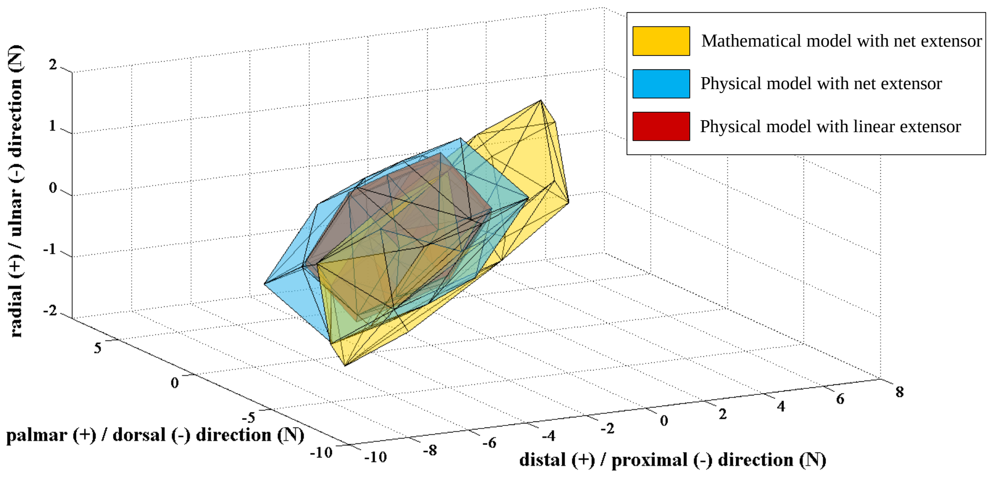
Considering that the force output of the fingertip equals to the vector sum of the measured fingertip force output contributed by each tendon, the feasible output force space can be directly generated.

5.2.2 Results of enlarged feasible force space

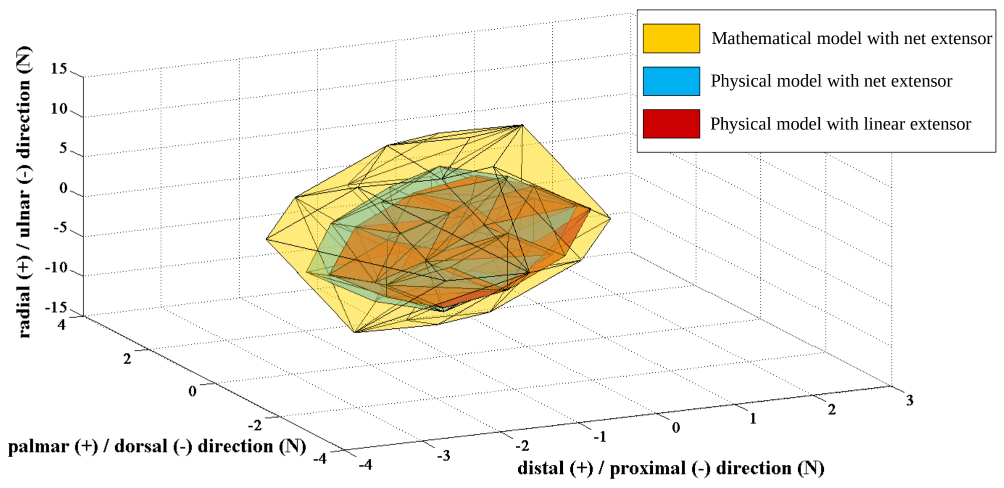
Consequently, the comparison results of the mathematical model with net extensor, and the physical model with net and linear extensor are presented in Figure 85(a), (b) and (c). The yellow blocks represent the feasible force space of the mathematical models with the net extensor. The blue and red blocks are the feasible force space results of the physical models with the net and linear extensor, respectively. From the figures, we can see that in terms of the models with the net extensor, the yellow blocks take up more space than the blue blocks. More apparent results are shown in Figure 85(b) and (c), where the blue and red blocks are nearly completely enveloped in the yellow blocks. And for the physical model results only, the blue blocks are always larger than the red blocks, especially in Figure 85(a) and (c). Besides, as to the feasible force space results in the three postures, the feasible force range in the distal-proximal direction of posture 1 shows the maximum magnitude, following with the posture 3 and 2. A similar result also appears in the palmar-dorsal direction, with the feasible force range of posture 1 showing the maximum magnitude. But for the feasible force range in the radial-ulnar direction, posture 2 has the maximum magnitude apparently. The detailed feasible force space volume results are presented in Table 6.

5.2.3 Discussion

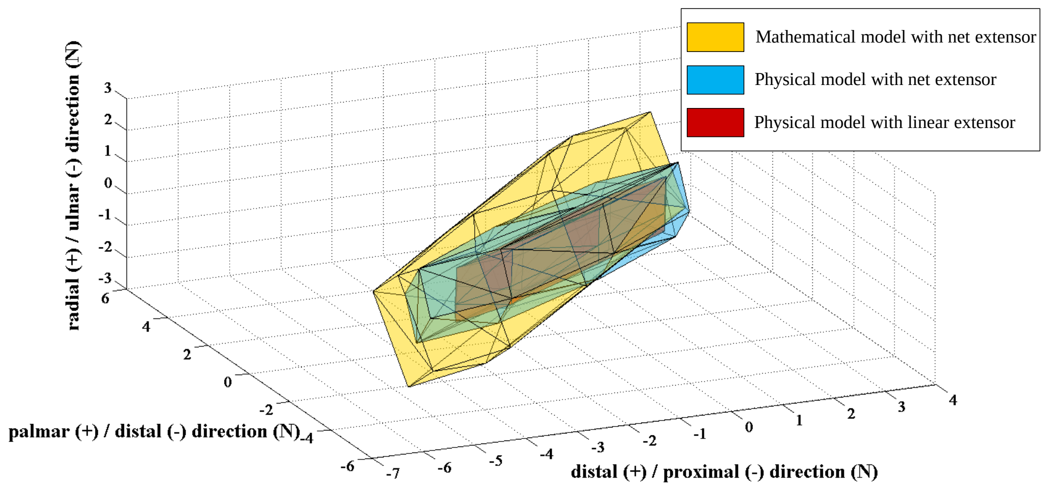
From the above results, we can see that in terms of all the three postures, the feasible force space of the mathematical model with net extensor is slightly larger than the ones of the physical models. And the feasible force space of the physical model with net extensor is apparently larger than that of the physical model with linear extensor. Especially in posture 3, as can be seen in Table 6, the volume of space with net extensor is more than two times larger than the linear extensor. Moreover, as the results in Table 6 shown, for the finger model with the net extensor, if a larger feasible force space is expected, the finger should make a posture 1 or 3 rather than 2. But as the Figure 85 shown, the finger can make a posture 2 when a larger radial-ulnar fingertip force range is needed. From the above research, it is evident that the net extensor mechanism can significantly enlarge the endpoint feasible force space compared with the linear extensor structure. And the enlarged fingertip feasible force space contributes to the practical performance by allowing exerting sufficient forces in any direction within the finger's workspace. This optimized mechanical output in the finger is particularly important when it interacts with the uncertain, unstructured and irregular environment since various forces in any direction may be potentially required.



(a)



(b)



(c)

Figure 85. Comparison result of the fingertip feasible force space in different postures. (a) The comparison results of the fingertip feasible force space in posture 1. (b) The comparison results of the fingertip feasible force space in posture 2. (c) The comparison results of the fingertip feasible force space in posture 3. (Yellow-mathematical model with net extensor; Blue-physical model with net extensor; Red- physical model with linear extensor)

Table 6. The volume of the feasible force space in different postures with net and linear extensor

Physical model \ Posture	Posture 1	Posture 2	Posture 3
Net extensor	39.8599	17.3209	34.4450
Linear extensor	19.8477	9.5869	9.6287

5.3Augmented force-velocity workspace

5.3.1 Experiment set-up and protocol

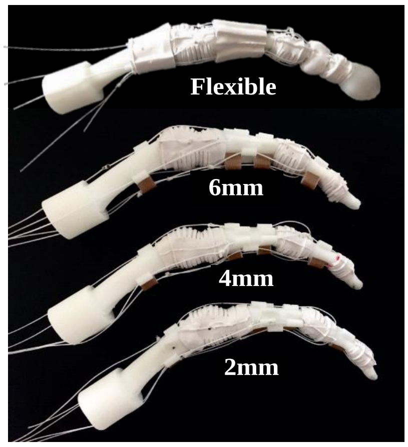
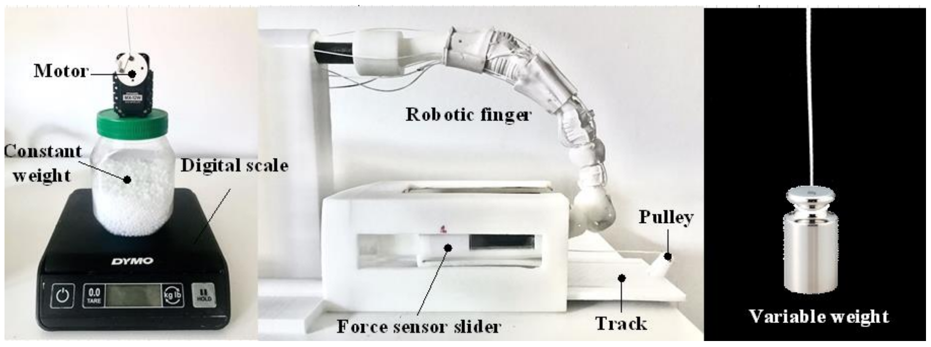
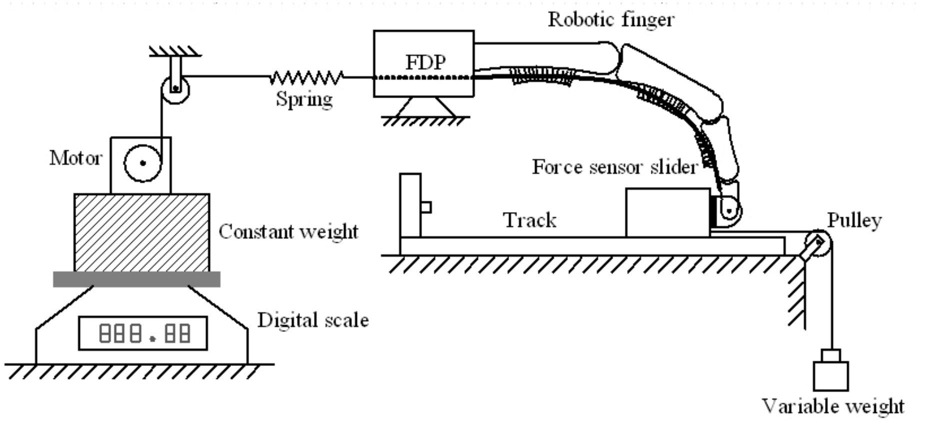


Figure 86. The robotic finger model with the flexible tendon sheath and three comparison robotic finger models with rigid tendon sheaths of different thicknesses (6mm, 4mm and 2mm).

Since the quasi-static force-velocity output can be obtained from the two ends of the dynamic working curve, only dynamic force-velocity characteristics are tested in this section. In order to illustrate the influence of the flexibility of tendon sheath on the force-velocity workspace of a finger, except for the proposed robotic finger with flexible tendon sheath, three additional robotic fingers with rigid tendon sheaths of thickness 2 mm, 4 mm and 6 mm were designed and fabricated for comparison purpose, as shown in Figure 86. To verify the prediction of the theoretical model, the quick release experiments were conducted to investigate the dynamic force-velocity characteristics resulted by different tendon sheath structures. A test rig was designed and developed (in Figure 87 (a) and (b)) for the proposed tests. From left to right in Figure 87 (b), the test rig contains a digital scale, a constant weight, a motor, a connection spring, a robotic finger with FDP tendon, a force sensor slider, a track, a pulley and a variable weigh. For the purpose of controlling the variables, the initial tendon force is set the same for different finger models tests which is realized by a constant weight bound with a motor and connected with a spring. The weight is placed on a digital scale to precisely obtain the value of the initial tendon force. The spring is linked with the FDP tendon of the robotic finger, and the fingertip is fixed with the press button of the force sensor through a revolute joint. The force sensor slider, which is a piece of calibrated force-sensing resistor (FSR) integrated with an Arduino Nano Every board, can only slide along the track horizontally so as to keep the testing press force and velocity in a constant direction. At the other end, a variable weight is connected with the force sensor slider through a pulley to provide a variable endpoint load condition.



(a)



(b)

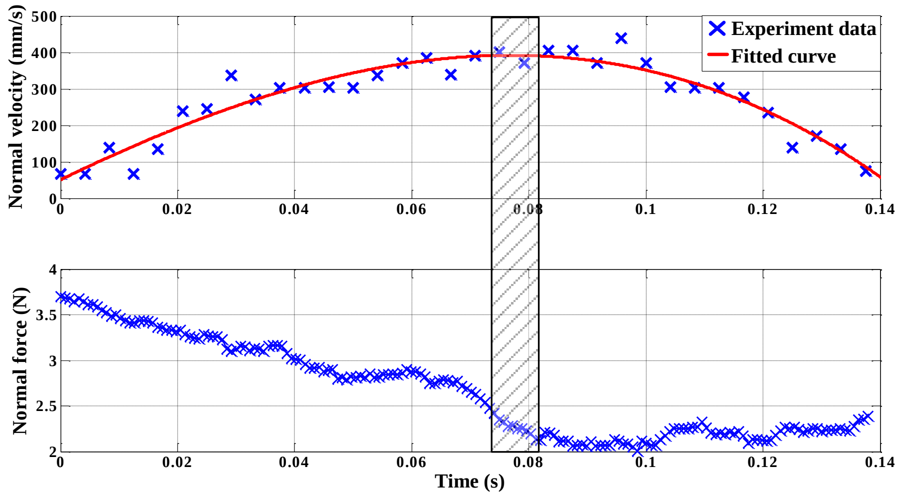
Figure 87. Experimental setup of the quick-release test. (a) Test rig for the quick-release experiment. (b) Schematic diagram of the quick-release test rig.

At the beginning of the test, the posture of the robotic finger was set as 10 ° DIP joint, 30 ° DIP joint, 30 ° MCP flexion joint and 0 ° abduction joint. The variable weight was fixed so that the position of the fingertip does not change after the initial tendon force is set. Then the motor attached on the constant weight started to pull the spring to exert the initial tendon force to the target value. The initial tendon force was set as 13 N such that the fingertip could reach a steady state during the motion with both small and large endpoint load applied. The digital scale shows the instantaneous force value which equals to the gravity of the constant weight subtracting the tendon force. For instance, if the constant weight used is 2kg, the motor should rotate to pull the spring until the digital scale shows 700g.

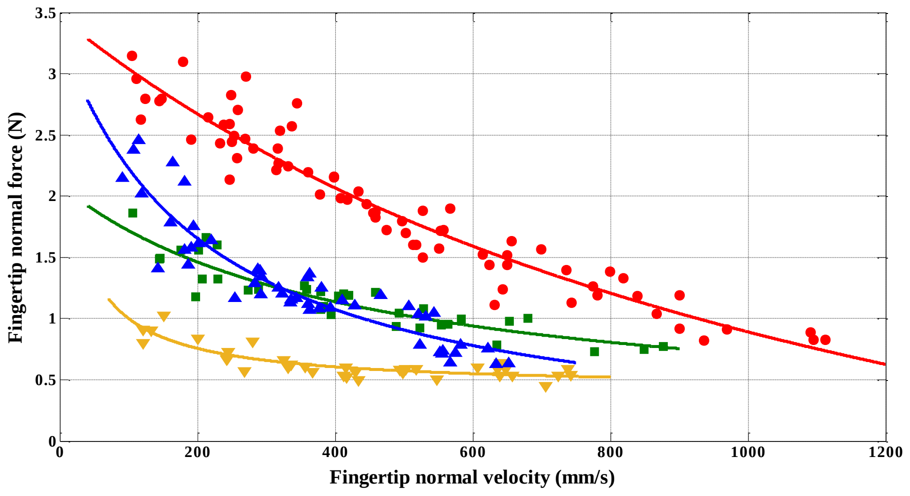
After that, the variable weight was released and the dynamic fingertip force and velocity were recorded. During the test, the Arduino board integrated into the FSR was set to collect the force data with 500000 baud rate in real time. And the velocity of the slider was recorded by a high-speed camera with 960 ftp and then analyzed in ImageJ with a Manual Tracking plug-in (Fabrice Cordelieres, 2005). It should be noted that the robotic finger was tested with only the FDP tendon pulling and all the other tendons were released and slack. The experiment was performed with variable weights from 50g to 300g in steps of around 25g with 5 repetitions each. Considering different load capabilities of the robotic fingers, the range of the weights would slightly vary accordingly.

5.3.2 Results of force-velocity characteristics

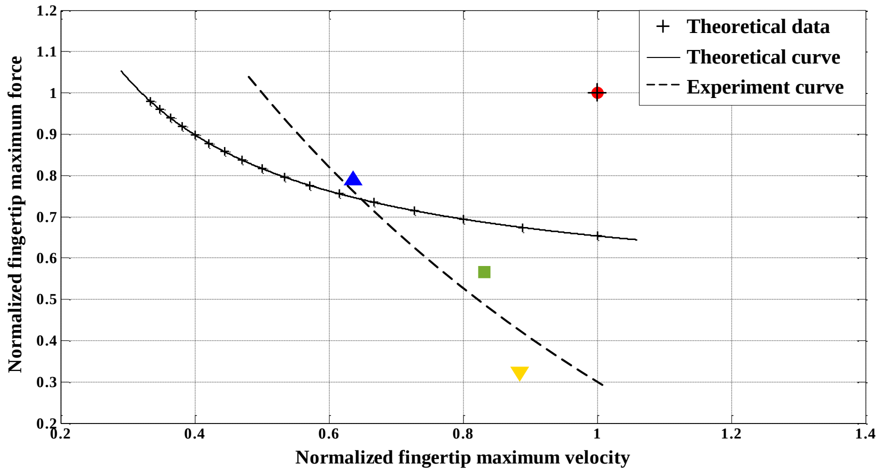
Figure 88 (a) shows one of the hardware experiment force and velocity results with the flexible tendon sheath in a 180g load. As can be seen that during the whole release process, the endpoint velocity increases at the initial stage and then reaches a quite short plateau, following with a drop at the end. We hypothesize that the steady-state force-velocity output occurs at the velocity plateau stage, and the average values of a small piece of force and velocity data in this period are selected as the force-velocity output in a certain load condition, which is shown as the shadowed block in Figure 88 (a). Take this average velocity output as the x-value and the average force output as the y-value, we got one corresponding point in Figure 88 (b). Taking the same procedure, we can obtain some point cloud data and their rational fitted curves to demonstrate the hyperbola-like dynamic force-velocity characteristics for different finger models, as shown in Figure 88 (b). In Figure 88 (b), the yellow inverted triangles represent the experiment data of the 2mm rigid tendon sheath and the corresponding fitted curve is the yellow curve. The green squares are the data of the 4mm rigid tendon sheath, fitted by the green curve. The data of the 6mm rigid tendon sheath are marked as the blue triangles and fitted by the blue curve. And the data of the flexible tendon sheath are shown as the red points and fitted by the red curve. It is obvious that the red curve is on the top of other curves and the yellow curve is at the bottom. Besides, the extreme values of the force and velocity near the ends of the red curve are also larger than other curves'. Its fingertip normal force can reach over 3N and its fingertip normal velocity can be over 1000mm/s. In the theoretical result of Figure 79, the workspace (within 500 to 1400 mm/s velocity range) with the flexible tendon sheath is nearly 35% larger than that with the rigid tendon sheaths in average. And the gap is even more pronounced in the experimental result (Figure 88 (b)), though the overall endpoint velocity is slightly lower than the theoretical result owing to the system friction. Within the range of the experiment data, the workspace with the flexible tendon sheath is approximately 50% larger than that with the rigid tendon sheaths in average.


Normal velocity (mm/s)
Experiment data
Fitted curve
Normal force (N)
Time (s)

(a)


Fingertip normal force (N)
Fingertip normal velocity (mm/s)

(b)


Theoretical data
Theoretical curve
Experiment curve
Normalized fingertip maximum force
Normalized fingertip maximum velocity

(c)

▼ 2mm rigid tendon sheath experiment data
■ 4mm rigid tendon sheath experiment data
▲ 6mm rigid tendon sheath experiment data
● Flexible tendon sheath experiment data
2mm rigid tendon sheath curve
4mm rigid tendon sheath curve
6mm rigid tendon sheath curve
Flexible tendon sheath curve

Figure 88. Experiment result of the fingertip force-velocity characteristics with different tendon sheaths. (a) Fingertip force and velocity experiment results with the flexible tendon sheath in a 180g load. (b) Dynamic force-velocity characteristics of the four physical robotic finger models (Yellow-2mm rigid tendon sheath; Green-4mm rigid tendon sheath; Blue-6mm rigid tendon sheath; Red-flexible tendon sheath). (c) The normalized results of the theoretical and experiment data about the quasi-static maximum force and velocity that different finger models can output.

Additionally, the quasi-static maximum fingertip force and velocity can also be approximately obtained from the dynamic experimental results. For one force-velocity curve in Figure 88 (b), the two ends close to the axes are the nearly maximal values of the force and velocity that the finger can output. To better approach the maximal values and not cause too much deviation away from the experiment data, the four fitted force-velocity curves are extended 5% of the experiment data range to the axes. Thus, the values of the extended curves' two ends are selected as the approximate maximum force and velocity. Likewise, all the results are normalized with the results of the flexible tendon sheath and marked as different shape and color blocks in Figure 88 (c). And in Figure 88 (c), the theoretical results are represented by the '+' marks and the solid curve. The experimental results of the rigid tendon sheath shown as different types of points are fitted by the dotted curve. Since all the data is normalized, the theoretical and experimental results of the flexible tendon sheath are overlapped in Figure 88 (c), shown as the '+' mark and the red point, respectively. It can be seen from both of the theoretical and experimental results that the flexible tendon sheath can obtain both a large maximum fingertip force and a large maximum fingertip velocity. In comparison, for the rigid tendon sheaths, the thicker of the sheath, the larger maximum fingertip force but the smaller maximum fingertip velocity.

5.3.3 Discussion

For the dynamic force-velocity characteristics, as can be seen from both the theoretical (Figure 79) and experimental results (Figure 88 (b)), in terms of the rigid tendon sheaths, the thicker tendon sheaths (such as the blue curve representing the 6mm thick tendon sheath), the larger fingertip output force at low velocity and the larger force-velocity workspace. However, compared with the rigid tendon sheaths, a larger endpoint force, endpoint velocity and force-velocity workspace can be obtained by the robotic finger with the flexible tendon sheath (shown as the red curve in Figure 79 and Figure 88 (b)) in the same condition, manifested as that the red curve totally laying above the other curves. These results demonstrate a dynamic and consecutive process to show the force and velocity characteristics that the finger can achieve simultaneously at work. Obviously, the finger with the flexible tendon sheath can work on more force-velocity points to adapt to more interaction environments.

And for the quasi-static force-velocity characteristics, as can be seen in Figure 88 (c), the flexible tendon sheath can achieve both a larger maximum fingertip force and a larger maximum fingertip velocity, showing better force and velocity generating capability compared with the rigid tendon sheaths. Therefore, it’s proven that the finger with the flexible tendon sheath performs better in both the quasi-static and dynamic force-velocity characteristics.

5.4 Human-finger-like grasping capability

5.4.1 Experiment set-up and protocol

To demonstrate performance of the robotic finger, grasping tests were conducted on a custom designed testbed. The testbed was shown in Figure 89, which was constructed by two robotic fingers as the end effector and eight Dynamixel MX-12W motors (RoboSavvy Ltd., UK) as the actuation system. Through the power hub and USB2Dynamixel connector, the motors can be controlled by PC. The index finger and the thumb were selected as the testing model since the basic grasping or picking up motion can be normally performed by these two fingers. In the eight-motor actuation system, four motors were used to drive the index finger and another four were for the thumb.

The grasping test was conducted with five different objects on three types of surfaces. The five target objects were selected according to their specific shapes and softness, including a nut, a pen, a pair of soft cubes, a handkerchief, and a playing card. And the interaction surfaces were set as flat, rough, and soft, shown in Figure 91.

The robotic fingers control system was constructed through the Matlab Dynamixel Software Development Kit. By using the same control algorithm, the grasping motion can be easily performed. Besides, the initial, intermediate, end positions, and the moving speed of the whole actuation platform were also set to the same and controlled by a robotic arm. The whole control flow chart is shown in Figure 90. At position 1 of the robotic arm, the robotic fingers changed the posture from the initial preparing state to the open state. Then the robotic arm moved down to approach the object, and the robotic fingers turned to the close posture when the robotic arm moved to position 2 where the interaction distance is proper for grasping. During this process, the final actual grasping posture depends on the collaborative effect of the target close posture of the robotic fingers and the target interaction objects and surfaces because of the intrinsic self-adaptation property of the robotic fingers. Finally, the robotic arm moved up back to the initial position 1 with the robotic fingers succeeding or failing to grasp the object. Each test was repeated 10 times by the robotic fingers and human fingers, of which the successful cases were recorded as shown in Figure 92. The videos of all the grasping tests can be found in Appendix I (1~15).

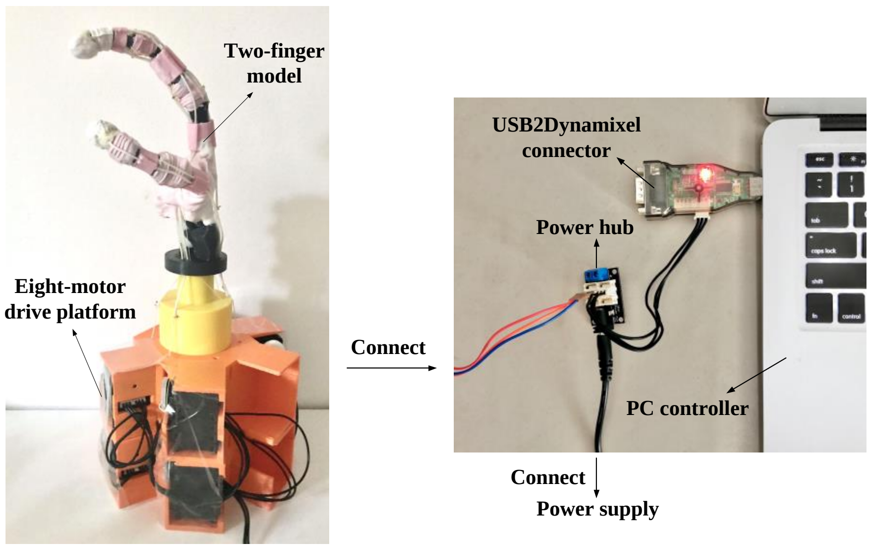


Figure 89. Two-finger testbed and the hardware setup.

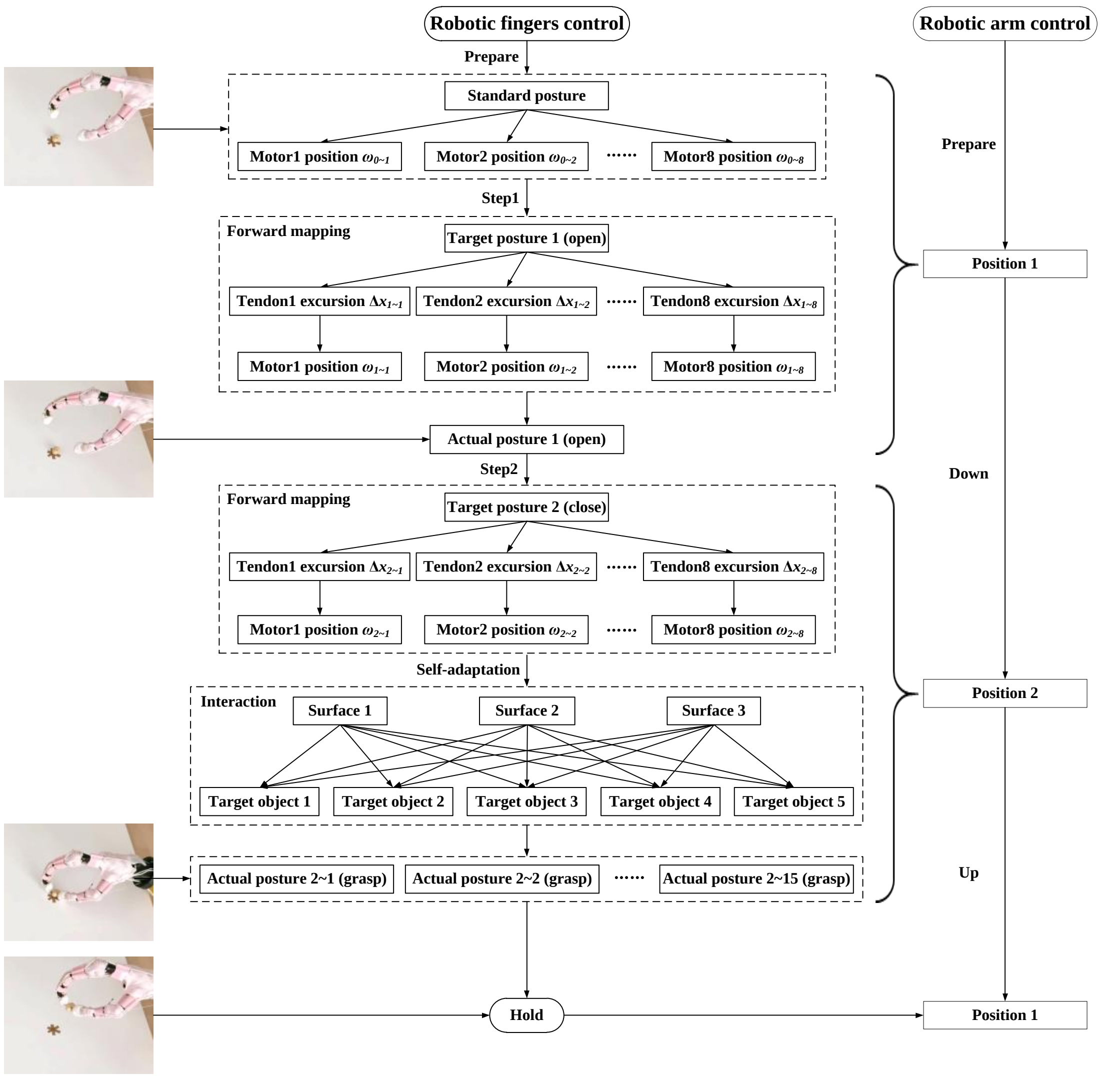

Robotic fingers control
Robotic arm control
Prepare
Standard posture
Motor1 position $\omega_{0\sim1}$
Motor2 position $\omega_{0\sim2}$
······
Motor8 position $\omega_{0\sim8}$
Prepare
Step1
Forward mapping
Target posture 1 (open)
Tendon1 excursion $\Delta x_{1\sim1}$
Tendon2 excursion $\Delta x_{1\sim2}$
Tendon8 excursion $\Delta x_{1\sim8}$
Motor1 position $\omega_{1\sim1}$
Motor2 position $\omega_{1\sim2}$
Motor8 position $\omega_{1\sim8}$
Position 1
Actual posture 1 (open)
Step2
Down
Forward mapping
Target posture 2 (close)
Tendon1 excursion $\Delta x_{2\sim1}$
Tendon2 excursion $\Delta x_{2\sim2}$
Tendon8 excursion $\Delta x_{2\sim8}$
Motor1 position $\omega_{2\sim1}$
Motor2 position $\omega_{2\sim2}$
Motor8 position $\omega_{2\sim8}$
Self-adaptation
Interaction
Surface 1
Surface 2
Surface 3
Position 2
Target object 1
Target object 2
Target object 3
Target object 4
Target object 5
Actual posture 2~1 (grasp)
Actual posture 2~2 (grasp)
Actual posture 2~15 (grasp)
Up
Hold
Position 1

Figure 90. Control flow chart for the robotic fingers and the robotic arm.

5.4.2 Results of grasping test

As shown in Figure 92, on the flat surface, the robotic fingers successfully grasp the peanut 9 times, the pen 8 times, the sponge blocks 8 times, the cloth 10 times and the playing card 10 times, respectively in 10-times tests each. And the human fingers correspondingly perform 9, 10, 9, 10, 10 times successful grasping in each 10-times tests. While, on the rough and soft surfaces, the average grasping success rate slightly drops, especially in the peanut, sponge blocks and playing card grasping. On the rough surface, the successful grasping cases of the robotic fingers appear 6, 9, 8, 10 and 6 times, respectively in the peanut, pen, sponge blocks, cloth and playing card grasping tests. Correspondingly, the human fingers respectively have 7, 9, 8, 8 and 8 times successful grasping. And on the soft surface, the peanut, pen, sponge blocks, cloth and playing card grasping are successfully performed 6, 9, 6, 10 and 7 times by the robotic fingers and 8, 10, 7, 10 and 9 times by the human fingers.

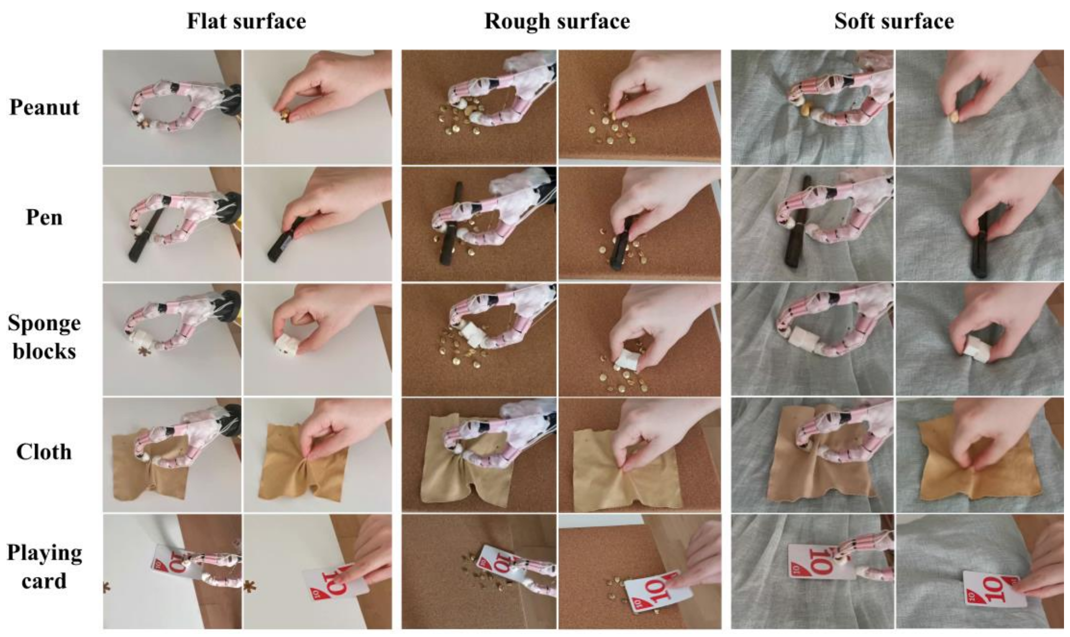


Figure 91. The robotic and human fingers grasping test with five target objects on three

different surfaces.

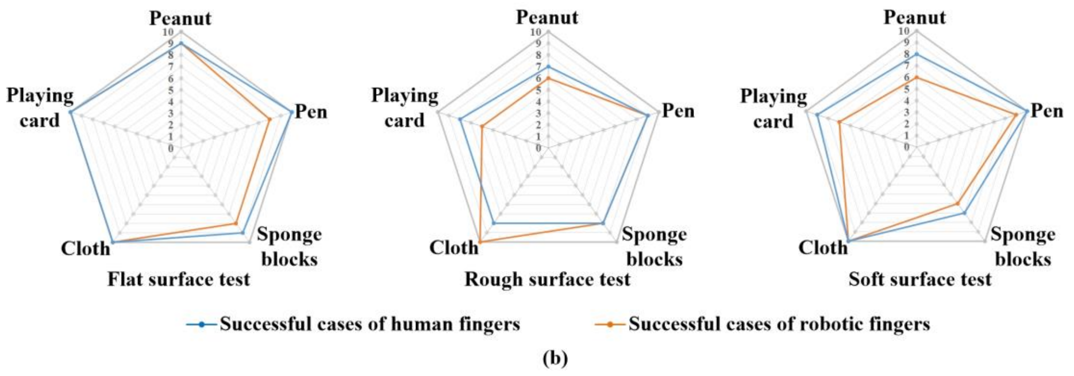


Figure 92. The comparison results of the successful cases of the robotic (orange line) and human (blue line) fingers.

5.4.3 Discussion

It can be seen that the overall grasping success rate of the robotic fingers is similar to that of the human fingers, both of which were at a high level. The above results illustrate that the shape and softness of the objects and interaction surfaces could influence the grasping qualities. The soft (sponge blocks), small (peanut) or thin (playing card) objects as well as the rough or soft surfaces all potentially complex the interaction process and result in decreasing the grasping success rate. For the robotic finger, the worst cases appear during grasping the peanut and the playing card on the rough surface and grasping the peanut and the sponge blocks on the soft surface. The lack of tactile sensors and the simple position control strategy can be the possible reasons. In most cases, the human finger performs better than the robotic finger. However, during grasping the cloth on the rough surface, the robotic finger shows a higher success rate. The reason can be the fact that the material property of the robotic fingertip may produce larger friction between the cloth and the fingertip. On the whole, despite the grasping difficulties brought by the target objects and the environments, the robotic and human fingers can both cope with them well, proving that the adopted structure design and materials of the proposed robotic finger can give it human-finger-like grasping capability.

5.5 Summary

To verify the three biomechanical properties and potential advantages brought by the ligamentous structure, the extensor mechanism and the flexible tendon sheath analyzed in the last chapter, several test rigs were constructed and the corresponding experiments were conducted.

The MCP joint stiffness was tested in both the flexion-extension and abduction-adduction directions on the robotic finger model. By comparing the experimental result with the theoretical prediction, a similar changing tendency of the joint stiffness was found, supporting that the variable joint stiffness behaviour exists in the ligamentous joint and can be realized on the robotic finger by proper joint design. This variable joint stiffness maintains the good dexterity of the finger within most motion range and simultaneously provide large load bearing capability with the high joint stiffness near the full flexion positions.

To compare the influence on the fingertip feasible force space by the net and linear extensor, two finger models with these two different extensors were designed and fabricated. A test rig for measuring the fingertip force generated by each tendon was developed. The result shows the net extensor leads to larger feasible force space compared with the linear extensor, with the volume of the former nearly twice as that of the latter in two positions and around four times in the standard posture of the finger. Similar results have been obtained through computational simulations. The verification result tested on the prototype provide another strong evidence that the human hand extensor mechanism can bring some biomechanical advantages and this structure is suggested to be adopted on the robotic hand to improve its performance.

A quick-release test was conducted in this chapter to demonstrate the dynamic as well as the resulted quasi-static force-velocity characteristics of the robotic finger with different tendon sheaths. Except for the proposed robotic finger model, three comparison models with rigid tendon sheaths (different thickness) were designed and fabricated. The experimental result shows that the flexible tendon sheaths do bring a larger force-velocity workspace compared with the rigid tendon sheaths as predicted in the theoretical results shown in Chapter 4. The difference is even more obvious in the experimental result, where the workspace associated with the flexible tendon sheaths is 50% larger than that with the rigid tendon sheaths in average. Better quasi-static force and velocity generating capabilities of the model with flexible tendon sheaths were also found in the test with the result consistent with the theoretical prediction.

With the three biomechanical advantages verified in the experiments, the grasping capability of a two-finger prototype was tested afterwards to investigate if these advantages can bring better practical functional performance. Through the tests of five different objects on three types of surfaces, the proposed robotic finger models show human-finger-like grasping capability with the grasping success rate analogous to the human fingers'. The successful design of the robotic finger paves the way for the development of the whole robotic hand, leading to the following chapters.

Chapter 6: Bio-inspired design and fabrication of the robotic hand

Based on the design of the robotic finger, a bio-inspired robotic hand was developed from designing, fabricating to assembling. In this chapter, the whole ligamentous-skeletal structure, tendon distribution and the artificial skin of the robotic hand are introduced. And the assembled robotic hand prototype is presented.

6.1 Ligamentous-skeletal structure

The ligamentous-skeletal structure of metacarpal and carpal bones is mainly discussed here. In this section, we reconstructed an entire human hand model from CT scanning and Mimics processing. The carpal and metacarpal bone models are shown in Figure 93. There are mainly eight separate carpal bones in a human hand. And they form two rows each of four bones, including a distal row with trapezium, trapezoid, capitate and hamate, as well as a proximal row with scaphoid, lunate, triquetral and pisiform. In the human hand, these carpal bones are bound together by plenty of ligaments as shown in Figure 94. As can be seen, the ligaments are classified into four groups which respectively connect the wrist and carpal bones, the carpal bones with one another, the carpal bones and the metacarpal bones, and the metacarpal bones with one another [150].

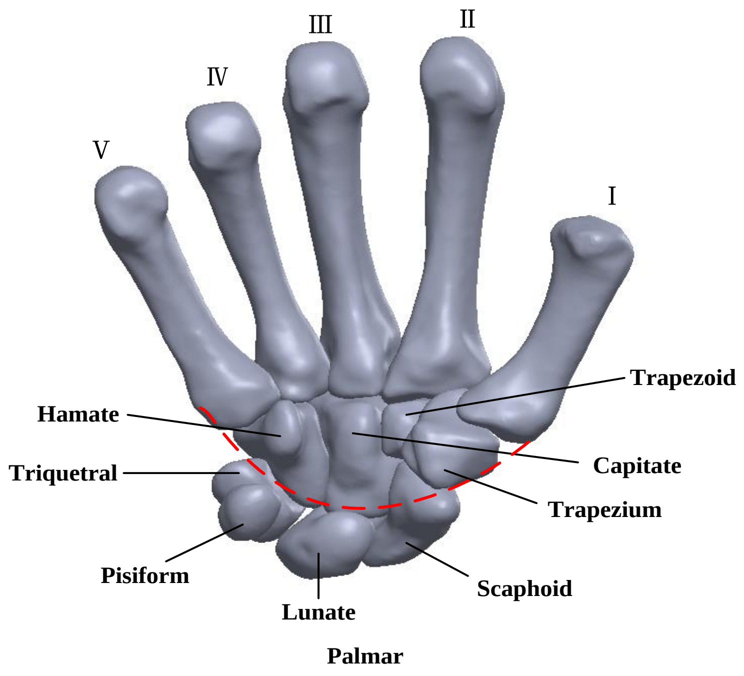


Figure 93. CAD model of human metacarpal and carpal bones

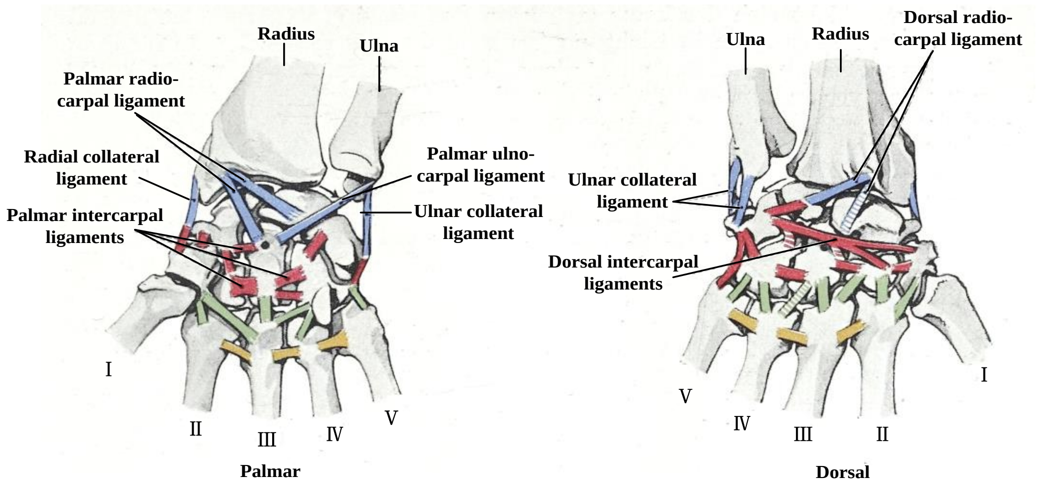

Radius
Ulna
Palmar radio-carpal ligament
Radial collateral ligament
Palmar intercarpal ligaments
Palmar ulno-carpal ligament
Ulnar collateral ligament
I
II
III
IV
V
Palmar
Ulna
Radius
Dorsal radio-carpal ligament
Ulnar collateral ligament
Dorsal intercarpal ligaments
V
IV
III
II
I
Dorsal

Figure 94. Metacarpal and carpal bones with ligaments (modified from reference [150])

In order to focus on the research of the hand part, the movement between the wrist and the carpals was ignored in this research. Thus there are only two groups of joints being studied — the carpometacarpal joints and the intercarpal joints [151].

The carpometacarpal joints are the articulation connection between the carpal and metacarpal bones, where little movement exists. Specifically, the index and middle finger metacarpals are nearly immobile, and there is only a slight gliding movement between the ring finger metacarpal and the hamate. However, between the little finger metacarpal and the hamate, an apparent relative movement can be observed because of the unique joint surface shape. Especially when we perform a tight grasp, this movement allows the opposition of the thumb to the little finger.

Furthermore, as can be seen in Figure 95, the carpal bones form two transverse rows- the proximal row and the distal row, between which is called the midcarpal joint. Though there is only slight movement between the carpals in each carpal row, a moderate gliding movement does exist in the midcarpal joint which can help form a fitted encircling contact between the palm and the object so as to perform a tight grasp.

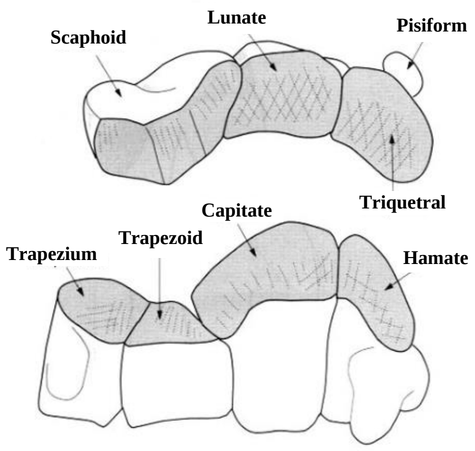


Figure 95. The midcarpal joint and its two single functional units (modified from reference [151])

Therefore, during the process of designing the physical prototype, we consider the carpal bones as two units—proximal and distal row, owing to the little relative movement between the carpal bones in each unit. On the other hand, for the reason that the index and middle metacarpals are nearly immobile, the metacarpal bones of the index finger and the middle finger connect to the carpals with fixed joints, but the ones of the ring finger and the little finger are different. They are fixed with some ligaments, allowing them to glide within certain limits.

Thus, the whole skeletal structure of the robotic hand is shown in Figure 96 (a). The metacarpal bones of the index and middle finger were printed together with the distal row of the carpals, shown as area 2. And the proximal row of the carpals was printed together with the wrist base (area 3), hence there will be no relative motion between the whole hand and the wrist. And area 1 in the figure represents the phalanges part of the prototype. Moreover, the distribution of the ligaments throughout the metacarpal-carpal bones is shown in Figure 96 (b). This design highly replicates the ligamentous-skeletal structure of the human hand. The CAD model of the skeletal structure of the robotic hand with ligaments and tendons insertions holes is shown in Appendix V.

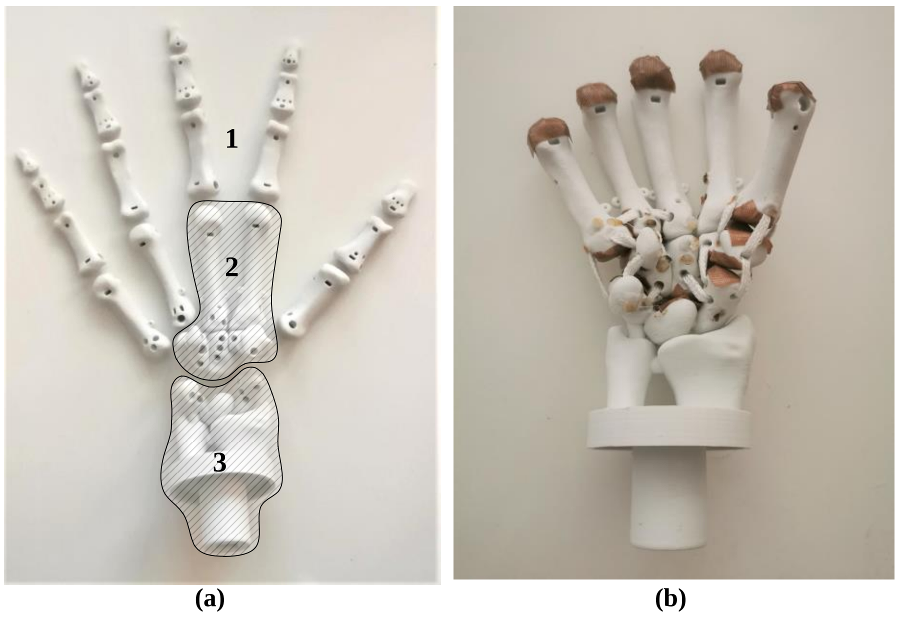


Figure 96. Skeletal model and carpal ligaments of the robotic hand. (a) Three parts of 3D printed skeletal models of the robotic hand. (b) Ligaments connection among the carpals and metacarpals.

In addition, there is a band-shape transverse carpal ligament structure on the palmar side of the hand near the wrist, covering over the carpal bones and forming a carpal tunnel which all the flexor tendons pass through [152]. On the other hand, it can hold all the flexor tendons to lie in the tunnel near the wrist, leading the tendons in proper directions. Since the proximal row of the carpal bones is designed and printed together, the impact on the tendons is only considered when we design the model. In order to make the transverse carpal ligament attached to the carpals more firmly, a cover with the band-shape ligament was designed and manufactured with silicone rubber and PTFE tape. As Figure 97 shows, the carpal cover was made from a carpal-shape mould which was brushed with liquid silicone rubber. After the silicone rubber was cured into a particular shape, a band of PTFE tape is rolled up into a cylinder shape and buried inside the silicone rubber. In this way, all the flexor tendons can be gathered and restrained near the wrist better, as shown in Figure 97 (right).

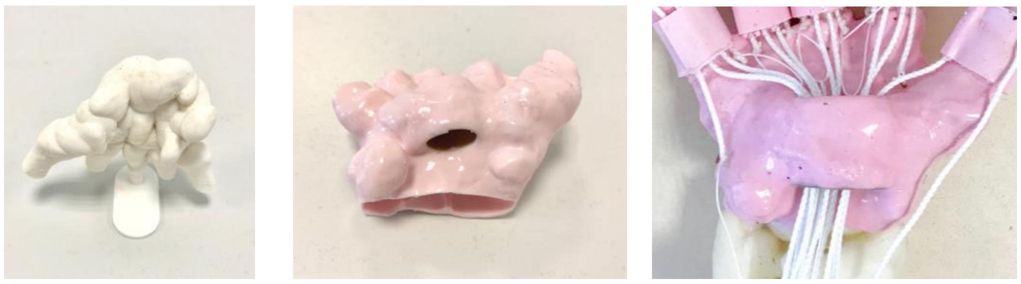


Figure 97. The carpal mould and the carpal cover with a band-shape ligament

6.2 Tendon distribution

As demonstrated in the literature review, there are two sets of tendons originating and branching out from the extrinsic muscle groups in the forearm and inserting to the base of PIP and DIP joints of the fingers-the flexors which bend the fingers and the extensors which straighten the fingers. A schematic diagram of the cross-section view of the human hand is shown in Figure 90, describing the distribution of the tendon tunnels around the wrist. We aimed to replicate all these external muscle tendons except the tendons actuating the wrist since the wrist is designed unmovable in this prototype.

Besides, the intrinsic muscles also play a crucial role in precise hand manipulation and power grasp, especially realizing thumb functions. There are nearly 20 intrinsic muscles distributing among the metacarpal and carpal bones and six of them drives the thumb whose volumes are also relatively huge compared with other intrinsic muscles. We found that some intrinsic muscles perform similar functions and some can be regarded as the combined results of another two or three muscles' actuation. To reduce the volume and complexity of the actuation system, we precisely replicated the major intrinsic muscle groups and integrated several intrinsic muscle tendons together or with the external muscle tendons into one tendon without losing the basic actuation functions. Specifically, for one finger, the palmar, dorsal interossei and the lumbrical muscle are mainly responsible for abduction-adduction and slightly assisting flexion. We integrated these muscle tendons into two tendons, called abductor and adductor. And the ulnar abductor of the middle finger is integrated with the adductor of the ring finger into one tendon, the abductor of the ring finger is integrated with the adductor of the little finger into one tendon. For the thumb, the abductor pollicis brevis, the opponens pollicis are integrated with the abductor pollicis longus into one thumb abductor tendon, the flexor pollicis brevis and the adductor pollicis are integrated into one thumb adductor tendon. For the little finger, the flexor digiti minimi brevis and the opponens digiti minimi are integrated into one little opponens tendon.

Considering the tendons and motors arrangement, a base with 19 guiding holes were designed and printed together with the wrist to lead the tendons in proper directions. The vertical view of the wrist base is shown in Figure 99. As can be seen, the external muscle tendons are mostly preserved and the integrated tendons are signed by blue texts.

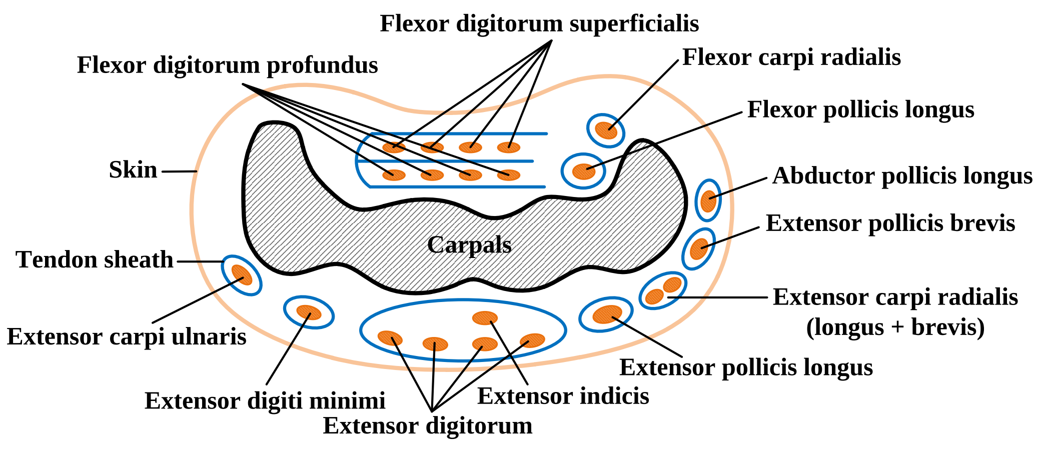


Figure 98. The cross-section view of the human hand near the wrist

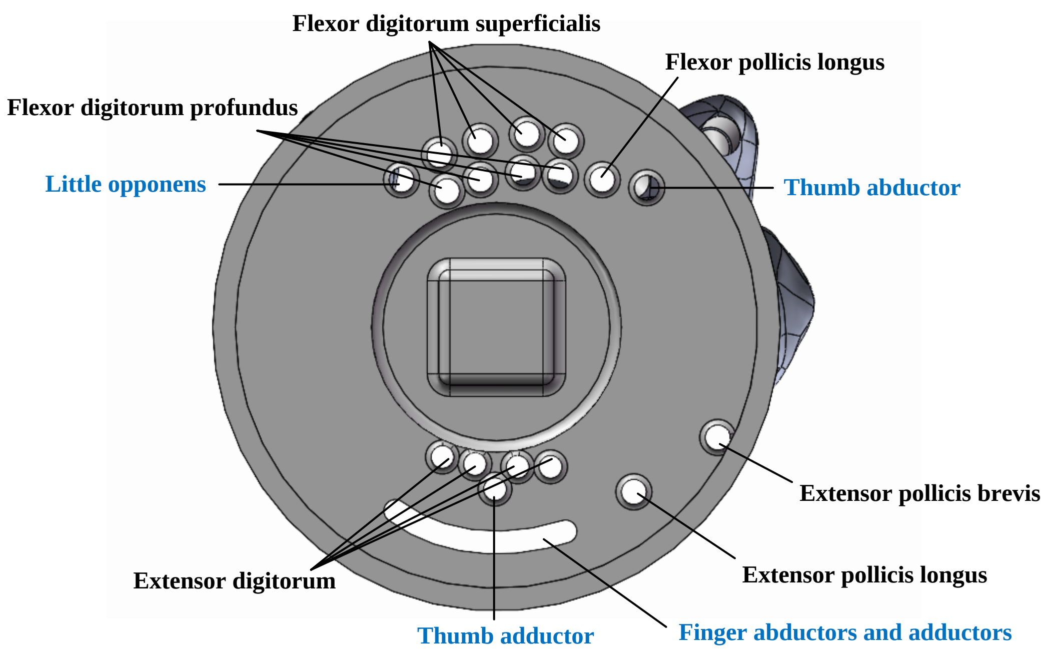


Figure 99. Robotic hand wrist base model with guiding holes

To fabricate the prototype of the robotic hand, a series of the joint capsule and tendon sheath moulds in different sizes were designed and 3D printed. Using the same silicone rubber curing method, the joint capsules and tendon sheaths fitted for different finger joints and phalanges were made, as shown in Figure 100. The assembling prototype with bones, ligaments, capsules, tendons and tendon sheaths is presented in Figure 101. In fact, this is the final version of the robotic hand design after four upgrades. All the five generations of the robotic hand are shown in Appendix VI.

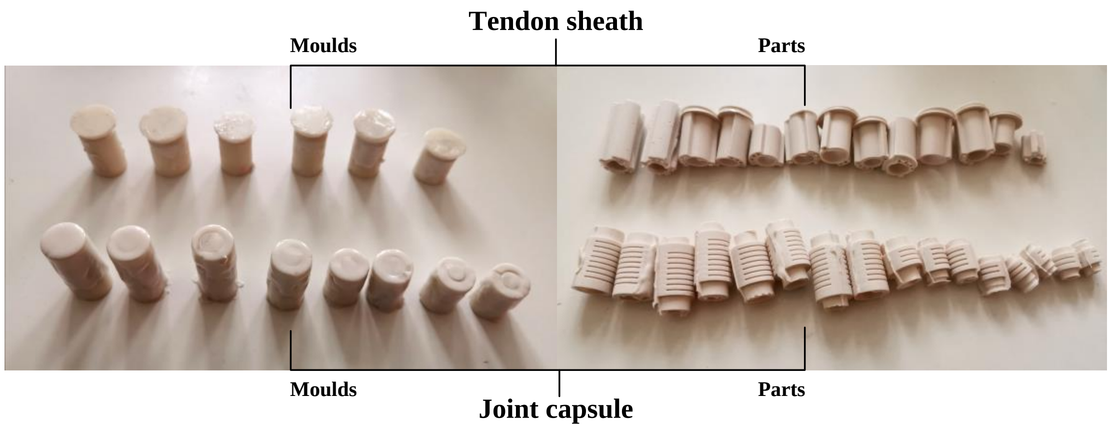


Figure 100. Joint capsule and tendon sheath moulds and cured parts

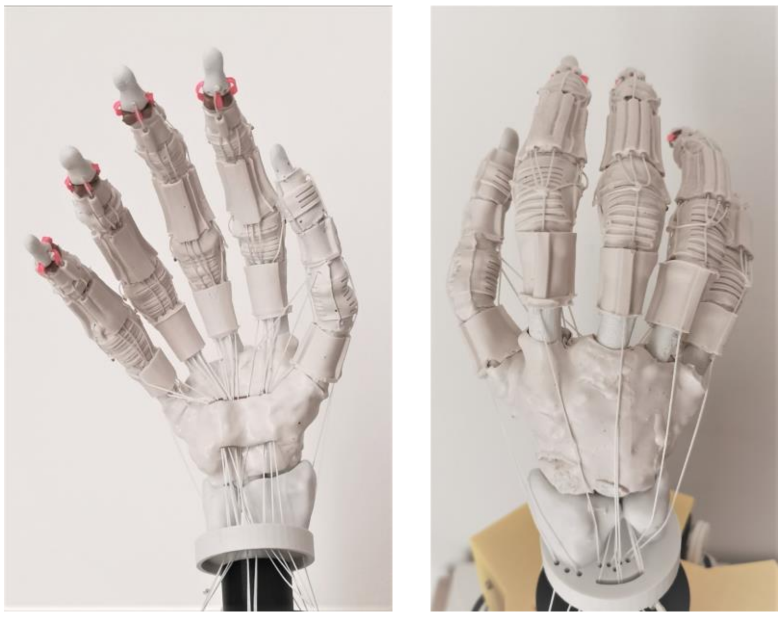


Figure 101. Assembled robotic hand model without skin

6.3 Artificial skin

The skin plays the function of not only protecting the soft tissues in hands but also increasing the friction between the hand and the objects to complete grasping and manipulation tasks. Accordingly, the mechanical properties of human hand skin are expected to be performed on the artificial skin for the robotic hand, not only replicating the appearance. Therefore, the artificial hand skin should be elastic and flexible.

Besides, there should be some wrinkles on the dorsal side of joint positions to reduce the tension resistance of the skin to the joint motion. Moreover, the fingerprint should also be preserved on the artificial skin to maintain the friction of the fingertips.

Considering all these conditions, a human-hand-based multi-layered silicone rubber curing and moulding method was adopted. Since the skeletal model was obtained through CT scanning from the right hand of a 23-year-old healthy male [127], the artificial skin was also moulded on the same human hand.

To make the mould, some vaseline was firstly brushed on the human hand to form a lubrication layer so that the mould can be easily removed from the human hand. Then the liquid silicone rubber Ecoflex-35 was evenly applied onto the human hand. After the first layer was cured, we applied the second layer. Normally this procedure should be repeated three to four times to make sure the hand was fully covered by a nearly 2mm thick silicone rubber layer. After that, we covered this silicone rubber layer with several pieces of water-soaked plaster bandages. After about ten minutes until the plaster bandages were totally dried and casted, we removed the plaster shell and silicone rubber layer together from the human hand. Prepared 80g liquid silicone rubber Ecoflex-10 mixed with several drops of fleshcolor pigment in a cup and poured it into the silicone rubber mould, and then slowly wobbled the mould to make Ecoflex-10 uniformly attached to the internal surface of the Ecoflex-35 silicone rubber layer. After the first layer was nearly cured onto the mould internal surface, we poured out the extra uncured liquid and poured into another 80g prepared Ecoflex-10. This procedure should be repeated three to four times. Until the last layer of Ecoflex-10 was thoroughly cured, we removed the silicone rubber layer out of the plaster shell and peeled Ecoflex-10 layer off the Ecoflex-35 layer. Finally, I got a 1mm thick artificial skin made with Ecoflex-10 silicone rubber. The plaster shell layer and the Ecoflex-35 layer were shown in Figure 102 and the artificial skin with zoomed surface texture is shown in Figure 103.

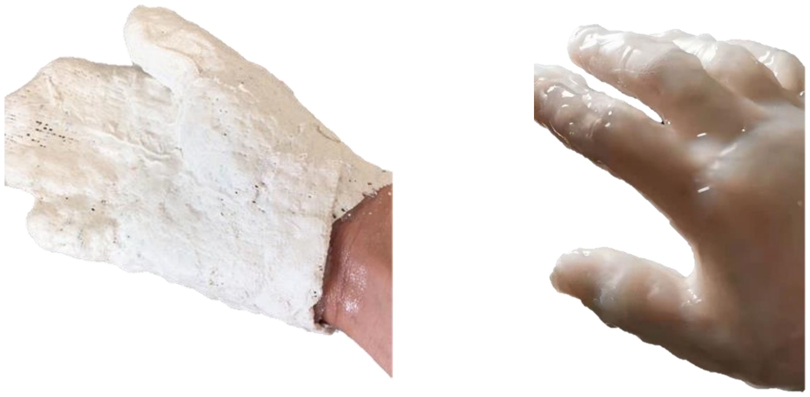


Figure 102. The plaster shell layer (1st layer) and the Ecoflex-35 silicone rubber layer (2nd layer) of the artificial skin mould

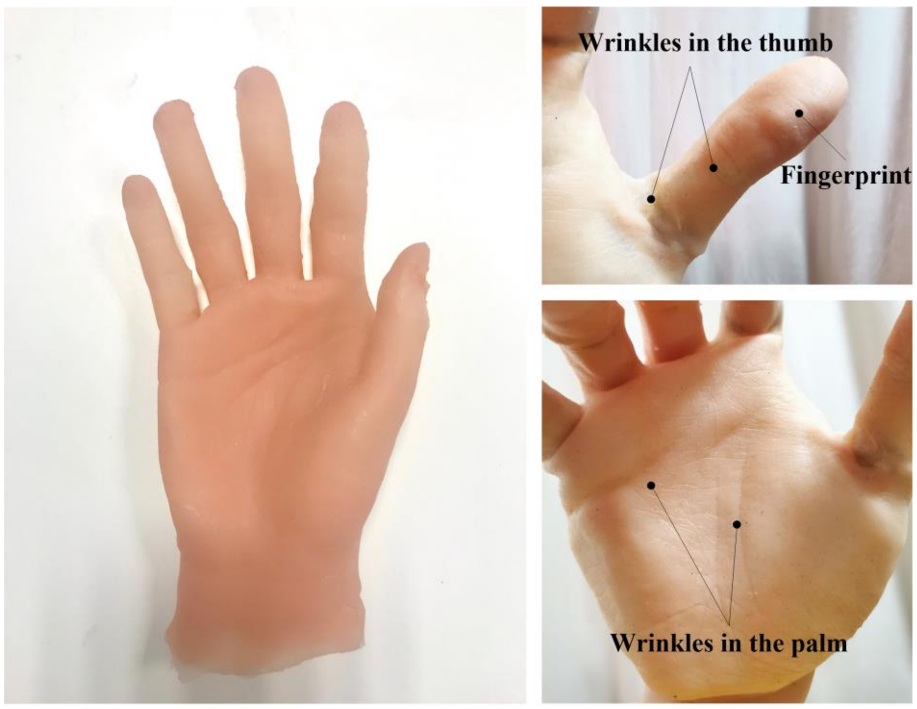


Figure 103. Artificial skin with fingerprint and wrinkles

6.4 Summary

Developed from the robotic finger, the design and fabrication process of the robotic hand were introduced in this chapter. The whole skeletal-ligamentous structure of the human hand was highly replicated in the robotic hand. The movement of the carpometacarpal joints and the midcarpal joint in the human hand was mostly reserved in the robotic hand by proper ligaments constraint. The muscle-tendon system was also implemented in the robotic hand but with some simplification such as integrating several tendons into one tendon. To simplify the actuation system, we connected all the extrinsic and intrinsic muscle tendons to the motors outside the hand with similar distribution of the tendons. Thus, a wrist base with 19 guiding holes was designed and 3D printed to lead the tendons in proper directions. In order to restore the appearance as well as the wrinkles around the joints and the fingerprint of the human hand, a 1mm thick Ecoflex-10 silicone rubber artificial skin was fabricated by using a human-hand-based multi-layered silicone rubber curing and moulding method. So far, the design and fabrication of the bio-inspired robotic hand were completed. In the next chapter, we are going to construct the actuation system and propose a control strategy for the robotic hand.

Chapter 7: Actuation system construction and control theory of the robotic hand

The actuation system and the control theory of the robotic hand are both developed from that of robotic fingers in chapter 5. In this chapter, a motor-tendon actuation system was constructed and the corresponding control theory of tendon-driven robotic finger with flexible tendon sheaths was introduced. And based on the control model, the data-glove-based position control was proposed.

7.1 Motor-tendon actuation system construction

The actuation system of the robotic hand consists of 24 Dynamixel motors, including sixteen MX-12W motors with 0.088°resolution and 0.2N·m (12*V* working voltage) stall torque and eight MX-28W motors with 55rpm no-load speed and 2.5N·m stall torque (12*V* working voltage). We chose the Dynamixel motors because of their small size and moderate actuation torque. Besides, they have many built-in sensors providing feedbacks of position, load, voltage, temperature, etc. From the aspect of system simplification, they can be connected with each other and simultaneously controlled by PC through an USB2Dynamiel converter.

To constrain the size of the actuation system, a customized modular actuation platform with human forearm length was designed to place the twenty-four motors. There are totally four modules and each module can accommodate six motors. The CAD model and the physical prototype of the robotic hand actuation platform are shown in Figure 104. And the CAD models of the actuation platform connector, as well as one module, are shown in Appendix IV.

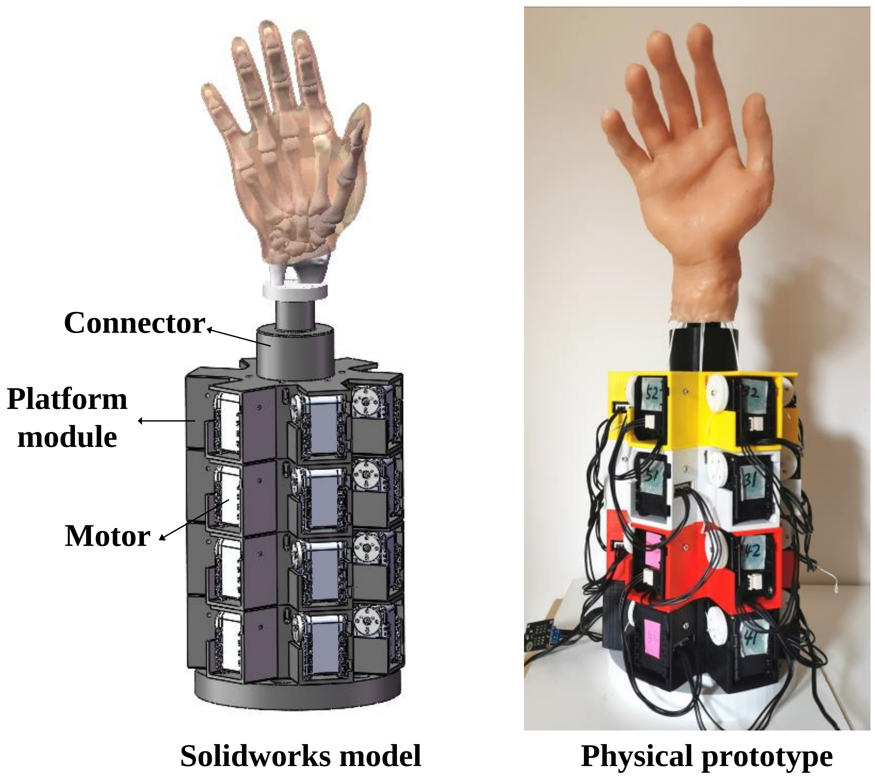


Figure 104. Assembled robotic hand model and prototype with actuation system

7.2 Control methods of the tendon-driven robotic finger with flexible tendon sheaths

There are two points that need to be considered when establishing the control model for the proposed robotic hand design. One is the action of the joint stiffness when the finger is actuated by a single tendon, and another is the action of the flexible tendon sheaths. Based on these conditions, two control models were established in this section.

7.2.1 Control of the single-tendon-driven robotic finger

According to the single-joint flexible tendon sheath model in chapter 4, the three joints finger model with flexible tendon sheaths can be simplified and established as shown in Figure 105.

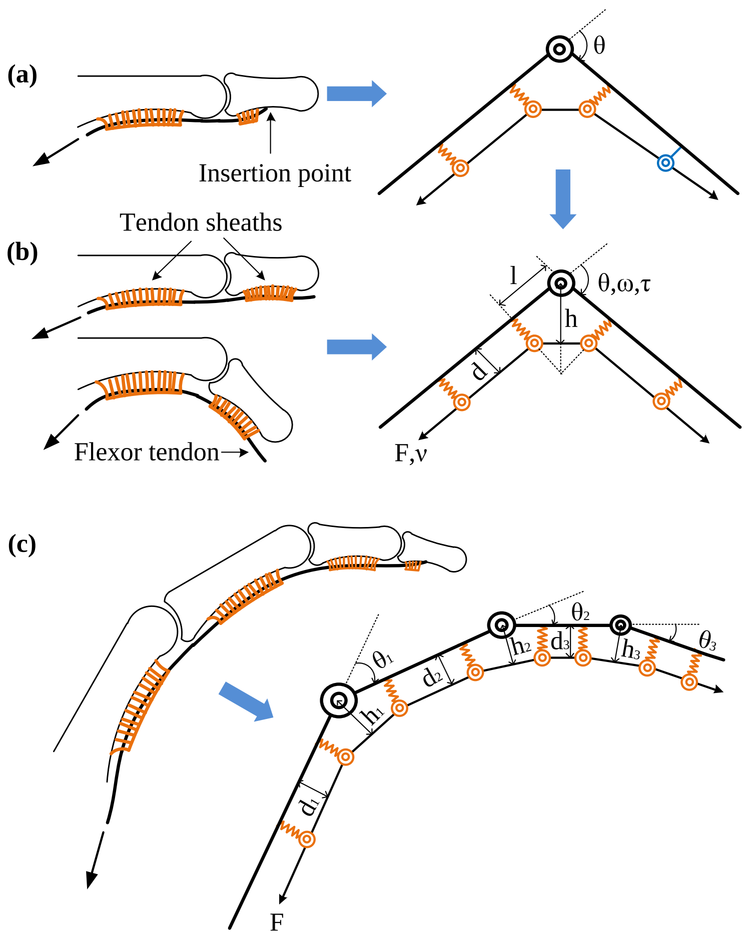


Figure 105. Simplified model of the single-tendon-driven robotic finger. (a) Model of the distal end joint with a tendon insertion. (b) Simplified model of model a. (c) Simplified model of the three-joint finger model with FDP tendon actuation.

Take FDP tendon as an example, it passes through three joints. The insertion is located at the distal end of the FDP tendon, as shown in Figure 105 (a). In this case, the tendon sheath model should involve a fixed rigid end part, however, which will dramatically complicate the problem. Therefore, we still consider it as the same case as the previous one, shown in Figure 105 (b). And for the FDP tendon, the tendon sheath model can be simplified as Figure 105 (c), which can also be considered as a combination of three groups of the model (b).

According to the analysis of the previous research flexible tendon sheath model, the distance *x* between the pulleys near the joint can be expressed as:

$$x = 2 \cdot \left(l \cdot \cos\frac{\theta}{2} - d_0 \cdot \sin\frac{\theta}{2} - \frac{F}{k} \cdot \sin^2\frac{\theta}{2} \right)$$

(56)

Thus the tendon excursion x_θ can be obtained when the joint angle increases from φ to θ with a constant tendon force F.

$$x_\theta = -2l \cdot \left(\cos\frac{\theta}{2} - \cos\frac{\phi}{2} \right) + 2d_0 \cdot \left(\sin\frac{\theta}{2} - \sin\frac{\phi}{2} \right) + \frac{2F}{k} \cdot \left(\sin^2\frac{\theta}{2} - \sin^2\frac{\phi}{2} \right)$$

(57)

Where a positive x_θ means the tendon is elongated, and a negative x_θ means the tendon is shortened.

To simplify the case, the distance l, the initial height of the tendon sheaths d_0 and the elastic coefficient k are set to be the same in these three joints. Thus, if the MCP, PIP and DIP joints move from the initial position φ_1, φ_2, φ_3 to the target position θ_1, θ_2, θ_3, the total excursion of the FDP tendon can be obtained.

$$x_{\theta t} = x_{\theta 1} + x_{\theta 2} + x_{\theta 3}$$

(58)

However, if we want to precisely control the angle of each joint, the excursion control of only one tendon is definitely not enough for three joints since it will bring infinite solutions. From the grasping test of the robotic fingers, we found one tendon actuation can actually realize the repeatable flexion motions since the joints have intrinsic stiffness resulted by the joint ligaments and capsules. Thus, the joint stiffness needs to be considered in this case and the minimum potential energy method was adopted to solve this problem.

According to the minimum potential energy theory, the total potential energy of the system Π is the sum of the strain energy of the system *U* and the total potential energy of the external force *W*. To minimize the value of *Π*, we need to set its first-order derivative as zero. Only in this condition, the system tends to be stable. For the robotic finger actuated by one tendon, there should be only one solution for the three joint angles as long as their stiffness is identified.

As to the finger model in Figure 105 (c), whose initial posture is with φ_1 joint 1 angle, φ_2 joint 2 angle and φ_3 joint 3 angle and the initial tendon force is zero, the strain energy of the system *U* contributed by the elastic tendon sheath and the joint stiffness can be expressed as:

$$U=\left(3\cdot\frac{1}{2}\cdot k\cdot\Delta d_1^{\,2}+2\cdot\frac{1}{2}\cdot k\cdot\Delta d_2^{\,2}+3\cdot\frac{1}{2}\cdot k\cdot\Delta d_3^{\,2}\right)+\left(\frac{1}{2}\cdot K_1\cdot\Delta\theta_1^{\,2}+\frac{1}{2}\cdot K_2\cdot\Delta\theta_2^{\,2}+\frac{1}{2}\cdot K_3\cdot\Delta\theta_3^{\,2}\right) \tag{59}$$

Where *k* is the elastic coefficient of the tendon sheaths, d_1, d_2, d_3 are the respective thickness of the tendon sheath group 1, group2 and group 3 after deformation, K_1, K_2, K_3 are the joint stiffness of joint 1, 2 and 3, respectively, and θ_1, θ_2, θ_3 are the target angles of joint 1, 2 and 3. Referring to Eq. (50) in Chapter 4, we have:

$$\begin{cases} d=d_0+\dfrac{F\cdot\sin\left(\dfrac{\theta}{2}\right)}{k}; & \Delta d_1=d_0+\dfrac{F\cdot\left(\sin\left(\dfrac{\theta_1}{2}\right)-\sin\left(\dfrac{\varphi_1}{2}\right)\right)}{k}; \\ \Delta d_2=d_0+\dfrac{F\cdot\left(\sin\left(\dfrac{\theta_2}{2}\right)-\sin\left(\dfrac{\varphi_2}{2}\right)\right)}{k}; & \Delta d_3=d_0+\dfrac{F\cdot\left(\sin\left(\dfrac{\theta_3}{2}\right)-\sin\left(\dfrac{\varphi_3}{2}\right)\right)}{k}; \\ \Delta\theta_1=\theta_1-\varphi_1; & \Delta\theta_2=\theta_2-\varphi_2; \qquad \Delta\theta_3=\theta_3-\varphi_3 \end{cases} \tag{60}$$

The total potential energy of the external force in this model comes from the actuation force on the tendon, it has:

$$W = x_{\theta t} \cdot F = \left(x_{\theta_1} + x_{\theta_2} + x_{\theta_3} \right) \cdot F$$

(61)

Where F is the tendon force after the finger moves to the target posture, $x_{\theta 1}$, $x_{\theta 2}$, $x_{\theta 3}$ are the tendon excursion at joint 1, 2 and 3, respectively. According to Eq. (57), it has:

$$\begin{cases} x_{\theta_1} = -2l\left(\cos\frac{\theta_1}{2} - \cos\frac{\varphi_1}{2}\right) + 2d_0\left(\sin\frac{\theta_1}{2} - \sin\frac{\varphi_1}{2}\right) + \frac{2F}{k}\sin^2\frac{\theta_1}{2} \\ x_{\theta_2} = -2l\left(\cos\frac{\theta_2}{2} - \cos\frac{\varphi_2}{2}\right) + 2d_0\left(\sin\frac{\theta_2}{2} - \sin\frac{\varphi_2}{2}\right) + \frac{2F}{k}\sin^2\frac{\theta_2}{2} \\ x_{\theta_3} = -2l\left(\cos\frac{\theta_3}{2} - \cos\frac{\varphi_3}{2}\right) + 2d_0\left(\sin\frac{\theta_3}{2} - \sin\frac{\varphi_3}{2}\right) + \frac{2F}{k}\sin^2\frac{\theta_3}{2} \end{cases}$$

(62)

Then we have:

$$\Pi = U + W$$

(63)

To minimize the total potential energy of the robotic finger to all the joints, we should make

$$\frac{\partial \Pi}{\partial \theta_1} = 0, \frac{\partial \Pi}{\partial \theta_2} = 0, \frac{\partial \Pi}{\partial \theta_3} = 0$$

(64)

which can be formulated as:

$$\begin{cases} \dfrac{3F^2}{2k}\sin\dfrac{\theta_1}{2}\cos\dfrac{\theta_1}{2}+K_1\left(\theta_1-\varphi_1\right)+F\left(-l\sin\dfrac{\theta_1}{2}-d_0\cos\dfrac{\theta_1}{2}-\dfrac{2F}{k}\sin\dfrac{\theta_1}{2}\cos\dfrac{\theta_1}{2}\right)=0 \\ \dfrac{3F^2}{2k}\sin\dfrac{\theta_2}{2}\cos\dfrac{\theta_2}{2}+K_2\left(\theta_2-\varphi_2\right)+F\left(-l\sin\dfrac{\theta_2}{2}-d_0\cos\dfrac{\theta_2}{2}-\dfrac{2F}{k}\sin\dfrac{\theta_2}{2}\cos\dfrac{\theta_2}{2}\right)=0 \\ \dfrac{3F^2}{2k}\sin\dfrac{\theta_3}{2}\cos\dfrac{\theta_3}{2}+K_3\left(\theta_3-\varphi_3\right)+F\left(-l\sin\dfrac{\theta_3}{2}-d_0\cos\dfrac{\theta_3}{2}-\dfrac{2F}{k}\sin\dfrac{\theta_3}{2}\cos\dfrac{\theta_3}{2}\right)=0 \end{cases} \tag{65}$$

In these equations, the values of k, l, d_1, d_2, d_3, K_1, K_2, K_3 are constants which are only related to the structure and material properties of the robotic finger. Besides, the values of φ_1, φ_2, φ_3 are already identified according to the initial posture of the robotic finger. Thus, with Eqs. (59), (60), (61), (62), (63) and (65), the target joint θ_1, θ_2, θ_3 can be uniquely obtained with a specific tendon total excursion input x.

From the analysis above, it should be noted that the finger with one tendon actuation is possibly not able to achieve some certain joint angle groups since the relationship between the tendon excursion and the three target joint angles depends on the joint stiffness properties. If the joint stiffness is constant or changes with a fixe trend (such as change with joint angles), it’s inevitable that some joint angles cannot be covered. However, it makes the control strategy easy and it’s enough for satisfying the simple grasp motions.

From another aspect, if the relationship between the tendon excursion and the joint angles is measured from the experiment, each joint stiffness property can be obtained. This method can be further used to reversely verify or optimize the joint stiffness of the designed model.

And when the finger touches the object, the normal contact force may need to be increased to f to avoid slip while the posture of the finger keeps stable. In this condition, the FDP tendon force *F* should increase from *F₁* to *F₂*, where the force *F₂* needs to meet the condition that:

$$\frac{F_2 \cdot h_1}{y_1} + \frac{F_2 \cdot h_2}{y_2} + \frac{F_2 \cdot h_3}{y_3} = f \tag{66}$$

In the equation (66),

$$\begin{cases} y_1 = l_1 \cdot \sin(\theta_1) + l_2 \cdot \sin(\theta_1 + \theta_2) + l_3 \cdot \sin(\theta_1 + \theta_2 + \theta_3) \\ y_2 = l_2 \cdot \sin(\theta_1 + \theta_2) + l_3 \cdot \sin(\theta_1 + \theta_2 + \theta_3) \\ y_3 = l_3 \cdot \sin(\theta_1 + \theta_2 + \theta_3) \end{cases} \tag{67}$$

Where h_i is the moment arm of the tendon force with respect to the joint i, and *yi* is the moment arm of the fingertip normal contact force with respect to the joint i, as shown in Figure 105 (c).

And according to equation (56), the tendon excursion x_f caused by the tendon force increasing from F_1 to F_2 can be expressed as:

$$x_f = -2 \cdot \frac{\sin^2 \frac{\theta}{2}}{k} \cdot (F_2 - F_1) \tag{68}$$

Therefore, the total tendon excursion resulted by the tendon force changing can be obtained.

$$x_{ft} = x_{f1} + x_{f2} + x_{f3} \tag{69}$$

Thus, the total tendon excursion in the whole approaching and contacting process with both joint angles and tendon forces involved is shown as follows:

$$x_t = x_{\theta t} + x_{ft} \tag{70}$$

Eventually, according to this finger model with flexible tendon sheaths and hinge joints, the target joint angles and fingertip contact force can be potentially reached by controlling the tendon excursion.

7.2.2 Control of the multi-tendon-driven robotic finger

However, there are two flexor tendons and one extensor tendon actuating one human finger so as to perform many complex manipulations. For the robotic finger, we can also adopt the same actuation method to improve its dexterity and manipulation capability. Therefore, a multi-tendon-driven robotic finger model was established and the corresponding control strategy was derived.

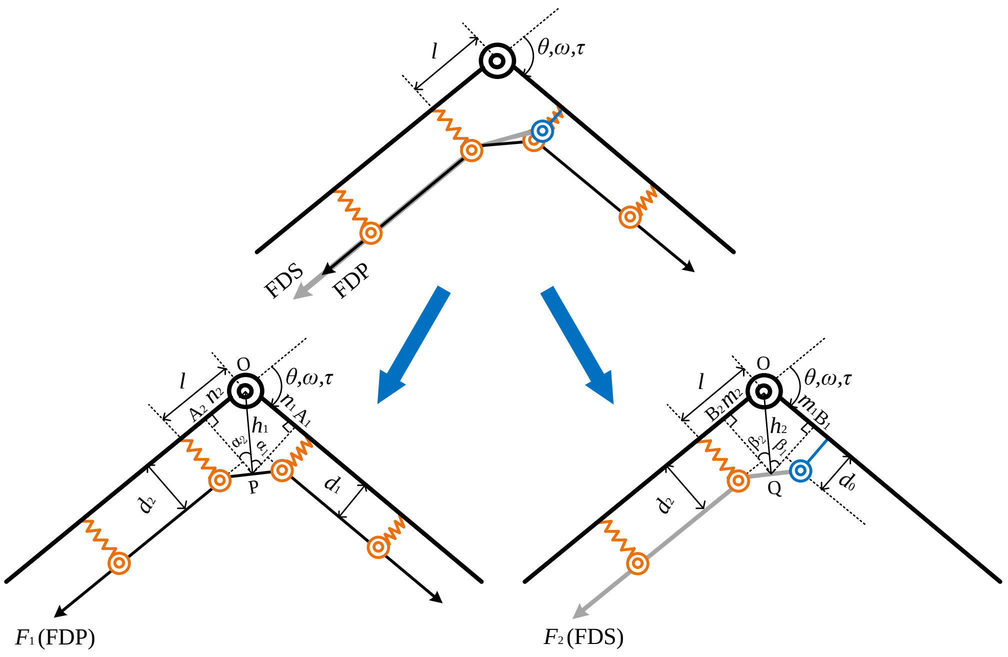


Figure 106. Single-joint model with two flexor tendons

In the case of two flexor tendons actuation, there must be a situation that two tendons pass one joint. As shown in Figure 106 (top figure), the joint is actuated by two flexor tendons-FDP and FDS. The tendon sheaths on the left phalanx are stretched by the forces of two tendons and the tendon sheaths on the right phalanx are only stretched by FDP tendon, resulting in different deformation of these two tendon sheath groups. Therefore, the case can be separated into two specific cases and there is only one tendon acting on the joint in each case, shown as the bottom two figures in Figure 106.

In the joint actuated by FDP tendon (bottom left in Figure 106), the tendon has a d_1 distance away from the right body, and a d_2 distance from the left body. The distance from the elastic pulley edges on both sides to the joint is l. The joint rotation axis is expressed as O. Make a perpendicular from point O to the tendon between the two pulleys close to the joint, forming a foot point P. The line segment OP is the moment arm from the tendon to the joint, whose length is h_1. Then make two perpendicular lines from point P to the right body with a foot point A_1 and to the left body with a foot point A_2. The length of OA_1 is identified as n_1, and OA_2 as n_2. The angle of $\angle OPA_1$ is α_1, and $\angle OPA_2$ is α_2.

In the joint actuated by FDS tendon, the insertion of FDS is located on a fixed point which is d_0 away from the body, from which the distance to the rotation axis is also l, shown as the blue pulley in Figure 106 (bottom right). With the similar definition of the geometrical relation, the line segment OQ is the moment arm from the FDS tendon to the joint with the length of h_2. With QB_1 perpendicular to OB_1 and QB_2 perpendicular to OB_2, the length of OB_1 is identified as m_1, and OB_2 as m_2. The angle of $\angle OQB_1$ is β_1, and $\angle OQB_2$ is β_2.

According to the geometrical relations identified in the figure, it has:

$$\begin{cases} \dfrac{\dfrac{n_1}{\tan\alpha_1}-d_1}{l-n_1}=\tan\alpha_1; & \dfrac{\dfrac{n_2}{\tan\alpha_2}-d_2}{l-n_2}=\tan\alpha_2; & \dfrac{\sin\alpha_1}{\sin\alpha_2}=\dfrac{n_1}{n_2}; \quad \alpha_1+\alpha_2=\theta; \\ \dfrac{\dfrac{m_1}{\tan\beta_1}-d_0}{l-m_1}=\tan\beta_1; & \dfrac{\dfrac{m_2}{\tan\beta_2}-d_2}{l-m_2}=\tan\beta_2; & \dfrac{\sin\beta_1}{\sin\beta_2}=\dfrac{m_1}{m_2}; \quad \beta_1+\beta_2=\theta; \\ d_1=d_0+\dfrac{F_1\sin\alpha_1}{k}; \quad d_2=d_0+\dfrac{F_1\sin\alpha_2+F_2\sin\beta_2}{k} \end{cases}$$

(71)

Within these ten equations, there are ten unknown parameters which are *n₁*, *n₂*, *α₁*, *α₂*, *m₁*, *m₂*, *β₁*, *β₂*, *d₁*, *d₂*. These parameters can be uniquely expressed by the other known parameters since this equation set has a group of unique solution. The known parameters includes *l*, *d₀*, *k*, *F₁* and *F₂*, which depend on the initial condition of the model. And the joint angle *θ*, as the input of the system, also needs to be identified so as to obtain the corresponding tendon excursion. According to the geometrical relation, it also has:

$$\begin{cases} x_1=\dfrac{l-n_1}{\cos\alpha_1}+\dfrac{l-n_2}{\cos\alpha_2}; & x_2=\dfrac{l-m_1}{\cos\beta_1}+\dfrac{l-m_2}{\cos\beta_2}; \\ h_1=\dfrac{n_1}{\sin\alpha_1}; & h_2=\dfrac{m_1}{\sin\beta_1} \end{cases}$$

(72)

Where x_1 is the distance between the pulley edges on the sides of the joint with FDP tendon, x_2 is the distance between the pulley edges on the sides of the joint with FDS tendon, h_1 is the moment arm from FDP tendon to the joint, and h_2 is the moment arm from FDS tendon to the joint. In the equation set (72), all the parameters come from the equation set (71), which can be expressed by θ. Thus, x_1, x_2, h_1, h_2 can all be regarded as the implicit expression of θ. Therefore, the equation (72) can be converted into the form as follows:

$$\begin{cases} x_1 = f_1(\theta); \quad x_2 = f_2(\theta); \\ h_1 = g_1(\theta); \quad h_2 = g_2(\theta) \end{cases}$$

(73)

Where $f_1(\theta)$, $f_2(\theta)$, $g_1(\theta)$ and $g_2(\theta)$ are the functions of θ. Δx_1 and Δx_2 are the tendon excursion of FDP and FDS generated from the joint angle change, respectively. Thus, the relationship between the joint angle change and each tendon excursion is obtained.

Basing on this single-joint model, a 3-joints robotic finger actuated by three tendons is modelled as shown in Figure 107

In terms of the two flexor tendons, only joint 2 is in the same condition as the model in Figure 106, and the other two joints can be regarded as the model in Figure 105 with different tendon force applied. As for the long extensor tendon, we assumed that it attached close to the dorsal of finger phalanges. Thus, its tendon excursion at each joint can be calculated as the arc generated by the joint rotation angles with the rotation radius of each joint geometric radius.

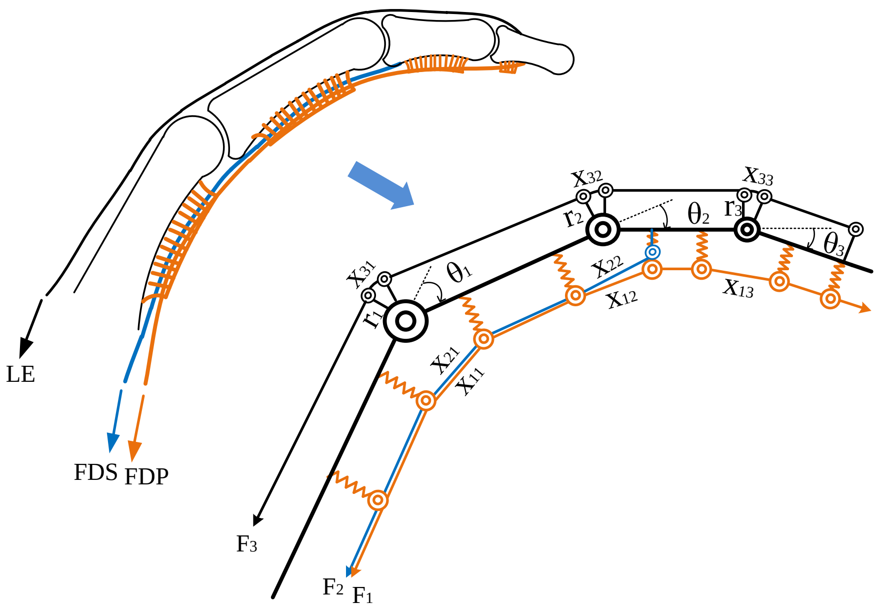


Figure 107. Simplified model of the robotic finger with three tendons actuation

With the Eqs. (56) and (73) it has:

$$\begin{cases} x_{11} = 2\cdot\left(l\cdot\cos\frac{\theta_1}{2} - d_0\cdot\sin\frac{\theta_1}{2} - \frac{F_1+F_2}{k}\cdot\sin^2\frac{\theta_1}{2}\right); \quad x_{12} = f_1(\theta_2); \quad x_{13} = 2\cdot\left(l\cdot\cos\frac{\theta_3}{2} - d_0\cdot\sin\frac{\theta_3}{2} - \frac{F_1}{k}\cdot\sin^2\frac{\theta_3}{2}\right); \\ x_{21} = 2\cdot\left(l\cdot\cos\frac{\theta_1}{2} - d_0\cdot\sin\frac{\theta_1}{2} - \frac{F_1+F_2}{k}\cdot\sin^2\frac{\theta_1}{2}\right); \quad x_{22} = f_2(\theta_2); \\ x_{31} = \theta_1\cdot r_1; \quad x_{32} = \theta_2\cdot r_2; \quad x_{33} = \theta_3\cdot r_3 \end{cases}$$

(74)

Thus, the total excursion of each actuation tendon can be obtained:

$$\begin{cases} \Delta x_1 = \Delta x_{11} + \Delta x_{12} + \Delta x_{13} \\ \Delta x_2 = \Delta x_{21} + \Delta x_{22} \\ \Delta x_3 = \Delta x_{31} + \Delta x_{32} + \Delta x_{33} \end{cases}$$

(75)

For instance, if the joint moves from the initial joint angles φ_1, φ_2, φ_3 to θ_1, θ_2, θ_3, then the tendon excursion of FDP, FDS and LE are:

$$\begin{cases} \Delta x_1 = 2l\left(\cos\frac{\varphi_1}{2} + \cos\frac{\varphi_3}{2} - \cos\frac{\theta_1}{2} - \cos\frac{\theta_3}{2}\right) - 2d_0\left(\sin\frac{\varphi_1}{2} + \sin\frac{\varphi_3}{2} - \sin\frac{\theta_1}{2} - \sin\frac{\theta_3}{2}\right) \\ \qquad - 2\frac{F_1+F_2}{k}\left(\sin^2\frac{\varphi_1}{2} - \sin^2\frac{\theta_1}{2}\right) - 2\frac{F_1}{k}\left(\sin^2\frac{\varphi_3}{2} - \sin^2\frac{\theta_3}{2}\right) + f_1(\varphi_2) - f_1(\theta_2) \\ \Delta x_2 = 2l\left(\cos\frac{\varphi_1}{2} - \cos\frac{\theta_1}{2}\right) - 2d_0\left(\sin\frac{\varphi_1}{2} - \sin\frac{\theta_1}{2}\right) - 2\frac{F_1+F_2}{k}\left(\sin^2\frac{\varphi_1}{2} - \sin^2\frac{\theta_1}{2}\right) + f_2(\varphi_2) - f_2(\theta_2) \\ \Delta x_3 = (\varphi_1 - \theta_1)r_1 + (\varphi_2 - \theta_2)r_2 + (\varphi_3 - \theta_3)r_3 \end{cases}$$

(76)

And if the value of tendon excursion is positive, it means the tendon needs to be shortened. Otherwise, a negative value of tendon excursion means the tendon needs to be elongated. In this case, the finger is flexed with positive Δx_1, Δx_2, and negative Δx_3 tendon excursion. Using multiple tendons to actuate the robotic hand is not only for performing various postures but also improving the force-production capabilities of the robotic hand which, in other words, enlarging the feasible force space [28]. The control model of the thumb is similar to the other fingers, except that there are only two joints, making the model even easier.

7.2.3 Data glove-based posture control of the robotic hand

The VMG 30 data glove (Virtual Realities, LLC, USA) was used to capture and record the hand's posture when grasp or manipulate the object. It can provide up to 30 high-accuracy joint-angle measurements, including flexion-extension angles of all the fingers and thumb joints, the relative abduction angles between every two fingers, the palm arch angle, and the orientation of the hand and wrist. Moreover, each fingertip is equipped with a quite thin pressure sensor, which will be used to test the human hand and the robotic hand fingertip pressure in future research. In this project, we mainly used its function of joint angle measurement.

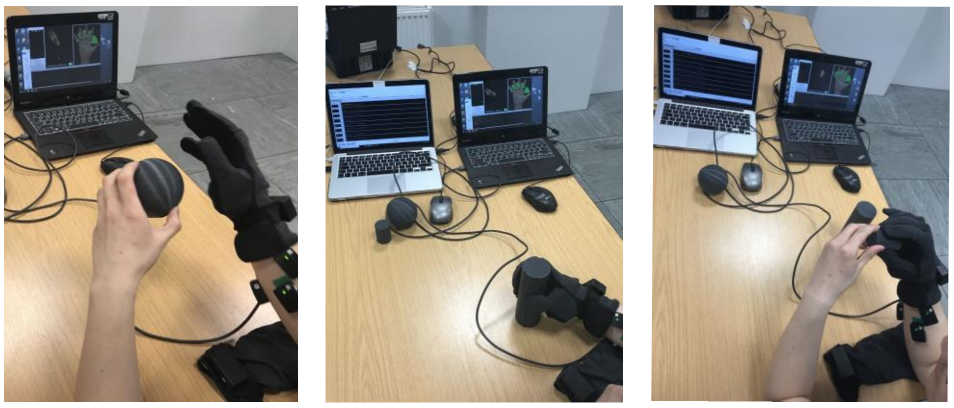


Figure 108. Posture identification of objects grasping with data glove

During the test, the subject was asked to wear one data glove to grasp a ball, a big cylinder and a small cylinder, which respectively corresponds to three postures spherical grasping, cylinder grasping and precision gripping, as shown in Figure 108. The customized software VMG_30 Firmware was used to show the motion status of the data glove and export the data to a csv format file. Once the data of finger joints angles were obtained, we input these joint angle values into the robotic hand control model (multi-tendon-driven). Then, we obtained each tendon's excursion for a specific hand posture. Theoretically, the robotic hand should perform the same posture as the subject's hand as long as the motors pull the tendons by these excursion values. To verify the validity of this data glove-based posture control strategy, the robotic hand grasping and manipulation functions based on this control method were tested in chapter 8.

7.3 Summary

To actuate the robotic hand, an actuation system with 24 Dynamixel motors was constructed. A modular motor accommodation platform was designed and 3D printed for easy assembling. Accordingly, the control models of tendon-driven robotic hand with flexible tendon sheaths were proposed in this chapter. In the simple grasp case, each finger of the robotic hand can be driven by only one tendon to realize the grasping (flexion) motion. In this case, the joint stiffness must be considered and the minimum potential energy theory was used to obtain the relation between one tendon excursion and three joints. However, the finger should be driven by multiple tendons to perform various postures during the manipulation. Thus, the multi-tendon-driven control model was proposed where the FDP, FDS and LE tendon were involved. Three tendons actuating three finger joints makes the kinematical equation have unique solutions. With the proposed control model, a data-glove-based position control strategy was developed for the function realization of the robotic hand. The angle of each finger joint was recorded by the data glove and substituted into the control model to obtain the excursion of each tendon basing on which the motor make the corresponding actuation. The next chapter will show the practical cases performed by the designed actuation system with this simple position control strategy.

Chapter 8: Human-hand-like grasping and manipulation

Grasping and manipulation capabilities are the basic functions of human hands and commonly used to evaluate the performance of robotic hands, which were demonstrated on the proposed robotic hand in this chapter. The 24 motors actuation system and the data-glove-based position control strategy introduced in the last chapter were used. The grasping test was based on the Cutkosky and Feix taxonomy. And the manipulation test was conducted basing on several common activities in daily lives.

8.1 Grasping capability test based on Cutkosky and Feix taxonomy

The Cutkosky and Feix grasping taxonomy are both widely used by researchers to test the grasping capability of the robotic hands.

The Cutkosky grasp taxonomy includes 16 grasp types which are classified based on the power/precision requirements, prehensile/non-prehensile contact, prismatic/circular grasp postures and the finger involved [153].

Extended from the previous grasp taxonomies, including the Cutkosky taxonomy, Feix grasp taxonomy contains more grasp types. Within this taxonomy, the grasp tasks are classified into 33 types according to the power/precision/intermediate requirements, the opposition types and the positions of thumb [154]. It is expected to test the grasping capability of the robotic hand more thoroughly.

Thus, the proposed robotic hand in this research was tested based on both of these two grasp taxonomies. The robotic hand was actuated by twenty-four motors with the data glove-based control strategy. The subject wore one data glove on his right hand and was asked to perform all the grasp tasks in these two taxonomies. The hand posture of each grasp type was recorded and the robotic hand was controlled to make the same posture by pulling the tendons based on the corresponding tendon excursion-joint angle relations.

The result is shown in Figure 109 and Figure 110. It can be seen that the proposed robotic hand completed all the 16 grasps in Cutkosky taxonomy and 33 grasps in Feix taxonomy. From the figure, we can see these grasps involve a wide variety of articles for daily use, including the pen, the cup, the baseball, the disk, the needle, the ID card, the coin, etc. These objects are with different shapes, dimensions and weights, and some of them are relatively heavy, such as the solid wood stick. The good performance of the robotic hand in the test indicates its great grasping capability. The videos in Appendix I (20~22) show several grasping cases.

However, the tests based on the Cutkosky and Feix taxonomy demonstrated a kind of passive grasping capability that needs the grasping objects to be properly placed into the robotic hand so that a successful grasp can be performed. To test its active grasping capability, a dynamic grasping experiment was conducted of which the process is shown in Figure 111. The robotic hand was mounted on a robotic arm which provided the up and down motion. The robotic arm control strategy for the robotic hand dynamic grasping is the same as that of the robotic finger grasping capability test in section 5.4. As can be seen from Figure 111, there are four objects with different shapes prepared for the test, which are a roll of small tape, a pen, a 3D printed ball and a 3D printed cylinder. The whole dynamic grasp process involves six steps, including preparing, opening the robotic hand, moving down to touch the object, grasping, adjusting (or interacting), and moving up to lift the object. During these four different objects grasping tests, the control command for the robotic hand is the same which are opening (extending the fingers) and closing (flexing the fingers). The demonstration videos of these four grasping processes are shown in Appendix I (16~19). And the ball grasping Matlab control code of the robotic hand can be found in Appendix VII.

The result shows that the robotic hand successfully grasped and lifted up the objects from the table in all four tests. It should be noted that there is no anti-collision or collision-reaction feedback algorithm programmed in the control strategy of the robotic hand. It all depends on the mechanical structure of the robotic hand to cope with the interaction of collision. This behaviour is obviously demonstrated in small objects grasping, such as grasping a roll of small tape and a pen. Besides, during the grasping, the position and orientation of the object are not precisely controlled but the initial position and posture of the robotic hand are predetermined, resulting in that the relative position and attitude between the robotic hand and the object may not be conducive to grasp. However, the proposed robotic hand can automatically adjust the relative position and attitude by allowing slight passive deformation during the interaction with the object. This behaviour can be easily observed in the pen and ball grasping tests, where the pen's orientation and the ball's position are adjusted before the whole grasping process is completed. The result also indicates that the robotic hand developed from the robotic finger can also perform strong adaptability.

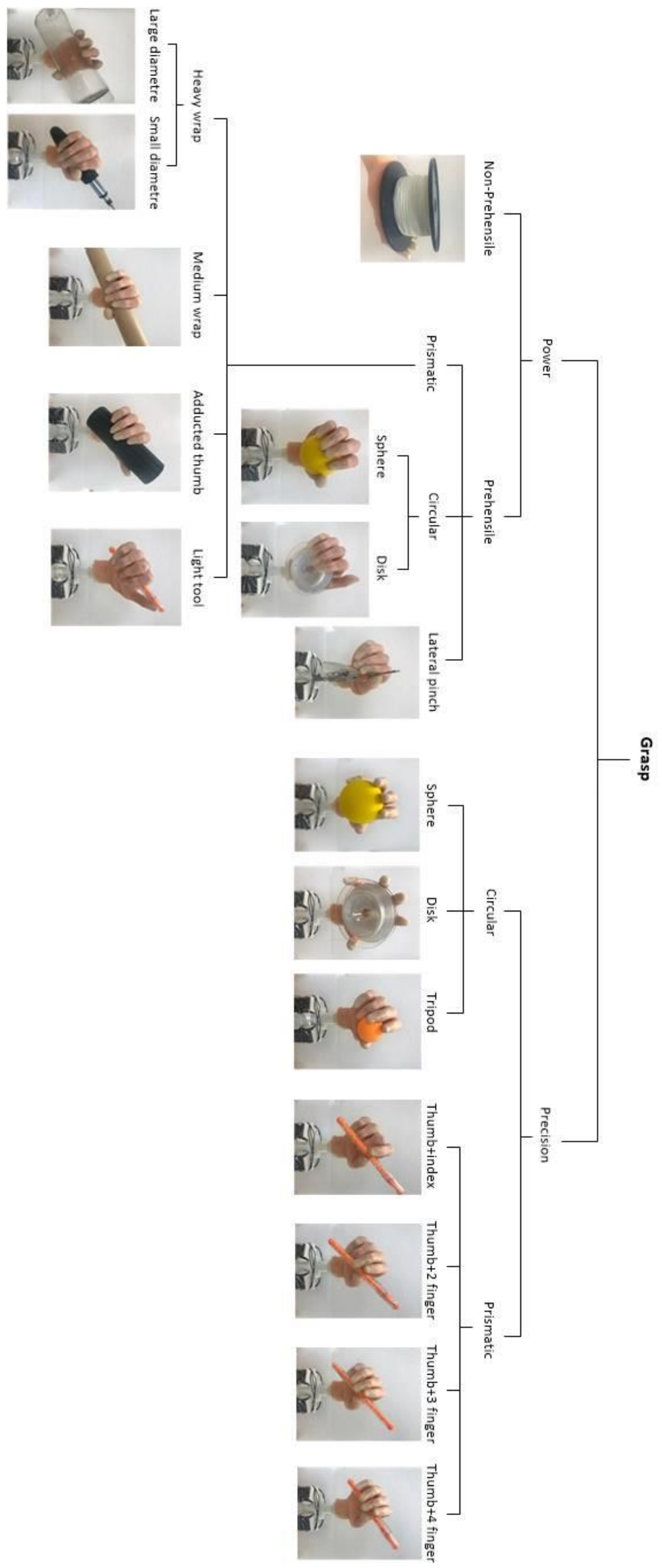


Figure 109. Grasping test based on Cutkosky taxonomy

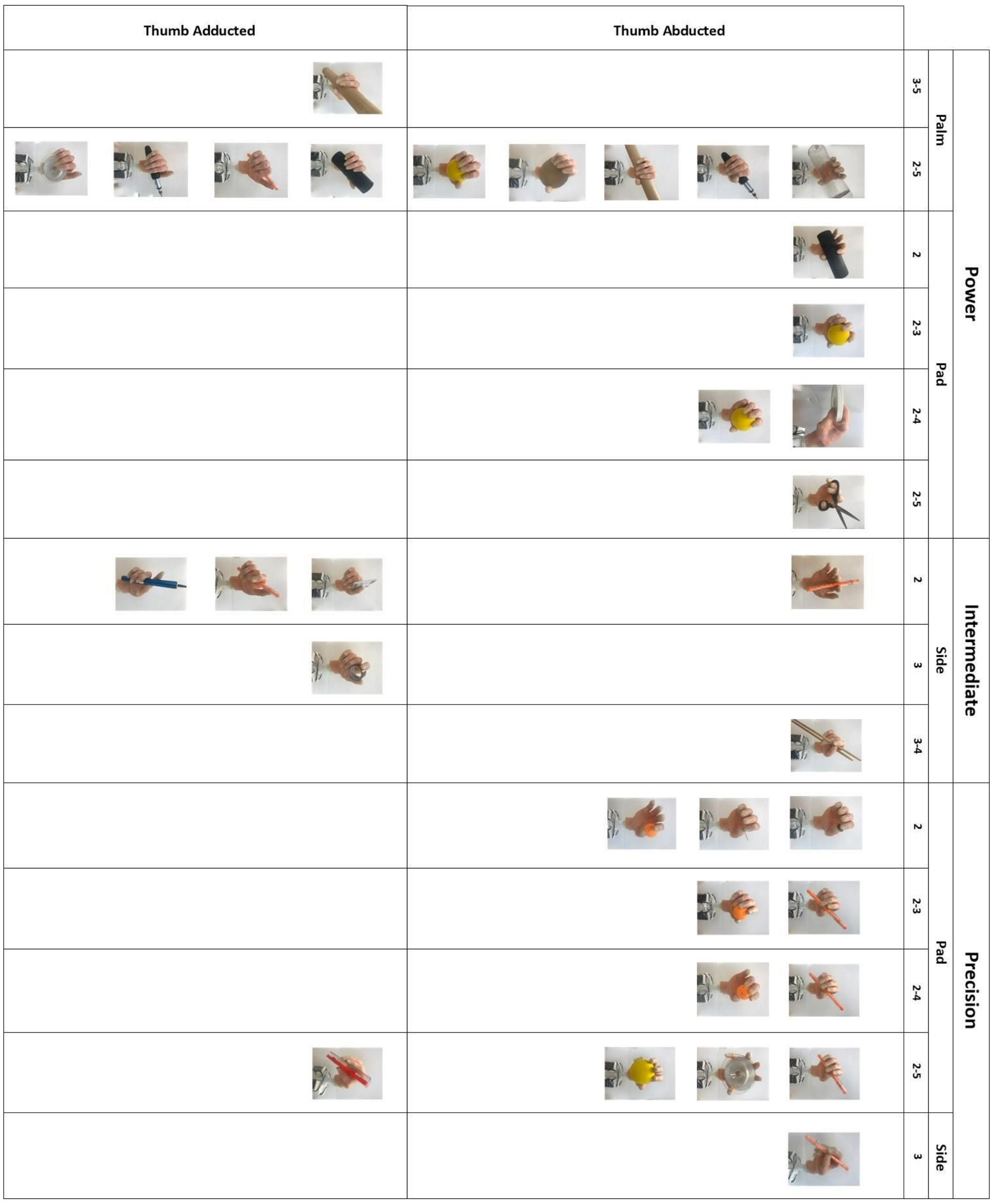


Figure 110. Grasping test based on Feix taxonomy

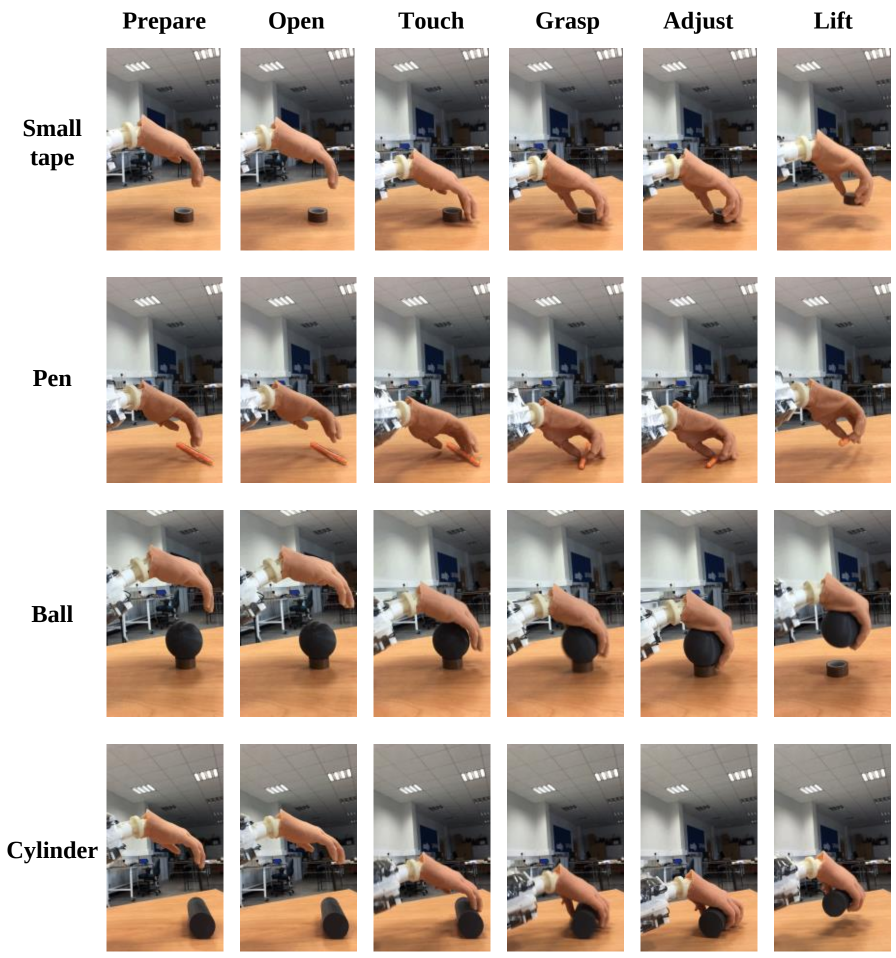


Figure 111. Dynamic grasping process of four objects

8.2 Manipulation capability test

To demonstrate the manipulation capability of the proposed robotic hand, we select several common tasks in daily lives to conduct the test, such as screwing off the bottle cap, using chopsticks, using scissors, pressing in the syringe, crumpling up a piece of paper and fanning out the playing cards. To a certain extent, these proposed tasks can test the finger synergetic capabilities, dexterity as well as force-applying capabilities of the robotic hand.

The control strategy is still based on the data glove posture measurement and the multi-tendon-driven control model. Unlike the grasping test, the manipulation test involves a series of different hand postures during the motion. To make a simplification, the consecutive motion is divided into several sequential or repetitive steps with different postures. The transition section between every two postures will be completed by the interaction between the robotic hand and the object. In this case, the data glove only needs to record a small number of postures for the robotic hand control.

The manipulation process of each test is shown in Figure 112. As can be seen, some of the manipulations need several motion cycles to accomplish the tasks, including screwing off the bottle cap (5 cycles), using scissors (6 cycles), crumpling a piece of paper (4 cycles) and fanning out the playing cards (5 cycles). In these cases, there are a few steps in one motion cycle and they are repeated in the following cycles. Take the case of screwing off the bottle cap as an example, there are five motion cycles throughout the manipulation. In each cycle, there are three steps: open the hand, hold the cap, and screw the cap. These steps are repeated five times until the bottle cap is screwed off eventually. Moreover, some of the manipulations only need a series of coherent steps, such as using chopsticks and pressing in the syringe. To complete the manipulation of pressing in the syringe, five steps are needed which are preparing the clipping posture, separating the index and middle fingers, clipping the syringe, adjusting the syringe to the working position and pressing in the syringe.

The robotic hand completed all the manipulations. Some tasks need to be accomplished by two hands, which are also successfully performed by the robotic hand with the assistance of the human hand. For instance, to screw off the bottle cap, the human hand needs to grasp the bottle. Besides, to use the scissors to cut a piece of paper, the paper also needs to be properly held by the human hand. Likewise, in the manipulation of fanning out the playing cards, the human hand needs to pinch the bottom corner of the playing cards like we normally do in our daily lives. However, this assistance from the human hand just plays the basic role of grasping or pinching, which does not influence the test of the robotic hand manipulation capabilities. The videos to show all these manipulations can be found in Appendix I (23~28). And the Matlab code for fanning out the playing cards of the robotic hand is presented in Appendix VIII.

In addition, it should be noted that these manipulations are completed by the proposed robotic hand only with the position control. As mentioned, this simple control strategy requires little computational cost for the reason that it separates the manipulation motion into several static postures. And the fact that the complex manipulation can be easily realized by this simple position control strategy benefits from the rigid-flexible bio-inspired design of the proposed robotic hand. The intrinsic passive behaviour of the structure helps the robotic hand smoothly and naturally complete the interaction with the object in the transition section between every two postures, implying that the bio-inspired design of the proposed robotic hand can bring some practical functional advantages with reduced control complexity.

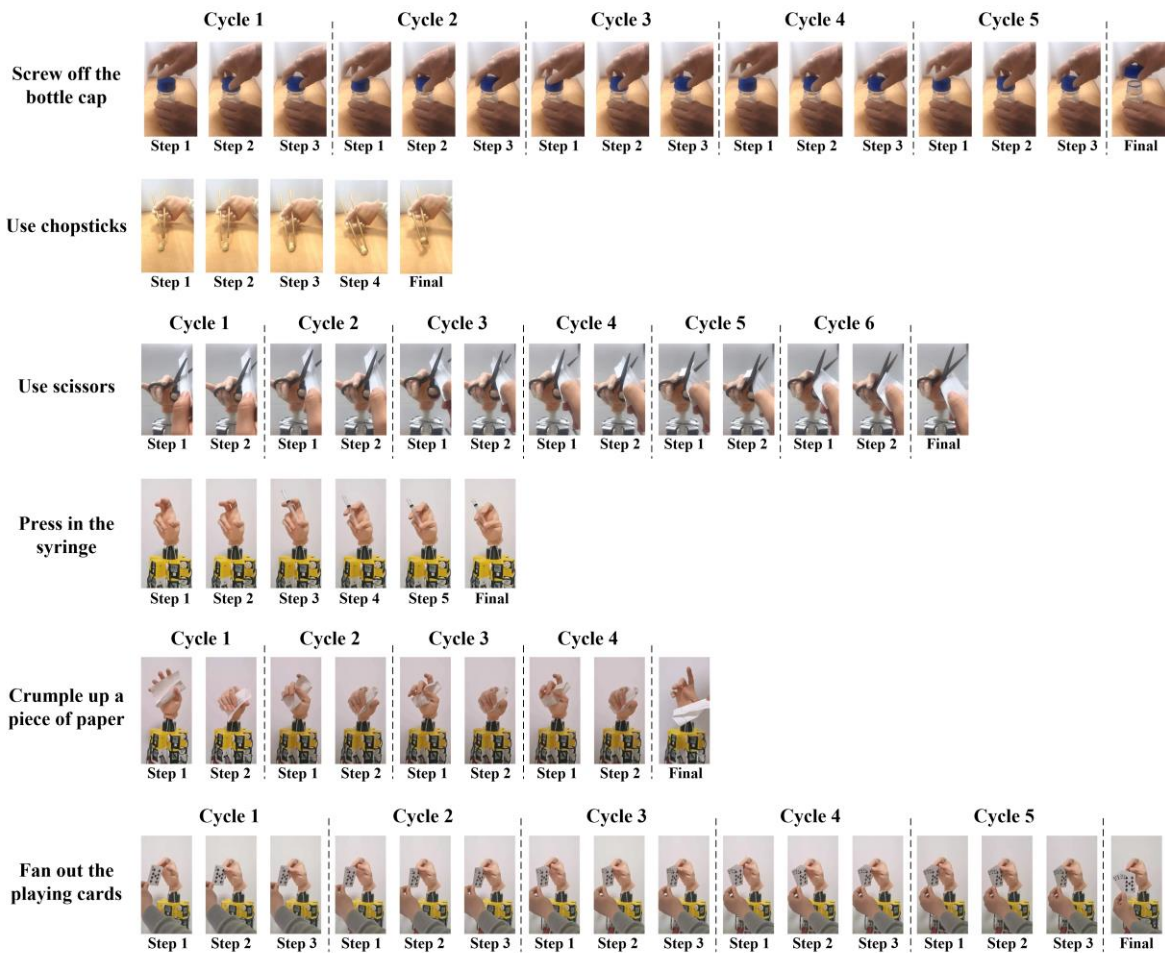


Figure 112. Manipulation test of the robotic hand

8.3 Summary

The grasping and manipulation capabilities are the most common and critical functions of the human hand, which were tested on the proposed robotic hand in this chapter.

Basing on Cutkosky and Feix taxonomy, we tested the grasping capability of the robotic hand with 16 and 33 grasp types, respectively. Most of the objects are from daily lives and cover a wide range of hand grasp postures. Besides, the active dynamic grasping process was also demonstrated with four normal objects involved. With the single-tendon-driven position control, all the grasp cases were successfully performed which benefitted a lot from the adaptivity coming from the bio-inspired rigid-flexible design of the robotic hand.

Subsequently, the manipulation capability was further tested. A data-glove-based position control strategy was adopted to perform six common manipulations in daily lives. The motors generated the corresponding tendon excursion to actuate the robotic hand to make the same posture sequence with the human hand whose postures during the whole manipulation process were recorded by the data glove. The robotic hand completed all the manipulations easily and naturally.

The positive results from the grasping and manipulation tests show the excellent performance of the proposed robotic hand and indicate that the biological structures adopted on the robotic hand can enhance its practical functions, validating the research significance of the project.

Chapter 9: Conclusion and future work

This chapter reviews the entire content of the project, including prototype design and fabrication, mathematical modelling, theoretical analysis, experimental verification, actuation system construction, control strategy development and performance test. Besides, several main findings in this project were concluded in this chapter. Three biomechanical advantages were found from the biological structures and realized on the proposed robotic finger with biomimetic design. Eventually, some research limitations were clarified and accordingly, the specific future works were planned proposed.

9.1 Thesis overview

By developing a novel bio-inspired multi-layered anthropomorphic robotic hand, the anatomical structures of the human hand are largely mimicked. However, it is definitely not a blind copy. The excellent functional performance of this robotic hand showed in the grasping and manipulation tests proves the validity and rationality of the structure design. One of the evaluation standards is the mechanical output of the system, which is the final result of the muscle actuation input and the regulation of the black-box-like end effector mechanical body. From the previous research, the mechanical regulation mainly comes from the property of the transmission support base (acted by the biomimetic joints) [6] [21], the morphology of the transmission main-body path (shaped by the extensor mechanism) [7] [31] [32] and the property of the transmission auxiliary structure (acted by the tendon sheaths pulley system) [38] [41]. This is also one of the triggers for this project.

Throughout the whole project, it started from the detailed investigation on the previous research about the human hand anatomy and biomechanics as well as structure design (especially the joints), actuation system types and control strategies of various robotic hands, which was presented in Chapter 2. Basing on this, a multi-layered structure design of one robotic finger was proposed in Chapter 3. It consists of three layers, including the base layer (phalanges and articular cartilage), the second layer (capsuloligamentous structures), and the third layer (tendons and tendon sheaths). After the fabrication of the prototype by using 3D print technology, silicone rubber moulding method and some unconventional soft materials, several basic kinematic indexes of the robotic finger were measured. Simultaneously, we suggested that some biomechanical advantages may exist in this biomimetic design just as in the biological structures of human hands. Therefore, in Chapter 4, we established mathematical models for the ligamentous joint, the robotic finger with extensor mechanism and the joint mechanism with flexible tendon sheaths to analyze how these structures will influence the mechanical output of the finger, such as the joint stiffness, the fingertip feasible forces space and the fingertip force-velocity characteristics. Besides, in the next chapter, the corresponding experiments were conducted to verify the theoretical analysis. And a stage test was conducted on the robotic finger prototype. A two-finger testbed was constructed and its grasping capability was tested. This procedure aimed to check if the bio-inspired design can benefit the practical performance of the robotic finger, providing a realistic foundation for the development of the robotic hand. In Chapter 6, the design of the whole robotic hand was presented, including the skeletal structure, the tendon distribution and the artificial skin. A 24 motors actuation system was then constructed and the control models for the robotic finger with flexible tendon sheaths were proposed in Chapter 7. And accordingly, the data-glove-based control strategy was developed and adopted to realize the functions tested in Chapter 8. The Cutkosky and Feix taxonomies were both used to test the grasping capability of the robotic hand. To test its manipulation capability, several common activities were selected to challenge the propose robotic hand. That is the overall work for now.

9.2 Main findings and contributions

By developing a bio-inspired robotic finger and hand, three biomechanical advantages were identified and investigated through theoretical analysis and experimental verification. These advantages contribute to the practical performance of the robotic hand from different aspects, including fingertip stiffness, force and velocity.

Variable stiffness of the ligamentous joint is one of the main findings. Indeed, coping with a variety of daily activities requires different joint stiffness conditions from fingers. Low stiffness in most of the flexion motion and the abduction-adduction motion with big flexion angles can help to easily move fingers to reach a large range of positions, enabling the hand to grasp objects with various sizes and shapes as well as complete some elaborate manipulations. Moreover, high stiffness can make fingers capable of bearing or resisting large lateral forces without many forces needed on the intrinsic muscles, e.g., pulling a rope or pinching a key [17]. We believe this is one of the primary biomechanical advantages of the human hand and can be applied to the robotic hand. This research also provides some insights on how to realize and analyze the variable stiffness of ligamentous joints.

Enlarged feasible force space is another interesting finding through the investigation of the extensor mechanism. Actually, the enlarged endpoint feasible force space contributes to the practical performance by allowing exerting enough forces in any space directions, which is particularly essential when interacting with the uncertain, unstructured and irregular environment since any amount of forces in all directions can be potentially required. More force feasibility could also benefit the complex hand manipulations by providing sufficient force conditions in every step of the motion. The result shown on our physical model provided another strong and intuitive evidence for the morphological advantage existing in the extensor mechanism [9] [28] [29] [31], providing a piece of supportive evidence for the advantages of applying the extensor mechanism to robotic hands.

One of the most important findings is the augmented force-velocity workspace contributed by the flexible tendon sheath, which means more force and velocity requirements can be satisfied quickly and continuously for different situations. Take the baseball catch as an example, our hand needs to close rapidly to catch the ball at the moment of the ball touching the hand and enough grasping forces are also needed to hold the ball so that it won't fall from the hand. The whole process happens in a short time and our hand needs to quickly adjust from the high-velocity mode to high-torque mode according to the external load, which holds out high requirements to the control system of the conventional robotic hands. The flexible tendon sheaths just simplify the problem by using the force-velocity self-adjustment from the unique structure and flexible material to substitute the complex control algorithm. This is another instance to show the mechanical intelligence of the biological body. However, there is little research testing the force-velocity workspace of robotic fingers. This research provides a novel method to optimize the robotic fingers' force-velocity characteristics by its own structural design while still in the same actuation conditions.

In fact, these unique biomechanical features can be performed tridimensionally in space and consecutively in time. The joint stiffness continuously varies in both flexion-extension and abduction-adduction directions. The feasible force of the endpoint could be enlarged in a 3D space through the extensor mechanism regulation. And the tendon sheath elastic pulley systems incessantly adjust the relationship of the force and velocity output according to the external load without delay, which is a spatial adjustment as well because of the distribution of the tendon sheath systems. As a result, the human-hand-like grasping and manipulation capabilities can be performed by the proposed robotic fingers and hand model thanks to these features.

Moreover, except for the findings in the research field of biomechanics, the proposed robotic hand with bio-inspired design is another contribution to the robotics field. The structure design, material selection and fabrication methods in this project provide some new inspirations for the future robotic system design, especially for the novel rigid-flexible coupled mechanisms. In addition, with the physical model, more characteristics of the human hand can be further explored efficiently. Take the biological joint as an example, together with the articular bones, the capsuloligamentous structure forms a tension-compression body with only the compression pressure born by the bones, and the tension only loaded on the joint ligaments and capsules. In the condition of the unidirectional force on each component, the overall load capacity of the joint can be enhanced. Besides, the unique ligamentous joint structure allows not only the flexion-extension and the abduction-adduction motion but also the supination-pronation motion [6], which improves dexterity and adaptivity of the finger, leading to better interaction with objects. Moreover, to prevent irreparable damage by the sudden impact, the joints can be dislocated out of their normal position and be easily repaired through reduction [155]. As to the elastic tendon sheaths, they can enhance the passive behaviours of the hand as well. For instance, the passive deformation of the tendons and muscles are needed when the posture tends to be changed by an external disturbance, which could be partially offset by the flexible tendon sheath's adjustment to the moment arm, simplifying the control to the tendon-muscle system. Additionally, the hand skin, as one of the most important functional unit of the human hand, plays a crucial role in grasping or manipulation tasks by providing proper friction between the hand and objects.

By comparison, other robotic hands can rarely show these biomechanical advantages mentioned in this article. In terms of the joint stiffness property, the conventional robotic hands can hardly perform this kind of variable stiffness because of their hinge joint design [46] [47] [48] [49] [50] [51] [52] [53]. And the recent robotic hands with elastic joints often used rubber-like materials or springs of which the joint stiffness can only vary in one direction individually [54] [55] [56]. The variable joint stiffness can be well performed by the biomimetic ligamentous joints but still lack sufficient studies and evidence [18] [19] [20] [21] [22]. As to the function of the extensor mechanism, many novel biomimetic robotic hands tended to adopt the extensor mechanism structure [18] [20] [32] [33]. However, little research investigated the feasible force space of robotic fingers benefiting from this structure. Likewise, quite a lot of tendon-driven robotic hands adopted the design of tendon sheaths or pulleys. Most of the robotic hands made this structure rigid only for guiding the tendons [19] [46] [55] [53] [75]. While some bio-inspired robotic hands used flexible tendon sheaths so as to highly mimic human hands [18] [22]. These flexible tendon sheaths are expected to improve the performance of robotic hands but still lack specific analysis and verification. In this research, these biomechanical advantages were systematically investigated, verified and eventually realized on the proposed robotic hand with strong evidence.

9.3 Limitation and future works

Some drawbacks and limitations do exist in this research. In this model design, the joint capsules and the tendon sheaths were moulded by using the silicone rubber. Some other materials and fabrication methods can be further explored since the silicone-rubber-made parts have limited strength and can be broken after performing functions for certain times. Some polymer fibres and the fibre braid technology can be considered to improve the performance of the functional parts. The mathematical models of the biological joint and the tendon sheaths system are simplified to clearly explain the basic theories. More details can be restored in computational models so as to obtain more accurate results. Besides, the thumb joint design is a little bit different from other finger joints, especially the CMC joint. This may bring some different joint stiffness properties and should be modelled and analyzed dependently. Moreover, in the design of the carpal bones, I considered the eight carpal bones as two units which are the proximal row and the distal row. Though there are some degrees of freedom between these two carpal bone rows, no theoretical analysis and experiments were conducted to explain the functional advantages of these extra degrees of freedom in this research, such as larger contact area when grasping. For the robotic fingers grasping section, only the comparison tests between the robotic fingers and the human fingers were conducted. However, to illustrate the advantages of the practical functions brought by the three human-like structures, additional comparison tests should be conducted, such as between the robotic fingers with hinge joints and ligamentous joints, with linear and net extensors, and with rigid and flexible tendon sheaths. The control strategy used in this research is the simple position control. But the whole control framework should be clearly summarized and presented. For example, the key-frame control strategy during the robotic hand manipulations is essential for completing the whole project, of which the principle and process need to be clarified. Additionally, the specific contribution of the biological structures to the practical performance of the human and robotic hand has not been studied in detail. Though the biomechanical advantages of the ligamentous joint, reticular extensor mechanism and flexible tendon sheath were found and analyzed, how these structures help the robotic hand realize the grasping and manipulation functions even without sensory feedback still remains uncovered, which leads to future work.

For future expansion work, several tasks need to be accomplished:

(1) The detailed mechanical properties of the robotic hand need to be tested, including the motion range, the maximum fingertip force, full flexion-extension speed, fingertip trajectory and fingertip stiffness of all the fingers and thumb, and the gripping force of the robotic hand. And accordingly, the design can be further improved.

(2) Compare the robotic hand with the human hand in terms of posture, fingertip contact pressure and contact area of the whole hand when grasping the same object to verify their behavioural similarities.

(3) Develop a systematic control strategy and comprehensive manipulation taxonomy for testing the performance of the proposed robotic hand and the future robotic hands. With the bio-inspired design, the robotic hand in this project is expected to complete larger amounts of manipulations as well as more complex manipulations. Thus, the

manipulation performance of the robotic hand will be further and continuously tested based on the proposed control strategy and manipulation taxonomy.

(4) Compare the dynamic grasping and manipulation success rate between the robotic hands with conventional hinge joints/ligamentous joints, with reticular extensor/linear extensor, with and without artificial skin to demonstrate the practical functional advantages coming from the bio-inspired structures.

A human only has two hands to cope with millions of daily activities. Except for the sophisticated neural control and sensing systems, the mechanical properties of the physical body also plays a significant role. Through this research, it can be found that some biomechanical advantages of the human hand are born in smart structures and soft materials, resulting in outstanding grasping and manipulation performance. The 3D printing technology and new materials allow relatively accurate reproduction of the human hand’s anatomical structures and biomechanical properties to better explore the inherent mechanism of these underlying advantages, and in turn, improving the design and functional performance of robotic hands.

In fact, a bigger picture can be further explored. How to make the robotic hand compare favourably with the human hand? It may be a sophisticated problem and a long process. Exploring and employing human hand features and advantages could be a good start. Honestly, one reason for the slow progress made in the robotic hand studies is that too much attention was paid to the implementation of functions rather than exploring the fundamental biomechanical principles. While the truth is that the investigation on the underlying biomechanical advantages of the biological systems could profoundly support the realization of practical functions, of which the research presented in this thesis is one block of the architecture. Nature is a treasure. We believe that there are more potential applications and research values in such anisotropic soft-rigid hybrid structures and tension-compression bodies inspired from human bodies requiring further study.

References

1. Ejaz, N., Hamada, M. and Diedrichsen, J. (2015) Hand use predicts the structure of representations in sensorimotor cortex. *Nature neuroscience*, *18*(7), pp.1034–1040.
2. Martuzzi, R., van der Zwaag, W., Farthouat, J., Gruetter, R., & Blanke, O. (2014) Human finger somatotopy in areas 3b, 1, and 2: a 7T fMRI study using a natural stimulus. *Human brain mapping*, *35*(1), 213-226.
3. Moscatelli, A., Bianchi, M., Ciotti, S., Bettelani, G. C., Parise, C. V., Lacquaniti, F., & Bicchi, A. (2019) Touch as an auxiliary proprioceptive cue for movement control. *Science advances*, *5*(6), eaaw3121.
4. Pruszynski, J.A., Johansson, R.S. and Flanagan, J.R. (2016) A Rapid Tactile-Motor Reflex Automatically Guides Reaching toward Handheld Objects. *Current Biology*, *26*(6), pp.788–792.
5. Lederman, S.J. and Klatzky, R.L. (1993) Extracting object properties through haptic exploration. *Acta psychologica*, *84*(1), pp.29–40.
6. Chao, E. Y. (1989) *Biomechanics of the Hand: A Basic Research Study*. World Scientific.
7. Valero-Cuevas, F.J., Zajac, F.E. and Burgar, C.G. (1998) Large index-fingertip forces are produced by subject-independent patterns of muscle excitation. *Journal of Biomechanics*, *31*(8), pp.693–703.
8. Valero-Cuevas, F.J., Towles, J.D. and Hentz, V.R. (2001) Quantification of fingertip force reduction in the forefinger floowing simulated paralysis of extensor and intrinsic muscles. *Journal of Biomechanics*, 34(1), pp.151–151.
9. Valero-Cuevas, F. J., Yi, J. W., Brown, D., McNamara, R. V., Paul, C., & Lipson, H. (2007) The tendon network of the fingers performs anatomical computation at a macroscopic scale. *IEEE Transactions on Biomedical Engineering*, *54*(6), 1161-1166.
10. Valero-Cuevas, F.J. (2009) A mathematical approach to the mechanical capabilities of limbs and fingers. *Advances in Experimental Medicine and Biology*, 629, pp.619–633.
11. Leger, A.B. and Milner, T.E. (2000) Passive and active wrist joint stiffness following eccentric exercise. *European journal of applied physiology*, *82*(5), pp.472–479.
12. Kubo, K., Miyazaki, D., Yamada, K., Yata, H., Shimoju, S., & Tsunoda, N. (2015) Passive and active muscle stiffness in plantar flexors of long distance runners. *Journal of biomechanics*, *48*(10), 1937-1943.
13. Kuo, P.H. and Deshpande, A.D. (2012) Muscle-tendon units provide limited contributions to the passive stiffness of the index finger metacarpophalangeal joint. *Journal of biomechanics*, *45*(15), pp.2531–2538.
14. Werner, D., Kozin, S. H., Brozovich, M., Porter, S. T., Junkin, D., & Seigler, S. (2003) The biomechanical properties of the finger metacarpophalangeal joints to

varus and valgus stress. *The Journal of hand surgery*, *28*(6), 1044-1051.

15. Lutsky, K., Matzon, J., Walinchus, L., Ross, D. A., & Beredjiklian, P. (2014). Collateral ligament laxity of the finger metacarpophalangeal joints: an in vivo study. *The Journal of hand surgery*, *39*(6), 1088-1093.
16. Kataoka, T., Moritomo, H., Miyake, J., Murase, T., Yoshikawa, H., & Sugamoto, K. (2011). Changes in shape and length of the collateral and accessory collateral ligaments of the metacarpophalangeal joint during flexion. *JBJS*, *93*(14), 1318-1325.
17. Grebenstein, M. (2014) *Approaching Human Performance : The Functionality-Driven Awiwi Robot Hand*. Cham: Springer International Publishing.
18. Zhe Xu and Todorov, E. (2016) Design of a highly biomimetic anthropomorphic robotic hand towards artificial limb regeneration. In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. *IEEE*, pp. 3485–3492
19. Chepisheva, M., Culha, U. and Iida, F. (2016) A biologically inspired soft robotic hand using chopsticks for grasping tasks. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. *Springer Verlag*, pp. 195–206.
20. Faudzi, A. A. M., Ooga, J., Goto, T., Takeichi, M., & Suzumori, K. (2017) Index finger of a human-like robotic hand using thin soft muscles. *IEEE Robotics and Automation Letters*, *3*(1), 92-99.
21. Hughes, J.A.E., Maiolino, P. and Iida, F. (2018) An anthropomorphic soft skeleton hand exploiting conditional models for piano playing. *Science Robotics*, *3*(25).
22. Tebyani, M., Robbins, A., Asper, W., Kurniawan, S., Teodorescu, M., Wang, Z., & Hirai, S. (2020) 3D Printing an Assembled Biomimetic Robotic Finger. In *2020 17th International Conference on Ubiquitous Robots (UR)*, pp. 526-532.
23. Valero-Cuevas, F.J. and Hentz, V.R. (2002) Releasing the A3 pulley and leaving flexor superficialis intact increases pinch force following the Zancolli lasso procedures to prevent claw deformity in the intrinsic palsied finger. *Journal of Orthopaedic Research*, *20*(5), pp.902–909.
24. Yokogawa, R. and Hara, K. (2002) Measurement of distribution of maximum index-fingertip force in all directions at fingertip in flexion/extension plane. *Journal of biomechanical engineering*, *124*(3), 302–307. https://doi.org/10.1115/1.1468637
25. Valero-Cuevas, F.J. (2005) An integrative approach to the biomechanical function and neuromuscular control of the fingers. *Journal of Biomechanics*, *38*(4), pp.673–684.
26. Inouye, J.M., Kutch, J.J. and Valero-Cuevas, F.J. (2012) A Novel Synthesis of Computational Approaches Enables Optimization of Grasp Quality of Tendon-Driven Hands. *IEEE Transactions on Robotics*, *28*(4), pp.958–966.
27. Shirafuji, S., Ikemoto, S. and Hosoda, K. (2014) Development of a tendon-driven robotic finger for an anthropomorphic robotic hand. *The International Journal of Robotics Research*, *33*(5), pp.677–693.

28. Inouye, J. and Valero-Cuevas, F.J. (2012) Asymmetric routings with fewer tendons can offer both flexible endpoint stiffness control and high force-production capabilities in robotic fingers. In: *2012 4th IEEE RAS & EMBS International Conference on Biomedical Robotics and Biomechatronics (BioRob). IEEE*, pp. 1273–1280.
29. Inouye, J.M. and Valero-Cuevas, F.J. (2014) Anthropomorphic tendon-driven robotic hands can exceed human grasping capabilities following optimization. *The International Journal of Robotics Research*, *33*(5), pp.694–705.
30. Valero-Cuevas, F. and Lipson, H. (2004) A computational environment to simulate complex tendinous topologies. In: *26th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. *IEEE*, pp. 4653–4656.
31. Synek, A. and Pahr, D. (2016) The effect of the extensor mechanism on maximum isometric fingertip forces: A numerical study on the index finger. *Journal of Biomechanics*, *49*(14), pp.3423–3429.
32. Wilkinson, D.D., Vande Weghe, M. and Matsuoka, Y. (2003) An extensor mechanism for an anatomical robotic hand. In: *Proceedings - IEEE International Conference on Robotics and Automation*. pp. 238–243.
33. Deshpande, A. D., Balasubramanian, R., Ko, J., & Matsuoka, Y. (2010) Acquiring variable moment arms for index finger using a robotic testbed. *IEEE Transactions on Biomedical Engineering*, *57*(8), 2034-2044.
34. Schmitt, S., Haeufle, D. F. B., Blickhan, R., & Günther, M. (2012) Nature as an engineer: one simple concept of a bio-inspired functional artificial muscle. *Bioinspiration & Biomimetics*, *7*(3), 036022.
35. Alcazar, J., Csapo, R., Ara, I., & Alegre, L. M. (2019) On the shape of the force-velocity relationship in skeletal muscles: The linear, the hyperbolic, and the double-hyperbolic. *Frontiers in physiology*, *10*, 769.
36. Asada, H. and Ro, I. H. (1985) A Linkage Design for Direct-Drive Robot Arms. *ASME. J. Mech., Trans., and Automation*, *107*(4): 536–540.
37. Bae, J.-H. and Arimoto, S. (2004) Important role of force/velocity characteristics in sensory-motor coordination for control design of object manipulation by a multi-fingered robot hand. *Robotica*, *22*(5), pp.479–491.
38. O'Brien, K. W., Xu, P. A., Levine, D. J., Aubin, C. A., Yang, H. J., Xiao, M. F., ... & Shepherd, R. F. (2018) Elastomeric passive transmission for autonomous force-velocity adaptation applied to 3D-printed prosthetics. *Science Robotics*, *3*(23).
39. Ralston, H. J., Polissar, M. J., Inman, V. T., Close, J. R., and Feinstein, B. (1949) Dynamic features of human isolated voluntary muscle in isometric and free contractions. *J. Appl. Physiol. 1*, 526–533. doi: 10.1152/jappl.1949.1.7.526
40. Abbott, B. C., and Wilkie, D. R. (1953) The relation between velocity of shortening and the tension-length curve of skeletal muscle. *J. Physiol. 120*, 214–223. doi: 10.1113/jphysiol.1953.sp004886

41. Amis, A. A. and Jones, M. M. (1988) The interior of the flexor tendon sheath of the finger. The functional significance of its structure. *The Journal of bone and joint surgery. British volume*, *70*(4), 583–587.
42. Lin, G. T., Cooney, W. P., Amadio, P. C., & An, K. N. (1990) Mechanical properties of human pulleys. *Journal of Hand Surgery*, *15*(4), 429-434.
43. Bianchi, M. and Moscatelli, A. (2016) *Human and robot hands: sensorimotor synergies to bridge the gap between neuroscience and robotics*. Cham: Springer.
44. Gama Melo, E. N., Aviles Sanchez, O. F., & Amaya Hurtado, D. (2014) Anthropomorphic robotic hands: a review. *Ingenier ú y desarrollo, 32*(2), 279-313.
45. Piazza, C., Grioli, G., Catalano, M. G. and Bicchi, A. (2019) A Century of Robotic Hands. *Annual Review of Control, Robotics, and Autonomous Systems*, *2*, 1-32
46. Jacobsen, S., Iversen, E., Knutti, D., Johnson, R., & Biggers, K. (1986, April). Design of the Utah/MIT dextrous hand. In *Proceedings. 1986 IEEE International Conference on Robotics and Automation*, Vol. 3, pp. 1520-1532.
47. Bekey G.A., Tomovic R., Zeljkovic I. (1990) Control Architecture for the Belgrade/USC Hand. In: *Venkataraman S.T., Iberall T. (eds)* Dextrous Robot Hands. *Springer*, New York, NY
48. Kyberd, P. J., Light, C., Chappell, P. H., Nightingale, J. M., Whatley, D., & Evans, M. (2001) The design of anthropomorphic prosthetic hands: A study of the Southampton Hand. *Robotica*, *19*(6), 593-600.
49. Mouri, T., Kawasaki, H., Yoshikawa, K., Takai, J. and Ito, S. (2002) Anthropomorphic robot hand: Gifu hand III.. *Proc. Int. Conf. ICCAS*: 1288-1293.
50. Carrozza, M. C., Cappiello, G., Micera, S., Edin, B. B., Beccai, L., & Cipriani, C. (2006) Design of a cybernetic hand for perception and action. *Biological cybernetics*, *95*(6), 629-644.
51. Liu, H., Wu, K., Meusel, P., Seitz, N., Hirzinger, G., Jin, M. H., ... & Chen, Z. P. (2008) Multisensory five-finger dexterous hand: The DLR/HIT Hand II. In *2008 IEEE/RSJ international conference on intelligent robots and systems*, pp. 3692-3697.
52. Bridgwater, L. B., Ihrke, C. A., Diftler, M. A., Abdallah, M. E., Radford, N. A., Rogers, J. M., ... & Linn, D. M. (2012) The Robonaut 2 hand - designed to do work with tools. In: *2012 IEEE International Conference on Robotics and Automation. IEEE*, pp. 3425–3430.
53. Shadow Robot Co. (2018) *Shadow Dexterous Hand.* Shadow Robot Company. https://www.shadowrobot. com/products/dexterous-hand.
54. Deimel, R. and Brock, O. (2016) A novel type of compliant and underactuated robotic hand for dexterous grasping. *The International journal of robotics research*, *35*(1-3), pp.161–185.
55. Lotti, F., Tiezzi, P., Vassura, G., Biagiotti, L., Palli, G., & Melchiorri, C. (2005) Development of UB Hand 3: Early Results. In: *Proceedings of the 2005 IEEE International Conference on Robotics and Automation. IEEE*, pp. 4488–449.
56. Kontoudis, G. P., Liarokapis, M. V., Zisimatos, A. G., Mavrogiannis, C. I., & Kyriakopoulos, K. J. (2015) Open-source, anthropomorphic, underactuated robot

hands with a selectively lockable differential mechanism: Towards affordable prostheses. In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE*, pp. 5857–5862.

57. Xu, Z., Todorov, E., Dellon, B., & Matsuoka, Y. (2011) Design and analysis of an artificial finger joint for anthropomorphic robotic hands. In *2011 IEEE International Conference on Robotics and Automation*, pp.5096–5102.
58. Drake, R.L., Vogl, A. Wayne & Mitchell, Adam W. M, (2014) *Gray's Anatomy for Students* 3rd ed., London: Elsevier Health Sciences.
59. Brunelli, Giovanni R. (1999) Stability of the first carpometacarpal joint. In Brüser, Peter; Gilbert, Alain. *Finger bone and joint injuries*. Taylor & Francis. pp. 167–170.
60. Marieb, Elaine Nicpon, and Katja Hoehn (2019) *Human Anatomy & Physiology*. Eleventh edition. Harlow, United Kingdom: Pearson Education Limited, 2019. Print.
61. Netter, F., (2014) *Atlas de Anatomia Humana*. 5th ed. Phialdelphia, PA: Saunders Elsevier, p.446.
62. Berger, Richard A.; Weiss, Arnold-Peter C., (2004) *Hand Surgery*. Lippincott Williams & Wilkins. ISBN 978-0-7817-2874-4.
63. Eaton, R. G., (1972) Joint Injuries Of The Hand. *Clinical Orthopaedics and Related Research (1976-2007), 84,* 280.
64. Martini, F., Timmons, M. J., Tallitsch, R. B., Ober, W. C., Garrison, C. W., Welch, K. B., & Hutchings, R. T. (2006) *Human anatomy* (p. 904). San Francisco, CA: Pearson/Benjamin Cummings.
65. Muscles of the hand. (2020) Retrieved March 1, 2021, from https://en.wikipedia.org/wiki/Muscles_of_the_hand#cite_note-LHC-2
66. Netter, F. (2011) *Atlas of human anatomy*. 4th ed. Phialdelphia, PA: Saunders Elsevier, p.464.
67. Anatomy (UL): Extensor Forearm and Hand Flashcards | Memorang. n.d. *Anatomy (UL): Extensor Forearm and Hand Flashcards | Memorang*. [online] Available at: <https://memorang.com/flashcards/67336/Anatomy+UL+Extensor+Forearm+and+Hand> [Accessed 1 March 2021].
68. Bradley S. Bowden, Joan M. Bowden (2002) *An Illustrated Atlas of the Skeletal Muscles*. CO: Morton Publishing Company.
69. https://en.wikipedia.org/wiki/Tendon_sheath#cite_note-2
70. Richard S. Snell (2011) *Clinical Anatomy by Regions*. Lippincott Williams & Wilkins, page 399.
71. Atlantaequine.com. n.d. AEC Client Education - PD Tenosynovitis. [online] Available at: <http://www.atlantaequine.com/pages/client_lib_PDsynovitis.html > [Accessed 13 March 2021].
72. Gurbuz H, Mesut R, Nesrin Turan F. (2006) Measurement of active abduction of metacarpophalangeal joints via electronic digital incinometric technique. *Ital J Anat Embryol*, *111*(1):9-14..
73. Harris, C. and Rutledge, A. L(1972) The functional anatomy of the extensor mechanism of finger, *J. Bone Joint Surg*, *54*-A: 713 -726.

74. Ueda, J., Kondo, M., & Ogasawara, T. (2010) The multifingered NAIST hand system for robot in-hand manipulation. *Mechanism and Machine Theory*, *45*(2), 224-238.

75. Deshpande, A. D., Xu, Z., Weghe, M. J. V., Brown, B. H., Ko, J., Chang, L. Y., ... & Matsuoka, Y. (2011) Mechanisms of the anatomically correct testbed hand. *IEEE/ASME Transactions on Mechatronics*, *18*(1), 238-250.

76. Dollar, A.M. & Howe, R.D. (2010) The Highly Adaptive SDM Hand: Design and Performance Evaluation. *The International Journal of Robotics Research*, *29*(5), pp.585–597.

77. Dollar, A.M. & Howe, R.D. (2006) A robust compliant grasper via shape deposition manufacturing. Mechatronics, *IEEE/ASME Transactions on*, *11*(2), pp.154–161.

78. Dollar, A.M. & Howe, R.D. (2005) Towards grasping in unstructured environments: grasper compliance and configuration optimization. *Advanced Robotics*, *19*(5), pp.523–543.

79. Mosadegh, B., Polygerinos, P., Keplinger, C., Wennstedt, S., Shepherd, R. F., Gupta, U., ... & Whitesides, G. M. (2014) Pneumatic networks for soft robotics that actuate rapidly. *Advanced functional materials*, *24*(15), 2163-2170.

80. Galloway, K. C., Polygerinos, P., Walsh, C. J., & Wood, R. J. (2013) Mechanically programmable bend radius for fiber-reinforced soft actuators. In *2013 16th International Conference on Advanced Robotics (ICAR)*, pp.1-6.

81. http://www.totalsmallbone.com/us/products/hand.php4

82. Xu, Z., Kumar, V., & Todorov, E. (2013) A low-cost and modular, 20-DOF anthropomorphic robotic hand: Design, actuation and modeling. In *2013 13th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, pp.368–375.

83. Kumar, V., Xu, Z., & Todorov, E. (2013) Fast, strong and compliant pneumatic actuation for dexterous tendon-driven hands. In *2013 IEEE international conference on robotics and automation*, pp.1512–1519.

84. Cura, V. O. D., Cunha, F. L., Aguiar, M. L., & Cliquet Jr, A. (2003) Study of the different types of actuators and mechanisms for upper limb prostheses. *Artificial organs*, *27*(6), 507-516.

85. Laurentis, K. J. D., & Mavroidis, C. (2002). Mechanical design of a shape memory alloy actuated prosthetic hand. *Technology and Health Care*, *10*(2), 91-106.

86. Cho, K. J., Rosmarin, J., & Asada, H. (2007) SBC hand: a lightweight robotic hand with an SMA actuator array implementing C-segmentation. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pp.921–926

87. Jung, S., Bae, J. & Moon, I. (2011) Lightweight prosthetic hand with five fingers using SMA actuator. *International Conference on Control, Automation and Systems*, pp.1797–1800.

88. Matsubara, S., Okamoto, S., & Lee, J. H. (2012) Prosthetic Hand Using Shape Memory Alloy Type Artificial Muscle. *Lecture Notes in Engineering and Computer Science*, *2196*(1), pp.873–876.

89. Andrianesis, K. & Tzes, A. (2015) Development and Control of a Multifunctional Prosthetic Hand with Shape Memory Alloy Actuators. *Journal of Intelligent & Robotic Systems*, *78*(2), pp.257–289.
90. Rasmussen, C.E. & Williams, Christopher K. I (2006) *Gaussian processes for machine learning*, Cambridge, Mass.: MIT Press.
91. Enge, Y., Szabo, P. & Volkinshtein, D. (2005) Learning to control an Octopus arm with Gaussian process temporal difference methods. *Advances in Neural Information Processing Systems*, pp.347–354.
92. Ko, J. & Fox, D. (2009) GP-Bayes Filters: Bayesian filtering using Gaussian process prediction and observation models. *Autonomous Robots*, *27*(1), pp.75–90.
93. Deshpande, A. D., Ko, J., Fox, D., & Matsuoka, Y. (2013). Control strategies for the index finger of a tendon-driven hand. *The International Journal of Robotics Research*, *32*(1), 115-128.
94. Melchiorri, C. (2000) Slip detection and control using tactile and force sensors. Mechatronics, *IEEE/ASME Transactions on*, *5*(3), pp.235–243.
95. Ikeda, A., Kurita, Y., Ueda, J., Matsumoto, Y., & Ogasawara, T. (2005) Grip Force Control of the Elastic Body based on Contact Surface Eccentricity During the Incipient Slip. *Journal of the Robotics Society of Japan*, *23*(3), pp.337–343.
96. Takahashi, T., Tsuboi, T., Kishida, T., Kawanami, Y., Shimizu, S., Iribe, M., ... & Fujita, M. (2008) Adaptive grasping by multi fingered hand with tactile sensor based on robust force and position control. In *2008 IEEE International Conference on Robotics and Automation*, pp.264–271.
97. Gunji, D., Araki, T., Namiki, A., Ming, A., & Shimojo, M. (2007) Grasping force control of multi-fingered robot hand based on slip detection using tactile sensor. *Journal of the Robotics Society of Japan*, *25*(6), 970-978.
98. Stachowsky, M., Moussa, M., & Abdullah, H. (2012) A locally adaptive online grasp control strategy using array sensor force feedback. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp.4060–4065.
99. Lippold, O. C. J. (1952) The relation between integrated action potentials in a human muscle and its isometric tension. *The Journal of physiology*, *117*(4), 492-499.
100. Kuriki, H. U., Mello, E. M., De Azevedo, F. M., Takahashi, L. S. O., Alves, N., & de Faria Negrão Filho, R. (2012) *The relationship between electromyography and muscle force*. INTECH Open Access Publisher.
101. Benjuya, N.B. & Kenney, S. (1990) Myoelectric Hand Orthosis. *JPO Journal of Prosthetics and Orthotics*, *2*(2), pp.149–154.
102. Dicicco, M., Lucas, L. & Matsuoka, Y. (2004) Comparison of control strategies for an EMG controlled orthotic exoskeleton for the hand. *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*, *2*, pp.1622–1627.
103. Reischl, M., Mikut, R., Pylatiuk, C., & Schulz, S. (2001) Control strategies for hand prostheses using myoelectric patterns. In *Proc. 9th Zittau Fuzzy Colloquium*, pp. 168-174.

104. Chen, C. H., & Naidu, D. S. (2011) Fusion of fuzzy logic and PD control for a five-fingered smart prosthetic hand. In *2011 IEEE International Conference on Fuzzy Systems, IEEE*, pp. 2108-2115.

105. Fausett, L. V. (2006) *Fundamentals of neural networks: architectures, algorithms and applications*. Pearson Education India.

106. Gawande, S. D., & Chopde, N. R. (2013) Neural network based hand gesture recognition. *International Journal of Emerging Research in Management and Technology*, *3*, 2278-9359.

107. Erim, Z., & Lin, W. (2008) Decomposition of intramuscular EMG signals using a heuristic fuzzy expert system. *IEEE Transactions on Biomedical Engineering*, *55*(9), 2180-2189.

108. Hargrove, L.J., Englehart, K. & Hudgins, B. (2007) A Comparison of Surface and Intramuscular Myoelectric Signal Classification. *Biomedical Engineering, IEEE Transactions on*, *54*(5), pp.847–853.

109. Srinivasan, H., Gupta, S., Sheng, W., & Chen, H. (2012) Estimation of hand force from surface Electromyography signals using Artificial Neural Network. In *Proceedings of the 10th World Congress on Intelligent Control and Automation*, pp.584–589.

110. Mattar, E. (2011) Dexterous robotics hands: ANN based artificial muscles control. In *2011 UkSim 13th International Conference on Computer Modelling and Simulation*, pp.224–229.

111. Mane, S. M., Kambli, R. A., Kazi, F. S., & Singh, N. M. (2015) Hand motion recognition from single channel surface EMG using wavelet & artificial neural network. *Procedia Computer Science*, *49*, 58-65.

112. Chaudhary, A., & Raheja, J. L. (2013) Bent fingers' angle calculation using supervised ANN to control electro-mechanical robotic hand. *Computers & Electrical Engineering*, *39*(2), 560-570.

113. Pasluosta, C.F. & Chiu, A.W.L. (2012) Evaluation of a Neural Network-Based Control Strategy for a Cost-Effective Externally-Powered Prosthesis. *Assistive Technology*, *24*(3), pp.196–208.

114. Huluta, E., Da Silva, R.F. & de Oliveira, T.E.A. (2014) Neural network-Based hand posture control of a humanoid Robot Hand. *Computational Intelligence and Virtual Environments for Measurement Systems and Applications (CIVEMSA), 2014 IEEE International Conference on*, pp.124–128

115. McFarland, D.J.J., Sarnacki, W.A.A. & Wolpaw, J.R.R. (2010) Electroencephalographic (EEG) control of three-dimensional movement. *Journal of Neural Engineering*, *7*(3).

116. Fabiani, G. E., McFarland, D. J., Wolpaw, J. R., & Pfurtscheller, G. (2004) Conversion of EEG activity into cursor movement by a brain-computer interface (BCI). *IEEE transactions on neural systems and rehabilitation engineering*, *12*(3), 331-338.

117. Scherer, R., Muller, G. R., Neuper, C., Graimann, B., & Pfurtscheller, G. (2004) An asynchronously controlled EEG-based virtual keyboard: improvement of the spelling rate. *IEEE Transactions on Biomedical Engineering*, *51*(6), 979-984.

118. Vidaurre, C., Schlogl, A., Cabeza, R., Scherer, R., & Pfurtscheller, G. (2007) Study of on-line adaptive discriminant analysis for EEG-based brain computer interfaces. *IEEE transactions on biomedical engineering*, *54*(3), 550-556.
119. Lehtonen, J., Jylanki, P., Kauhanen, L., & Sams, M. (2008) Online classification of single EEG trials during finger movements. *IEEE Transactions on Biomedical Engineering*, *55*(2), 713-720.
120. Mahmoudi, B., & Erfanian, A. (2002) Single-channel EEG-based prosthetic hand grasp control for amputee subjects. In *Proceedings of the Second Joint 24th Annual Conference and the Annual Fall Meeting of the Biomedical Engineering Society][Engineering in Medicine and Biology*, *3*, pp.2406–2407.
121. Hazrati, M. K., & Erfanian, A. (2010) An online EEG-based brain–computer interface for controlling hand grasp using an adaptive probabilistic neural network. *Medical engineering & physics*, *32*(7), 730-739.
122. Pichiorri, F., Morone, G., Petti, M., Toppi, J., Pisotta, I., Molinari, M., ... & Mattia, D. (2015) Brain–computer interface boosts motor imagery practice during stroke recovery. *Annals of neurology*, *77*(5), 851-865.
123. Ramos-Murguialday, A., Broetz, D., Rea, M., Läer, L., Yilmaz, Ö., Brasil, F. L., ... & Birbaumer, N. (2013) Brain–machine interface in chronic stroke rehabilitation: a controlled study. *Annals of neurology*, *74*(1), 100-108.
124. Werth, B. (2015) *An Armless Man Raises His Hand*. [online] The Wall Street Journal. Available at: <https://www.wsj.com/articles/an-armless-man-raises-his-hand-1447803110> [Accessed 29 October 2021].
125. Sarasola-Sanz, A., López-Larraz, E., Irastorza-Landa, N., Klein, J., Valencia, D., Belloso, A., ... & Ramos-Murguialday, A. (2017) An EEG-Based Brain-Machine Interface to Control a 7-Degrees of Freedom Exoskeleton for Stroke Rehabilitation. In *Biosystems and Biorobotics*, pp. 1127–1131.
126. Neumann, D. A. (2010) *Kinesiology of the musculoskeletal system; Foundation for rehabilitation.* Mosby & Elsevier.
127. Wei, Y., Zou, Z., Wei, G., Ren, L., & Qian, Z. (2020) Subject-specific finite element modelling of the human hand complex: muscle-driven simulations and experimental validation. *Annals of biomedical engineering*, *48*(4), 1181-1195.
128. Buckwalter, J. A., Rosenberg, L. A. and Hunziker, E. B. (1990) Articular cartilage and knee joint function: Basic science and arthroscopy. In: *Bristol-Myers/Zimmer Orthopaedic Symposium*, pp. 19-56. Raven Press New York.
129. Sophia Fox, A. J., Bedi, A. and Rodeo, S. A. (2009) The basic science of articular cartilage: structure, composition, and function. *Sports health*, *1*(6), 461–468.
130. Nordin, M., Frankel, V. H. and Leger, D. (2012) *Basic Biomechanics of the Musculoskeletal System*, Wolters Kluwer/Lippincott Williams & Wilkins Health.
131. Ogura, T., Hirata, A., Hayashi, N., Ito, H., Takenaka, S., Fujisawa, Y., ... & Kameda, H. (2017) SAT0661 Finger joint cartilage thickness evaluated by ultrasound in patients with rheumatoid arthritis (RA). *Annals of the Rheumatic Diseases*, *76*(s2), pp.1024–1025.

132. Graffte, K. (2005) Fluoropolymers: Fitting the bill for medical applications. *Medical Device and Diagnostic Industry*, pp.25–31.
133. Teng, H. (2012) Overview of the Development of the Fluoropolymer Industry. *Applied Sciences*, *2*(2), pp.496–512.
134. Sattar, M., Patel, M. and Alani, A. (2017) Clinical applications of polytetrafluoroethylene (PTFE) tape in restorative dentistry. *British dental journal*, *222*(3), pp.151–158.
135. Olcay, K., Steier, L. and Erdogan, H. (2015) Polytetrafluoroetylene tape as temporary restorative material: a fluid filtration study. *Journal of Istanbul University Faculty of Dentistry*, *49*(3), pp.17–22.
136. Moráguez, O.D. and Belser, U.C. (2010) The use of polytetrafluoroethylene tape for the management of screw access channels in implant-supported prostheses. *The Journal of Prosthetic Dentistry*, *103*(3), pp.189–191.
137. Wan, C., Hao, Z. and Wen, S. (2013) A review on research on development of ligament constitutive relations on macro, meso, and micro levels. *Acta Mechanica Solida Sinica*, *26*(4), pp.331–343.
138. Ralphs, J. R. and Benjamin, M. (1994) The joint capsule: structure, composition, ageing and disease. *Journal of anatomy*, *184* (Pt 3), 503–50.
139. Watkins, J., Mathieson, I. (2009) CHAPTER 5 - The articular system, The Pocket Podiatry Guide: Functional Anatomy, *Churchill Livingstone*, Pages 157-181
140. Thompson, J. C. (2015) *Netter's Concise Orthopaedic Anatomy E-Book*. Elsevier Health Sciences.
141. Buschmann, J. and Bürgisser, G.M. (2017) *Biomechanics of tendons and ligaments : tissue reconstruction and regeneration*. Cambridge, Massachusetts: Woodhead Publishing.
142. Zancolli, E. (1979) *Structural and Dynamic Bases of Hand Surgery*, Lippincott Williams & Wilkins.
143. Goodman, H. J. and Choueka, J. (2005) Biomechanics of the flexor tendons. *Hand clinics*, *21*(2), pp.129–149.
144. Botte, M. J. (2003) *Surgical anatomy of the hand and upper extremity*. Lippincott Williams & Wilkins.
145. Chase, R. and White, W. (2015) YouTube. [online] Youtube.com. Available at: https://www.youtube.com/watch?v=9V1ZzAiB1rY [Accessed 23 Nov. 2019].
146. Alexander, R.M. and Bennett, M.B. (1987) Some principles of ligament function, with examples from the tarsal joints of the sheep. Ovis aries. *Journal of Zoology*, *211*(3), pp.487–504.
147. Valero-Cuevas, F.J. (2016) *Fundamentals of Neuromechanics*. 1st ed. London: Springer London.
148. Van Soest, A.J. and Bobbert, M.F. (1993) The contribution of muscle properties in the control of explosive movements. *Biological cybernetics*, *69*(3), pp.195–204.
149. Haeufle, D.F.B., Grimmer, S. and Seyfarth, A. (2010) The role of intrinsic muscle properties for stable hopping—stability is achieved by the force–velocity relation. *Bioinspiration & Biomimetics*, *5*(1), p.016004–.

150. https://en.wikipedia.org/wiki/Carpal_bones
151. Palastanga, N., Field, D., & Soames, R. (2006) *Anatomy and human movement: structure and function* (Vol. 20056). Elsevier Health Sciences.
152. Vanhees, M., Verstreken, F., & van Riet, R. (2015) What does the transverse carpal ligament contribute to carpal stability?. *Journal of wrist surgery, 4*(01), 031-034.
153. Cutkosky, M. (1989) On grasp choice, grasp models, and the design of hands for manufacturing tasks. *IEEE Transactions on Robotics and Automation*, *5*(3), pp.269-279.
154. Feix, T., Romero, J., Schmiedmayer, H., Dollar, A. and Kragic, D. (2016) The GRASP Taxonomy of Human Grasp Types. *IEEE Transactions on Human-Machine Systems*, *46*(1), pp.66-77.
155. Seo, C. (2019) Human-mimetic soft robot joint for shock absorption through joint dislocation. *Bioinspiration & biomimetics*, *15*(1), pp.016001–016001.
156. Virtualmotionlabs.com. n.d. *VMG 30 Virtual Reality Glove - Virtual Motion Labs*. [online] Available at: <https://www.virtualmotionlabs.com/vr-gloves/vmg-30/> [Accessed 19 October 2021].
157. WILLIAMS, N. (1997) The Virtual Hand. *Journal of Hand Surgery, 22*(5), pp.560-567.
158. CyberGlove Systems. (2007) *CyberGlove Data Glove User Guide*. 1st ed. San Jose, California: CyberGlove Systems LLC, p.13.
159. CyberGlove Systems LLC. n.d. *CyberGlove II — CyberGlove Systems LLC*. [online] Available at: <http://www.cyberglovesystems.com/cyberglove-ii> [Accessed 19 October 2021].
160. CyberGlove Systems LLC. n.d. *CyberGlove III — CyberGlove Systems LLC*. [online] Available at: <http://www.cyberglovesystems.com/cyberglove-iii> [Accessed 19 October 2021].
161. Caeiro-Rodr ǵuez, M., Otero-Gonz ález, I., Mikic-Fonte, F. and Llamas-Nistal, M. (2021) A Systematic Review of Commercial Smart Gloves: Current Status and Applications. *Sensors*, *21*(8), p.2667.
162. FIFTH DIMENSION TECHNOLOGIES, n.d. *5DT Data Glove Ultra - 5DT*. [online] 5DT. Available at: <https://5dt.com/5dt-data-glove-ultra/> [Accessed 19 October 2021].
163. Su, Y., Allen, C., Geng, D., Burn, D., Brechany, U., Bell, G. and Rowland, R. (2003) 3-D motion system ("data-gloves"): application for Parkinson's disease. *IEEE Transactions on Instrumentation and Measurement*, *52*(3), pp.662-674.
164. Dipietro, L., Sabatini, A. and Dario, P. (2003) Evaluation of an instrumented glove for hand-movement acquisition. *The Journal of Rehabilitation Research and Development*, *40*(2), p.181.
165. Wang, Y., & Zhang, W. (2015) Data glove control of robot hand with force telepresence. In *2015 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pp. 314-319.

166. Kazi, M., & Bill, M. (2020) Robotic Hand Controlled by Glove Using Wireless Communication.
167. Akhlaghi, N., Baker, C., Lahlou, M., Zafar, H., Murthy, K., Rangwala, H., Kosecka, J., Joiner, W., Pancrazio, J. and Sikdar, S. (2016) Real-Time Classification of Hand Motions Using Ultrasound Imaging of Forearm Muscles. *IEEE Transactions on Biomedical Engineering*, *63*(8), pp.1687-1698.
168. Hettiarachchi, N., Ju, Z., & Liu, H. (2015) A new wearable ultrasound muscle activity sensing system for dexterous prosthetic control. In *2015 IEEE International Conference on Systems, Man, and Cybernetics*, pp. 1415-1420.
169. McIntosh, J., Marzo, A., Fraser, M., & Phillips, C. (2017) Echoflex: Hand gesture recognition using ultrasound imaging. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, pp. 1923-1934.
170. Ortenzi, V., Tarantino, S., Castellini, C., & Cipriani, C. (2015) Ultrasound imaging for hand prosthesis control: a comparative study of features and classification methods. In *2015 IEEE International Conference on Rehabilitation Robotics (ICORR)*, pp. 1-6.
171. McIntosh, J., Marzo, A., Fraser, M., & Phillips, C. (2017) Echoflex: Hand gesture recognition using ultrasound imaging. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, pp. 1923-1934.
172. Akhlaghi, N., Baker, C., Lahlou, M., Zafar, H., Murthy, K., Rangwala, H., Kosecka, J., Joiner, W., Pancrazio, J. and Sikdar, S. (2016) Real-Time Classification of Hand Motions Using Ultrasound Imaging of Forearm Muscles. *IEEE Transactions on Biomedical Engineering*, *63*(8), pp.1687-1698.
173. Sikdar, S., Rangwala, H., Eastlake, E., Hunt, I., Nelson, A., Devanathan, J., Shin, A. and Pancrazio, J. (2014) Novel Method for Predicting Dexterous Individual Finger Movements by Imaging Muscle Activity Using a Wearable Ultrasonic System. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, *22*(1), pp.69-76.
174. Hettiarachchi, N., Ju, Z., & Liu, H. (2015) A new wearable ultrasound muscle activity sensing system for dexterous prosthetic control. In *2015 IEEE International Conference on Systems, Man, and Cybernetics*, pp. 1415-1420.
175. Georgia Tech. n.d. *The Force is Strong: Amputee Controls Individual Prosthetic Fingers*. [online] Available at: <https://news.gatech.edu/news/2017/12/11/force-strong-amputee-controls-individual-prosthetic-fingers> [Accessed 19 October 2021].
176. Napier, J. (1956) THE PREHENSILE MOVEMENTS OF THE HUMAN HAND. *The Journal of Bone and Joint Surgery. British volume*, *38*-B(4), pp.902-913.
177. Stival, F., Michieletto, S., Cognolato, M., Pagello, E., Müller, H. and Atzori, M. (2019) A quantitative taxonomy of human hand grasps. *Journal of NeuroEngineering and Rehabilitation*, *16*(1).
178. Mehrkish, A. and Janabi-Sharifi, F. (2021) A comprehensive grasp taxonomy of continuum robots. *Robotics and Autonomous Systems*, *145*, p.103860.

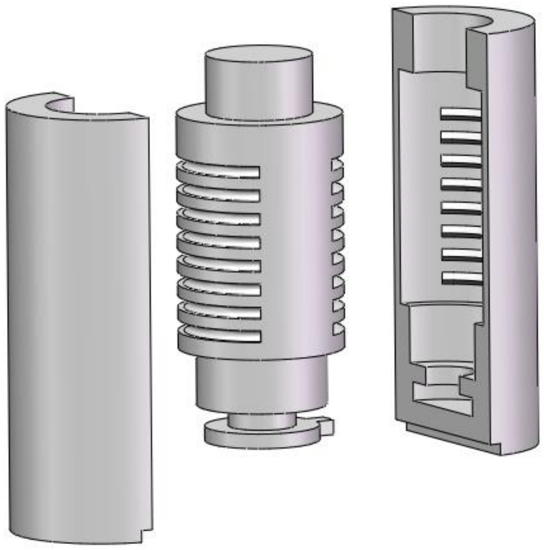
179. Elliott, J. and Connolly, K. (1984) A CLASSIFICATION OF MANIPULATIVE HAND MOVEMENTS. *Developmental Medicine & Child Neurology*, *26*(3), pp.283-296.
180. Case-Smith, J. and Pehoski, C. (1992) *Development of hand skills in children*. Rockville, MD: American Occupational Therapy Association, pp.35-45.
181. Bullock, I., Ma, R. and Dollar, A. (2013) A Hand-Centric Classification of Human and Robot Dexterous Manipulation. *IEEE Transactions on Haptics*, *6*(2), pp.129-144.
182. Calli, B., Singh, A., Bruce, J., Walsman, A., Konolige, K., Srinivasa, S., Abbeel, P. and Dollar, A. (2017) Yale-CMU-Berkeley dataset for robotic manipulation research. *The International Journal of Robotics Research*, *36*(3), pp.261-268.
183. Paulius, D., Huang, Y., Meloncon, J., & Sun, Y. (2019) Manipulation motion taxonomy and coding for robots. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5596-5601.
184. Festo.com. (2021). *BionicSoftHand | Festo USA*. [online] Available at: <https://www.festo.com/us/en/e/about-festo/research-and-development/bionic-learning-network/bionicsofthand-id_68106/> [Accessed 29 October 2021].
185. Santina, C., Piazza, C., Grioli, G., Catalano, M. and Bicchi, A. (2018) Toward Dexterous Manipulation With Augmented Adaptive Synergies: The Pisa/IIT SoftHand 2. *IEEE Transactions on Robotics*, *34*(5), pp.1141-1156.
186. Youtube.com. (2019) *Using Haptic Gloves to Control an Amazing Telepresence Robot!*. [online] Available at: <https://www.youtube.com/watch?v=rWHk4ht-boM> [Accessed 29 October 2021].
187. Youtube.com. (2013) *Teleoperation and Manipulation with DLR/HIT II Robot Hand using a Low Cost Force Feedback Device*. [online] Available at: <https://www.youtube.com/watch?v=MmK1QmLHajk&t=6s> [Accessed 29 October 2021].

Appendices

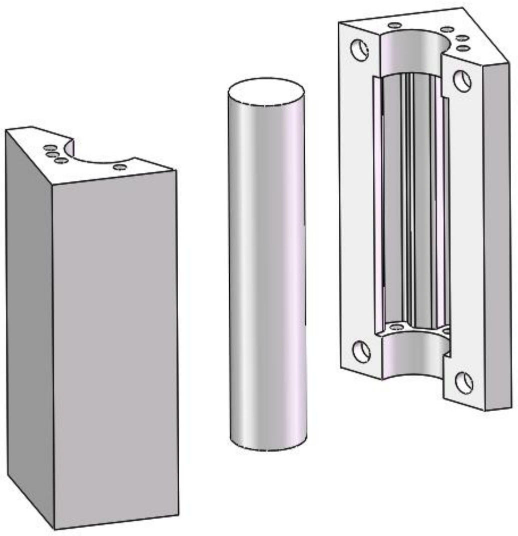
Appendix I: Grasping and manipulation video list of the robotic finger and the robotic hand

1. Appendix/fingergrasp-cloth-flat.mp4
2. Appendix/fingergrasp-cloth-rough.mp4
3. Appendix/fingergrasp-cloth-soft.mp4
4. Appendix/fingergrasp-peanut-flat.mp4
5. Appendix/fingergrasp-peanut-rough.mp4
6. Appendix/fingergrasp-peanut-soft.mp4
7. Appendix/fingergrasp-pen-flat.mp4
8. Appendix/fingergrasp-pen-rough.mp4
9. Appendix/fingergrasp-pen-soft.mp4
10. Appendix/fingergrasp-playing card-flat.mp4
11. Appendix/fingergrasp-playing card-rough.mp4
12. Appendix/fingergrasp-playing card-soft.mp4
13. Appendix/fingergrasp-sponge blocks-flat.mp4
14. Appendix/fingergrasp-sponge blocks-rough.mp4
15. Appendix/fingergrasp-sponge blocks-soft.mp4
16. Appendix/hand grasp-dynamic-ball.mp4
17. Appendix/hand grasp-dynamic-cylinder.mp4
18. Appendix/hand grasp-dynamic-pen.mp4
19. Appendix/hand grasp-dynamic-small tape.mp4
20. Appendix/hand grasp-static-big ball.mp4
21. Appendix/hand grasp-static-pen.mp4
22. Appendix/hand grasp-static-small ball.mp4
23. Appendix/hand manipulation-crumple up a piece of paper.mp4
24. Appendix/hand manipulation-fan out the playing cards.mp4
25. Appendix/hand manipulation-press in the syringe.mp4
26. Appendix/hand manipulation-screw off bottle cap.mp4
27. Appendix/hand manipulation-use chopsticks.mp4
28. Appendix/hand manipulation-use scissors.mp4

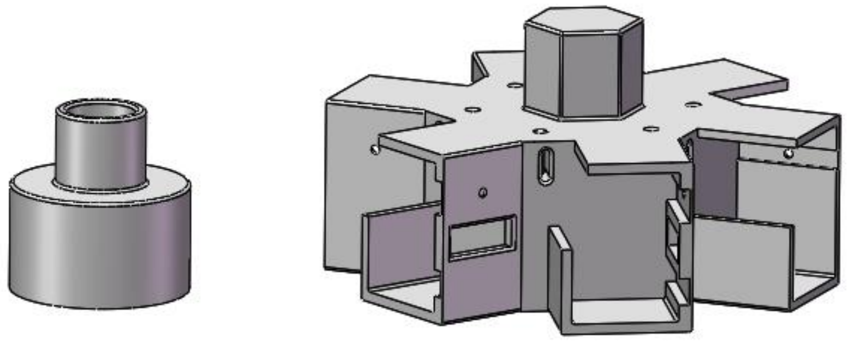
Appendix II: CAD model of joint capsule moulds



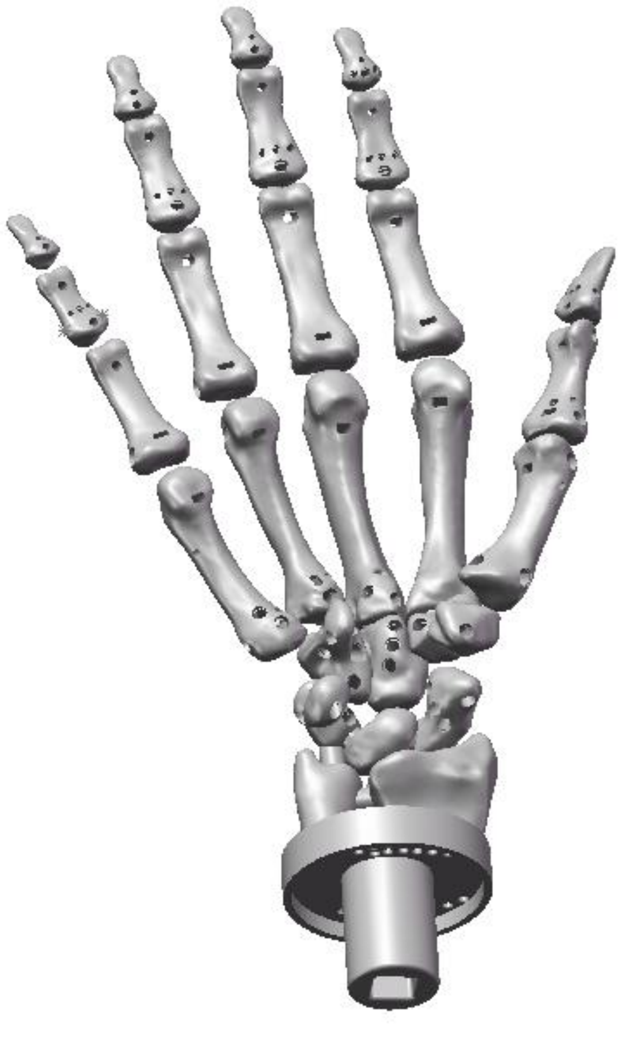
Appendix III: CAD model of tendon sheath moulds



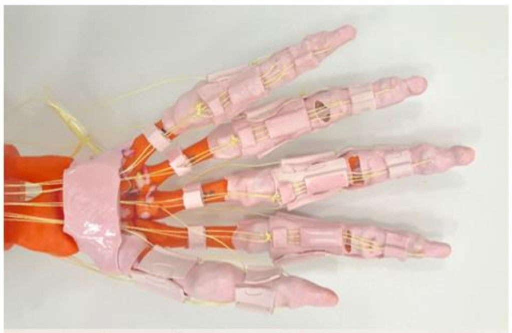
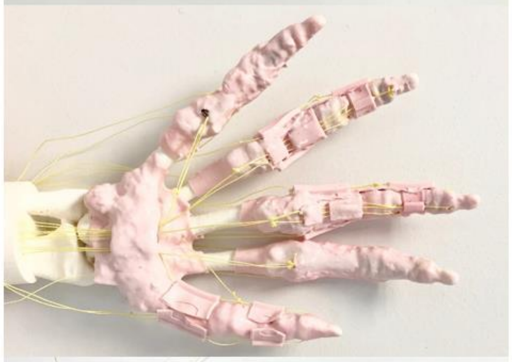
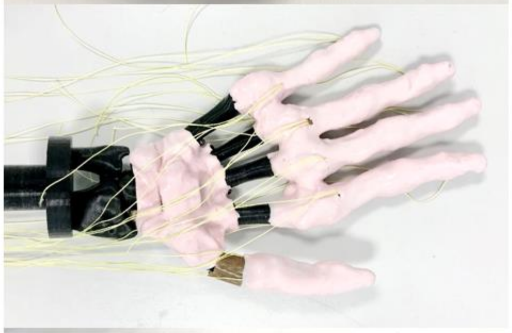
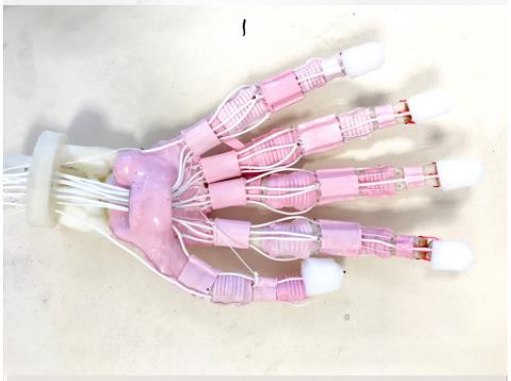
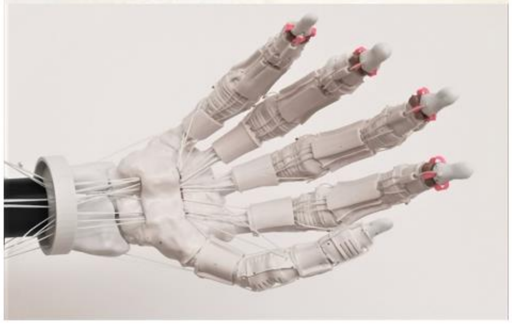
Appendix IV: CAD model of actuation system connector and module



Appendix V: CAD model of the skeletal structure of the robotic hand with ligaments and tendons insertion holes



Appendix VI: Five generations of the robotic hand

Appendix VII: Matlab code for ball grasping of the robotic hand

```
clc;

clear all;


lib_name = '';


if strcmp(computer, 'PCWIN')

  lib_name = 'dxl_x86_c';

elseif strcmp(computer, 'PCWIN64')

  lib_name = 'dxl_x64_c';

elseif strcmp(computer, 'GLNX86')

  lib_name = 'libdxl_x86_c';

elseif strcmp(computer, 'GLNXA64')

  lib_name = 'libdxl_x64_c';

elseif strcmp(computer, 'MACI64')

  lib_name = 'libdxl_mac_c';

end


% Load Libraries

if ~libisloaded(lib_name)

    [notfound, warnings] = loadlibrary(lib_name, 'dynamixel_sdk.h',

'addheader', 'port_handler.h', 'addheader', 'packet_handler.h',

'addheader', 'group_sync_write.h');

end


% Control table address

ADDR_MX_TORQUE_ENABLE       = 24;           % Control table address is

different in Dynamixel model

ADDR_MX_GOAL_POSITION       = 30;
```

```
ADDR_MX_PRESENT_POSITION    = 36;
ADDR_MX_GOAL_SPEED          = 32;



% Data Byte Length
LEN_MX_GOAL_POSITION        = 2;
LEN_MX_PRESENT_POSITION     = 2;
LEN_MX_GOAL_SPEED           = 2;

% Protocol version
PROTOCOL_VERSION            = 1.0;          % See which protocol
version is used in the Dynamixel

% Default setting
DXL1_ID                     = 1;            % Dynamixel#1 ID: 6
DXL2_ID                     = 2;
DXL3_ID                     = 3;
DXL4_ID                     = 4;
DXL5_ID                     = 5;
DXL6_ID                     = 6;
DXL7_ID                     = 7;            % Dynamixel#5 ID: 7
DXL8_ID                     = 8;            % Dynamixel#1 ID: 8
DXL9_ID                     = 9;            % Dynamixel#5 ID: 9
DXL10_ID                     = 10;            % Dynamixel#1 ID: 10
DXL11_ID                     = 11;            % Dynamixel#5 ID: 11
DXL12_ID                     = 12;            % Dynamixel#1 ID: 12
DXL13_ID                     = 13;            % Dynamixel#5 ID: 13
DXL14_ID                     = 14;            % Dynamixel#1 ID: 14
DXL15_ID                     = 15;
```

```
DXL18_ID                     = 18;            % Dynamixel#5 ID: 16
DXL19_ID                     = 19;
DXL21_ID                     = 21;

BAUDRATE                    = 1000000;
DEVICENAME                  = 'COM3';       % Check which port is being
used on your controller
                                             % ex) Windows: 'COM1'
Linux: '/dev/ttyUSB0' Mac: '/dev/tty.usbserial-*'

TORQUE_ENABLE               = 1;            % Value for enabling the
torque
TORQUE_DISABLE              = 0;            % Value for disabling the
torque



DXL1_MINIMUM_POSITION_VALUE  = 1926;
DXL1_MAXIMUM_POSITION_VALUE  = 1926;

DXL2_MINIMUM_POSITION_VALUE  = 2225;          %1025
DXL2_MAXIMUM_POSITION_VALUE  = 800;

DXL3_MINIMUM_POSITION_VALUE  = 1909;          %1259
DXL3_MAXIMUM_POSITION_VALUE  = 650;

DXL4_MINIMUM_POSITION_VALUE  = 2720;
DXL4_MAXIMUM_POSITION_VALUE  = 2720;
```

```
DXL5_MINIMUM_POSITION_VALUE  = 3200;
DXL5_MAXIMUM_POSITION_VALUE  = 3200;

DXL6_MINIMUM_POSITION_VALUE  = 2490;          %1094
DXL6_MAXIMUM_POSITION_VALUE  = 3800;



DXL7_MINIMUM_POSITION_VALUE  = 1629;
DXL7_MAXIMUM_POSITION_VALUE  = 1629;



DXL8_MINIMUM_POSITION_VALUE  = 1828;         %
DXL8_MAXIMUM_POSITION_VALUE  = 1828;         % and this value (note that
the Dynamixel would not move when the position value is out of movable
range. Check e-manual about the range of the Dynamixel you use.)
%2250

DXL9_MINIMUM_POSITION_VALUE  = 1077;          %923    Dynamixel will
rotate between this value
DXL9_MAXIMUM_POSITION_VALUE  = 2200;
%3440

DXL10_MINIMUM_POSITION_VALUE  = 1536;          %1264
DXL10_MAXIMUM_POSITION_VALUE  = 2800;         % and this value (note
that the Dynamixel would not move when the position value is out of movable
range. Check e-manual about the range of the Dynamixel you use.)
%3000

DXL11_MINIMUM_POSITION_VALUE  = 1350;          % Dynamixel will rotate
```

```
between this value
DXL11_MAXIMUM_POSITION_VALUE  = 1350;
%1250

DXL12_MINIMUM_POSITION_VALUE  = 1746;          %544
DXL12_MAXIMUM_POSITION_VALUE  = 2290;         % and this value (note
that the Dynamixel would not move when the position value is out of movable
range. Check e-manual about the range of the Dynamixel you use.)
%1100

DXL13_MINIMUM_POSITION_VALUE  = 1900;          %560
DXL13_MAXIMUM_POSITION_VALUE  = 2460;
%2050

DXL14_MINIMUM_POSITION_VALUE  = 3039;          % Dynamixel will rotate
between this value
DXL14_MAXIMUM_POSITION_VALUE  = 3039;         % and this value (note
that the Dynamixel would not move when the position value is out of movable
range. Check e-manual about the range of the Dynamixel you use.)
%1500


DXL15_MINIMUM_POSITION_VALUE  = 3850 ;          %1557    Dynamixel
will rotate between this value
DXL15_MAXIMUM_POSITION_VALUE  = 2293;

DXL18_MINIMUM_POSITION_VALUE  = 1669;          %419
DXL18_MAXIMUM_POSITION_VALUE  = 1250;
%1300
```

```
DXL19_MINIMUM_POSITION_VALUE  = 1608;          %275

DXL19_MAXIMUM_POSITION_VALUE  = 1333;


DXL21_MINIMUM_POSITION_VALUE  = 1946;

DXL21_MAXIMUM_POSITION_VALUE  = 1946;


DXL_MOVING_STATUS_THRESHOLD = 10;           % Dynamixel moving status

threshold


DXL2_GOAL_SPEED_VALUE        =

round(0.07*abs(DXL2_MINIMUM_POSITION_VALUE -

DXL2_MAXIMUM_POSITION_VALUE));          % Dynamixel moving SPEED

DXL3_GOAL_SPEED_VALUE        =

30+round(0.07*abs(DXL3_MINIMUM_POSITION_VALUE -

DXL3_MAXIMUM_POSITION_VALUE));

DXL6_GOAL_SPEED_VALUE        = 300;          % Dynamixel moving SPEED

thumb!

DXL9_GOAL_SPEED_VALUE        =

round(0.07*abs(DXL9_MINIMUM_POSITION_VALUE -

DXL9_MAXIMUM_POSITION_VALUE));

DXL10_GOAL_SPEED_VALUE        =

20+round(0.07*abs(DXL10_MINIMUM_POSITION_VALUE -

DXL10_MAXIMUM_POSITION_VALUE));           % Dynamixel moving SPEED

DXL12_GOAL_SPEED_VALUE        =

round(0.07*abs(DXL12_MINIMUM_POSITION_VALUE -

DXL12_MAXIMUM_POSITION_VALUE));

DXL13_GOAL_SPEED_VALUE        =

round(0.07*abs(DXL13_MINIMUM_POSITION_VALUE -
DXL13_MAXIMUM_POSITION_VALUE));           % Dynamixel moving SPEED
DXL18_GOAL_SPEED_VALUE        =
round(0.07*abs(DXL18_MINIMUM_POSITION_VALUE -
DXL18_MAXIMUM_POSITION_VALUE));
DXL19_GOAL_SPEED_VALUE        =
50+round(0.07*abs(DXL19_MINIMUM_POSITION_VALUE -
DXL19_MAXIMUM_POSITION_VALUE));           % Dynamixel moving SPEED
DXL15_GOAL_SPEED_VALUE        =150;

DXL1_GOAL_SPEED_VALUE        = 70;
DXL4_GOAL_SPEED_VALUE        = 70;
DXL5_GOAL_SPEED_VALUE        = 70;
DXL7_GOAL_SPEED_VALUE        = 70;
DXL8_GOAL_SPEED_VALUE        = 70;
DXL11_GOAL_SPEED_VALUE        = 70;
DXL14_GOAL_SPEED_VALUE        = 70;
DXL21_GOAL_SPEED_VALUE        = 300;


ESC_CHARACTER               = 'e';          % Key for escaping loop

COMM_SUCCESS                = 0;            % Communication Success
result value
COMM_TX_FAIL                = -1001;        % Communication Tx Failed

% Initialize PortHandler Structs
% Set the port path
% Get methods and members of PortHandlerLinux or PortHandlerWindows
port_num = portHandler(DEVICENAME);
```

```
% Initialize PacketHandler Structs
packetHandler();


% Initialize Groupsyncwrite instance
group1_num = groupSyncWrite(port_num, PROTOCOL_VERSION,
ADDR_MX_GOAL_POSITION, LEN_MX_GOAL_POSITION);
group2_num = groupSyncWrite(port_num, PROTOCOL_VERSION,
ADDR_MX_GOAL_SPEED, LEN_MX_GOAL_SPEED);
group3_num = groupSyncWrite(port_num, PROTOCOL_VERSION,
ADDR_MX_GOAL_POSITION, LEN_MX_GOAL_POSITION);
index = 1;
dxl_comm_result = COMM_TX_FAIL;             % Communication result
dxl_addparam_result1 = false;                % AddParam result



dxl1_goal_position = [DXL1_MINIMUM_POSITION_VALUE
DXL1_MAXIMUM_POSITION_VALUE];
dxl2_goal_position = [DXL2_MINIMUM_POSITION_VALUE
DXL2_MAXIMUM_POSITION_VALUE];
dxl3_goal_position = [DXL3_MINIMUM_POSITION_VALUE
DXL3_MAXIMUM_POSITION_VALUE];
dxl4_goal_position = [DXL4_MINIMUM_POSITION_VALUE
DXL4_MAXIMUM_POSITION_VALUE];
dxl5_goal_position = [DXL5_MINIMUM_POSITION_VALUE
DXL5_MAXIMUM_POSITION_VALUE];
dxl6_goal_position = [DXL6_MINIMUM_POSITION_VALUE
DXL6_MAXIMUM_POSITION_VALUE];         % Goal position
dxl7_goal_position = [DXL7_MINIMUM_POSITION_VALUE
DXL7_MAXIMUM_POSITION_VALUE];
dxl8_goal_position = [DXL8_MINIMUM_POSITION_VALUE
DXL8_MAXIMUM_POSITION_VALUE];         % Goal position
dxl9_goal_position = [DXL9_MINIMUM_POSITION_VALUE
DXL9_MAXIMUM_POSITION_VALUE];
dxl10_goal_position = [DXL10_MINIMUM_POSITION_VALUE
DXL10_MAXIMUM_POSITION_VALUE];         % Goal position
dxl11_goal_position = [DXL11_MINIMUM_POSITION_VALUE
DXL11_MAXIMUM_POSITION_VALUE];
dxl12_goal_position = [DXL12_MINIMUM_POSITION_VALUE
DXL12_MAXIMUM_POSITION_VALUE];         % Goal position
dxl13_goal_position = [DXL13_MINIMUM_POSITION_VALUE
DXL13_MAXIMUM_POSITION_VALUE];
dxl15_goal_position = [DXL15_MINIMUM_POSITION_VALUE
DXL15_MAXIMUM_POSITION_VALUE];         % Goal position
dxl18_goal_position = [DXL18_MINIMUM_POSITION_VALUE
DXL18_MAXIMUM_POSITION_VALUE];
dxl19_goal_position = [DXL19_MINIMUM_POSITION_VALUE
DXL19_MAXIMUM_POSITION_VALUE];
dxl21_goal_position = [DXL21_MINIMUM_POSITION_VALUE
DXL21_MAXIMUM_POSITION_VALUE];

dx11_extend_position= 800; %thumb extend
dx1_extend_position= 1900;
dx2_extend_position= 1680;            %index extend;
dx3_extend_position= 1777;            %middle extend;
dx4_extend_position= 2650;
dx5_extend_position= 3500;
dx6_extend_position= 2400;
```

```
dx7_extend_position= 1440;

dx8_extend_position= 1800;

dx9_extend_position= 1000;

dx19_extend_position= 2100;               %ring extend;

dx18_extend_position= 2200;               %little extend;

dx10_extend_position= 1400;

dx12_extend_position= 1680;

dx13_extend_position= 1777;

dx21_extend_position= 1500;




dxl_error = 0;                              % Dynamixel error

dxl1_present_position = 0;                  % Present position

dxl5_present_position = 0;



% Open port

if (openPort(port_num))

    fprintf('Succeeded to open the port!\n');

else

    unloadlibrary(lib_name);

    fprintf('Failed to open the port!\n');

    input('Press any key to terminate...\n');

    return;

end
```

```
% Set port baudrate
if (setBaudRate(port_num, BAUDRATE))
    fprintf('Succeeded to change the baudrate!\n');
else
    unloadlibrary(lib_name);
    fprintf('Failed to change the baudrate!\n');
    input('Press any key to terminate...\n');
    return;
end



% Enable Dynamixel#6 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL1_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_ENABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
else
    fprintf('Dynamixel has been successfully connected \n');
end

% Enable Dynamixel#6 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL2_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_ENABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
```

```
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
else
    fprintf('Dynamixel has been successfully connected \n');
end

% Enable Dynamixel#6 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL3_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_ENABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
else
    fprintf('Dynamixel has been successfully connected \n');
end



% Enable Dynamixel#6 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL4_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_ENABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
```

```
elseif dxl_error ~= 0

    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));

else

    fprintf('Dynamixel has been successfully connected \n');

end



% Enable Dynamixel#6 Torque

write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL5_ID,

ADDR_MX_TORQUE_ENABLE, TORQUE_ENABLE);

dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);

dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);

if dxl_comm_result ~= COMM_SUCCESS

    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));

elseif dxl_error ~= 0

    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));

else

    fprintf('Dynamixel has been successfully connected \n');

end




% Enable Dynamixel#6 Torque

write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL6_ID,

ADDR_MX_TORQUE_ENABLE, TORQUE_ENABLE);

dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);

dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);

if dxl_comm_result ~= COMM_SUCCESS

    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
```

```
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
else
    fprintf('Dynamixel has been successfully connected \n');
end

% Enable Dynamixel#7 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL7_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_ENABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
else
    fprintf('Dynamixel has been successfully connected \n');
end

% Enable Dynamixel#8 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL8_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_ENABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
else
```

```
    fprintf('Dynamixel has been successfully connected \n');
end


% Enable Dynamixel#9 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL9_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_ENABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
else
    fprintf('Dynamixel has been successfully connected \n');
end


% Enable Dynamixel#10 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL10_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_ENABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
else
    fprintf('Dynamixel has been successfully connected \n');
end
```

```
% Enable Dynamixel#11 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL11_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_ENABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
else
    fprintf('Dynamixel has been successfully connected \n');
end

% Enable Dynamixel#12 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL12_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_ENABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
else
    fprintf('Dynamixel has been successfully connected \n');
end

% Enable Dynamixel#13 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL13_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_ENABLE);
```

```
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
else
    fprintf('Dynamixel has been successfully connected \n');
end



% Enable Dynamixel#15 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL15_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_ENABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
else
    fprintf('Dynamixel has been successfully connected \n');
end



% Enable Dynamixel#16 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL18_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_ENABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
else
    fprintf('Dynamixel has been successfully connected \n');
end

% Enable Dynamixel#6 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL19_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_ENABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
else
    fprintf('Dynamixel has been successfully connected \n');
end


% Enable Dynamixel#6 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL21_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_ENABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
```

```
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
else
    fprintf('Dynamixel has been successfully connected \n');
end


while 1

    x = input('Press any key to continue! (or input e to quit!)\n', 's');

    if x == ESC_CHARACTER
        break;
    end
    if x == 'p' && index == 2;



     % Add Dynamixel#11 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL1_ID,
DXL1_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL1_ID);
        return;
    end
```

```
      % Add Dynamixel#11 goal position value to the Syncwrite parameter
storage
     dxl_addparam_result3 = groupSyncWriteAddParam(group3_num, DXL1_ID,
dx1_extend_position, LEN_MX_GOAL_POSITION);
    if dxl_addparam_result3 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL1_ID);
        return;
    end

     % Add Dynamixel#11 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL2_ID,
DXL2_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL2_ID);
        return;
    end



      % Add Dynamixel#11 goal position value to the Syncwrite parameter
storage
     dxl_addparam_result3 = groupSyncWriteAddParam(group3_num, DXL2_ID,
dx2_extend_position, LEN_MX_GOAL_POSITION);
    if dxl_addparam_result3 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL2_ID);
        return;
    end

     % Add Dynamixel#11 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL3_ID,
DXL3_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL3_ID);
        return;
    end


      % Add Dynamixel#11 goal position value to the Syncwrite parameter
storage
     dxl_addparam_result3 = groupSyncWriteAddParam(group3_num, DXL3_ID,
dx3_extend_position, LEN_MX_GOAL_POSITION);
    if dxl_addparam_result3 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL3_ID);
        return;
    end



     % Add Dynamixel#11 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL4_ID,
DXL4_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL4_ID);
        return;
    end


      % Add Dynamixel#11 goal position value to the Syncwrite parameter
```

```
storage
     dxl_addparam_result3 = groupSyncWriteAddParam(group3_num, DXL4_ID,
dx4_extend_position, LEN_MX_GOAL_POSITION);
    if dxl_addparam_result3 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL4_ID);
        return;
    end



     % Add Dynamixel#11 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL5_ID,
DXL5_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL5_ID);
        return;
    end



      % Add Dynamixel#11 goal position value to the Syncwrite parameter
storage
     dxl_addparam_result3 = groupSyncWriteAddParam(group3_num, DXL5_ID,
dx5_extend_position, LEN_MX_GOAL_POSITION);
    if dxl_addparam_result3 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL5_ID);
        return;
    end



     % Add Dynamixel#11 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL6_ID, 
DXL6_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL6_ID);
        return;
    end



      % Add Dynamixel#11 goal position value to the Syncwrite parameter 
storage
     dxl_addparam_result3 = groupSyncWriteAddParam(group3_num, DXL6_ID, 
dx6_extend_position, LEN_MX_GOAL_POSITION);
    if dxl_addparam_result3 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL6_ID);
        return;
    end



     % Add Dynamixel#11 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL7_ID, 
DXL7_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL7_ID);
        return;
    end



      % Add Dynamixel#11 goal position value to the Syncwrite parameter 
storage
```

```
     dxl_addparam_result3 = groupSyncWriteAddParam(group3_num, DXL7_ID, 
dx7_extend_position, LEN_MX_GOAL_POSITION);
    if dxl_addparam_result3 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL7_ID);
        return;
    end


     % Add Dynamixel#11 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL8_ID, 
DXL8_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL8_ID);
        return;
    end


      % Add Dynamixel#11 goal position value to the Syncwrite parameter 
storage
     dxl_addparam_result3 = groupSyncWriteAddParam(group3_num, DXL8_ID, 
dx8_extend_position, LEN_MX_GOAL_POSITION);
    if dxl_addparam_result3 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL8_ID);
        return;
    end


     % Add Dynamixel#11 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL9_ID, 
DXL9_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL9_ID);
        return;
    end



      % Add Dynamixel#11 goal position value to the Syncwrite parameter
storage
     dxl_addparam_result3 = groupSyncWriteAddParam(group3_num, DXL9_ID,
dx9_extend_position, LEN_MX_GOAL_POSITION);
    if dxl_addparam_result3 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL9_ID);
        return;
    end

     % Add Dynamixel#11 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL10_ID,
DXL10_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL10_ID);
        return;
    end



      % Add Dynamixel#11 goal position value to the Syncwrite parameter
storage
     dxl_addparam_result3 = groupSyncWriteAddParam(group3_num,
DXL10_ID, dx10_extend_position, LEN_MX_GOAL_POSITION);
```

```
    if dxl_addparam_result3 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL10_ID);
        return;
    end




     % Add Dynamixel#11 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL11_ID,
DXL11_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL11_ID);
        return;
    end

     % Add Dynamixel#11 goal position value to the Syncwrite parameter
storage
     dxl_addparam_result3 = groupSyncWriteAddParam(group3_num,
DXL11_ID, dx11_extend_position, LEN_MX_GOAL_POSITION);
    if dxl_addparam_result3 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL11_ID);
        return;
    end

     % Add Dynamixel#12 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL12_ID,
DXL12_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL12_ID);
        return;
    end



     % Add Dynamixel#12 goal position value to the Syncwrite parameter
storage
     dxl_addparam_result3 = groupSyncWriteAddParam(group3_num,
DXL12_ID, dx12_extend_position, LEN_MX_GOAL_POSITION);
    if dxl_addparam_result3 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL12_ID);
        return;
    end

         % Add Dynamixel#13 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL13_ID,
DXL13_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL13_ID);
        return;
    end

     % Add Dynamixel#13 goal position value to the Syncwrite parameter
storage
     dxl_addparam_result3 = groupSyncWriteAddParam(group3_num,
DXL13_ID, dx13_extend_position, LEN_MX_GOAL_POSITION);
    if dxl_addparam_result3 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL13_ID);
        return;
    end
```

```
     % Add Dynamixel#16 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL18_ID,
DXL18_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL18_ID);
        return;
    end

     % Add Dynamixel#18 goal position value to the Syncwrite parameter
storage
     dxl_addparam_result3 = groupSyncWriteAddParam(group3_num,
DXL18_ID, dx18_extend_position, LEN_MX_GOAL_POSITION);
    if dxl_addparam_result3 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL18_ID);
        return;
    end

      % Add Dynamixel#16 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL19_ID,
DXL19_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL19_ID);
        return;
    end


     % Add Dynamixel#19 goal position value to the Syncwrite parameter
```

```
storage
     dxl_addparam_result3 = groupSyncWriteAddParam(group3_num,
DXL19_ID, dx19_extend_position, LEN_MX_GOAL_POSITION);
    if dxl_addparam_result3 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL19_ID);
        return;
    end



    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL21_ID,
DXL21_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL21_ID);
        return;
    end



     % Add Dynamixel#19 goal position value to the Syncwrite parameter
storage
     dxl_addparam_result3 = groupSyncWriteAddParam(group3_num,
DXL21_ID, dx21_extend_position, LEN_MX_GOAL_POSITION);
    if dxl_addparam_result3 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL21_ID);
        return;
    end



   groupSyncWriteTxPacket(group2_num);  %WRITE SPEED FOR DYMX
    groupSyncWriteTxPacket(group3_num);
    dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
    if dxl_comm_result ~= COMM_SUCCESS
        fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION,
dxl_comm_result));
    end

    % Clear syncwrite parameter storage
    groupSyncWriteClearParam(group2_num);
    groupSyncWriteClearParam(group3_num);




    pause(1);



     % Add Dynamixel#6 goal position value to the Syncwrite storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL1_ID,
dxl1_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL1_ID);
        return;
    end

    % Add Dynamixel#6 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL1_ID,
DXL1_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL1_ID);
        return;
    end

     % Add Dynamixel#6 goal position value to the Syncwrite storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL2_ID,
dxl2_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL2_ID);
        return;
    end

    % Add Dynamixel#6 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL2_ID,
DXL2_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL2_ID);
        return;
    end

     % Add Dynamixel#6 goal position value to the Syncwrite storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL3_ID,
dxl3_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL3_ID);
        return;
    end

    % Add Dynamixel#6 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL3_ID,
DXL3_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
```

```
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL3_ID);
        return;
    end

     % Add Dynamixel#6 goal position value to the Syncwrite storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL4_ID,
dxl4_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL4_ID);
        return;
    end

    % Add Dynamixel#6 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL4_ID,
DXL4_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL4_ID);
        return;
    end

     % Add Dynamixel#6 goal position value to the Syncwrite storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL5_ID,
dxl5_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL5_ID);
        return;
    end
    % Add Dynamixel#6 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL5_ID,
DXL5_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL5_ID);
        return;
    end

    % Add Dynamixel#6 goal position value to the Syncwrite storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL6_ID,
dxl6_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL6_ID);
        return;
    end

    % Add Dynamixel#6 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL6_ID,
DXL6_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL6_ID);
        return;
    end

    % Add Dynamixel#7 goal position value to the Syncwrite parameter
storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL7_ID,
dxl7_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
```

```
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL7_ID);
        return;
    end

     % Add Dynamixel#7 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL7_ID,
DXL7_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL7_ID);
        return;
    end

     % Add Dynamixel#8 goal position value to the Syncwrite parameter
storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL8_ID,
dxl8_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL8_ID);
        return;
    end

     % Add Dynamixel#8 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL8_ID,
DXL8_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL8_ID);
        return;
    end
```

```
     % Add Dynamixel#9 goal position value to the Syncwrite parameter
storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL9_ID,
dxl9_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL9_ID);
        return;
    end

     % Add Dynamixel#9 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL9_ID,
DXL9_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL9_ID);
        return;
    end

     % Add Dynamixel#10 goal position value to the Syncwrite parameter
storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL10_ID,
dxl10_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL10_ID);
        return;
    end

     % Add Dynamixel#10 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL10_ID,
DXL10_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
```

```
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL10_ID);
        return;
    end

     % Add Dynamixel#11 goal position value to the Syncwrite parameter
storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL11_ID,
dxl11_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL11_ID);
        return;
    end

     % Add Dynamixel#11 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL11_ID,
DXL11_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL11_ID);
        return;
    end




     % Add Dynamixel#12 goal position value to the Syncwrite parameter
storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL12_ID,
dxl12_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL12_ID);
        return;
    end

     % Add Dynamixel#12 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL12_ID,
DXL12_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL12_ID);
        return;
    end




     % Add Dynamixel#13 goal position value to the Syncwrite parameter
storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL13_ID,
dxl13_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL13_ID);
        return;
    end

     % Add Dynamixel#13 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL13_ID,
DXL13_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
```

```
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL13_ID);
        return;
    end



     % Add Dynamixel#15 goal position value to the Syncwrite parameter
storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL15_ID,
dxl15_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL15_ID);
        return;
    end

     % Add Dynamixel#14 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL15_ID,
DXL15_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL15_ID);
        return;
    end

     % Add Dynamixel#16 goal position value to the Syncwrite parameter
storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL18_ID,
dxl18_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
```

```
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL18_ID);
        return;
    end

     % Add Dynamixel#16 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL18_ID,
DXL18_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL18_ID);
        return;
    end



       % Add Dynamixel#16 goal position value to the Syncwrite parameter
storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL19_ID,
dxl19_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL19_ID);
        return;
    end

     % Add Dynamixel#16 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL19_ID,
DXL19_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL19_ID);
        return;
    end



       % Add Dynamixel#16 goal position value to the Syncwrite parameter
storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL21_ID,
dxl21_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL21_ID);
        return;
    end

     % Add Dynamixel#16 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL21_ID,
DXL21_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL21_ID);
        return;
    end

    % Syncwrite goal position



    groupSyncWriteTxPacket(group2_num);  %WRITE SPEED FOR DYMX
    groupSyncWriteTxPacket(group1_num);
    dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
    if dxl_comm_result ~= COMM_SUCCESS
        fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION,
dxl_comm_result));
    end

    % Clear syncwrite parameter storage
    groupSyncWriteClearParam(group2_num);
    groupSyncWriteClearParam(group1_num);

    else x == 'p';

         % Add Dynamixel#6 goal position value to the Syncwrite storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL1_ID,
dxl1_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL1_ID);
        return;
    end

    % Add Dynamixel#6 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL1_ID,
DXL1_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL1_ID);
        return;
    end

     % Add Dynamixel#6 goal position value to the Syncwrite storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL2_ID,
dxl2_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
```

```
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL2_ID);
        return;
    end

    % Add Dynamixel#6 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL2_ID,
DXL2_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL2_ID);
        return;
    end

     % Add Dynamixel#6 goal position value to the Syncwrite storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL3_ID,
dxl3_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL3_ID);
        return;
    end

    % Add Dynamixel#6 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL3_ID,
DXL3_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL3_ID);
        return;
    end

     % Add Dynamixel#6 goal position value to the Syncwrite storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL4_ID,
dxl4_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL4_ID);
        return;
    end

    % Add Dynamixel#6 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL4_ID,
DXL4_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL4_ID);
        return;
    end

     % Add Dynamixel#6 goal position value to the Syncwrite storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL5_ID,
dxl5_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL5_ID);
        return;
    end

    % Add Dynamixel#6 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL5_ID,
DXL5_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL5_ID);
        return;
    end

    % Add Dynamixel#6 goal position value to the Syncwrite storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL6_ID,
dxl6_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL6_ID);
        return;
    end

    % Add Dynamixel#6 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL6_ID,
DXL6_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL6_ID);
        return;
    end

    % Add Dynamixel#7 goal position value to the Syncwrite parameter
storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL7_ID,
dxl7_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL7_ID);
        return;
    end

     % Add Dynamixel#7 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL7_ID,
DXL7_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL7_ID);
        return;
    end

     % Add Dynamixel#8 goal position value to the Syncwrite parameter
storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL8_ID,
dxl8_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL8_ID);
        return;
    end

     % Add Dynamixel#8 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL8_ID,
DXL8_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL8_ID);
        return;
    end

     % Add Dynamixel#9 goal position value to the Syncwrite parameter
storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL9_ID,
dxl9_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL9_ID);
        return;
    end

     % Add Dynamixel#9 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL9_ID, 
DXL9_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL9_ID);
        return;
    end

     % Add Dynamixel#10 goal position value to the Syncwrite parameter 
storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL10_ID, 
dxl10_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL10_ID);
        return;
    end

     % Add Dynamixel#10 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL10_ID, 
DXL10_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL10_ID);
        return;
    end

     % Add Dynamixel#11 goal position value to the Syncwrite parameter
```

```
storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL11_ID,
dxl11_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL11_ID);
        return;
    end

     % Add Dynamixel#11 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL11_ID,
DXL11_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL11_ID);
        return;
    end




     % Add Dynamixel#12 goal position value to the Syncwrite parameter
storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL12_ID,
dxl12_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL12_ID);
        return;
    end
```

```
     % Add Dynamixel#12 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL12_ID,
DXL12_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL12_ID);
        return;
    end



     % Add Dynamixel#13 goal position value to the Syncwrite parameter
storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL13_ID,
dxl13_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL13_ID);
        return;
    end

     % Add Dynamixel#13 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL13_ID,
DXL13_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL13_ID);
        return;
    end
```

```
     % Add Dynamixel#14 goal position value to the Syncwrite parameter
storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL15_ID,
dxl15_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL15_ID);
        return;
    end

     % Add Dynamixel#14 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL15_ID,
DXL15_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL15_ID);
        return;
    end

     % Add Dynamixel#16 goal position value to the Syncwrite parameter
storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL18_ID,
dxl18_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL18_ID);
        return;
    end

     % Add Dynamixel#16 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL18_ID,
DXL18_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL18_ID);
        return;
    end



       % Add Dynamixel#16 goal position value to the Syncwrite parameter
storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL19_ID,
dxl19_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL19_ID);
        return;
    end

     % Add Dynamixel#16 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL19_ID,
DXL19_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL19_ID);
        return;
    end


       % Add Dynamixel#16 goal position value to the Syncwrite parameter
storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL21_ID,
dxl21_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL21_ID);
        return;
    end

     % Add Dynamixel#16 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL21_ID,
DXL21_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL21_ID);
        return;
    end

    % Syncwrite goal position



    groupSyncWriteTxPacket(group2_num);  %WRITE SPEED FOR DYMX
    groupSyncWriteTxPacket(group1_num);
    dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
    if dxl_comm_result ~= COMM_SUCCESS
        fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION,
dxl_comm_result));
    end

    % Clear syncwrite parameter storage
    groupSyncWriteClearParam(group2_num);
    groupSyncWriteClearParam(group1_num);
    % Change goal position
    end

    if index == 1
        index = 2;
    else
        index = 1;
    end

end




% Disable Dynamixel#6 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL1_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_DISABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
end


% Disable Dynamixel#6 Torque
```

```
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL2_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_DISABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
end


% Disable Dynamixel#6 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL3_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_DISABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
end


% Disable Dynamixel#6 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL4_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_DISABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
```

```
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
end



% Disable Dynamixel#6 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL5_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_DISABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
end




% Disable Dynamixel#6 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL6_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_DISABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
end
```

```
% Disable Dynamixel#7 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL7_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_DISABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
end

% Disable Dynamixel#8 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL8_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_DISABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
end

% Disable Dynamixel#9 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL9_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_DISABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
```

```
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
end

% Disable Dynamixel#10 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL10_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_DISABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
end

% Disable Dynamixel#11 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL11_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_DISABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
end

% Disable Dynamixel#12 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL12_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_DISABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
end

% Disable Dynamixel#13 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL13_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_DISABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
end

% Disable Dynamixel#14 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL15_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_DISABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
```

```
end


% Disable Dynamixel#16 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL18_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_DISABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
end



% Disable Dynamixel#6 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL19_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_DISABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
end



% Disable Dynamixel#6 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL21_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_DISABLE);
```

```
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
end



% Close port
closePort(port_num);


% Unload Library
unloadlibrary(lib_name);


close all;
clear all;
```

Appendix VIII: Matlab code for fanning out the playing cards of the robotic hand

```
clc;

clear all;


lib_name = '';


if strcmp(computer, 'PCWIN')

  lib_name = 'dxl_x86_c';

elseif strcmp(computer, 'PCWIN64')

  lib_name = 'dxl_x64_c';

elseif strcmp(computer, 'GLNX86')

  lib_name = 'libdxl_x86_c';

elseif strcmp(computer, 'GLNXA64')

  lib_name = 'libdxl_x64_c';

elseif strcmp(computer, 'MACI64')

  lib_name = 'libdxl_mac_c';

end


% Load Libraries

if ~libisloaded(lib_name)

    [notfound, warnings] = loadlibrary(lib_name, 'dynamixel_sdk.h',

'addheader', 'port_handler.h', 'addheader', 'packet_handler.h',

'addheader', 'group_sync_write.h');

end


% Control table address

ADDR_MX_TORQUE_ENABLE       = 24;           % Control table address is
```

```
different in Dynamixel model
ADDR_MX_GOAL_POSITION       = 30;
ADDR_MX_PRESENT_POSITION    = 36;
ADDR_MX_GOAL_SPEED          = 32;



% Data Byte Length
LEN_MX_GOAL_POSITION        = 2;
LEN_MX_PRESENT_POSITION     = 2;
LEN_MX_GOAL_SPEED           = 2;

% Protocol version
PROTOCOL_VERSION            = 1.0;          % See which protocol
version is used in the Dynamixel

% Default setting

%thumb
DXL11_ID                     = 11;            % Thumb flex
DXL12_ID                     = 12;            % Thumb abd
DXL13_ID                     = 13;            % Thumb ext DIP
DXL14_ID                     = 14;            % Thumb ext PIP
DXL15_ID                     = 15;            % Thumb add

%index
DXL21_ID                     = 21;            % FDP
DXL22_ID                     = 22;            % FDS
DXL23_ID                     = 23;            % LE
DXL24_ID                     = 24;            % RI
```

```
DXL25_ID                     = 25;           % UI

%middle
DXL31_ID                     = 31;           % FDP
DXL32_ID                     = 32;           % FDS
DXL33_ID                     = 33;           % LE
DXL34_ID                     = 34;           % RI

DXL134_ID                     = 134;           % MIDDLE UI; RING RI

%ring
DXL41_ID                     = 41;           % FDP
DXL42_ID                     = 42;           % FDS
DXL43_ID                     = 43;           % LE

DXL145_ID                     = 145;           % RING UI; LITTLE RI

%little
DXL51_ID                     = 51;           % FDP
DXL52_ID                     = 52;           % FDS
DXL53_ID                     = 53;           % LE
DXL54_ID                     = 54;           % UI
DXL55_ID                     = 55;           % OPP


BAUDRATE                    = 1000000;
DEVICENAME                  = 'COM3';       % Check which port is being
used on your controller
                                             % ex) Windows: 'COM1'
```

```
Linux: '/dev/ttyUSB0' Mac: '/dev/tty.usbserial-*'

TORQUE_ENABLE               = 1;            % Value for enabling the
torque
TORQUE_DISABLE              = 0;            % Value for disabling the
torque

%POSITION 3 bsc stps

DXL11_1_POSITION_VALUE  = 1036;          % Dynamixel will rotate
between this value
DXL11_2_POSITION_VALUE  = 1658;
DXL11_3_POSITION_VALUE  = 2526;
DXL11_4_POSITION_VALUE  = 1820;
DXL11_5_POSITION_VALUE  = 2025;
%1250
DXL12_1_POSITION_VALUE  = 1505;          % Dynamixel will rotate
between this value
DXL12_2_POSITION_VALUE  = 1810;         % and this value (note that the
Dynamixel would not move when the position value is out of movable range.
Check e-manual about the range of the Dynamixel you use.)
DXL12_3_POSITION_VALUE  = 1778;
DXL12_4_POSITION_VALUE  = 1681;
DXL12_5_POSITION_VALUE  = 2025;
%1100
DXL13_1_POSITION_VALUE  = 2590;          % Dynamixel will rotate
between this value
DXL13_2_POSITION_VALUE  = 2200;
DXL13_3_POSITION_VALUE  = 1970;
```

```
DXL13_4_POSITION_VALUE  = 1436;

DXL13_5_POSITION_VALUE  = 1888;

%2050e

DXL14_1_POSITION_VALUE  = 2400;          % Dynamixel will rotate
between this value

DXL14_2_POSITION_VALUE  = 1750;          % and this value (note that the
Dynamixel would not move when the position value is out of movable range.
Check e-manual about the range of the Dynamixel you use.)

DXL14_3_POSITION_VALUE  = 1550;

DXL14_4_POSITION_VALUE  = 2945;

DXL14_5_POSITION_VALUE  = 2185;

%1500

DXL15_1_POSITION_VALUE  = 850;           % Dynamixel will rotate between
this value

DXL15_2_POSITION_VALUE  = 1475;          % and this value (note that the
Dynamixel would not move when the position value is out of movable range.
Check e-manual about the range of the Dynamixel you use.)

DXL15_3_POSITION_VALUE  = 1889;

DXL15_4_POSITION_VALUE  = 713;

DXL15_5_POSITION_VALUE  = 1125;

%1500



DXL21_1_POSITION_VALUE  = 2150;          % Dynamixel will rotate
between this value

DXL21_2_POSITION_VALUE  = 2383;          % and this value (note that the
Dynamixel would not move when the position value is out of movable range.
Check e-manual about the range of the Dynamixel you use.)

DXL21_3_POSITION_VALUE  = 2283;
```

```
DXL21_4_POSITION_VALUE  = 2936;

DXL21_5_POSITION_VALUE  = 2955;

%3520

DXL22_1_POSITION_VALUE  = 1709;          % Dynamixel will rotate
between this value

DXL22_2_POSITION_VALUE  = 2033;         % and this value (note that the
Dynamixel would not move when the position value is out of movable range.
Check e-manual about the range of the Dynamixel you use.)

DXL22_3_POSITION_VALUE  = 1933;

DXL22_4_POSITION_VALUE  = 1376;

DXL22_5_POSITION_VALUE  = 1376;

%2250

DXL23_1_POSITION_VALUE  = 2465;          % Dynamixel will rotate
between this value

DXL23_2_POSITION_VALUE  = 2008;         % and this value (note that the
Dynamixel would not move when the position value is out of movable range.
Check e-manual about the range of the Dynamixel you use.)

DXL23_3_POSITION_VALUE  = 2108;

DXL23_4_POSITION_VALUE  = 2870;

DXL23_5_POSITION_VALUE  = 1880;

%3000

DXL24_1_POSITION_VALUE  = 1227;          % Dynamixel will rotate
between this value

DXL24_2_POSITION_VALUE  = 1177;         % and this value (note that the
Dynamixel would not move when the position value is out of movable range.
Check e-manual about the range of the Dynamixel you use.)

DXL24_3_POSITION_VALUE  = 1577;

DXL24_4_POSITION_VALUE  = 300;

DXL24_5_POSITION_VALUE  = 1655;
```

```
%3000
DXL25_1_POSITION_VALUE  = 650;          % Dynamixel will rotate between
this value
DXL25_2_POSITION_VALUE  = 1711;         % and this value (note that the
Dynamixel would not move when the position value is out of movable range.
Check e-manual about the range of the Dynamixel you use.)
DXL25_3_POSITION_VALUE  = 1711;
DXL25_4_POSITION_VALUE  = 400;
DXL25_5_POSITION_VALUE  = 2245;
%3000


DXL31_1_POSITION_VALUE  = 1031;         % Dynamixel will rotate
between this value
DXL31_2_POSITION_VALUE  = 1828;
DXL31_3_POSITION_VALUE  = 1728;
DXL31_4_POSITION_VALUE  = 2915;
DXL31_5_POSITION_VALUE  = 2915;
%1180
DXL32_1_POSITION_VALUE  = 1610;         % Dynamixel will rotate
between this value
DXL32_2_POSITION_VALUE  = 2189;
DXL32_3_POSITION_VALUE  = 2089;
DXL32_4_POSITION_VALUE  = 2514;
DXL32_5_POSITION_VALUE  = 2514;
%3440
DXL33_1_POSITION_VALUE  = 2789;         % Dynamixel will rotate
between this value
DXL33_2_POSITION_VALUE  = 2424;
```

```
DXL33_3_POSITION_VALUE  = 2524;

DXL33_4_POSITION_VALUE  = 1342;

DXL33_5_POSITION_VALUE  = 1342;

%1300

DXL34_1_POSITION_VALUE  = 1694;           % Dynamixel will rotate
between this value

DXL34_2_POSITION_VALUE  = 1590;

DXL34_3_POSITION_VALUE  = 1590;

DXL34_4_POSITION_VALUE  = 538;

DXL34_5_POSITION_VALUE  = 538;

%1300




DXL134_1_POSITION_VALUE  = 2300;           % Dynamixel will rotate
between this value

DXL134_2_POSITION_VALUE  = 2405;

DXL134_3_POSITION_VALUE  = 2405;

DXL134_4_POSITION_VALUE  = 1535;

DXL134_5_POSITION_VALUE  = 1535;

%1300




DXL41_1_POSITION_VALUE  = 1502;           % Dynamixel will rotate
between this value

DXL41_2_POSITION_VALUE  = 2231;

DXL41_3_POSITION_VALUE  = 2231;

DXL41_4_POSITION_VALUE  = 2750;

DXL41_5_POSITION_VALUE  = 2750;
```

```
%1180
DXL42_1_POSITION_VALUE  = 1198;          % Dynamixel will rotate
between this value
DXL42_2_POSITION_VALUE  = 1535;
DXL42_3_POSITION_VALUE  = 1435;
DXL42_4_POSITION_VALUE  = 1950;
DXL42_5_POSITION_VALUE  = 1950;
%3440
DXL43_1_POSITION_VALUE  = 1977;          % Dynamixel will rotate
between this value
DXL43_2_POSITION_VALUE  = 1516;
DXL43_3_POSITION_VALUE  = 1516;
DXL43_4_POSITION_VALUE  = 950;
DXL43_5_POSITION_VALUE  = 950;
%1300


DXL145_1_POSITION_VALUE  = 1896;          % Dynamixel will rotate
between this value
DXL145_2_POSITION_VALUE  = 1896;
DXL145_3_POSITION_VALUE  = 1896;
DXL145_4_POSITION_VALUE  = 1900;
DXL145_5_POSITION_VALUE  = 1900;
%1300


DXL51_1_POSITION_VALUE  = 1437;          % Dynamixel will rotate
between this value
DXL51_2_POSITION_VALUE  = 2485;         % and this value (note that the
Dynamixel would not move when the position value is out of movable range.
Check e-manual about the range of the Dynamixel you use.)
DXL51_3_POSITION_VALUE  = 2485;
DXL51_4_POSITION_VALUE  = 3242;
DXL51_5_POSITION_VALUE  = 3242;
%3520
DXL52_1_POSITION_VALUE  = 1548;          % Dynamixel will rotate
between this value
DXL52_2_POSITION_VALUE  = 2094;         % and this value (note that the
Dynamixel would not move when the position value is out of movable range.
Check e-manual about the range of the Dynamixel you use.)
DXL52_3_POSITION_VALUE  = 1994;
DXL52_4_POSITION_VALUE  = 2632;
DXL52_5_POSITION_VALUE  = 2632;
%2250
DXL53_1_POSITION_VALUE  = 1948;          % Dynamixel will rotate
between this value
DXL53_2_POSITION_VALUE  = 1758;         % and this value (note that the
Dynamixel would not move when the position value is out of movable range.
Check e-manual about the range of the Dynamixel you use.)
DXL53_3_POSITION_VALUE  = 1758;
DXL53_4_POSITION_VALUE  = 1200;
DXL53_5_POSITION_VALUE  = 1200;
%3000
DXL54_1_POSITION_VALUE  = 2416;          % Dynamixel will rotate
between this value
DXL54_2_POSITION_VALUE  = 2416;         % and this value (note that the
Dynamixel would not move when the position value is out of movable range.
Check e-manual about the range of the Dynamixel you use.)
```

```
DXL54_3_POSITION_VALUE  = 2416;

DXL54_4_POSITION_VALUE  = 3820;

DXL54_5_POSITION_VALUE  = 3820;

%3000

DXL55_1_POSITION_VALUE  = 2266;          % Dynamixel will rotate
between this value

DXL55_2_POSITION_VALUE  = 2266;          % and this value (note that the
Dynamixel would not move when the position value is out of movable range.
Check e-manual about the range of the Dynamixel you use.)

DXL55_3_POSITION_VALUE  = 2266;

DXL55_4_POSITION_VALUE  = 2900;

DXL55_5_POSITION_VALUE  = 2900;

%3000



DXL_MOVING_STATUS_THRESHOLD = 10;          % Dynamixel moving status
threshold



%SPEED

DXL11_GOAL_SPEED_VALUE        =
50+round(abs((DXL11_2_POSITION_VALUE-DXL11_1_POSITION_VALUE)/50));

DXL12_GOAL_SPEED_VALUE        = 1000;          % Dynamixel moving SPEED

DXL13_GOAL_SPEED_VALUE        =
50+round(abs((DXL13_2_POSITION_VALUE-DXL13_1_POSITION_VALUE)/50));

DXL14_GOAL_SPEED_VALUE        =
50+round(abs((DXL14_2_POSITION_VALUE-DXL14_1_POSITION_VALUE)/50));
       % Dynamixel moving SPEED

DXL15_GOAL_SPEED_VALUE        = 1000;          % Dynamixel moving SPEED
```

```
DXL21_GOAL_SPEED_VALUE        =
round(abs((DXL21_2_POSITION_VALUE-DXL21_1_POSITION_VALUE)/50));
    % Dynamixel moving SPEED
DXL22_GOAL_SPEED_VALUE        = 800;          % Dynamixel moving SPEED
DXL23_GOAL_SPEED_VALUE        =
round(abs((DXL23_2_POSITION_VALUE-DXL23_1_POSITION_VALUE)/50));
    % Dynamixel moving SPEED
DXL24_GOAL_SPEED_VALUE        =
40+round(abs((DXL24_2_POSITION_VALUE-DXL24_1_POSITION_VALUE)/50));
       % Dynamixel moving SPEED
DXL25_GOAL_SPEED_VALUE        =
40+round(abs((DXL25_2_POSITION_VALUE-DXL25_1_POSITION_VALUE)/50));
       % Dynamixel moving SPEED


DXL31_GOAL_SPEED_VALUE        =
round(abs((DXL31_2_POSITION_VALUE-DXL31_1_POSITION_VALUE)/50));
DXL32_GOAL_SPEED_VALUE        = 800;
DXL33_GOAL_SPEED_VALUE        =
round(abs((DXL33_2_POSITION_VALUE-DXL33_1_POSITION_VALUE)/50));
DXL34_GOAL_SPEED_VALUE        =
40+round(abs((DXL34_2_POSITION_VALUE-DXL34_1_POSITION_VALUE)/50));
       % Dynamixel moving SPEED


DXL134_GOAL_SPEED_VALUE        =
40+round(abs((DXL134_2_POSITION_VALUE-DXL134_1_POSITION_VALUE)/50));
```

```
         % Dynamixel moving SPEED


DXL41_GOAL_SPEED_VALUE        =
round(abs((DXL41_2_POSITION_VALUE-DXL41_1_POSITION_VALUE)/50));
DXL42_GOAL_SPEED_VALUE        = 1000;
DXL43_GOAL_SPEED_VALUE        =
round(abs((DXL43_2_POSITION_VALUE-DXL43_1_POSITION_VALUE)/50));


DXL145_GOAL_SPEED_VALUE        = 1000;          % Dynamixel moving
SPEED


DXL51_GOAL_SPEED_VALUE        =
round(abs((DXL51_2_POSITION_VALUE-DXL51_1_POSITION_VALUE)/50));
    % Dynamixel moving SPEED
DXL52_GOAL_SPEED_VALUE        = 1000;          % Dynamixel moving SPEED
DXL53_GOAL_SPEED_VALUE        =
round(abs((DXL53_2_POSITION_VALUE-DXL53_1_POSITION_VALUE)/50));
    % Dynamixel moving SPEED
DXL54_GOAL_SPEED_VALUE        =
round(abs((DXL54_2_POSITION_VALUE-DXL54_1_POSITION_VALUE)/50));
    % Dynamixel moving SPEED
DXL55_GOAL_SPEED_VALUE        = 1000;          % Dynamixel moving SPEED


ESC_CHARACTER               = 'e';          % Key for escaping loop
```

```
COMM_SUCCESS                = 0;            % Communication Success
result value
COMM_TX_FAIL                = -1001;        % Communication Tx Failed

% Initialize PortHandler Structs
% Set the port path
% Get methods and members of PortHandlerLinux or PortHandlerWindows
port_num = portHandler(DEVICENAME);

% Initialize PacketHandler Structs
packetHandler();

% Initialize Groupsyncwrite instance
group1_num = groupSyncWrite(port_num, PROTOCOL_VERSION,
ADDR_MX_GOAL_POSITION, LEN_MX_GOAL_POSITION);
group2_num = groupSyncWrite(port_num, PROTOCOL_VERSION,
ADDR_MX_GOAL_SPEED, LEN_MX_GOAL_SPEED);
index = 1;
dxl_comm_result = COMM_TX_FAIL;             % Communication result
dxl_addparam_result1 = false;                % AddParam result

dxl11_goal_position = [DXL11_1_POSITION_VALUE DXL11_2_POSITION_VALUE
DXL11_3_POSITION_VALUE DXL11_4_POSITION_VALUE DXL11_5_POSITION_VALUE];
dxl12_goal_position = [DXL12_1_POSITION_VALUE DXL12_2_POSITION_VALUE
DXL12_3_POSITION_VALUE DXL12_4_POSITION_VALUE
DXL12_5_POSITION_VALUE];         % Goal position
dxl13_goal_position = [DXL13_1_POSITION_VALUE DXL13_2_POSITION_VALUE
DXL13_3_POSITION_VALUE DXL13_4_POSITION_VALUE DXL13_5_POSITION_VALUE];
```

```
dxl14_goal_position = [DXL14_1_POSITION_VALUE DXL14_2_POSITION_VALUE
DXL14_3_POSITION_VALUE DXL14_4_POSITION_VALUE
DXL14_5_POSITION_VALUE];         % Goal position
dxl15_goal_position = [DXL15_1_POSITION_VALUE DXL15_2_POSITION_VALUE
DXL15_3_POSITION_VALUE DXL15_4_POSITION_VALUE
DXL15_5_POSITION_VALUE];         % Goal position


dxl21_goal_position = [DXL21_1_POSITION_VALUE DXL21_2_POSITION_VALUE
DXL21_3_POSITION_VALUE DXL21_4_POSITION_VALUE
DXL21_5_POSITION_VALUE];         % Goal position
dxl22_goal_position = [DXL22_1_POSITION_VALUE DXL22_2_POSITION_VALUE
DXL22_3_POSITION_VALUE DXL22_4_POSITION_VALUE
DXL22_5_POSITION_VALUE];         % Goal position
dxl23_goal_position = [DXL23_1_POSITION_VALUE DXL23_2_POSITION_VALUE
DXL23_3_POSITION_VALUE DXL23_4_POSITION_VALUE
DXL23_5_POSITION_VALUE];         % Goal position
dxl24_goal_position = [DXL24_1_POSITION_VALUE DXL24_2_POSITION_VALUE
DXL24_3_POSITION_VALUE DXL24_4_POSITION_VALUE
DXL24_5_POSITION_VALUE];         % Goal position
dxl25_goal_position = [DXL25_1_POSITION_VALUE DXL25_2_POSITION_VALUE
DXL25_3_POSITION_VALUE DXL25_4_POSITION_VALUE
DXL25_5_POSITION_VALUE];         % Goal position


dxl31_goal_position = [DXL31_1_POSITION_VALUE DXL31_2_POSITION_VALUE
DXL31_3_POSITION_VALUE DXL31_4_POSITION_VALUE DXL31_5_POSITION_VALUE];
dxl32_goal_position = [DXL32_1_POSITION_VALUE DXL32_2_POSITION_VALUE
DXL32_3_POSITION_VALUE DXL32_4_POSITION_VALUE DXL32_5_POSITION_VALUE];
```

```
dxl33_goal_position = [DXL33_1_POSITION_VALUE DXL33_2_POSITION_VALUE
DXL33_3_POSITION_VALUE DXL33_4_POSITION_VALUE DXL33_5_POSITION_VALUE];
dxl34_goal_position = [DXL34_1_POSITION_VALUE DXL34_2_POSITION_VALUE
DXL34_3_POSITION_VALUE DXL34_4_POSITION_VALUE
DXL34_5_POSITION_VALUE];         % Goal position


dxl134_goal_position = [DXL134_1_POSITION_VALUE
DXL134_2_POSITION_VALUE DXL134_3_POSITION_VALUE
DXL134_4_POSITION_VALUE DXL134_5_POSITION_VALUE];         % Goal
position


dxl41_goal_position = [DXL41_1_POSITION_VALUE DXL41_2_POSITION_VALUE
DXL41_3_POSITION_VALUE DXL41_4_POSITION_VALUE DXL41_5_POSITION_VALUE];
dxl42_goal_position = [DXL42_1_POSITION_VALUE DXL42_2_POSITION_VALUE
DXL42_3_POSITION_VALUE DXL42_4_POSITION_VALUE DXL42_5_POSITION_VALUE];
dxl43_goal_position = [DXL43_1_POSITION_VALUE DXL43_2_POSITION_VALUE
DXL43_3_POSITION_VALUE DXL43_4_POSITION_VALUE DXL43_5_POSITION_VALUE];


dxl145_goal_position = [DXL145_1_POSITION_VALUE
DXL145_2_POSITION_VALUE DXL145_3_POSITION_VALUE
DXL145_4_POSITION_VALUE DXL145_5_POSITION_VALUE];         % Goal
position


dxl51_goal_position = [DXL51_1_POSITION_VALUE DXL51_2_POSITION_VALUE
DXL51_3_POSITION_VALUE DXL51_4_POSITION_VALUE
DXL51_5_POSITION_VALUE];         % Goal position
dxl52_goal_position = [DXL52_1_POSITION_VALUE DXL52_2_POSITION_VALUE
DXL52_3_POSITION_VALUE DXL52_4_POSITION_VALUE
DXL52_5_POSITION_VALUE];         % Goal position
dxl53_goal_position = [DXL53_1_POSITION_VALUE DXL53_2_POSITION_VALUE
DXL53_3_POSITION_VALUE DXL53_4_POSITION_VALUE
DXL53_5_POSITION_VALUE];         % Goal position
dxl54_goal_position = [DXL54_1_POSITION_VALUE DXL54_2_POSITION_VALUE
DXL54_3_POSITION_VALUE DXL54_4_POSITION_VALUE
DXL54_5_POSITION_VALUE];         % Goal position
dxl55_goal_position = [DXL55_1_POSITION_VALUE DXL55_2_POSITION_VALUE
DXL55_3_POSITION_VALUE DXL55_4_POSITION_VALUE
DXL55_5_POSITION_VALUE];         % Goal position



dxl_error = 0;                              % Dynamixel error
dxl1_present_position = 0;                  % Present position
dxl5_present_position = 0;



% Open port
if (openPort(port_num))
    fprintf('Succeeded to open the port!\n');
else
    unloadlibrary(lib_name);
    fprintf('Failed to open the port!\n');
    input('Press any key to terminate...\n');
    return;
```

```
end



% Set port baudrate
if (setBaudRate(port_num, BAUDRATE))
    fprintf('Succeeded to change the baudrate!\n');
else
    unloadlibrary(lib_name);
    fprintf('Failed to change the baudrate!\n');
    input('Press any key to terminate...\n');
    return;
end



%ENABLE MOTOR

% Enable Dynamixel#11 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL11_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_ENABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
else
    fprintf('Dynamixel has been successfully connected \n');
end
```

```
% Enable Dynamixel#12 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL12_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_ENABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
else
    fprintf('Dynamixel has been successfully connected \n');
end

% Enable Dynamixel#13 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL13_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_ENABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
else
    fprintf('Dynamixel has been successfully connected \n');
end

% Enable Dynamixel#14 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL14_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_ENABLE);
```

```
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);

dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);

if dxl_comm_result ~= COMM_SUCCESS

    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));

elseif dxl_error ~= 0

    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));

else

    fprintf('Dynamixel has been successfully connected \n');

end


% Enable Dynamixel#14 Torque

write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL15_ID,

ADDR_MX_TORQUE_ENABLE, TORQUE_ENABLE);

dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);

dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);

if dxl_comm_result ~= COMM_SUCCESS

    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));

elseif dxl_error ~= 0

    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));

else

    fprintf('Dynamixel has been successfully connected \n');

end




% Enable Dynamixel#21 Torque

write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL21_ID,

ADDR_MX_TORQUE_ENABLE, TORQUE_ENABLE);

dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
```

```
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
else
    fprintf('Dynamixel has been successfully connected \n');
end

% Enable Dynamixel#8 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL22_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_ENABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
else
    fprintf('Dynamixel has been successfully connected \n');
end

% Enable Dynamixel#10 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL23_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_ENABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
```

```
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
else
    fprintf('Dynamixel has been successfully connected \n');
end

% Enable Dynamixel#10 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL24_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_ENABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
else
    fprintf('Dynamixel has been successfully connected \n');
end

% Enable Dynamixel#10 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL25_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_ENABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
else
```

```
    fprintf('Dynamixel has been successfully connected \n');
end




% Enable Dynamixel#7 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL31_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_ENABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
else
    fprintf('Dynamixel has been successfully connected \n');
end

% Enable Dynamixel#9 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL32_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_ENABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
else
    fprintf('Dynamixel has been successfully connected \n');
```

```
end


% Enable Dynamixel#16 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL33_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_ENABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
else
    fprintf('Dynamixel has been successfully connected \n');
end


% Enable Dynamixel#16 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL34_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_ENABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
else
    fprintf('Dynamixel has been successfully connected \n');
end
```

```
% Enable Dynamixel#16 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL134_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_ENABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
else
    fprintf('Dynamixel has been successfully connected \n');
end



% Enable Dynamixel#7 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL41_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_ENABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
else
    fprintf('Dynamixel has been successfully connected \n');
end
```

```
% Enable Dynamixel#9 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL42_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_ENABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
else
    fprintf('Dynamixel has been successfully connected \n');
end

% Enable Dynamixel#16 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL43_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_ENABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
else
    fprintf('Dynamixel has been successfully connected \n');
end



% Enable Dynamixel#16 Torque
```

```
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL145_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_ENABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
else
    fprintf('Dynamixel has been successfully connected \n');
end



% Enable Dynamixel#21 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL51_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_ENABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
else
    fprintf('Dynamixel has been successfully connected \n');
end

% Enable Dynamixel#8 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL52_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_ENABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
else
    fprintf('Dynamixel has been successfully connected \n');
end

% Enable Dynamixel#10 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL53_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_ENABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
else
    fprintf('Dynamixel has been successfully connected \n');
end

% Enable Dynamixel#10 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL54_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_ENABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
```

```
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
else
    fprintf('Dynamixel has been successfully connected \n');
end

% Enable Dynamixel#10 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL55_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_ENABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
else
    fprintf('Dynamixel has been successfully connected \n');
end



%MAIN LOOP   3steps
p=1;
while 1
    if input('Press any key to continue! (or input e to quit!)\n', 's')
== ESC_CHARACTER
        break;
    end


    while p<6
        index=1;
    while index<4
      % Add Dynamixel#11 goal position value to the Syncwrite parameter
storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL11_ID,
dxl11_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL11_ID);
        return;
    end

     % Add Dynamixel#11 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL11_ID,
DXL11_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL11_ID);
        return;
    end

     % Add Dynamixel#12 goal position value to the Syncwrite parameter
storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL12_ID,
dxl12_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL12_ID);
        return;
    end

     % Add Dynamixel#12 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL12_ID,
DXL12_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL12_ID);
        return;
    end


     % Add Dynamixel#13 goal position value to the Syncwrite parameter
storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL13_ID,
dxl13_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL13_ID);
        return;
    end

     % Add Dynamixel#13 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL13_ID,
DXL13_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL13_ID);
        return;
    end

     % Add Dynamixel#14 goal position value to the Syncwrite parameter
```

```
storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL14_ID,
dxl14_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL14_ID);
        return;
    end

     % Add Dynamixel#14 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL14_ID,
DXL14_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL14_ID);
        return;
    end

         % Add Dynamixel#14 goal position value to the Syncwrite
parameter storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL15_ID,
dxl15_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL15_ID);
        return;
    end

     % Add Dynamixel#14 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL15_ID,
DXL15_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
```

```
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL15_ID);
        return;
    end




    % Add Dynamixel#6 goal position value to the Syncwrite storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL21_ID,
dxl21_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL21_ID);
        return;
    end

    % Add Dynamixel#6 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL21_ID,
DXL21_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL21_ID);
        return;
    end

     % Add Dynamixel#8 goal position value to the Syncwrite parameter
storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL22_ID,
dxl22_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL22_ID);
        return;
    end

     % Add Dynamixel#8 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL22_ID,
DXL22_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL22_ID);
        return;
    end

        % Add Dynamixel#10 goal position value to the Syncwrite parameter
storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL23_ID,
dxl23_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL23_ID);
        return;
    end

     % Add Dynamixel#10 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL23_ID,
DXL23_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL23_ID);
        return;
    end

        % Add Dynamixel#10 goal position value to the Syncwrite parameter
storage
```

```
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL24_ID,
dxl24_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL24_ID);
        return;
    end

     % Add Dynamixel#10 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL24_ID,
DXL24_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL24_ID);
        return;
    end

        % Add Dynamixel#10 goal position value to the Syncwrite parameter
storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL25_ID,
dxl25_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL25_ID);
        return;
    end

     % Add Dynamixel#10 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL25_ID,
DXL25_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL25_ID);
        return;
    end




       % Add Dynamixel#7 goal position value to the Syncwrite parameter
storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL31_ID,
dxl31_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL31_ID);
        return;
    end

     % Add Dynamixel#7 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL31_ID,
DXL31_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL31_ID);
        return;
    end

     % Add Dynamixel#9 goal position value to the Syncwrite parameter
storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL32_ID,
dxl32_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL32_ID);
        return;
    end

     % Add Dynamixel#9 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL32_ID,
DXL32_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL32_ID);
        return;
    end

     % Add Dynamixel#16 goal position value to the Syncwrite parameter
storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL33_ID,
dxl33_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL33_ID);
        return;
    end

     % Add Dynamixel#16 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL33_ID,
DXL33_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL33_ID);
        return;
    end

        % Add Dynamixel#16 goal position value to the Syncwrite parameter
storage
```

```
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL34_ID,
dxl34_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL34_ID);
        return;
    end

     % Add Dynamixel#16 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL34_ID,
DXL34_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL34_ID);
        return;
    end



       % Add Dynamixel#16 goal position value to the Syncwrite parameter
storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num,
DXL134_ID, dxl134_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL134_ID);
        return;
    end

     % Add Dynamixel#16 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num,
DXL134_ID, DXL134_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
```

```
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL134_ID);
        return;
    end




       % Add Dynamixel#7 goal position value to the Syncwrite parameter
storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL41_ID,
dxl41_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL41_ID);
        return;
    end

     % Add Dynamixel#7 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL41_ID,
DXL41_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL41_ID);
        return;
    end

     % Add Dynamixel#9 goal position value to the Syncwrite parameter
storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL42_ID,
dxl42_goal_position(index), LEN_MX_GOAL_POSITION);
```

```
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL42_ID);
        return;
    end

     % Add Dynamixel#9 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL42_ID,
DXL42_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL42_ID);
        return;
    end

     % Add Dynamixel#16 goal position value to the Syncwrite parameter
storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL43_ID,
dxl43_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL43_ID);
        return;
    end

     % Add Dynamixel#16 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL43_ID,
DXL43_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL43_ID);
        return;
    end
```

```
        % Add Dynamixel#16 goal position value to the Syncwrite parameter
storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num,
DXL145_ID, dxl145_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL145_ID);
        return;
    end

     % Add Dynamixel#16 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num,
DXL145_ID, DXL145_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL145_ID);
        return;
    end



        % Add Dynamixel#6 goal position value to the Syncwrite storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL51_ID,
dxl51_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL51_ID);
        return;
    end
    % Add Dynamixel#6 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL51_ID,
DXL51_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL51_ID);
        return;
    end

     % Add Dynamixel#8 goal position value to the Syncwrite parameter
storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL52_ID,
dxl52_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL52_ID);
        return;
    end

     % Add Dynamixel#8 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL52_ID,
DXL52_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL52_ID);
        return;
    end

        % Add Dynamixel#10 goal position value to the Syncwrite parameter
storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL53_ID,
dxl53_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL53_ID);
        return;
    end

     % Add Dynamixel#10 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL53_ID, 
DXL53_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL53_ID);
        return;
    end

        % Add Dynamixel#10 goal position value to the Syncwrite parameter 
storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL54_ID, 
dxl54_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL54_ID);
        return;
    end

     % Add Dynamixel#10 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL54_ID, 
DXL54_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL54_ID);
        return;
    end

        % Add Dynamixel#10 goal position value to the Syncwrite parameter
storage
    dxl_addparam_result1 = groupSyncWriteAddParam(group1_num, DXL55_ID,
dxl55_goal_position(index), LEN_MX_GOAL_POSITION);
    if dxl_addparam_result1 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL55_ID);
        return;
    end

     % Add Dynamixel#10 goal SPEED value to the Syncwrite storage
    dxl_addparam_result2 = groupSyncWriteAddParam(group2_num, DXL55_ID,
DXL55_GOAL_SPEED_VALUE, LEN_MX_GOAL_SPEED);
    if dxl_addparam_result2 ~= true
        fprintf('[ID:%03d] groupSyncWrite addparam failed', DXL55_ID);
        return;
    end




    % Syncwrite goal position
    groupSyncWriteTxPacket(group2_num);  %WRITE SPEED FOR DYMX
    groupSyncWriteTxPacket(group1_num);
    dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
    if dxl_comm_result ~= COMM_SUCCESS
        fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION,
dxl_comm_result));
```

```
    end

    % Clear syncwrite parameter storage
    groupSyncWriteClearParam(group2_num);
    groupSyncWriteClearParam(group1_num);

    pause(1);



    index=index+1;
    end
    p=p+1;
    end
end



% Disable Dynamixel#11 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL11_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_DISABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
end

% Disable Dynamixel#12 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL12_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_DISABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
end

% Disable Dynamixel#13 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL13_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_DISABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
end

% Disable Dynamixel#14 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL14_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_DISABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
```

```
end

% Disable Dynamixel#14 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL15_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_DISABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
end



% Disable Dynamixel#6 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL21_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_DISABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
end

% Disable Dynamixel#8 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL22_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_DISABLE);
```

```
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);

dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);

if dxl_comm_result ~= COMM_SUCCESS

    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));

elseif dxl_error ~= 0

    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));

end


% Disable Dynamixel#10 Torque

write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL23_ID,

ADDR_MX_TORQUE_ENABLE, TORQUE_DISABLE);

dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);

dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);

if dxl_comm_result ~= COMM_SUCCESS

    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));

elseif dxl_error ~= 0

    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));

end


% Disable Dynamixel#10 Torque

write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL24_ID,

ADDR_MX_TORQUE_ENABLE, TORQUE_DISABLE);

dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);

dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);

if dxl_comm_result ~= COMM_SUCCESS

    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));

elseif dxl_error ~= 0

    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));

end
```

```
% Disable Dynamixel#10 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL25_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_DISABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
end



% Disable Dynamixel#7 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL31_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_DISABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
end

% Disable Dynamixel#9 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL32_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_DISABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
```

```
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
end

% Disable Dynamixel#16 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL33_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_DISABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
end

% Disable Dynamixel#16 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL34_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_DISABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
end
```

```
% Disable Dynamixel#16 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL134_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_DISABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
end



% Disable Dynamixel#7 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL41_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_DISABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
end

% Disable Dynamixel#9 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL42_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_DISABLE);
```

```
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
end

% Disable Dynamixel#16 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL43_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_DISABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
end



% Disable Dynamixel#16 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL145_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_DISABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
```

```
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
end




% Disable Dynamixel#6 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL51_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_DISABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
end

% Disable Dynamixel#8 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL52_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_DISABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
end

% Disable Dynamixel#10 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL53_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_DISABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
end

% Disable Dynamixel#10 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL54_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_DISABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
end

% Disable Dynamixel#10 Torque
write1ByteTxRx(port_num, PROTOCOL_VERSION, DXL55_ID,
ADDR_MX_TORQUE_ENABLE, TORQUE_DISABLE);
dxl_comm_result = getLastTxRxResult(port_num, PROTOCOL_VERSION);
dxl_error = getLastRxPacketError(port_num, PROTOCOL_VERSION);
if dxl_comm_result ~= COMM_SUCCESS
    fprintf('%s\n', getTxRxResult(PROTOCOL_VERSION, dxl_comm_result));
elseif dxl_error ~= 0
    fprintf('%s\n', getRxPacketError(PROTOCOL_VERSION, dxl_error));
```

```
end




% Close port
closePort(port_num);

% Unload Library
unloadlibrary(lib_name);

close all;
clear all;
```