

2016-04-27

Actuation, Control and Environment Setup for a bi-joint hydro-muscle driven leg structure

Amaid Zia

Worcester Polytechnic Institute

Follow this and additional works at: <https://digitalcommons.wpi.edu/etd-theses>

Repository Citation

Zia, Amaid, "Actuation, Control and Environment Setup for a bi-joint hydro-muscle driven leg structure" (2016). *Masters Theses (All Theses, All Years)*. 390.

<https://digitalcommons.wpi.edu/etd-theses/390>

This thesis is brought to you for free and open access by [Digital WPI](#). It has been accepted for inclusion in Masters Theses (All Theses, All Years) by an authorized administrator of Digital WPI. For more information, please contact wpi-etd@wpi.edu.

Actuation, Control and Environment Setup for a bi-joint hydro-muscle driven leg structure

by

AMAID ZIA

A Thesis

Submitted to the Faculty

of the

WORCESTER POLYTECHNIC INSTITUTE

in partial fulfillment of the requirements for the

Degree of Master of Science

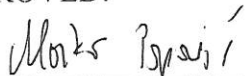
in

Robotics Engineering

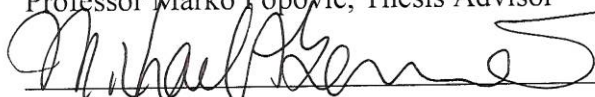
By

MAY 2016

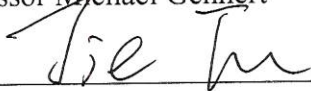
APPROVED:



Professor Marko Popovic, Thesis Advisor



Professor Michael Gennert



Professor Jie Fu

Abstract

About 74 million of the world population needs assistive leg devices on daily basis¹ on account of some form of disability. Although the standard wheelchairs perform well on level ground terrains but they prove ineffective on soft ground and in crossing large obstacles. For dealing with the advance challenges of navigating the human environment a biped walker seems to a more suitable choice. The research presented here is focused on building and actuating a two-joint leg structure that is an integral part of ongoing leg-chair project in Popovic Labs. The project provides a proof of concept that how the synthetic hydro muscles (also developed by Popovic Labs) can be used for the control of large artificial limb joints. Here we discuss the designing, testing and corresponding refining of electronics design, actuation and control of the bi-joint hydro muscle actuated leg structure .We will also elaborate on the requirements, design, problems and refinement of some of the important mechanical components like Coupler/Decoupler and Force Multipliers.

¹ "Fact sheet on wheelchairs - World Health Organization ..." 2013. 22 Apr. 2016
<http://www.searo.who.int/entity/disabilities_injury_rehabilitation/wheelchair_factsheet.pdf>

Acknowledgments

I would like to express my sincere gratitude towards Professor Marko Popovic for giving me the opportunity to work on the projects mentioned in this thesis and his invaluable support and direction throughout the duration of this work. I would also like to thank Professor Jie Fu and Professor Michael Gennert for being a part of my advising committee. Popovic Labs has been a great place to collaborate and learn. Thanks to Ananth Jonnavittula, Saivimal Sridar, Seiichiro Uda, Matt Bowers , Matt Rafferety, John Kelly and Lynn Koesterman for their support and help throughout the project.

Lastly, I would like to thank my parents for the motivation and guidance .

Contents

Table of Contents

Actuation, Control and Environment Setup for a bi-joint hydro-muscle driven leg structure	1
Abstract.....	i
Acknowledgments.....	ii
Contents.....	iii
List of figures.....	vii
List of Tables	viii
Chapter 1	1
Introduction	1
1.1 Hip and Knee Anatomy	2
1.1.1 Knee Anatomy.....	3
1.1.2 Hip Joint.....	4
1.2 Synthetic Muscles.....	6
1.3 Hydro Muscles	8
Chapter 2	10
Design Requirements	10
2.1 Need Analysis of the Whole System.....	10
2.1.1 Critical Needs.....	11

2.2 System Requirements	11
2.3 Single Joint Requirement.....	12
2.4 Electronics and Control Requirements	13
2.5 Features.....	14
2.5.1 Required.....	14
2.5.2 Desired/Optional.....	14
Chapter 3	15
System Overview	15
3.1 Hardware.....	15
3.1.1 Mechanical Components Arrangement.....	15
3.1.2 Hydraulics	16
3.2 Electronics.....	17
3.2.1 Sensors	18
3.2.2 Actuators	19
Chapter 4	20
Hardware Design and Specification.....	20
4.1 Basic Skeleton.....	20
.....	21
4.2 Coupler/Decoupler Design Iterations	21
4.2.1 Version 1	21

4.2.2 Version 2	22
4.2.3 Version 3	24
4.2.4 Version 4	24
4.2.5 Version 5	25
4.2.6 The final version.....	27
4.3 Force Multiplier Designing.....	27
4.4 Placement of Components on Frame.....	29
4.5 Manufacturing Test Frame.....	29
Chapter 5	32
Software Design and Specification.....	32
5.1 Overview of Architecture	32
5.2 Choosing a Microcontroller Research	33
5.3 Design Iterations for electronics	37
5.3.1 All YUNs	37
5.3.1 YUNs + Dagu Spider Board.....	38
5.4 MATLAB (Master) Software Architecture (Design Iterations).....	40
5.4.1 Overview of Code.....	40
5.4.2 Controller / Actuator.....	42
5.4.3 Communication Protocol	44
5.5 Controller's Logic	45

5.5.1 Sensor Controller	45
5.5.2 Actuator Controller	46
Chapter 6	48
Conclusions and Future Works.....	48
6.1 Results.....	48
6.1.1 Effect of Coupler/Decoupler on Joint Angle.....	49
6.1.2 Force and Elongation result.....	50
6.1.3 Position Control.....	51
6.2 Suggested Future Works.....	53
Appendix	55
A.1 The MATLAB main loop	55
A.2 The main block of Data acquire Module	57
A.3 The Main block of controller	59
A.4 Arduino actuation code.....	62
A.5 Sensor Code on Arduino	66
References	67

List of figures

1.1	The structure of a simple joint (elbow)	2
1.2	The structure of knee joint	3
1.3	The bone structure of hip joint (Posterior side)	4
1.4	The muscles on hip joint	5
1.5	The muscles on hip joint	5
1.6	The state of Mckibben muscle as it is pressurized.....	7
1.7	The pressurized and relaxed state of a Hydro muscle	8
1.8	The elongation and pressure state of Hydro muscle.....	9
3.1	The arrangement of components on the leg structure	16
3.2	The conceptual diagram of the hydraulics of the system.	17
3.3	The simplified view of electronics of the System.....	18
4.1	The basic skeleton of the leg structure	21
4.2	The first design iteration of Coupler/Decoupler.....	22
4.3	The second design iteration of Coupler/Decoupler.....	23
4.4	The third design iteration of Coupler/Decoupler.....	24
4.5	The fourth design iteration of Coupler/Decoupler.....	25
4.6	The fifth design iteration of Coupler/Decoupler.....	26
4.7	The final design iteration of Coupler/Decoupler.....	27
4.8	The first force multiplier design.....	28
4.9	The final force multiplier design.....	28
4.10	The placement of all components on the leg structure.....	29
4.11	The test structure for testing of just knee joint	30
4.12	The test structure for the knee and the hip joint.....	31
5.1	The conceptual view of electronics	33
5.2	The detailed electronic architecture with all the Yuns.....	38
5.3	The electronic architecture with Yuns and Spider Board	39
5.4	The conceptual program flow between the MATLAB modules.....	40
5.5	The flow diagram of the actuator's logic	43
5.6	The command based communication protocol	44
5.7	Bulk Data transfer communication protocol	45
5.8	The flow diagram of Sensor Controller's logic.....	46
5.9	The flow diagram of Actuator Controller's logic.....	47
6.1	The effect of coupling and decoupling on joint angle.....	49
6.2	Force vs Hydro muscle length result	50
6.3	The position control result on a Joint.....	51
6.4	The leg structure picture with both hip and knee actuated.....	52
6.5	Proposed new structure of Force Multiplier	54

List of Tables

5.1 Microcontroller selection research.....34

Chapter 1

Introduction

The purpose of this project is to provide a proof of concept of how a synthetic hydro-muscle can be used for actuation that mimics the actuation of joints in living organisms .For this purpose we introduce a leg structure with a hydro-muscle driven hip and a knee joint. In this section we'll see how these joints works in human being and later we'll see how the structure of our leg draws its inspiration from human anatomy. We'll also see what other kinds of synthetic muscle options

are available and lastly, we'll discuss the hydro-muscles and see why it is a good idea to have a hydro-muscle actuated system.

1.1 Hip and Knee Anatomy

In general, in humans and other vertebrates, a single degree of freedom of movement on a joint is often produced by an antagonistic pair of muscles connected to 2 different limbs bones around the joint which they actuate. Take the simple example of an elbow joint:

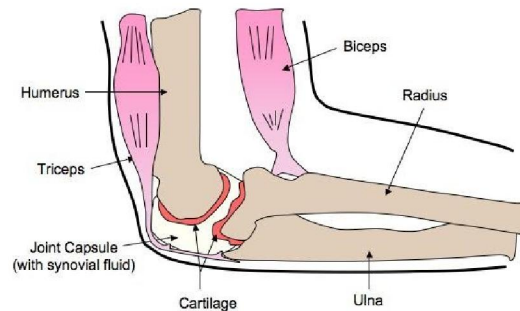


Fig. 1.1: The structure of a simple joint (elbow)

The figure above² shows that on elbow there are two main muscles that are responsible for flexion and extension movement at elbow. The contraction of triceps exerts a pulling force that becomes responsible extension movement. The contraction of biceps is similarly responsible for flexion movement. One of the major benefit of two muscle sets working against each other is

² "Human Elbow Anatomy – Humananatomychart.info." 2016. 22 Apr. 2016 <<http://anatomydiagram.info/human-elbow-anatomy/>>

that we can vary the stiffness of the joints as we pleased by modulating the force exerted by both the sets of the muscles.

We now see the anatomy of knee and the hip joint and see which muscles are responsible for movement of the said joints in the anterior-posterior plane.

1.1.1 Knee Anatomy

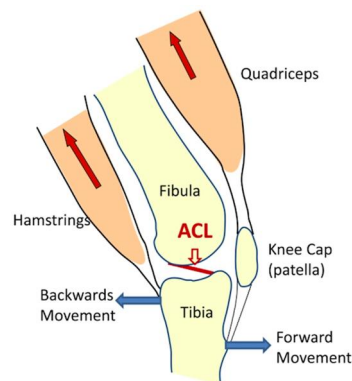


Fig. 1.2: The structure of knee joint

The figure above³ shows the basic structure of the human knee. The upper limb bone is called femur and the lower bone is called tibia⁴. The front muscles that are called Quadriceps. They extend over the knee cap and connect to tibia. The knee cap acts as a pulley. The forward motion produced by the quadriceps is called extension⁵. The backward motion called the inflexion⁴ and is produced by the motion of the hamstring muscles at the back. There are a couple of other

³ "Research Review: Weak in the knees? | Precision Nutrition." 2009. 22 Apr. 2016

<<http://www.precisionnutrition.com/quad-hamstring-ratio>>

⁴ "Knee (Human Anatomy): Images, Function ... - WebMD." 2010. 22 Apr. 2016 <<http://www.webmd.com/pain-management/knee-pain/picture-of-the-knee>>

⁵ "Muscles that Cause Movement at the Knee Joint - Boundless." 2016. 22 Apr. 2016

<<https://www.boundless.com/physiology/textbooks/boundless-anatomy-and-physiology-textbook/muscular-system-10/muscles-of-the-lower-limb-107/muscles-that-cause-movement-at-the-knee-joint-579-9335/>>

muscle groups as well that wrap around the back of femur and attach on the side of tibia and are responsible for rotational movement.⁴

1.1.2 Hip Joint

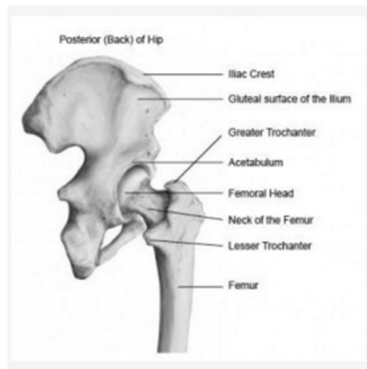


Fig. 1.3: The bone structure of hip joint (Posterior side)

The figure above⁶ shows the bone structure of the hip joint. As we can see the joint kind of looks like a ball and a socket joint⁷. There are many muscles on the hip joint and the combination of these can produce many complex movements.

⁶ "Hip Anatomy, Function and Common Problems." 2010. 22 Apr. 2016 <<http://www.healthpages.org/anatomy-function/hip-structure-function-common-problems/>>

⁷ "Ball & Socket Joint - Anatomy Pictures and Information." 2007. 22 Apr. 2016 <http://www.innerbody.com/image_skel07/skel34.html>

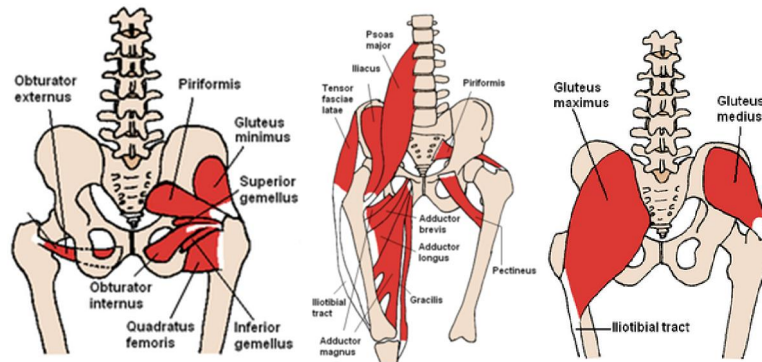


Fig. 1.4: The muscles on hip joint

The pictures above⁸ shows muscles of the hip joints .As you can see that there are a large number of muscles there. The muscles of the hip are often divided into 4 groups by anatomists ⁷ :

1. Gluteal group,
2. The lateral rotator group,
3. The adductor group, and
4. The iliopsoas group.

The movement that we are interested in the hip are extension and flexion. These are shown in the figure below⁹ shows these movements:



Fig. 1.5: The movements of hip joint

⁸ "Muscles of the hip - Wikipedia, the free encyclopedia." 2011. 22 Apr. 2016
<https://en.wikipedia.org/wiki/Muscles_of_the_hip>

⁹ "Hip rotation - Shotokan Karate Union." 2015. 22 Apr. 2016 <<http://www.shotokankarateunion.com/T122.html>>

The muscles that are responsible for flexion are Psoas muscles and iliacus⁷. The muscles responsible for the extension⁷ are Gluteus maximus, and Hamstrings.

1.2 Synthetic Muscles

The idea of using synthetic, muscles to actuate and produce natural like movement has been around for a couple of decades now. Over the years there has been many versions of the synthetic muscles. Some of them are discussed below.

One of the earliest synthetic muscles developed were called Pneumatic Rubber Artificial Muscle (PRAM)^{10 11}. They were developed in Okayama University. These muscles consisted of a silicon based pneumatic actuator and a polyester bellow system. The inner and outer bellow's surfaces limited radial expansion of silicon tube. They were reinforced with tape on one side such that actuator naturally bends in one direction when it was pressurized. These muscles were

¹⁰ Noritsugu, Toshiro et al. "Wearable power assist device for hand grasping using pneumatic artificial rubber muscle." *SICE-ANNUAL CONFERENCE*- 4 Aug. 2004: 420.

¹¹ "Hydro Muscle –A Novel Soft Fluidic Actuator." 2016. 22 Apr. 2016 ,Sridar, Saivimal (WPI), Majeika, Corey (WPI), Bowers, Matthew (Worcester Polytechnic Institute), Schaffer, Phillip (Worcester Polytechnic Intitute), Ueda, Seiichiro (WPI), Barth, Andrew (WPI), Sorrells, Jon (WPI), Wu, Jon (WPI), Hunt, Thane (WPI), Popovic, Marko (Worcester Polytechnic Institute)<<https://ras.papercept.net/conferences/scripts/abstract.pl?ConfID=119&Number=2771>>

pressurized using air. PRAM has been used for both the upper⁹ and lower¹² extremities assistive mechanisms.

Other synthetic muscles examples include PneuFlex¹³, developed in TU Berlin, and Fiber Reinforced Actuator (FRA)¹⁴, originally developed at the Harvard Wyss Institute. Both PneuFlex and FRA utilized air, were silicon based, and have typically helix-like fibers either bonded to the surface (PneuFlex) or embedded deeper within silicon (FRA).

Perhaps the most popular artificial muscle in use has been McKibben artificial muscles¹⁵¹⁶¹⁷¹⁸ in use since the 1950s. Like the biological muscles, the McKibben muscle soften radially while elongating axially and they bulge and stiffen radially while contracting axially. The figure below¹⁶ shows the state of the McKibben muscles as they are expanded.



Fig. 1.6: The state of Mckibben muscle as it is pressurized

¹² T.Noritsugu, et al., "Wearable Power Assist Device for Standing up Motion Using Pneumatic Rubber Artificial Muscles," *Journal of Robotics and Mechatronics*, Vol.19, No.6, pp.6 19-628, 2007.

¹³ Deimel, Raphael, and Oliver Brock. "A compliant hand based on a novel pneumatic actuator." *Robotics and Automation (ICRA), 2013 IEEE International Conference on* 6 May. 2013: 2047-2053.

¹⁴ Galloway, Kevin C et al. "Mechanically programmable bend radius for fiber-reinforced soft actuators." *Advanced Robotics (ICAR), 2013 16th International Conference on* 25 Nov. 2013: 1-6.

¹⁵ Chou, Ching-Ping, and Blake Hannaford. "Measurement and modeling of McKibben pneumatic artificial muscles." *Robotics and Automation, IEEE Transactions on* 12.1 (1996): 90-102.

¹⁶ Chou, Ching-Ping, and Blake Hannaford. "Static and dynamic characteristics of McKibben pneumatic artificial muscles." *Robotics and Automation, 1994. Proceedings., 1994 IEEE International Conference on* 8 May. 1994: 281-286.

¹⁷ Klute, Glenn K, Joseph M Czerniecki, and Blake Hannaford. "McKibben artificial muscles: pneumatic actuators with biomechanical intelligence." *Advanced Intelligent Mechatronics, 1999. Proceedings. 1999 IEEE/ASME International Conference on* 1999: 221-226.

¹⁸ M. B. Popovic, "Biomechanics and Robotics", book 364 pages, Copyright © 2014 Pan Stanford Publishing Pte. Ltd., Singapore, ISBN 978-981-4411-37-0 (Hardcover), 978-981-4411-38-7 (eBook). [www.panstanford.com]

1.3 Hydro Muscles

The Hydro-muscle is a new kind of muscle that was developed in the Worcester Polytechnic Institute (WPI) by the Popovic Labs¹⁹. At its core, the hydro muscle consists of an elastic latex tube which expands when pressurized by a fluid. The radial expansion of the hydro muscle is limited by an inelastic polyester sleeve, forcing the inner elastic tube to just expand linearly¹⁹. The figure below¹¹ shows the hydro muscle in its expanded and relaxed state.

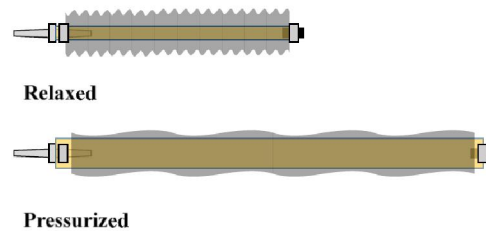


Fig. 1.7: The pressurized and relaxed state of a Hydro muscle

The muscle can be charged by any fluid but, we use water here to charge the muscle. When pressurized the muscle will store elastic potential energy which can quickly be turned into kinetic energy when the outlet is opened and the water is allowed to rush out the muscle¹¹.

This means that hydro muscle is essentially like a spring that can be charged with pressurized water. The graphs below²⁰ shows the simulated data of elongation (in m) and the pressure changes when the valve on hydro muscle end was (a) opened at once and (b) opened 1% per millisecond.

¹⁹ McCarthy, Gregory et al. "Hydraulically Actuated Muscle (HAM) Exo-Musculature." *Proc. Robot Makers Workshop, Robotics: Science Systems Conf* 2014.

²⁰ "Hydro-muscle actuated knee for biped Leg Chair" Mar 2016, Matthew P. Bowers, Seiichiro Ueda, Amair Zia, R. Matthew Rafferty, Varun V. Verlencar Saivimal Sridar, Ananth Jonnavittula, Chinmay V. Harmalkar, Lynn R. Koesterman, John D. Kelly, Thane R. Hunt, and Marko Popovic

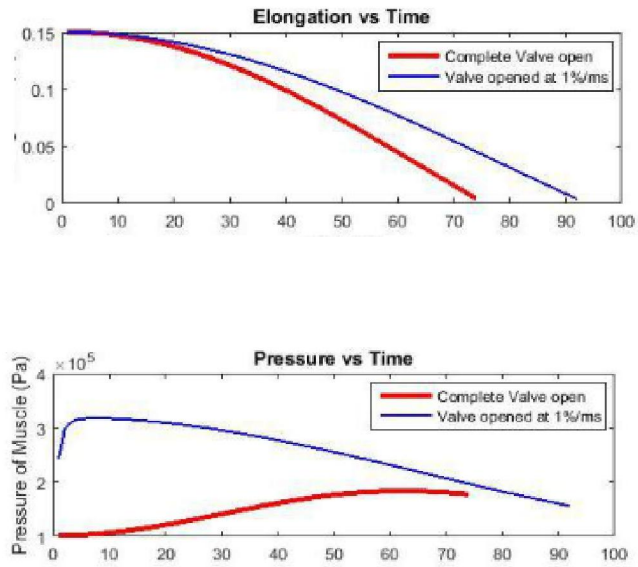


Fig. 1.8: The elongation and pressure state of Hydro muscle as its valve is opened at once and at a rate of 1% per millisecond.

Chapter 2

Design Requirements

The System on the whole is meant for the use of disabled, who have little or no functionality in their legs. We will discuss the requirements of the bigger project of leg chairs on the whole and then use those requirements for deriving the requirements of smaller system of 2 joints presented here.

2.1 Need Analysis of the Whole System

The list of the need that the intended user requires from the whole System are as follow:

1. The system needs to bear the weight of a normal sized human
2. The system needs navigate around obstacles.
3. The system needs carry a person on stairs.
4. The system needs to be stable while walking and standing.
5. The system needs to mimic human gait as closely as possible to give a feel of natural gait.

6. The System needs to be reactive.
7. The System needs to be safe.

2.1.1 Critical Needs

The critical needs of the system are:

1. Stability ,
2. Bearing weight.
3. Safety

Now that we have the needs for the system on whole we will translate these needs to the system requirements.

2.2 System Requirements

The following are the derived system requirements for the whole system.

1. The system should be able to bear the weight up to 150 kg.
2. The System should be able to hold any stable pose while maintaining the above described weight.
3. The range of motion should not be less than 40 degrees for the joints in general, if not mentioned explicitly.
4. The System should be able to balance itself, with the weight on one leg.
5. The System should be able to conform to natural limb movement speeds.
6. The System should have a reaction time no more than 200 ms.

7. The System should be able to shut-down while maintaining pose in case of a serious failure.
8. The System should not be a fire hazard.
9. The System should be able to work in the temperature range of 0-60 °C.

2.3 Single Joint Requirement

Using the System Requirements some the general requirements for the single joint can be derived as follow:

1. The joint should be able to provide at least a torque of 190 Nm at any angle. The highest torque experienced by a joint during climbing the stairs is 1.26 Nm/kg²¹ on the hip muscle. For a 150 kg person this translates to a torque of about 189 Nm.
2. To provide the max. Force and/or Torque at any pose of the joint their needs to be a mechanism that can couple and decouple muscles from the joint.
3. Since the force provided by the single muscle is not enough for the required torque so we should have either multiple muscles or a force magnification component or a combination of both.²⁰
4. The dimensions of the pulley at the joint should be suitable enough to get the required angle. With our current pulley diameter of 7 inches and displacement of 10 cm after Force multiplier, the range of angle that we get is : $S = r\theta \Rightarrow \theta = S/r \approx 64 \text{ degrees}$

²¹ KIM, SM. "Study of Knee and Hip Joints' Moment Estimation ... - IAENG." 2009. <http://www.iaeng.org/publication/WCECS2009/WCECS2009_pp785-788.pdf>

5. There should be a mechanism to control the flow of water escaping from muscle so that speed of the muscle contraction could be controlled.

2.4 Electronics and Control Requirements

The following are the requirements that could be deduced from the system requirements for the electronics:

1. The max. delay due to electronic components plus the controls cannot be more than 200 ms.
2. The electronic equipment should operate below its max. Limits so that they are transient to the electronic spikes.
3. The communication between different components should be reliable.
4. The electronics should be able to operate under the operating temperatures.
5. The actuators should be able to provide required torque/force while withstanding the high pressure where appropriate.
6. Since the System, on the whole, is being designed to react to complex real-life situations so it should have a powerful enough processing unit either on-board or off board to command and control the system.

2.5 Features

2.5.1 Required

The required features are as follow:

1. The control system loop should be at max 200ms.
2. More than 40° of motion range for the hip and knee joint.
3. Position Control.
4. Presence of Coupler/Decoupler mechanism
5. Presence of Force multiplier.

2.5.2 Desired/Optional

1. Control and actuation delay should be less than 200ms.
2. Wireless communication with the smaller onboard controllers so that the device could be light and its testing can be done at multiple places.
3. Some of the joints may have passive components like springs.
4. A part of gait to be passive.
5. On- structure rechargeable batteries.
6. Force and variable stiffness.

Chapter 3

System Overview

3.1 Hardware

3.1.1 Mechanical Components Arrangement

In general each joint has 2 pairs of muscles for movement. 1 pair for extension and the other for inflexion. These muscles start at one limb and connect to the other limb across the joint through a force multiplier and a coupler/decoupler. So each joint has the following 3 components that make it move. They are:

- **2 muscle pairs:** 1 pair for extension and the other one inflexion.
- **Force Multiplier:** Magnify the force generated by muscles by a factor of 2.
- **Coupler/Decoupler:** This component can couple and decouple the muscle movement to the joint movement.

The diagram below shows this simple arrangement:

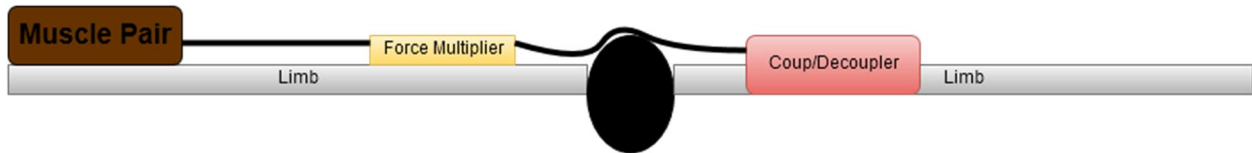


Fig. 3.1: The arrangement of components on the leg structure

Now we look at how the hydraulics are put together and what makes the muscle actuate.

3.1.2 Hydraulics

²²Each muscle pair at one of the end (a fixed end) is connected with 2 pairs of pipes. 1 pair for water to enter the muscle and the other to let the water leave the muscle. The other end of the muscle is sealed and free to extend. Each of the pair of pipes has a fine flow control butterfly valve, an on/off solenoid valve and a check valve in series. These states of these valves help in controlling flow and deciding which muscle to expand and which to contract. The diagram below shows the arrangement:

²² The hydraulics were designed and implemented by Matthew Bowers

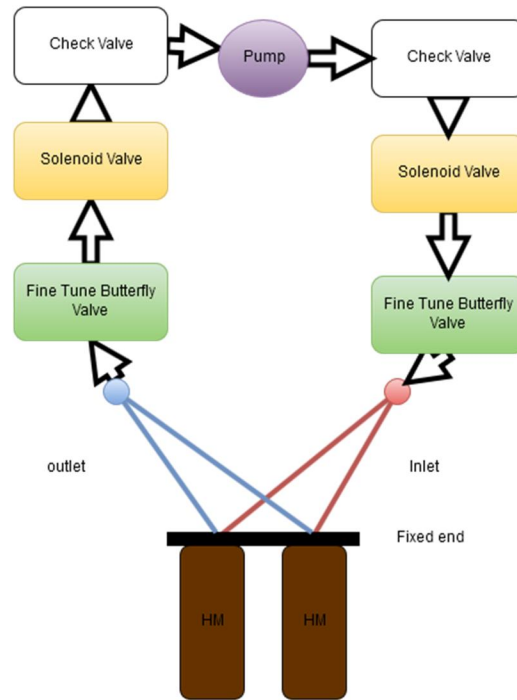


Fig. 3.2: The conceptual diagram of the hydraulics of the system.

3.2 Electronics

²³The diagram below shows an oversimplified view of the electronics in the System:

²³ The sensor section of the project was done by Seiichiro Uda.

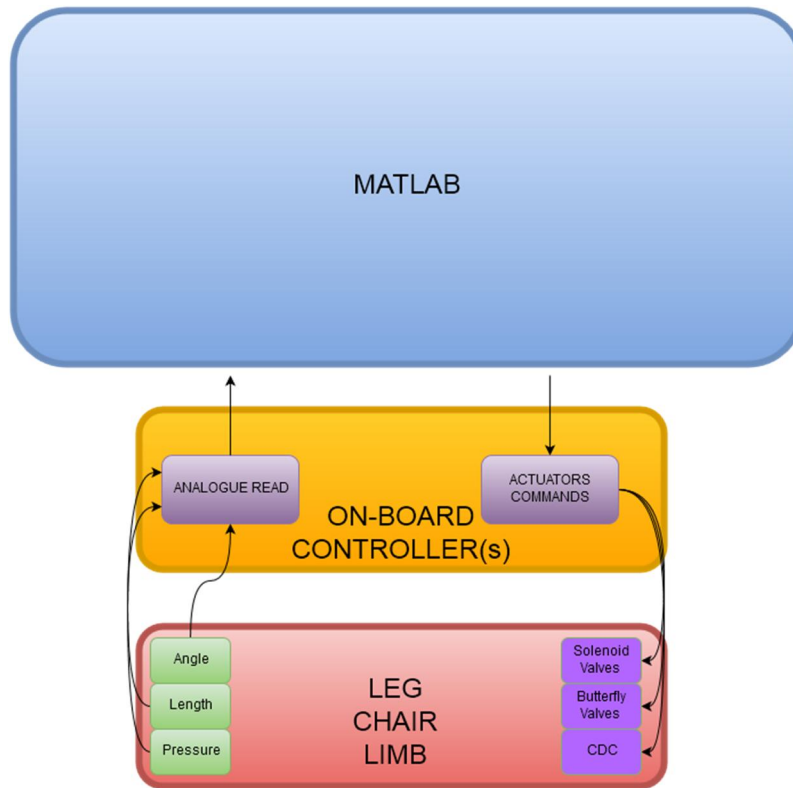


Fig. 3.3: The simplified view of electronics of the System.

3.2.1 Sensors

Currently, there are 3 types of sensors in the system:

1. **Angle Sensors:** For measuring the angle of the joints we have used a simple 10 kilo-ohm potentiometer.
2. **Length Sensor:** For measuring the length of the muscle we have used an IR range sensor.
3. **Pressure Sensor:** These are measured by the load cells which are embedded inside the muscle so that we can monitor their pressure.

3.2.2 Actuators

The 3 types of the actuators that are present in the system are:

- 1. Solenoid Valves:** We have a combination of latching and non-latching solenoid valves on the inlet and outlets of muscles. These control whether or not to allow water to flow through the pipe on which they are installed. The differences between the latching and the non-latching valve is that the non-latching valve needs constant power to remain on whereas the latching one requires either a positive or a negative pulse to change the state and it maintains its state if the power is turned off.
- 2. Butterfly Valves:** We have mounted a small, fast and high torque servo on top of the butterfly valves to fine control the flow of water flowing through the pipe. The motor is able to provide $60^\circ / 0.05s$ at 8.4V.
- 3. CDC:** The design of this coupler decoupler is inspired by the design of the ratchet. We use a servo to engage or disengage this mechanism.

Chapter 4

Hardware Design and Specification

4.1 Basic Skeleton

²⁴The basic skeleton was built so that it cannot only bear the forces that are exerted by the muscles and/or environment but also support the various components and the combined weight. This meant that the frame needs to be light yet durable. This is why the limb frame was built by 3 parallel, vertical 80/20 aluminum bars connected by horizontal bars. As a side benefit this provided us convenient v-slots for easy placements of other mechanical components. The length of each limb is 0.61m

²⁴ For the frame I decided the size parameters but Matthew Rafferty drew and implemented the CAD drawing.

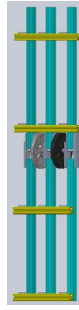


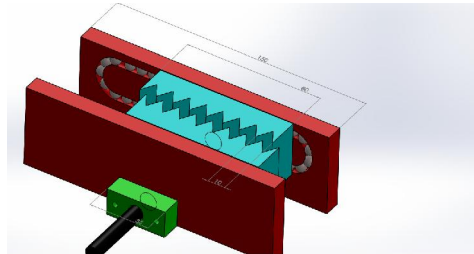
Fig. 4.1: The basic skeleton of the leg structure

4.2 Coupler/Decoupler Design Iterations

The Coupler/Decoupler connection with the muscles is perhaps the most novel aspect of the whole project. We went through a couple of design iterations of this mechanical component till we arrived at the current one. Here I represent a few of the early versions of the Coupler/Decoupler that I designed (with another project member²⁵).

4.2.1 Version 1

The figure below shows the crude diagram of our very first iteration of Coupler/Decoupler.



²⁵ Varun Vishnudas

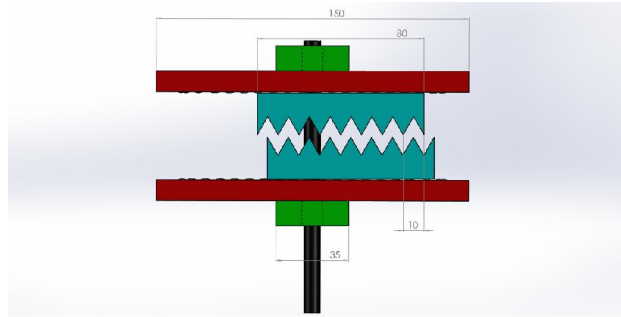


Fig. 4.2: The first design iteration of Coupler/Decoupler

The arrangement was supposed to fit on the frame with both teeth racks side by side. The racks were placed on a plate with a ball bearing plate that helped them slide with ease. The cable from the force multiplier was to be connected to one the rack pieces. The racks were supposed to come into contact with each other on the basis of the action of the lead screw.

The advantage of this iteration was that it was simple. The problems with it were that installing the rack onto the bearing plate in such a way that it could also slide was difficult. Also the lead screw method was slow for our purposes.

4.2.2 Version 2

The diagram below shows the 2nd iteration of the Coupler/Decoupler which was based on the improvements of the first design.

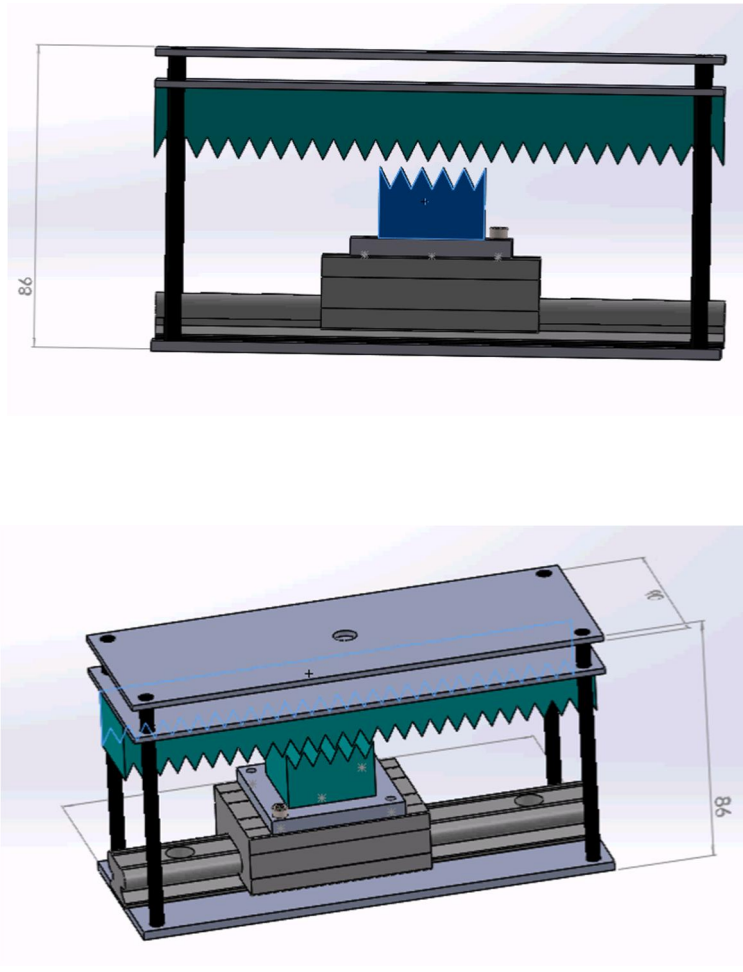


Fig. 4.3: The second design iteration of Coupler/Decoupler

This design also contains 2 rack pieces. One of them is mounted onto the linear guide and hence is free to move horizontally. This rack piece will be connected to the cable that is coming from the force multiplier. The other piece of the rack is free to move vertically. The vertical slides guide its movement. This piece was supposed to move up and down by action of either a servo connected rack or a solenoid.

The advantage of this design is that it is much smoother and easier to build than the first one. Also it has a nice casing. The problems with this design is that the actuator on the top will be constantly bearing the weight of a heavy rack piece and hence consuming large amounts of

energy. Also it is not able to fit between the frame and the height of the design is a bit too much for our liking.

4.2.3 Version 3

This is almost the same as the 2nd version with most of the design components being the same.

The only difference is that this one uses 80/20 as the slide with the help of the rollers.

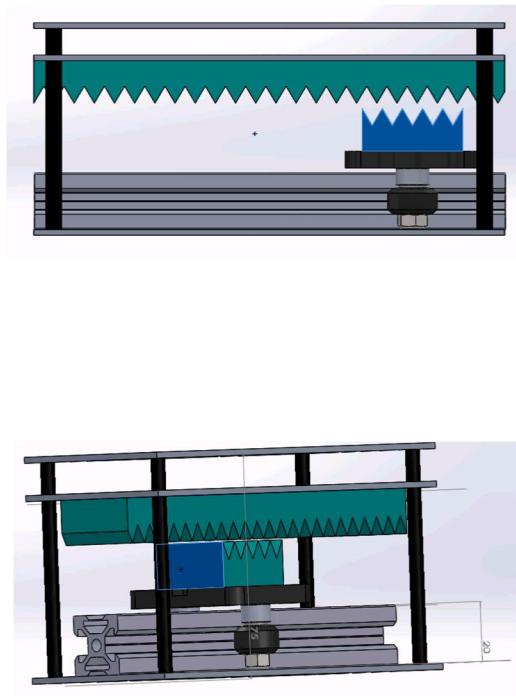


Fig. 4.4: The third design iteration of Coupler/Decoupler

This one has the same advantages and disadvantages as the 2nd version.

4.2.4 Version 4

In version 4 we again replaced the guide to make the Coupler/Decoupler even thinner.

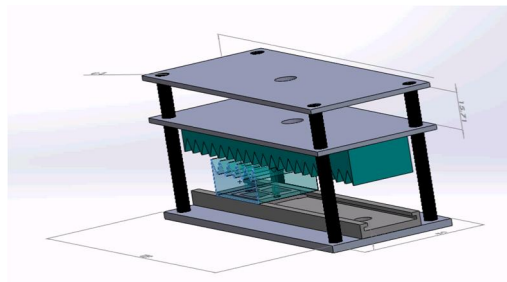
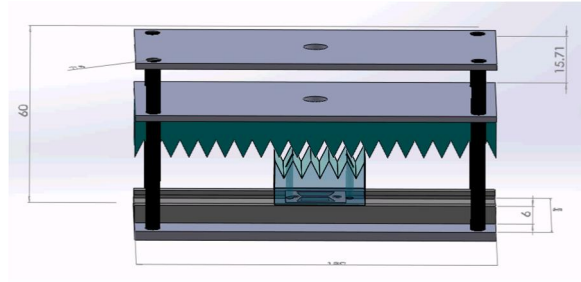


Fig. 4.5: The fourth design iteration of Coupler/Decoupler

The advantage of this design is that it is thinner. The disadvantage is that T-slot that is used as a guide here is less smooth than the other guides used here. Other than that this version has same benefits and failings as the previous 2 versions.

4.2.5 Version 5

This is the newer design for the Coupler/Decoupler. Here instead of a vertically actuated rack falling on another piece of rack, here we have a rack rotating that can slide into the teeth of the

horizontally fixed rack. As shown in the figure, the teeth on both the rack pieces of the rack have a hexagonal cross-section so that they can slide into each other with ease.

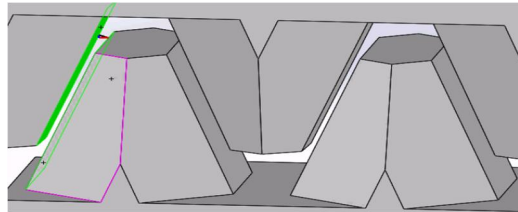
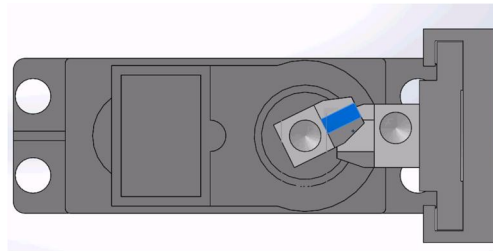
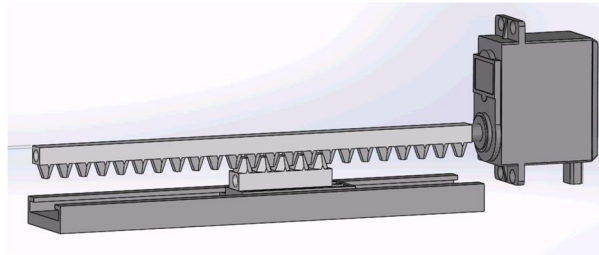


Fig. 4.6: The fifth design iteration of Coupler/Decoupler

The advantage of this design is that rotation controlled by the servo is much faster and compact. The disadvantages are that the servo will always experience a moment because of the weight of the rack. Also the shape of the teeth will require precise and specialized machinery.

4.2.6 The final version

The version that we end up using is designed by Prof. Popovic with inspiration from the ratchet mechanism

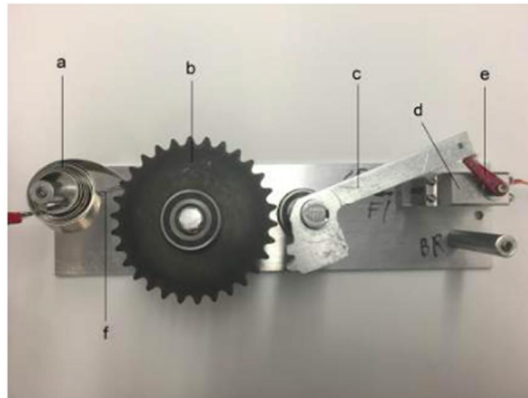


Fig. 4.7: The final design iteration of Coupler/Decoupler. Here (a): linear force tension spring, (b): Steel gear, (c): Servo controlled lever, (d): Servo, (e): Connector horn between servo and lever

4.3 Force Multiplier Designing

The first design iteration of the force multiplier that we designed is shown in the figure below.

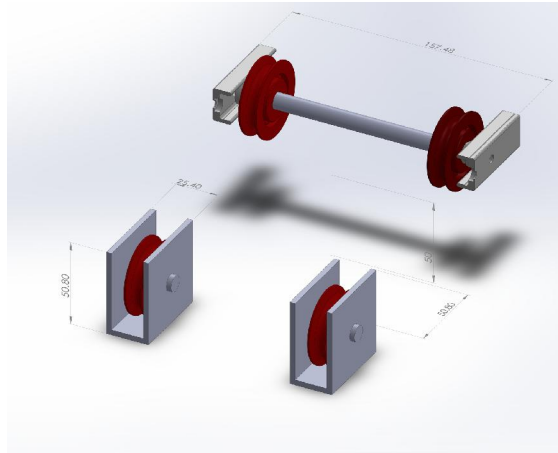


Fig. 4.8: The first force multiplier design

The design consist of a fixed pair of pulleys and a movable pair of pulleys. The cable is then wound around the pulleys to provide the force magnification. The advantage of this design was that it provided very smooth motion. The disadvantage is that because of non-linear and unequal expansion of the muscles and the shaft inside the linear guides will come out often.

Because of this problem we replaced the c-channel guides with first the horizontal shafts. That idea wasn't so successful either because of unequal forces from the muscles in the muscle pair.

We then replaced that by free hanging pulleys.

This is shown below:



Fig 4.9: The final force multiplier design employing free hanging pulleys

4.4 Placement of Components on Frame

The overall placement of components on the whole frame is shown by the figure below

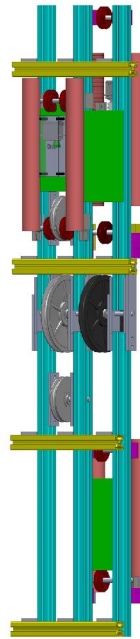
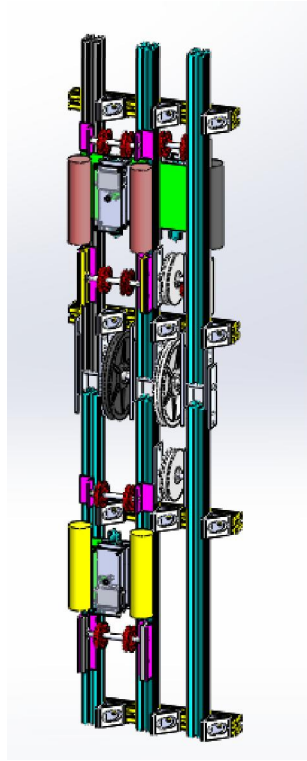


Fig. 4.10: The placement of all components on the leg structure.

4.5 Manufacturing Test Frame

We manufactured the test frame using the beams of wood. It is basically a cuboidal structure on which we supported the leg structure with the help of clamps (so that it is removable

whenever we need it to be). The first version, shown below was just for the support of the knee joint:

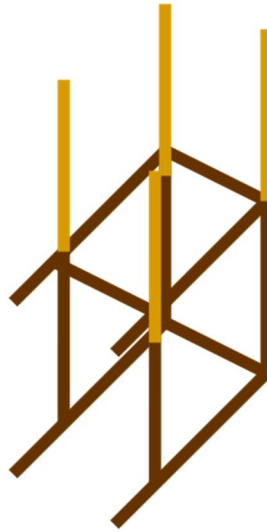


Fig. 4.11: The test structure for testing of just knee joint

For supporting the hip joint we changed the design little bit by building another 45° inclined structure on the top of the above structure. The hip was supported at an inclined angle so that we can reduce the overall height of the structure, while not bending the hydraulics horribly.

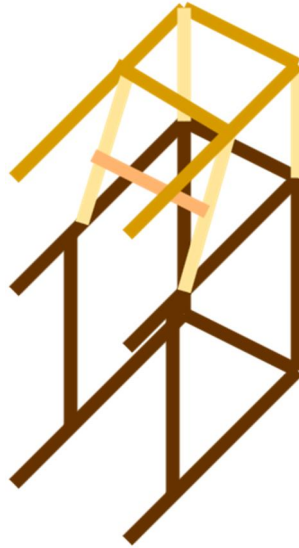


Fig. 4.12: The test structure for the knee and the hip joint

Chapter 5

Software Design and Specification

5.1 Overview of Architecture

As far as the electronics and software is concerned we had 3 main processing units working in conjunction with each other to make the leg actuate.

1. Sensor controller to input the sensor data
2. MATLAB, running on a PC that was used to process the information and take the decisions.
3. Actuator Controller that was used to actuate different pieces of hardware.

This diagram below (same diagram from chapter shows the communication flow and the components present:

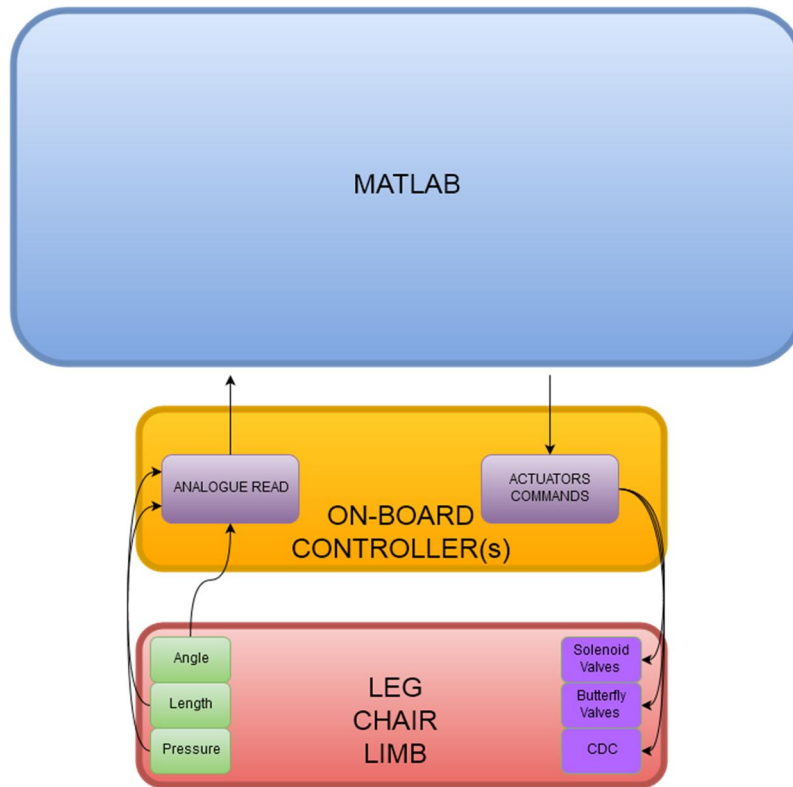


Fig. 5.1: The conceptual view of electronics

5.2 Choosing a Microcontroller Research

In the beginning, when the project was started we wanted a micro-controller that was cheap and will also have the wireless (Wi-Fi) capabilities so that our main processing unit does not have to be near the leg structure. This was because the system works on water and any leaks could harm the main processing unit. Also it would provide for ease of testing.

The table below shows the factors on which we judged some of the micro-controller candidates.

The number in violet denotes the relative priority of each of these factors (out of 5). The number in red denotes the score the board gets in each category (out of 5). In the end we multiply each of the score with its priority number and add to get a total score. The board with the largest total score is the best candidate.

	Costs	RTOS? / OS	WiFi	Speed	Mem ory	Ease of programming	Debugging	Comments	Sc ore
	5	2	4	1	1	4	5		
Intel Edis on	\$75 With break out kit	Linux	Yes	500MHz	1GB	Libraries for IoT and IDEs	Not real time	Firmware may limit kind of sensors we can use. I'm guessing that programming wouldn't be hard	72
	4	3	5	5	5	3	2		
Ardu ino Yun	\$71	Bare- metal	Yes	16MHz	(32- 4)KB	IDE	Real time is a possibility	Easiest programming, but peripherals we use we have to initialize them ourselves	93
	4	4	5	5	5	5	3		

Kinetic Board with tower	Very expensive Good board : \$40-~\$200 Tower: \$39 ADC module: \$119 Wifi: ~\$70	QMX	With module (built in inexpensive board)	Variable (depend on board)	Variable and expandable	Somewhat difficult (considering only I have experience with embedded systems programming)	Yes, real time(comes with a USB Jtag segger)	Highly customizable. Expensive than rest. If anytime in future we decide to make marketable products I'll recommend using it but not for now.	66
Beaglebone	\$89 3	Linux Angstrom. But it	With a separate	700MHz	256MB	PRU support is new so if we get stuck on that , then	Seems that JTAG is configurable with CCS	Seems like a good idea on first look but if you study it deeply then you'll realize	65

		also got PRU's 4	modul e 2	5	5	we'll remain stuck, or python or linux if real time is not required 3	but material and support on it is pretty much non- existent 4	that support/documentat ion of certain essential peripherals is not available	
RB- pi	\$35 5	Linux distro raspbi an , or bare metal 3	Separ ate modul e 2	900MHz , but RAM works at 450 MHz 5	1GB 5	Python for linux if real time is not required 3	NO 2	I consider this because it seems like we can import MATLAB code here... I've seen it done for image processing applications. Another problem here is that there are no built-in ADC's (as far as I know)	71

Table 5.1: Micro-controller selection research

As you can see that Arduino Yun gets the largest score so we decided to go with it.

Later we found out that there were certain problems with the Yun.

1. First of all the Yun didn't have enough pins to handle all the sensors and all the actuators.
2. 1 Yun cannot handle more than 2 servos.

3. Yun has 2 processors on-board. One an AVR core and the other one a 32-bits MIPS controller that handles a couple of high level modules that cannot be handled by the AVR core. Now to communicate the sensor data over wireless you first have to transfer data from the AVR core to the MIPS core. With the default code I found out that the time to transfer data between the cores was in the range 140-240 ms. After tweaking the linux files on the MIPS controller I was able reduce this time to 120-130 ms for all the packets. Previously, the device would open a new socket for every data transfer but I changed it so that I opened the socket only once in the beginning of transfer and close it only at the end. However, I also find out that if you keep on bombarding packets at the AVR core you can crash it.

First two problems indicated to us that if we are going to use the Yuns we might need multiple Yuns. I wanted to use Yuns at least for the sensor reading because their sensor read period was very less than the other controllers.

So for our first try we made a software architecture that used all the Yuns. And then we created another architecture that used a mega - based spider board that had a lot more pins and could handle multiple servos at once (48 according to the claim of the manufacturer)

5.3 Design Iterations for electronics

5.3.1 All YUNs

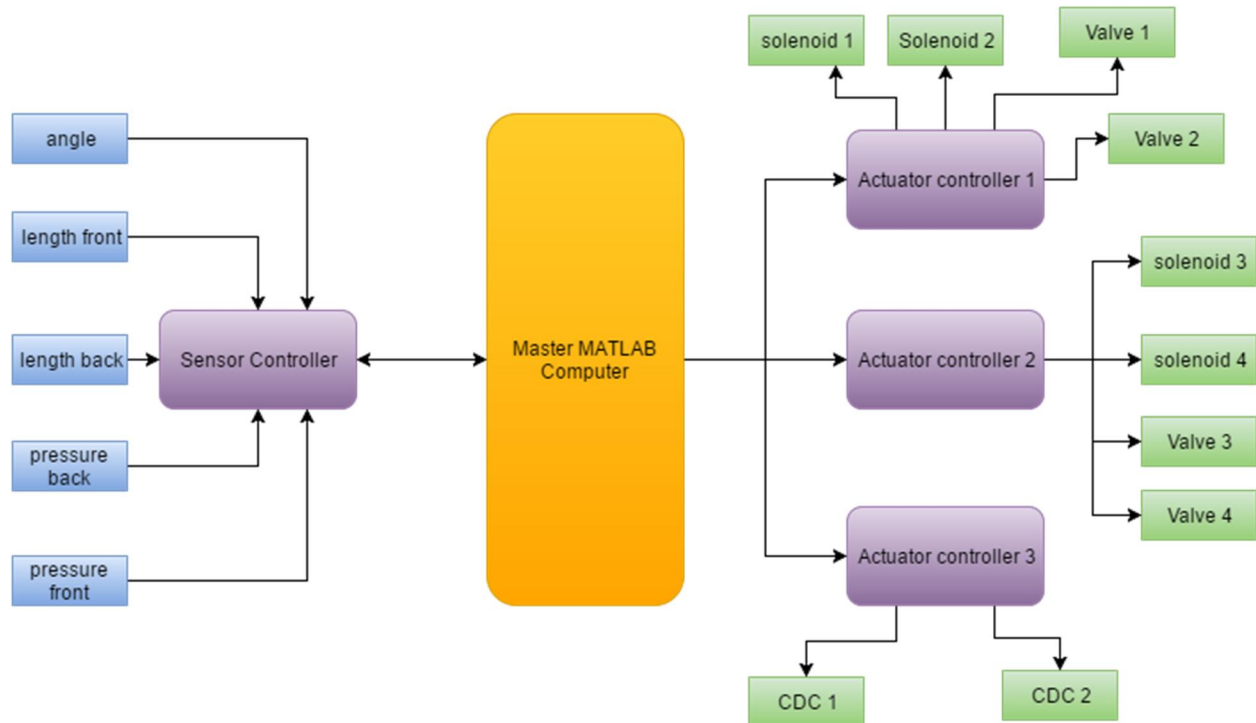


Fig. 5.2 : The detailed electronic architecture with all the Yuns

The diagram above shows the electronic architecture (when using just Yuns) for just one joint that shows how the different controllers, PC, sensors and actuators are connected together. See on the actuator side we need multiple controllers to interface all of the actuators.

5.3.1 YUNs + Dagu Spider Board

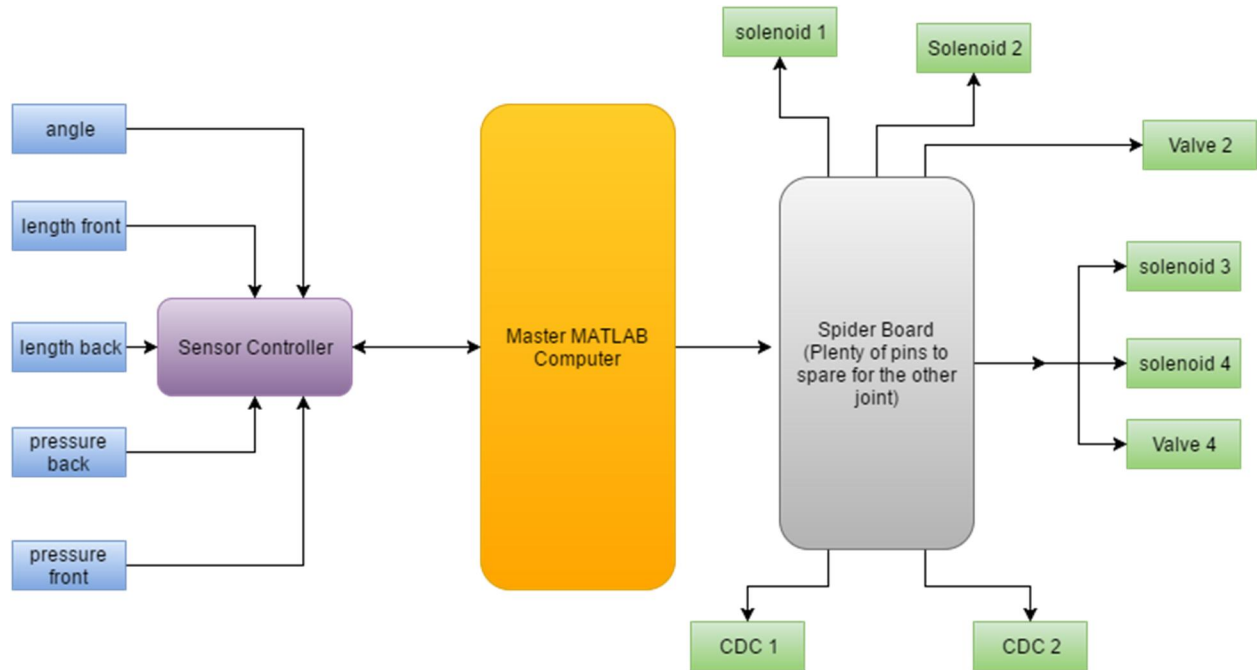


Fig. 5.3: The electronic architecture with Yuns and Spider Board

To simplify the actuator side of the control we replaced the multiple Yuns with a single Dagu spider board controller²⁶. The Spider board has enough pins and timer modules that it can handle the actuation of at least 3 joints at once.

²⁶ <https://www.sparkfun.com/products/11498>

5.4 MATLAB (Master) Software Architecture (Design Iterations)

5.4.1 Overview of Code

The MATLAB code is designed to be modular so that in future it would be fairly simple when it is desired to switch out a module with a newer version without effecting all the other modules.

The diagram below shows the modules and the program flow.

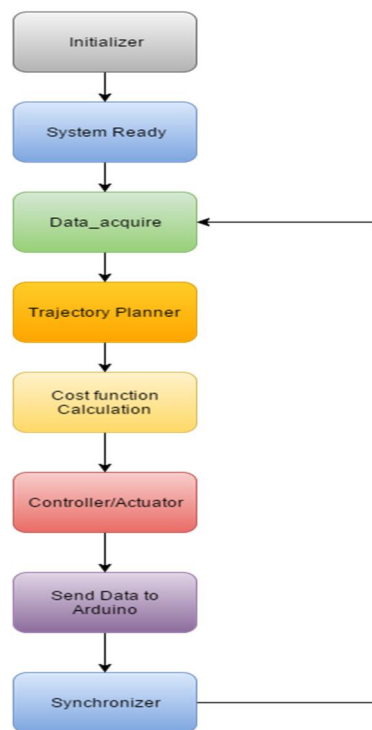


Fig. 5.4: The conceptual program flow between the MATLAB modules.

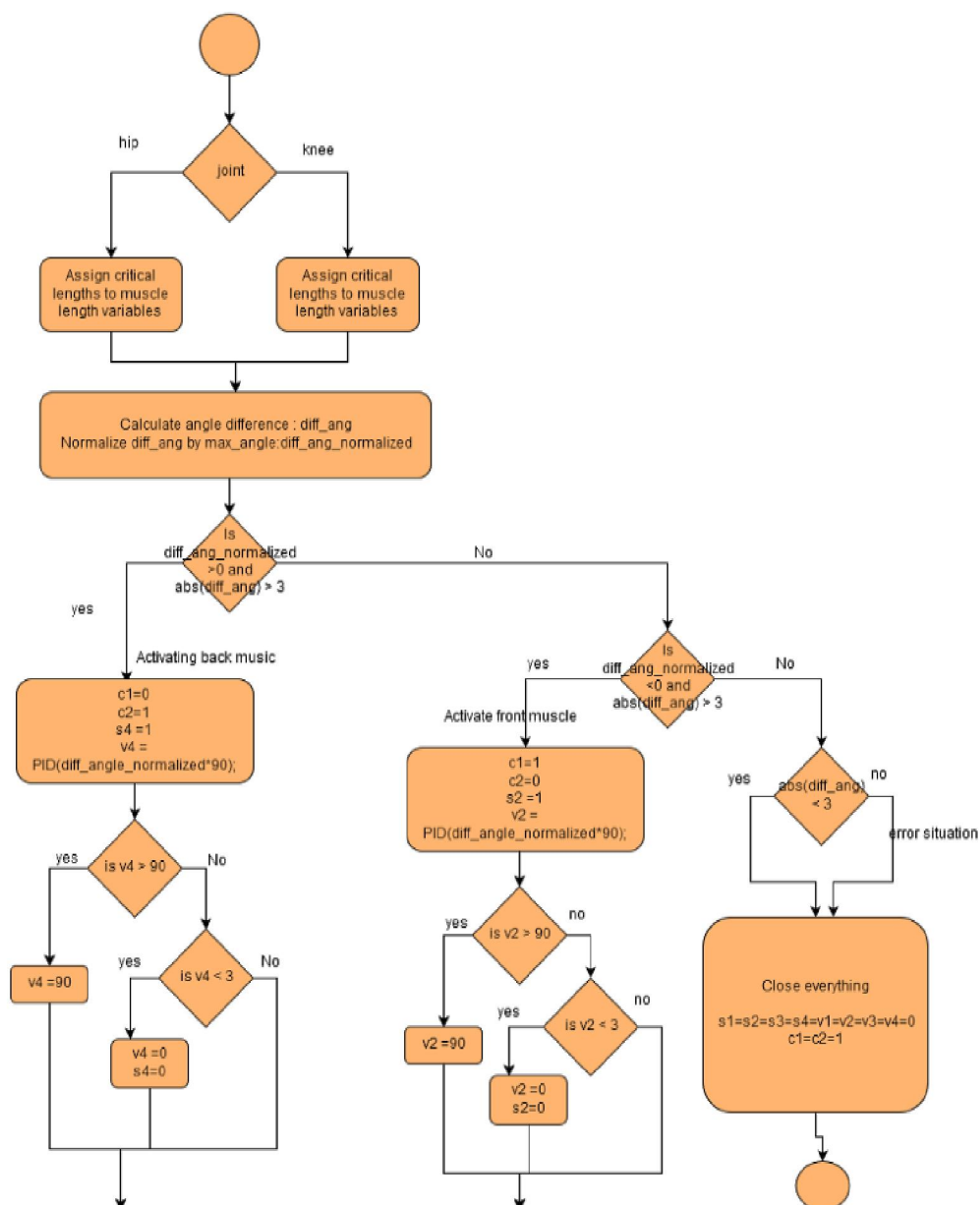
The explanation of the modules is as follow:

1. **Initializer:** This module initialize the system, create communication channels, create global variables and initialize other important system variables.

2. **System Ready:** This function will give command to the actuators so as to charge all the muscles. Initially, the logic I implemented was to charge all the muscles in parallel but due to pressure changes some muscles were preferred more than the others. It became clear that all the inlet valves should not be open all at once. So then I changed the logic to charge the muscles serially and then check all the lengths at the end. The function finishes only when all the lengths are not less than some critical length.
3. **Data Acquire:** This module gets the data from all the sensors. More on the communication protocol to come later.
4. **Trajectory Planner:** This module contains the formulas that help generating the next desired state of the leg structure.
5. **Cost function:** This module contain the optimization criteria formula according to what is required. Currently, it calculates a simple difference.
6. **Controller/ Actuator:** Based on the input from the sensors and cost function this function will generate the desired states of all the actuators. More on this in the section below.
7. **Send to Arduino:** This will send the actuator states generated by controller/Actuator module to the actuator controller present on the leg structure.
8. **Synchronizer:** This section will see how much time has passed and wait, if necessary, to generate a delay for the control loop.

5.4.2 Controller / Actuator

The flow diagram below shows the logic of the controller /Actuator portion of the MATLAB code.



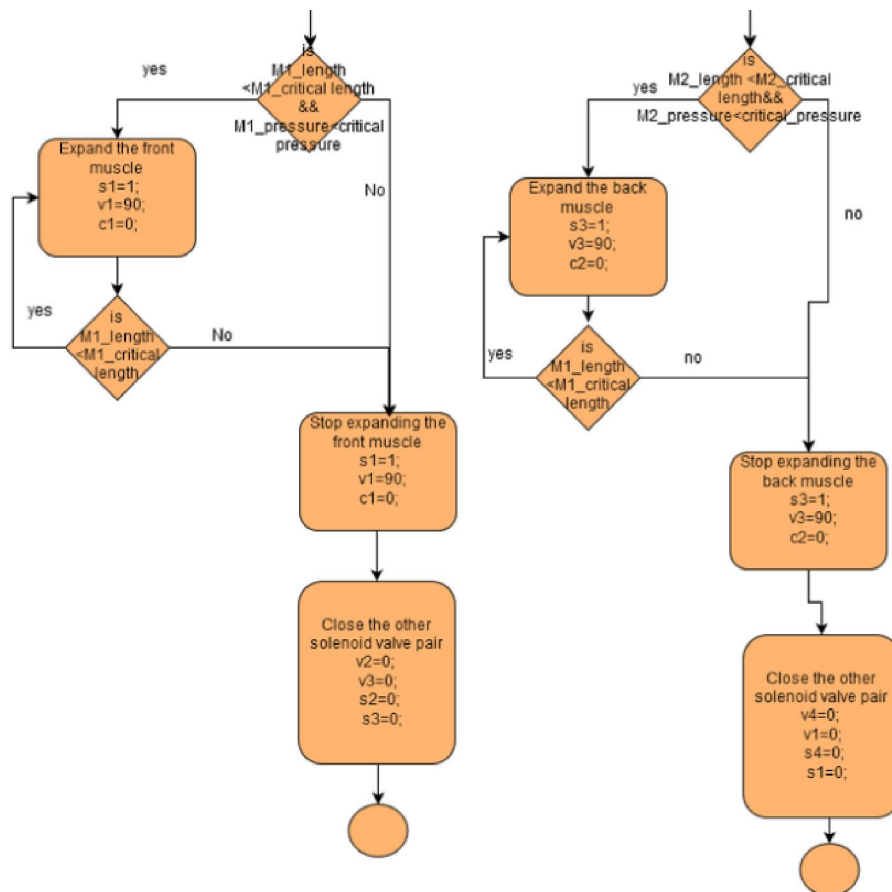


Fig. 5.5: The flow diagram of the actuator's logic

The controller refresh rate is 10 Hz .The controller defines the state of the outlet fine tune valve, controlling the flow in the outlet pipe and hence the rate at which the muscle contract and exert force.

5.4.3 Communication Protocol

For the communication the MATLAB acts as the Master to all the controllers on the leg structure. MATLAB and the controllers communicate serially at 115200 baud rate.

The initial communication protocol was such that on the sensor side MATLAB would give separate commands for reading each of the sensor. Similarly MATLAB would give commands for each of the actuators separately.

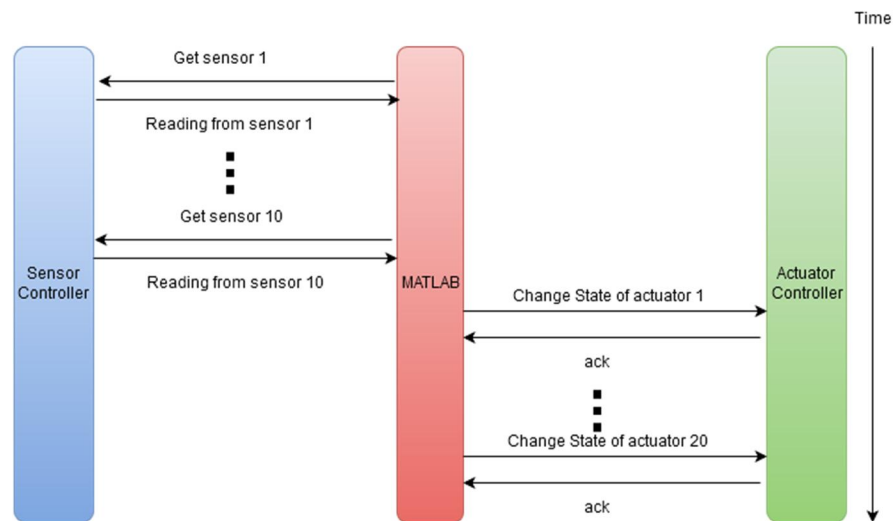


Fig. 5.6: The command based communication protocol

The problem with this protocol was that it was slow because you get data from individual commands. Similarly, the actuator communication was also slow for the same reason. This caused our control loop to take time longer than we desired because on MATLAB side it kept on timing out.

To overcome this problem, I developed a communication protocol that could read the sensor data in bulk and give commands to actuator controller in bulk as well. The diagram below shows this protocol:

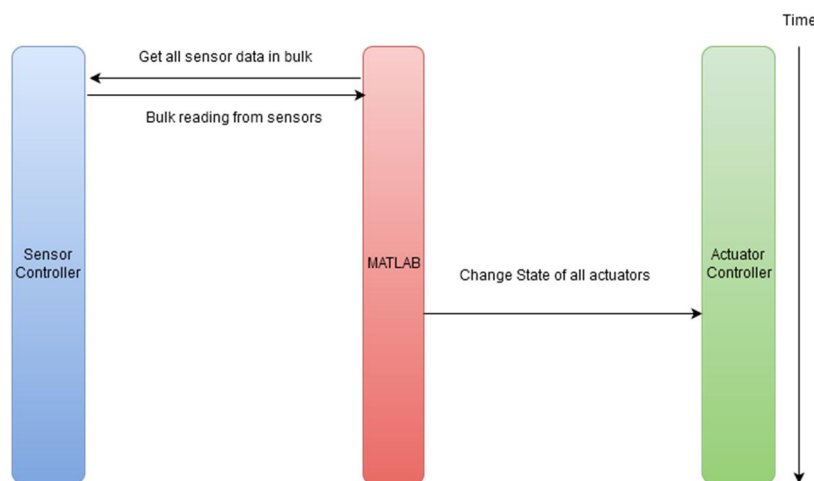


Fig. 5.7: Bulk Data transfer communication protocol

5.5 Controller's Logic

The controller's on the leg structure act in a slave capacity. They get their commands from MATLAB and does a series of simpler tasks.

5.5.1 Sensor Controller

The diagram below shows the program loop in simpler terms that the sensor controller follows. Currently, we are using Arduino Yuns for sensor controller. This is because analog read on Yun is much faster than the other options.

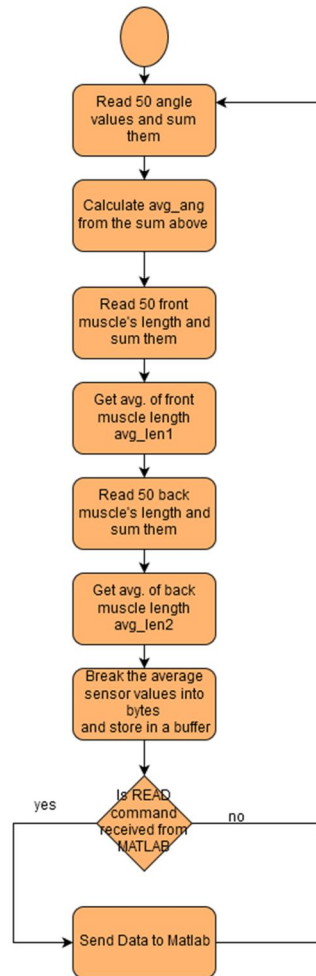


Fig. 5.8: The flow diagram of Sensor Controller's logic

5.5.2 Actuator Controller

The actuator controller gets the data from MATLAB in bulk. Then the controller reads that data and based on value of each byte it manipulates the state of the corresponding actuator.

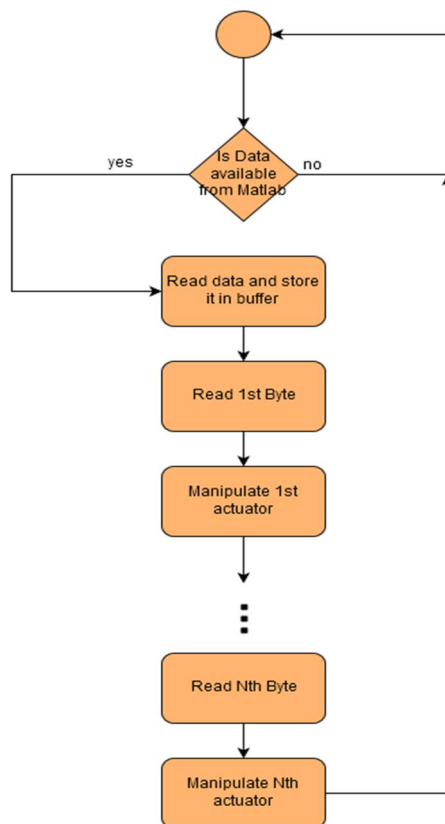


Fig. 5.9: The flow diagram of Actuator Controller's logic

Currently, we are using the Dagu Spider Board controller. According to manufacturer's claim it can support 48 servos at once.

Chapter 6

Conclusions and Future Works

6.1 Results

For the results we conducted a multiple of tests. This section will explain the results of the following experiments:

1. The effects of Coupler/Decoupler on the angle of the joint
2. Muscle elongation and the force generated
3. Position control of the joint.

6.1.1 Effect of Coupler/Decoupler on Joint Angle

For this experiment we expanded the muscle manually and got 2 readings of angles as it contracted: (1) with the coupler/decoupler disengaged, (2) with the coupler/decoupler disengaged.

The results of the experiment are shown below:

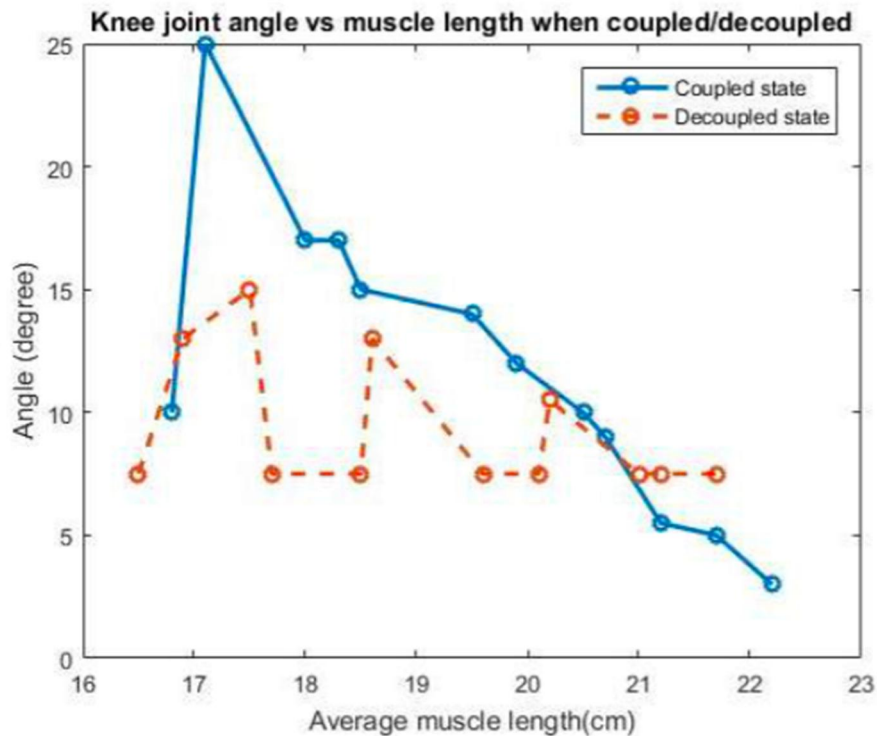


Fig. 6.1: The effect of coupling and decoupling on joint angle

As you can see that when the CDC is engaged the angle of the corresponding joint varies almost linearly. Now, when the CDC is disengaged the angle of the joint almost remains same. The fluctuations in both these graphs due to friction between cable and pulleys.

6.1.2 Force and Elongation result

In this experiment we elongated the muscles, coupled them and then took a reading of their length and the force they were exerting. We used a digital scale to measure the force reading statically. The result below shows the relationship of force and elongation for both the front (quadriceps) and back (hamstrings) muscles.

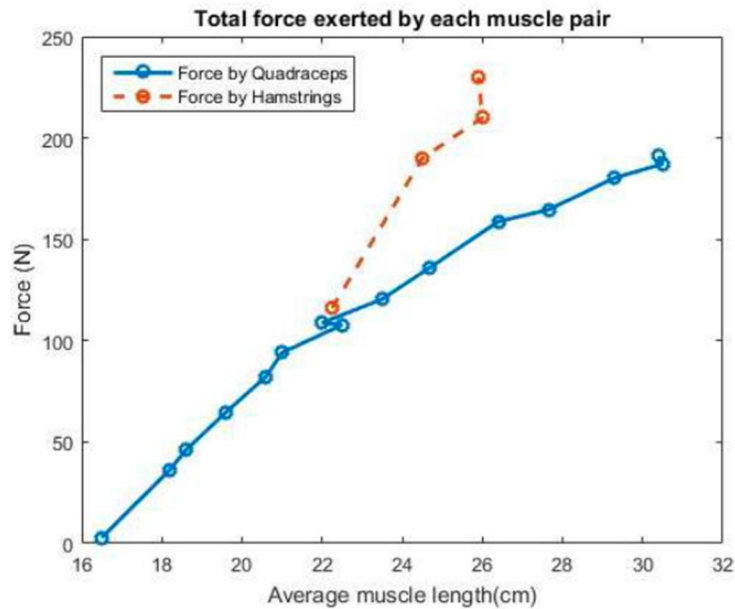


Fig. 6.2: Force vs Hydro muscle length result of the front (Quadriceps) and back (Hamstrings) muscles

The result shows a near linear relationship for both the front and the back muscles. For the front muscle the max. Force comes out to be $\sim 191.1\text{N}$. This corresponds to a torque of ($r = 0.61\text{m}$) 116.6 Nm . For the back muscle the max force was $\sim 230\text{N}$ which corresponds to a torque of 140.3 Nm .

6.1.3 Position Control

After hooking up the sensors and actuators and multiple hit and trial errors with different gains, we were able to control the position of both the hip and knee joint, individually and collectively as well.

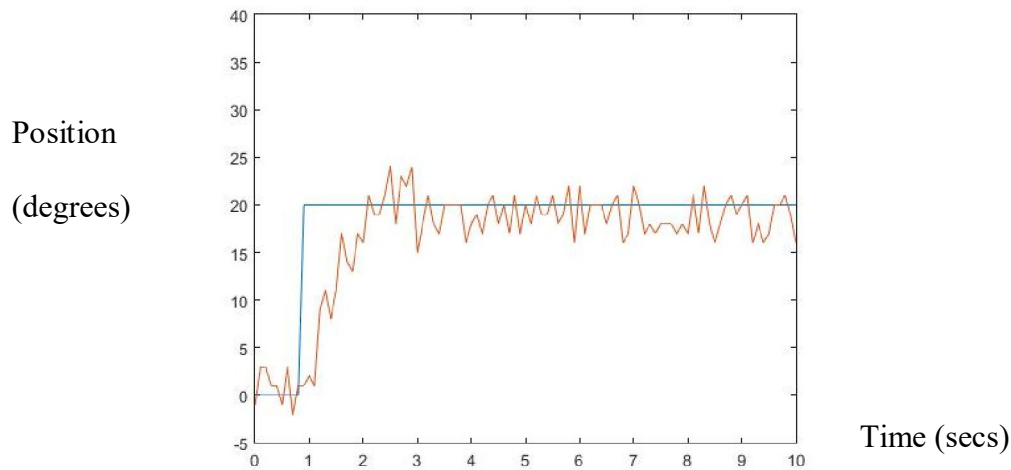


Fig. 6.3: The position control result on a Joint

The result above shows the result of the position sensor when a desired angle of 20° was given to the joint. The noise is due to the noisy signal from the position sensor (potentiometer) plus the vibrations in the whole system as a whole. The error in the position varies in the range $+3^{\circ}$ to -3° .

We have given this error range as acceptable in our code...

The figure below shows the situation in which we are controlling both the joints: knee and hip. Here the knee is at almost 50° and the hip is at almost 20° degrees

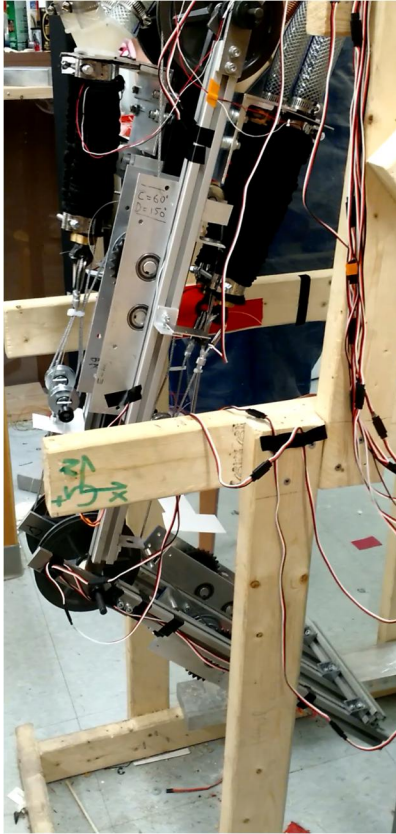


Fig. 6.4: The leg structure picture with both hip and knee actuated.

6.2 Suggested Future Works

I suggest the following to be done for the future works:

1. The data from the sensors is noisy. Although we took the average of the consecutive data samples, and put capacitors and resistors in parallel to sensors but these measures are still not extremely effective. In future, either a good filter should be designed for the existing sensors or sensors should be replaced by more expensive, reliable digital (to avoid spikes in signal from analogue sensors) sensors.
2. The lever on CDC should be of a harder metal because the steel gear eats into it. Also while testing we saw gear mismatch cases between gear and lever. We may see better performance finer teeth and more number of teeth on the lever to compensate for it.
3. The free hanging pulley design while works mechanically better but since it is free to move around, tilt and twist it is bad for the length sensor because you don't have a smooth motion. The pulleys twist and turn because of unequal and non-linear expansion of muscles in a single muscle pair. A fix for that could be to join the movable end of both the muscles with a plate and then a single cable goes from that plate to force multiplier. Since there is only one cable we'll need only one pulley which can be fixed in the middle of its shaft. Now because of one cable we can fix the force multipliers fixed inside a

linear guide . In this way we could get better readings from the IR sensors

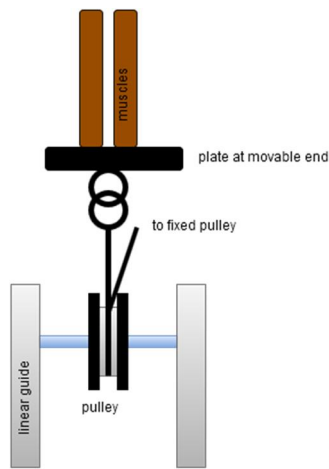


Fig.6.5 : Proposed new structure of Force Multiplier

4. Work on the making mathematical dynamic model of the whole leg and use it for better control.
5. Implementation of controls for walking and balancing.
6. Build a semi actuated ankle joint.
7. Build the other leg and interfacing both the legs together.
8. Investigate why some muscles expand faster than the other ones (knee muscles expand faster than hip muscles).

Appendix

A.1 The MATLAB main loop

```

%%*****
% Main Code file
% function calls:
% - [ang,length,pressure,flow]=Data_acquire()
%
%
%
%*****

%% Global Variables
% Global_Variables

function Main()
% Global_Variables

    i=0;

    Ready_System();
% Start_timer();

    Sensor_Data = zeros(1,5);
    Desired_traj = zeros(1,5);
    Cost_Outputs = zeros(1,5);
    Arduino_Outputs=[0,1,0,1,0,1,35,65,45,10];%zeros(1,10);%
    hip_flag=0; knee_flag=0;
% fid = fopen('G:\Git_repo\Lab_work\data_logger\logger.txt','at+');
% fprintf(fid,'%d %d %d %d %d %d',clock);
% fprintf('\n\r');

% Empty line handle
% IHandle = line(nan, nan); %# Generate a blank line and return the line handle
% IHandle2 = line(nan, nan);
% figure(1)
% axis([0 inf 0 20000]);

```

```

% Sensor_Data
% Desired_traj
% Cost_Outputs
tic;
while(1)
tic;
Sensor_Data = Data_acquire();
Desired_traj = Generate_desired_trajectory(i)
Cost_Outputs = Calculate_cost(Desired_traj,Sensor_Data); %

% if(hip_flag ==0)
[Arduino_Outputs_hip,flag] = Controller(Cost_Outputs(1:5),Sensor_Data(1:5),1); %
% hip_flag = flag;
% end

% if(knee_flag == 0)
[Arduino_Outputs_knee,flag] = Controller(Cost_Outputs(6:10),Sensor_Data(6:10),2);
% knee_flag = flag;
% end
Arduino_Outputs=[Arduino_Outputs_hip Arduino_Outputs_knee];
% Data_logger(Desired_traj,Sensor_Data,Arduino_Outputs,i,fid,lHandle,lHandle2); % Not yet implemented,
dump data in file and plot
% the graphs
%*****
% Experimental stuff
% Arduino_Outputs(13)=0;
% Arduino_Outputs(11)=0;
% Arduino_Outputs(3)=0;
% Arduino_Outputs(5)=0;
% *****

ack = Send_to_arduino(Arduino_Outputs);%
i=i+1;
Pause_till_next_cycle(i); % implemented
% pause(0.02);
Sensor_Data
end

end

```

A.2 The main block of Data acquire Module

```

%%*****
% [ang,length,pressure,flow] = Data_acquire().m
% This file contains code for data acquisition
%
%
%
% *****

function Sensor_Data = Data_acquire()

%     ang =  Get_angle();
%     length1 = Get_length(1);
%     length2 = Get_length(2);
%     pressure1 = Get_pressure(1);
%     pressure2 = Get_pressure(2);
%     %flow =      Get_flow();

%     fprintf('data read \n \r');

    s = instrfind('Tag','the_other_serial');
    s1 = instrfind('Tag','ip2');%----->new

    fprintf(s,'read');
    msg = fread(s,10);

% tic;

    fprintf(s1,'read'); %----->new
    msg_knee=fread(s1,10);%----->new
%     msg_knee = zeros(1,10);

%     disp(msg);
%     disp(toc);
%     fprintf('\n');
%     flushinput(s);
%     flushinput(s1);%----->new

    Data     = Reassemble_data(msg);
    Data_knee = Reassemble_data(msg_knee);%----->new

% for hip section
    ang = ang_mapping(Data(1),1);
    length1 = length_mapping(Data(2));
    length2 = length_mapping(Data(3));
    pressure1 = pressure_mapping(Data(4));
    pressure2 = pressure_mapping(Data(5));

```

```
% for knee section
ang_knee = ang_mapping(Data_knee(1),2);%----->new
length1_knee = length_mapping(Data_knee(2));%----->new
length2_knee = length_mapping(Data_knee(3));%----->new
pressure1_knee = pressure_mapping(Data_knee(4));%----->new
pressure2_knee = pressure_mapping(Data_knee(5));%----->new

Sensor_Data = [ang length1 length2 pressure1 pressure2 ang_knee length1_knee length2_knee
pressure1_knee pressure2_knee];

End
```

A.3 The Main block of controller

```

%%*****
% Arduino_Outputs = Controller(Cost_Outputs)
% - Main controller program + calculates the outputs to send to arduino
% - Cost_Outputs = [ang,length,pressure,flow]
% - Arduino_Outputs = [c1,c2,s1,s2,s3,s4,v1,v2,v3,v4],
% c=coup,v=value,s=solenoid
%
%
%
%*****

function [Arduino_Outputs,flag] = Controller(Cost_Outputs,Sensor_Data,a)
% Important Variables%*****
Kp=1;

if(a == 1)
M1_expansion_length = 26;
M1_pressure = Sensor_Data(4);
M2_expansion_length = 26;
M2_pressure = Sensor_Data(5);

elseif (a == 2)
M1_expansion_length = 21;
M1_pressure = Sensor_Data(9);
M2_expansion_length = 24;
M2_pressure = Sensor_Data(10);
end
critical_P = 115;

c1=0;c2=0;v1=0;v2=0;v3=0;v4=0;s1=0;s2=0;s3=0;s4=0;
%%*****

diff_ang = Cost_Outputs(1);
%If the max angle diff can be 40 degrees then we normalize it as follow
diff_angle_normalized = diff_ang/70;

%*****
% DEBUG

```

```

fprintf('diff_ang: %d \t diff_ang_normalized: %d \n',diff_ang,diff_angle_normalized);
%*****

%%
% Check to see if the diff is > 0
if (diff_angle_normalized > 0 && abs(diff_ang)>3 )
% Shorten M2
c1=0;
c2=1;
s4 =1;
v4 = Kp*(diff_angle_normalized*90);
if (v4>90) %i.e. it exceeds the upper limit of value angle
v4=90;
elseif (v4<3)
v4=0;
s4=0;

end

% Now expand M1 (@ full speed)until it reaches certain length
length1 = Sensor_Data(2);
if(length1 < M1_expansion_length && M1_pressure < critical_P)
s1=1;
v1=90;
c1=0;
else
s1=0;
v1=0;
c1=0;
end

%Close the other values-solenoid pair
v2=0;
v3=0;
s2=0;
s3=0;
flag =0;

elseif(diff_angle_normalized < 0 && abs(diff_ang)>3)
% The Strategy here is to let the limb fall and when it is within
% some tolerance couple it so that it doesn't move anymore.
% c2=0;
% c1=0;
%Shorten M1
c2=0;
c1=1;
s2 =1;
v2 = Kp*(abs(diff_angle_normalized)*90);

if (v2>90) %i.e. it exceeds the upper limit of value angle
v2=90;
elseif (v2<3)

```

```
v2=0;
s2=0;

end

% Now expand M2 (@ full speed)until it reaches certain length
length2 = Sensor_Data(3);
if(length2 < M2_expansion_length M2_pressure < critical_P)
s3=1;
v3=90;
c2=0;
fprintf('<<<<<<<<<<<<<<<<<<In the back expansion loop>>>>>>>>>>>>>>>>>');
else
s3=0;
v3=0;
c2=0;
end

%Close the other values-solenoid pair
v4=0;
v1=0;
s4=0;
s1=0;
flag =0;

elseif(abs(diff_ang)<3)
% Close all the values and engage all the coup/decoup
c1=1;
c2=1;
s1=0;
s2=0;
s3=0;
s4=0;
v1=0;
v2=0;
v3=0;
v4=0;
flag=1;
else
% Can come here for erroneous reading . In this case shutdown everything and stop it in its current position
s1=0;
s2=0;
s3=0;
s4=0;
v1=0;
v2=0;
v3=0;
v4=0;
c1=1;
c2=1;
flag =1;
fprintf('error error');

end
```

```
Arduino_Outputs = [c1,c2,s1,s2,s3,s4,v1,v2,v3,v4];
```

```
end
```

A.4 Arduino actuation code

```
#include <Servo.h>

#define S1_red 30
#define S2_red 32
#define S3_red 34
#define S4_red 36
#define S5_red 38
#define S6_red 40
#define S7_red 42
#define S8_red 44
#define dummy_one 28

#define S1_black 31
#define S2_black 33
#define S3_black 35
#define S4_black 37
#define S5_black 39
#define S6_black 41
#define S7_black 43
#define S8_black 45
#define dummy_two 29

Servo c1_servo, c2_servo, c3_servo, c4_servo, v1_servo, v2_servo, v3_servo,
v4_servo, v5_servo, v6_servo, v7_servo, v8_servo ;

void setup()
{
  Serial.begin(115200);
  Serial1.begin(115200);
  while(!Serial);
  int i=0;

  for (i=1;i<13;i++)
  {
    pinMode(i, OUTPUT);
    digitalWrite(i, LOW);
  }
  for (i=30;i<46;i++)
  {
    pinMode(i, OUTPUT);
    digitalWrite(i, LOW);
  }
  Serial.setTimeout(10);
  Serial1.setTimeout(10);

  c1_servo.attach(12);
  c2_servo.attach(11);
  c3_servo.attach(3);
  c4_servo.attach(4);
  v1_servo.attach(5);
  v2_servo.attach(6);
  v3_servo.attach(7);
  v4_servo.attach(8);
}
```



```

void loop()
{
  byte d_data[20]={};
  int pull_high[8] = {},int_data[20]={};
  int i=0;

  while(!Serial1.available());

  if (Serial1.available() > 0)
  {
    Serial1.readBytes(d_data,20);
    if(d_data[0]==0)
    {
      c1_servo.write(150);
    }

    else if(d_data[0]==1)
    {
      c1_servo.write(50);
    }
    if(d_data[1]==0)
    {
      c2_servo.write(100);
    }
    else if(d_data[1]==1)
    {
      c2_servo.write(30);
    }
    if(d_data[2]==0)
    {
      pull_high[0]=S1_black;
    }
    else if(d_data[2]==1)
    {
      pull_high[0]=S1_red;
    }
    if(d_data[3]==0)
    {
      pull_high[1]=S2_black;
    }
    else if(d_data[3]==1)
    {
      pull_high[1]=S2_red;
    }
    if(d_data[4]==0)
    {
      pull_high[2]=S3_black;
    }
    else if(d_data[4]==1)
    {
      pull_high[2]=S3_red;
    }
    if(d_data[5]==0)
    {
      pull_high[3]=S4_black;
    }
    else if(d_data[5]==1)
    {
      pull_high[3]=S4_red;
    }
    v1_servo.write(d_data[6]+30);
    v2_servo.write(d_data[7]+30);
    v3_servo.write(d_data[8]+30);
    v4_servo.write(d_data[9]+30);
    if(d_data[10]==0)
    {
      c3_servo.write(110);
    }
    else if(d_data[10]==1)

```

```

    {
        c3_servo.write(20);
    }
    if(d_data[11]==0)
    {
        c4_servo.write(130);
    }
    else if(d_data[11]==1)
    {
        c4_servo.write(30);
    }
    if(d_data[12]==0)
    {
        pull_high[4]=S5_black;
    }
    else if(d_data[12]==1)
    {
        pull_high[4]=S5_red;
    }
    if(d_data[13]==1)
    {
        digitalWrite(S6_red,HIGH);
        pull_high[5]=dummy_one;
    }
    else if(d_data[13]==0)
    {
        digitalWrite(S6_red,LOW);
        pull_high[5]=dummy_two;
    }
    if(d_data[14]==0)
    {
        pull_high[6]=S7_black;
    }
    else if(d_data[14]==1)
    {
        pull_high[6]=S7_red;
    }
    if(d_data[15]==0)
    {
        pull_high[7]=S8_black;
    }
    else if(d_data[15]==1)
    {
        pull_high[7]=S8_red;
    }
    v5_servo.write(d_data[16]+30);
    v6_servo.write(d_data[17]+30);
    v7_servo.write(d_data[18]+30);
    v8_servo.write(d_data[19]+30);

    for (i=0;i<8;i++)
    {
        digitalWrite(pull_high[i],HIGH);
    }
    delay(50);
    for (i=0;i<8;i++)
    {
        digitalWrite(pull_high[i],LOW);
    }
    //    for(int j=0;j<20;j++)
    //    {
    //        int_data[j]=atoi(d_data[j]);
    //    }
    for(int j=0;j<20;j++)
    {
        Serial.print(d_data[j],DEC);
        Serial.print(" ");
    }
    Serial.print("\n");
}

```

```
else
{
  Serial.println("serial not received");
  digitalWrite(13, LOW);
}
}
```

A.5 Sensor Code on Arduino

```

String act_str="";
int ang_pin = A0;
int ir1_pin = A1;
int ir2_pin = A2;

void setup()
{
  // put your setup code here, to run once:
  Serial.begin(115200);
  while(!Serial);
  pinMode(13, OUTPUT);
  Serial.setTimeout(1);
}

void loop()
{
  // put your main code here, to run repeatedly:
  String c1e = "send";
  String c2e = "read";
  c1e += '\n';
  c2e += '\n';
  char d_data[10]={};
  char c_data[10]={0,1,0,1,0,1,35,65,45,10};

  int i_data[5]={};
  char b_data[10]={};
  long unsigned int avg_ang=0;
  // for(int i=0;i<100;i++)
  // {
  //   avg_ang = avg_ang+analogRead(ang_pin);
  // }
  // i_data[0] = avg_ang/100;

  i_data[0] = analogRead(ang_pin);
  for(int i=0;i<50;i++)
  {
    i_data[1] = i_data[1]+analogRead(ir1_pin);
  }
  i_data[1] = i_data[1]/50;

  for(int i=0;i<50;i++)
  {
    i_data[2] = i_data[2]+analogRead(ir2_pin);
  }
  i_data[2] = i_data[2]/50;

  b_data[0]=char(i_data[0]/256);
  b_data[1]=char(i_data[0]%256);
  b_data[2]=char(i_data[1]/256);
  b_data[3]=char(i_data[1]%256);
  b_data[4]=char(i_data[2]/256);
  b_data[5]=char(i_data[2]%256);
  b_data[6]=char(i_data[3]/256);
  b_data[7]=char(i_data[3]%256);
  b_data[8]=char(i_data[4]/256);
  b_data[9]=char(i_data[4]%256);

  if (Serial.available() > 0)
  {
    act_str = Serial.readString();
  }
}

```

```
if(act_str == c1e)
{
    Serial.readBytes(d_data,10);

    Serial.write(d_data,10);
    Serial.print("\n");

    if(d_data == c_data)
    {
        digitalWrite(13, HIGH);
    }

    else
    {
        digitalWrite(13, LOW);
    }

}
else if (act_str == c2e)
{
    /* code */
    Serial.write(b_data,10);
    Serial.print("\n");

}

    act_str="";
    Serial.flush();
}
else
{
    Serial.println("nothing received");
}
}
```

References

1. "Fact sheet on wheelchairs - World Health Organization ..." 2013. 22 Apr. 2016
<http://www.searo.who.int/entity/disabilities_injury_rehabilitation/wheelchair_factsheet.pdf>

2. "Human Elbow Anatomy – Humananatomychart.info." 2016. 22 Apr. 2016
<<http://anatomydiagram.info/human-elbow-anatomy/>>
3. "Research Review: Weak in the knees? | Precision Nutrition." 2009. 22 Apr. 2016
<<http://www.precisionnutrition.com/quad-hamstring-ratio>>
4. "Knee (Human Anatomy): Images, Function ... - WebMD." 2010. 22 Apr. 2016
<<http://www.webmd.com/pain-management/knee-pain/picture-of-the-knee>>
5. "Muscles that Cause Movement at the Knee Joint - Boundless." 2016. 22 Apr. 2016
<<https://www.boundless.com/physiology/textbooks/boundless-anatomy-and-physiology-textbook/muscular-system-10/muscles-of-the-lower-limb-107/muscles-that-cause-movement-at-the-knee-joint-579-9335/>>
6. "Hip Anatomy, Function and Common Problems." 2010. 22 Apr. 2016
<<http://www.healthpages.org/anatomy-function/hip-structure-function-common-problems/>>
7. "Ball & Socket Joint - Anatomy Pictures and Information." 2007. 22 Apr. 2016
<http://www.innerbody.com/image_skel07/skel34.html>
8. "Muscles of the hip - Wikipedia, the free encyclopedia." 2011. 22 Apr. 2016
<https://en.wikipedia.org/wiki/Muscles_of_the_hip>
9. "Hip rotation - Shotokan Karate Union." 2015. 22 Apr. 2016
<<http://www.shotokankarateunion.com/T122.html>>
10. Noritsugu, Toshiro et al. "Wearable power assist device for hand grasping using pneumatic artificial rubber muscle." *SICE-ANNUAL CONFERENCE*- 4 Aug. 2004: 420.
11. "Hydro Muscle –A Novel Soft Fluidic Actuator." 2016. 22 Apr. 2016 ,Sridar, Saivimal (WPI), Majeika, Corey (WPI), Bowers, Matthew (Worcester Polytechnic Institute), Schaffer, Phillip (Worcester Polytechnic Intitute), Ueda, Seiichiro (WPI), Barth, Andrew (WPI), Sorrells, Jon (WPI), Wu, Jon (WPI), Hunt, Thane (WPI), Popovic, Marko (Worcester Polytechnic Institute)<<https://ras.papercept.net/conferences/scripts/abstract.pl?ConfID=119&Number=2771>>
12. T.Noritsugu, et al., "Wearable Power Assist Device for Standing up Motion Using Pneumatic Rubber Artificial Muscles," *Journal of Robotics and Mechatronics*, Vol.19, No.6, pp.6 19-628, 2007.

13. Deimel, Raphael, and Oliver Brock. "A compliant hand based on a novel pneumatic actuator." *Robotics and Automation (ICRA), 2013 IEEE International Conference on* 6 May. 2013: 2047-2053.
14. Galloway, Kevin C et al. "Mechanically programmable bend radius for fiber-reinforced soft actuators." *Advanced Robotics (ICAR), 2013 16th International Conference on* 25 Nov. 2013: 1-6.
15. Chou, Ching-Ping, and Blake Hannaford. "Measurement and modeling of McKibben pneumatic artificial muscles." *Robotics and Automation, IEEE Transactions on* 12.1 (1996): 90-102.
16. Chou, Ching-Ping, and Blake Hannaford. "Static and dynamic characteristics of McKibben pneumatic artificial muscles." *Robotics and Automation, 1994. Proceedings., 1994 IEEE International Conference on* 8 May. 1994: 281-286.
17. Klute, Glenn K, Joseph M Czerniecki, and Blake Hannaford. "McKibben artificial muscles: pneumatic actuators with biomechanical intelligence." *Advanced Intelligent Mechatronics, 1999. Proceedings. 1999 IEEE/ASME International Conference on* 1999: 221-226.
18. M. B. Popovic, "Biomechanics and Robotics", book 364 pages, Copyright © 2014 Pan Stanford Publishing Pte. Ltd., Singapore, ISBN 978-981-4411-37-0 (Hardcover), 978-981-4411-38-7 (eBook). [www.panstanford.com]
19. McCarthy, Gregory et al. "Hydraulically Actuated Muscle (HAM) Exo-Musculature." *Proc. Robot Makers Workshop, Robotics: Science Systems Conf* 2014.
20. "Hydro-muscle actuated knee for biped Leg Chair" Mar 2016, Matthew P. Bowers, Seiichiro Ueda, Amair Zia, R. Matthew Rafferty, Varun V. Verlencar
21. Saivimal Sridar, Ananth Jonnavittula, Chinmay V. Harmalkar, Lynn R. Koesterman, John D. Kelly, Thane R. Hunt, and Marko Popovic
22. KIM, SM. "Study of Knee and Hip Joints' Moment Estimation ... - IAENG." 2009. <http://www.iaeng.org/publication/WCECS2009/WCECS2009_pp785-788.pdf>

