AY 2022-23

PROJECT REPORT
ON

OLD AGE ASSISTIVE INTERACTIVE ROBOT

Submitted for

ECL405: Practicum IV

By
SEJAL JAIN (21240)

IV Semester
B. Tech (ECE)

SCHOOL OF ELECTRONICS

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY UNA
HIMACHAL PRADESH

APRIL 2023

BONAFIDE CERTIFICATE

This is to certify that the project titled **OLD AGE ASSISTIVE INTERACTIVE ROBOT** is a bonafide record of the work done by

SEJAL JAIN (21240)

in partial fulfilment of the requirements for the award of the degree of Bachelor of Technology in **ELECTRONIC AND COMMUNICATION ENGINEERING** of the INDIAN INSTITUTE OF INFORMATION TECHNOLOGY UNA, HIMACHAL PRADESH, during the year 2021 - 2025.

under the guidance of

DR. TANU WADHERA

Project viva-voce held on: _____

Internal Examiner External Examiner

ORIGNALITY / NO PLAGARISM DECLARATION

I certify that this project report is our original report and no part of it is copied from any published reports, papers, books, articles, etc. I certify that all the contents in this report are based on our personal findings and research and we have cited all the relevant sources which have been required in the preparation of this project report, whether they be books, articles, reports, lecture notes, and any other kind of document. I also certify that this report has not previously been submitted partially or as whole for the award of degree in any other university in India and/or abroad.

I hereby declare that, we are fully aware of what constitutes plagiarism and understand that if it is found at a later stage to contain any instance of plagiarism, our degrees may be cancelled.

SEJAL JAIN (21240)

ABSTRACT

In rapid change with technology a huge development of robots is competing equally with humans. The technology development has brought many changes in the field of manufacturing and research. Initially, the robots were used by humans to make their work easier, but today the robots are developed to replicate humans and substitute the labor in the fields where even human presence is difficult. artificial intelligence (AI) and robotics solutions to support older adults in maintaining their independence and improving their quality of life. The type of application defines the robots construction and the requirement of systems. This paper covers background of the robotics such as variety of robots, kinematics of robots, localization and mapping scenario, the trajectory of the robot's path. Also, the control systems used in robots like actuators, sensors and processing of the actuators, robots learning has been presented.

***Keywords*:** Humanoid robots, Robot sensing systems, robot kinematics, robots learning, robotics, robots construction, locomotion, design, programming

ACKNOWLEDGEMENT

I would like to thank the following people for their support and guidance without whom the completion of this project in fruition would not be possible.

I would like to express our sincere gratitude and heartfelt thanks to Dr. Tanu for her unflinching support and guidance, valuable suggestions and expert advice. Their words of wisdom and expertise in subject matter were of immense help throughout the duration of this project.

I also take the opportunity to thank our Director and all the faculty of School of Computing/Electronics, IIIT Una for helping us by providing necessary knowledge base and resources.

I would also like to thank our parents and friends for their constant support.

SEJAL JAIN (21240)

TABLE OF CONTENTS

LIST OF ACRONYMS

IDE	Integrated Development Environment
USB	Universal service Bus
SPP	Serial Port Protocol
RS	Register Select
I/O	Input Output
R/W	Read Write

LIST OF FIGURES

LIST OF TABLES

Chapter 1

Introduction

Humanoid robots are expected to exist and work in a close relationship with human beings in the everyday world and to serve the needs of physically handicapped people. These robots must be able to cope with the wide variety of tasks and objects encountered in dynamic unstructured environments. Humanoid robots for personal use for elderly and disabled people must be safe and easy to use. Therefore, humanoid robots need a lightweight body, high flexibility, many kinds of sensors and high intelligence. Advanced, high-performance robots are used today in automobile manufacturing and aircraft assembly, and electronics firms use robotic devices together with other computerized instruments to sort or test finished products. The modeling of a sophisticated hand is one of the challenges in the design of humanoid robots and artificial arms. The basic idea in all projects is to create an integrated system to monitor the activities and health of an old person and to provide support services in daily life. Humanoid robots are professional service robots built to mimic human motion and interaction. Like all service robots, they provide value by automating tasks in a way that leads to cost-savings and productivity. Humanoid robots are a relatively new form of professional service robot. While long-dreamt about, they're now starting to become commercially viable in a wide range of applications.

The main advantage is that a humanoid robot can act as human beings in such an environment without any previous adjustment for the robot. On the other hand, human friendly and functional machinery become more neccesary as robots are used closer to human beings to care.

Cognitive robots can be used in several scenarios: industrial robots, medical robots, domestic robots etc; one of the extreme expressions of such a paradigm, anyway, is when the task of the robot is the interaction with humans. Such robots are called social robots, and can be profitably used, just to mention a few examples, as automatic shop assistance, as virtual assistant on the production lines or as robotic museum guides.

1. Hardware Parts

The hardware part consists of the mechanical design of the robot, robot kinematics, the adequate choice of the motors, and the electronic devices to properly the robot joints.

1.1 Arduino

Figure 1 shows Arduino UNO is an open source micro controller board placed on the micro chip ATmega328p micro controller and developed by Adruino.cc. The board has 6 Analog pins, 14 digital pins programmable with Arduino IDE via a Type B USB cable. It can power by external main volt battery.

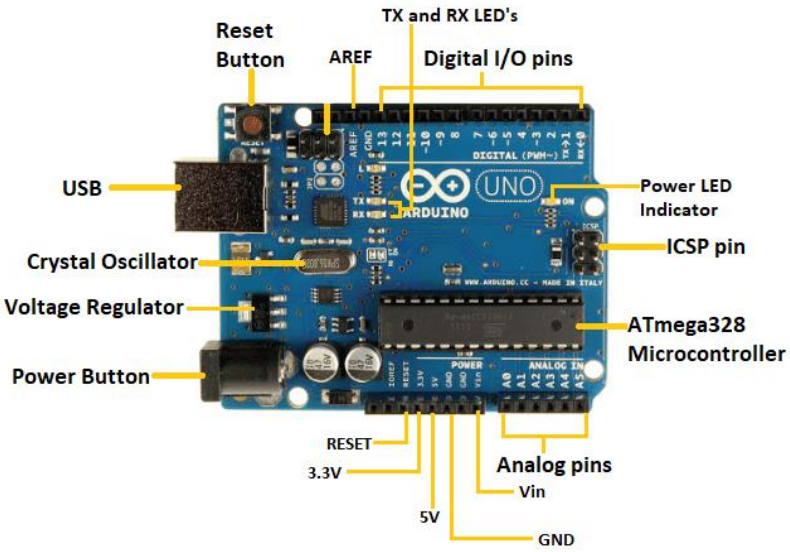

Figure 1: Arduino

- A number of pins, which are used to connect with various components you might want to use with the Arduino. These pins come in two varieties:
  - Digital pins, which can read and write a single state, on or off. Most Arduinos have 14 digital I/O pins.
  - Analog pins, which can read a range of values, and are useful for more fine-grained control. Most Arduinos have six of these analog pins.

  These pins are arranged in a specific pattern, so that if you buy an add-on board designed to fit into them, typically called a “shield”, it should fit into most Arduino-compatible devices easily.

- A power connector, which provides power to both the device itself, and provides a low voltage which can power connected components like LEDs and various sensors, provided their power needs are reasonably low. The power connector can connect to either an AC adapter or a small battery.
- A microcontroller, the primary chip, which allows you to program the Arduino in order for it to be able to execute commands and make decisions based on various input. The exact chip varies depending on what type of Arduino you buy, but they are generally Atmel controllers, usually a ATmega8, ATmega168, ATmega328, ATmega1280, or ATmega2560. The differences between these chips are subtle, but the biggest difference a beginner will notice is the different amounts of onboard memory.
- A serial connector, which on most newer boards is implemented through a standard USB port. This connector allows you to communicate to the board from your computer, as well as load new programs onto the device. Often times Arduinos can also be powered through the USB port, removing the need for a separate power connection.
- A variety of other small components, like an oscillator and/or a voltage regulator, which provide important capabilities to the board, although you typically don't interact with these directly; just know that they are there.

1.2 HC-05 Bluetooth module

Figure 3. Shows HC05 module is a simple Bluetooth module is a simple Bluetooth serial port protocol (SPP) module designed for wireless serial connection setup. It has a footprint as small as 12.7mm X 27mm. It will simplify the overall design cycle.

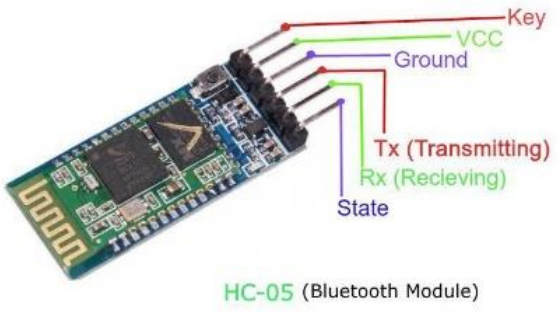

Figure 2: Bluetooth module

HC 05/06 works on serial communication. Here the android app is designed to send serial data to the Bluetooth module when a certain button is pressed. The Bluetooth module at the

other end receives the data and sends it to Arduino through the TX pin of the Bluetooth module (RX pin of Arduino). The Code fed to Arduino checks the received data and compares it. If received data is 1 the LED turns on turns OFF when received data is 0.

1.3 LCD (Liquid Crystal Display)

Figure 3: LCD

The LCDs have a parallel interface, meaning that the microcontroller has to manipulate several interface pins at once to control the display. The interface consists of the following pins:

- A **register select (RS) pin** that controls where in the LCD's memory you're writing data to. You can select either the data register, which holds what goes on the screen, or an instruction register, which is where the LCD's controller looks for instructions on what to do next.
- A **Read/Write (R/W) pin** that selects reading mode or writing mode.
- An **Enable pin** that enables writing to the registers.
- 8 **data pins (D0 -D7)**. The states of these pins (high or low) are the bits that you're writing to a register when you write, or the values you're reading when you read.

1.4 Steeper motor

Stepper motors are DC motors that divide a full rotation into a number of equal steps. One pulse causes the motor to increment one precise angle of motion. By energizing each step in sequence, the motor will rotate, one step at a time. Increasing the current or pulse increases motor torque, and the higher the step frequency, the higher the motor speed. The motor's position can be

commanded to move and hold at one of these steps without any position sensor, as long as the motor is correctly sized to the application.

Figure 4: Steeper Motor

2. Software parts

The software part contains the high level algorithms that convert the desired word to a sequence of target points, and the control algorithms that ultimately make the robot move according to the specifications. Here the writing mechanism is made by speech recognition technique. This speech recognition can be provided through either by using microphone or by using android applications.

3. Advancements in technology

- **System Hardware:** Designing a fully electromechanical humanoid, including hands. The goal is to develop hardware with the physical capabilities of a nonexpert human. We are measuring this in terms of range of motion, payload, torque, cost of transport and speed, and will continue to improve through rapid cycles of development, each cycle as part of a continuum.
- **Unit Cost:** Reduce individual humanoid unit costs through high-rate volume manufacturing, working towards a sustainable economy of scale. We are measuring our costs through the fully burdened operating cost per hour. At high rates of volume manufacturing, optimistic unit cost will come down to affordable levels.
- **Safety:** It's essential that our humanoids will be able to interact with humans in the workplace safely. We will design them to be able to adhere to industry standards and corporate requirements.

- **Volume Manufacturing:** We foresee not only needing to deliver a high-quality product but also needing to deliver it at an exceptionally high volume. We anticipate a steep learning curve as we exit prototyping and enter volume manufacturing. We are preparing for this by being thoughtful about design for manufacturing, system safety, reliability, quality, and other production planning.
- **Artificial Intelligence:** Building an AI system that enables our humanoids to perform everyday tasks autonomously is arguably one of the hardest problems we face long-term. We are tackling this by building intelligent embodied agents that can interact with complex and unstructured real-world environments.

4. Walking Control Algorithm

In this section, the authors proposed an online controller design and suitable use of online controllers in a walking cycle. The proposed walking control method is based on a controller switching strategy, and thus it is important to divide the walking cycle into several walking stages. In each walking stage, suitable online controllers are activated. Fig. 10 shows the walking stages and their descriptions are as follows:

1) 1st stage : lift the left leg to its maximum flexion and height

2) 2nd stage : lower the left leg until it makes complete contact with the ground

3) 3rd stage : lift the right leg to its maximum flexion and height

4) 4th stage : lower the right leg until it makes complete contact with the ground

5) 5th stage : this stage follows the 1st or 3rd stage, and brings the robot to a standing pose with both legs landed on the ground.

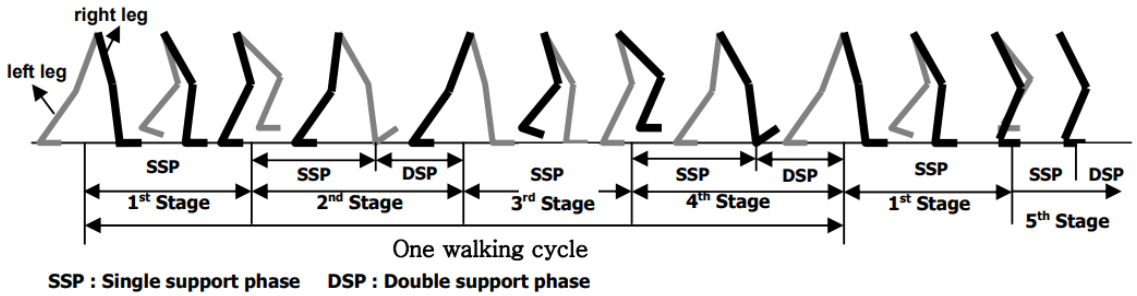

Figure 5: Walking algorithm of humanoid robot

Control Scheme	Online Controller (working period)	Objective
Real-Time Balance Control	Damping controller (1^{st} and 3^{rd} Stages, SSPs of 2^{nd} and 4^{th} Stages)	Eliminate the upper body oscillations in single support phase by imposing damping at the ankle joints
	ZMP compensator (1^{st} and 3^{rd} Stages, SSPs of 2^{nd} and 4^{th} Stages)	Maintain dynamic balance by horizontal motions of the pelvis
	Soft landing controllers (DSPs of 2^{nd} and 4^{th} Stages)	Absorb landing impact and adapt the foot to the ground surface
Walking Pattern Control	Pelvis swing amplitude controller (DSPs of 2^{nd} and 4^{th} Stages)	Compensate the lateral swing amplitude of the pelvis by considering the amplitude of the ZMP
	Torso pitch/roll controller (DSPs of 2^{nd} and 4^{th} Stages)	Compensate the center position of the pelvis swing to balance the pitch & roll inclinations of the torso
Predicted Motion Control	Tilt over controller (1^{st} and 3^{rd} Stages)	Compensate the ankle joint trajectories to prevent "tilt over" in roll direction
	Landing position controller (2^{nd} and 4^{th} Stages)	Compensate the landing position to prevent unstable landing

Table1: Summary of Walking algorithm

4.1. Upright Pose Controller

This new online controller was designed to allow the robot to walk stably on a globally inclined floor. If the floor is globally inclined, the robot is tilted in the direction of floor inclination, so the robot walking becomes unstable. To prevent the tilting of the robot during walking, the authors proposed the upright pose controller, which makes the robot upright all the time. In general, the global inclination of the floor continuously changes slowly as the robot proceeds across the floor surface, and hence it is relatively easy to handle. Here, a measurement of the global inclination of

the floor is a key factor. The authors have measured this through the use of an inertial sensor at the torso. A 2-axis accelerometer is installed in the inertial sensor, and it can be employed as an inclinometer by using a very heavy low pass filter. That is, signals due to the fast movement of the robot can be filtered, and the floor inclination is then measured from the robot torso's inclination, which is caused by the slowly changing floor inclination. By measuring the time varying global inclination, the upright pose controller keeps the robot's torso upright all the time. Figure shows the schematics of the upright pose control.

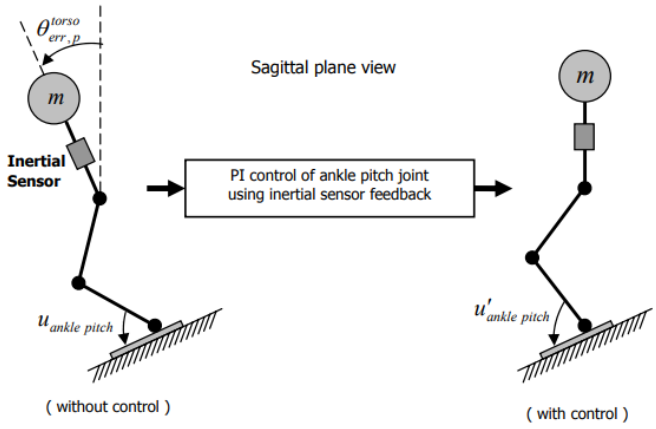

Figure6: Upright pose Controller

4.2. Landing Angular Momentum Control

It is difficult for the robot to walk on an uneven floor even if the floor is not globally inclined. A typical room floor has a local inclination of about 1~2 degrees. It is important to note that the small amount of the inclination can cause a serious instability. Moreover, the local inclination does not have any tendency and is changeable at every step. Due to the local inclination of the floor, the robot may fall down easily. Hence, it is necessary to overcome the local inclination of the floor. The most fundamental preparation for the local inclination is that the sole has to be supported by 4 points as a flat sole cannot be fixed at the moment of landing. To be robust against the local inclination of the floor, the authors proposed the control strategy which uses the angular momentum.

4.3. Landing Shock Absorber

When a robot walks on an uneven floor, the altitude of the floor is always changing. For example, even if the robot lowers the landing foot to the prescribed altitude, the foot may not contact the

floor. On the contrary, the landing foot may contact the floor before the foot is lowered to the prescribed position. In general, the second case occurs frequently because the supporting frame between the pelvis and the hip joint can be deflected due to the joint compliance and flexibility of the aluminum frame when the leg is swinging. Therefore, the foot of swinging leg can not be raised to desired height. If the foot lands early, the leg strongly pushes down on the floor with large impact. This large impact makes the robot to shake, thus the walking becomes unstable. Therefore, for the stable walking, the vertical displacement of the foot should be controlled in order to reduce the landing impact.

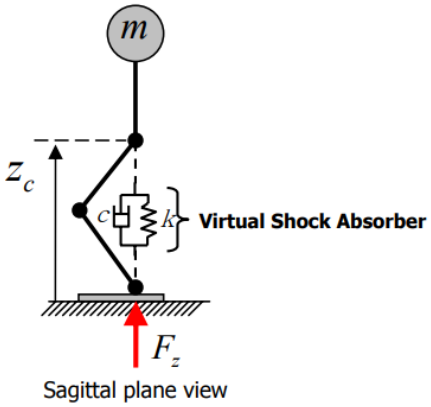

Figure7: Landing shock observer

4.4. Landing Timing Controller

During walking, landing timing of each foot is prescribed by walking pattern design. It is important to note that prescribed landing timing does not provide excellent landing at every step because landing timing of prescribed walking pattern is just for ideal case that the floor is perfectly flat with no inclination and the robot is not tilted. Hence, it is necessary to compensate the landing timing for the best landing. Our conventional landing timing controller only suspended the walking pattern flow if the foot does not contact the floor at the end of foot lowering. This late landing is caused by tilting outside of the robot due to local inclination of the supporting foot.

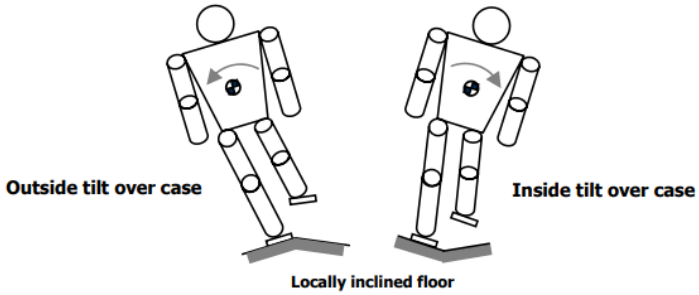

Figure8: Landing Timing Controller

4.5. Landing Position Controller

Similar to the landing timing, it is important to compensate the landing position. The landing timing and position are the most important factors in biped walking control as the walking is ultimately realized by these two factors. If we consider a human walking on a single position, the landing position on the floor is continuously changing. This phenomenon represents humans determine the landing position which produces suitable angular momentum through the use of vestibular organ. For example, if the angular velocity of the human is so much large during landing, the human will land the foot far away from the (Coronal Plane View) ω Prescribed left foot altitude Prescribed right foot altitude Angular velocity, ω Stable region Zero crossing point E (inside tilt over case) E A B Compensated right foot altitude Time [sec] Zero crossing point (normal case) Angular Velocity[deg/s] or Displacement[mm] body center in order to reduce the angular velocity after landing. Likewise, a biped walking robot also has to compensate the landing position from the prescribed landing position through the sensor feedback. The authors compensate the landing position before landing by using the angular momentum so that the robot can land the foot at the suitable position which is predicted to generate moderate angular momentum at every step. The rate gyro is attached on the hip because the angular velocity signal of the hip is clearer than that of the torso.

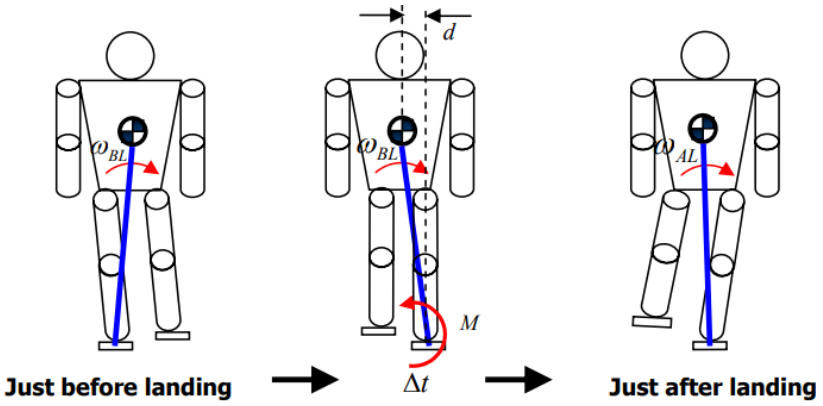

Figure9: Landing position Controller

4.6. Vibration Reduction Controller

When a biped humanoid robot is walking, the swinging leg vibrates. This is because the connecting frame between the leg and pelvis is deflecting and the reduction gear is slightly compliant. While the vibrations are not large, they disturb the precise landing position control of the foot. If the position control accuracy is not within ± 3 mm, the performance of the landing position controller is diminished. Therefore, the vibration reduction is fundamentally necessary for the landing position controller, so the authors proposed the vibration reduction controller in order to reduce the vibrations of foot. Fig. 18 represents a mathematical model of a swinging leg.

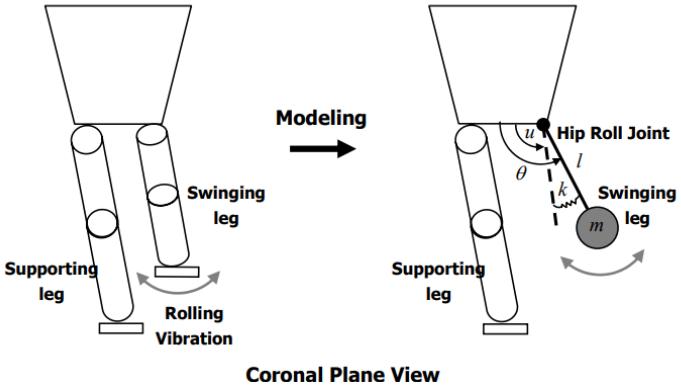

Figure10: Vibration Reduction Controller

Chapter 2

Review of Literature

2.1 Literature Survey

The paper presents an approach to design rapid and fluid movements of a universal robot to perform robot writing tasks. The design of the proposed robot comprises both hardware and software. The hardware part consists of the mechanical design of the robot, the adequate choice of the motors, and the electronic devices to properly drive the robot joints. The software part contains the high level algorithms that convert the desired word to a sequence of target points, and the control algorithms that ultimately make the robot move according to the specifications. Here the writing mechanism is made by speech recognition technique. This speech recognition can be provided through either by using microphone or by using android applications. Thereby the robot can make the writing mechanism according to the user's input.

This paper reviews several approaches that have been adopted for the optimization of the mechanical structure and the locomotion modes of artificial legged systems. Among them are referred the mechatronic mimic of the characteristics of biological animals (Quinn, et al., 2001; Breithaupt, et al., 2002; Alexander, 2005; Albiez and Berns, 2005; Hennion, et al., 2005), the use of genetic algorithms (GAs) for the optimization of the legged structure parameters (Ziegler and Banzhaf, 2001; Reyes and Gonzalez, 2005), the adoption of sound mechanical design rules (Yoneda, 2001), or the optimization of power / energy-based indices (Silva and Machado, 2005).

2.2 Mechanical Controls

a. The approaches to the systems design discussed in the two previous sections are inspired in the strategies found in nature. However, it is important to keep in mind that humanoid robots are machines. Therefore, the first aspect to consider in their design phase should be the adequate implementation from the mechanical and physical viewpoints. The characteristics of several structures that can be adopted for the legs of artificial locomotion systems, several concepts to be adopted during the design of legged vehicles. The main idea is to maximize the power developed in the system and to maximize the energy efficiency. The technique of

actuator gravitational decoupling was adopted in several robots and can be implemented not only during the system design, but also in the posture during locomotion.

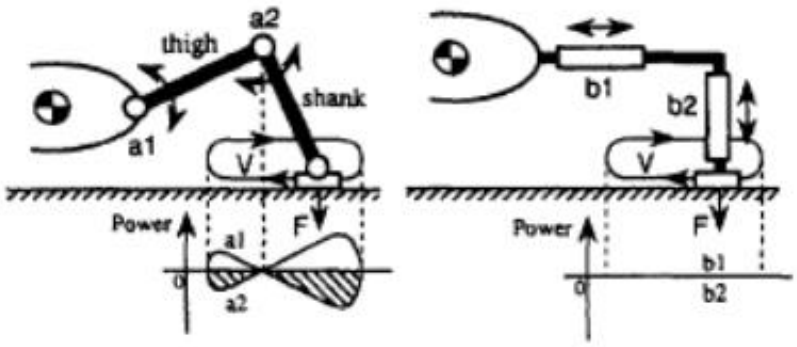

Figure 10: Robot from paper of reference 1

b. The maximum torque needed for the hip motor of a monopod robot as a way to design the transmission parameters of this joint. In a second phase, they used three indices, namely the absolute values of the instantaneous mechanical power on the hip joint of the monopod robot, its integral over time, and the specific resistance index, to analyse the robot performance when moving at different speeds, and when adopting different hip trajectories. These authors compared also the specific resistance of their robot, with those of other legged, wheeled and tracked vehicles and living beings.

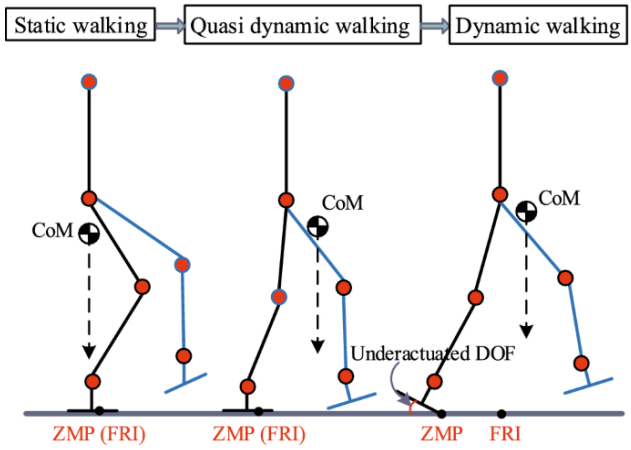

Figure 11: Robot from paper of reference 2

c. Concerning the weakness of artificial locomotion systems, one of the most serious problems faced by humanoid robot designers is the optimization of the energy consumption. It is verified that the locomotion efficiency of the existing robots has been very low when compared either with the living animals, or with the locomotion through wheels. As an example, Kajita, et al. (2002) performed a simulation of the HRP1 humanoid robot running gait (Figure 8) and concluded that the robot would need actuators from 28 up to 56 times more powerful than those available in the real humanoid. Besides this result, they also estimated that the energy consumption would be about ten times higher than that of a human adopting the same gait.

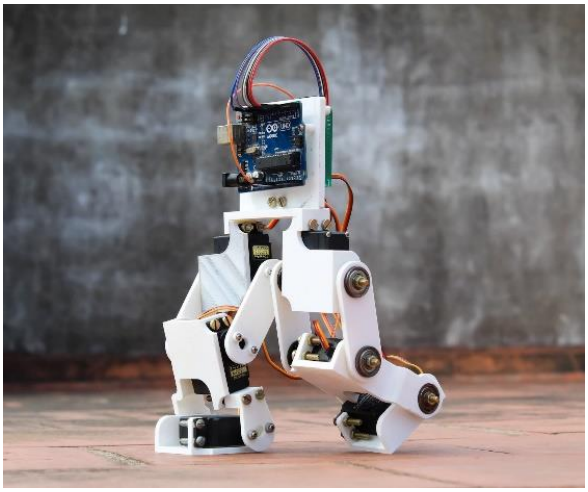

Figure 12: Robot from paper of reference 3

d. The design of a walking machine equipped with a one DOF leg mechanism. Two design objectives were simultaneous considered in the model: to minimize the peak crank torque, for the entire foot-point path, and the leg size at a prescribed leg posture. The algorithm also included constraint functions, namely concerning the stride length, the foot-path height, the fourbar transmission angle, the skew-pantograph transmission angle and the orientation angle of the lowest link. The optimization results showed a dramatic reduction in the mechanism size and a moderate reduction in the actuating torque from its preliminary design.

Chapter 3

Methodology

3.1 Circuit Diagram

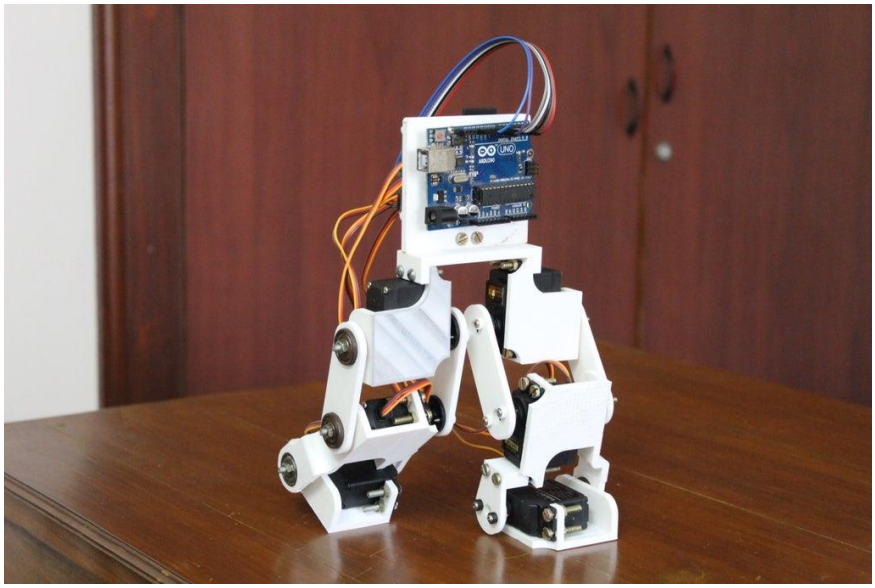

Figure 13: walking robot

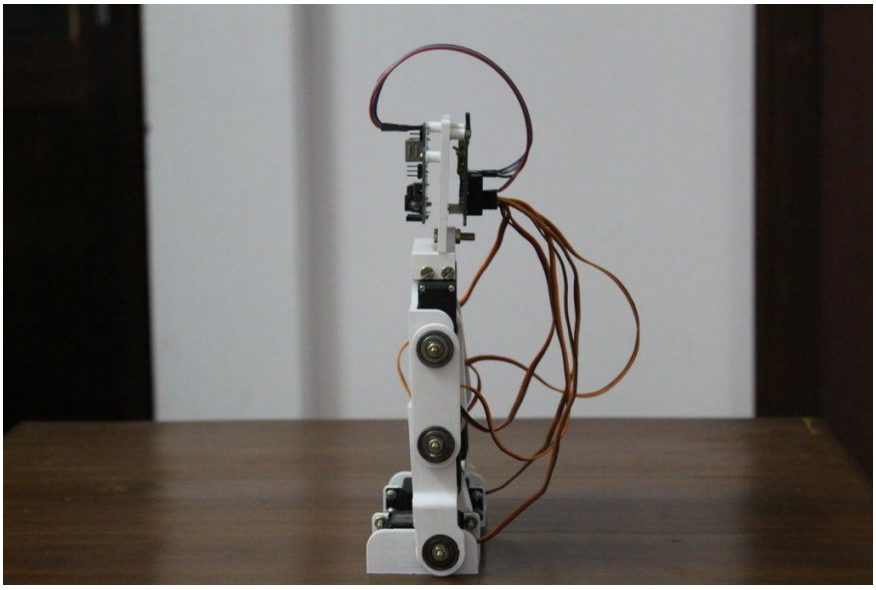

Figure 14: Arduino connection with robot

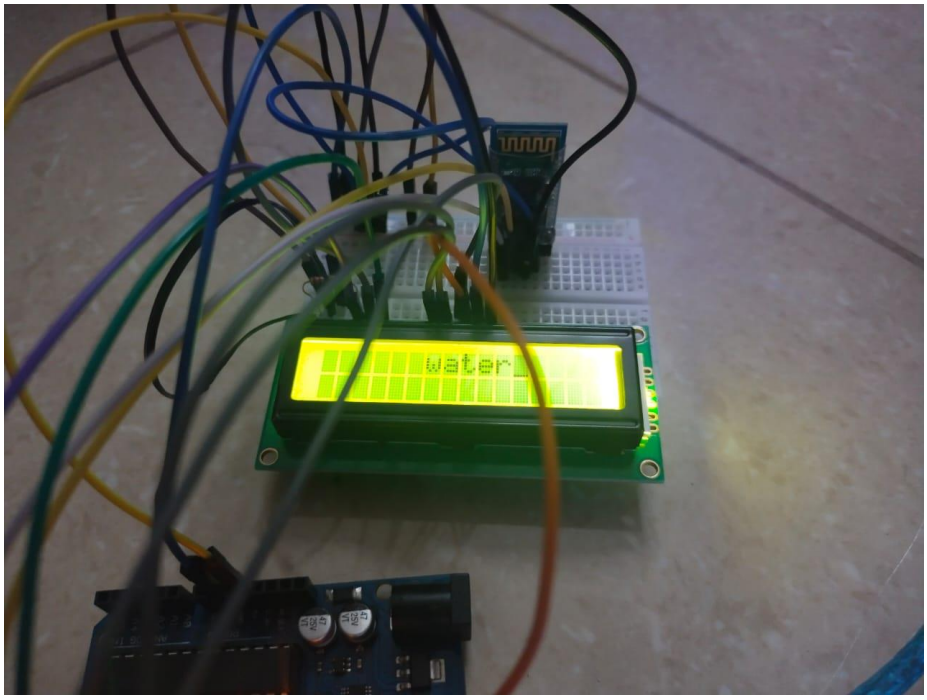

Figure 15: Instruction-1 to robot

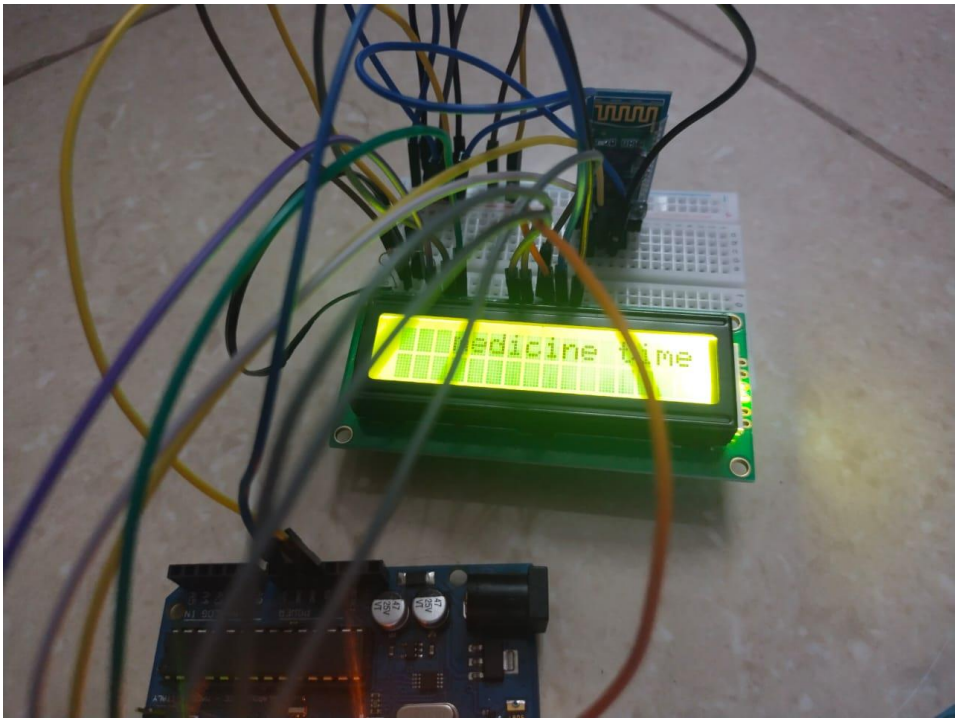

Figure 16: Instruction-2 to robot

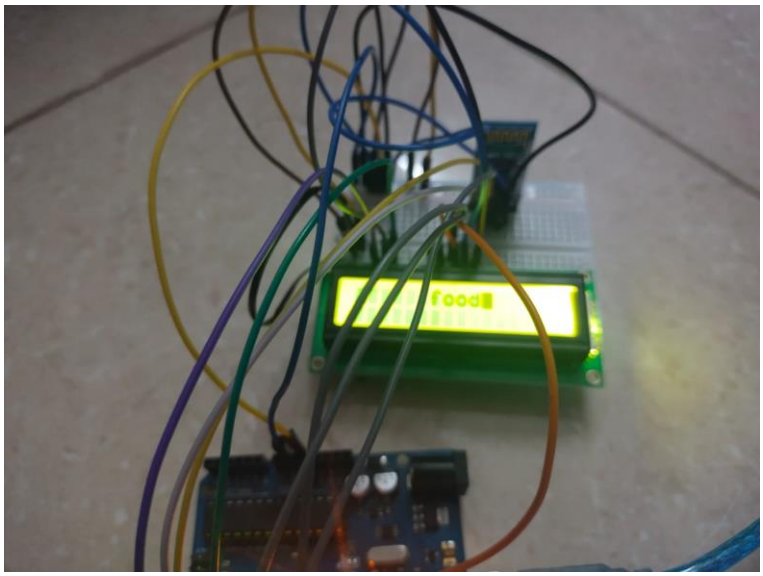

Figure 17: Instruction-3 to robot

3.2 Conclusion and Future Work

This paper described a walking control algorithm for biped humanoid robots that considers an uneven and inclined floor. In the case of the author's previous algorithm, the online controllers were designed without the consideration of the local and global inclinations of the floor. That is, the floor was assumed to be comparatively flat. The previous online controllers worked well for a slightly uneven and inclined floor, but the robot immediately fell down when the floor inclinations exceeded a certain threshold. Hence, six online controllers (Upright pose controller, landing angular momentum controller, landing shock absorber, landing timing controller, landing position controller, and vibration reduction controller) were developed and added to the previous algorithm. These online controllers were designed through simple mathematical models and experiments, and then suitable activation periods were planned in a walking cycle. Each online controller has a clear objective and the controllers are decoupled from each other. To validate the performance of the online controllers, walking experiments on an uneven and inclined aluminum plate were performed. After stable walking on the aluminum plate was confirmed, a forward walking experiment on an uneven room floor was carried out. Successful forward walking was realized, and the effectiveness of the proposed walking algorithm was thereby verified. As future research,

it is necessary to develop a more human-like sole that is ground shape adaptive and ground reaction force absorptive. In the case of a human sole, the thick and soft skin absorbs landing impact and adapts to uneven ground. The authors' biped humanoid robots, KHR-2, HUBO, and Albert HUBO have hard and flat soles with four rubber supporters. If the floor's curvature is large, one rubber supporter among the four rubber supporters may not be in contact with the floor, and consequently the foot landing becomes unstable. In addition, structural vibrations of the sole are generated because it is hard and has little damping property. From this point of view, it is necessary to attach

damping systems that have a good force of restitution on the hard aluminum sole.

References

[1] Kato I. Pneumatically powered artificial legs walking automatically under various circumstances. Proceedings of the 4th International Symposium in External Control of Human Extremities; Dubrovnik, Yugoslav Committee for Electronics and Automation; 1972. p. 458–470.

[2] Cham, Jorge G.; Bailey, Sean A.; Clark, Jonathan E.; Full, Robert J. & Cutkosky, Mark R.. (2002), Fast and Robust: Hexapedal Robots via Shape Deposition Manufacturing, International Journal of Robotics Research, 21(10 – 11), 869 – 882.

[3] Ahmadi, M. & M. Buehler (1999), The ARL Monopod II Running Robot: Control and Energetics, Proceedings of the 1999 IEEE Int. Conf. on Robotics and Automation, Detroit, Michigan, USA, pp. 1689 – 1694.

[4] Antonio Greco, Antonio Roberto, Alessia Saggese, Mario Vento, "Efficient Transformers for on-robot Natural Language Understanding", *2022 IEEE-RAS 21st International Conference on Humanoid Robots (Humanoids)*, pp.823-828, 2022

[5] K. Nagasaka, H. Inoue, M. Inaba. : Dynamic walking pattern generation for a humanoid robot based on optimal gradient method. Paper presented at the IEEE international conference on systems, man, and cybernetics, 12-15 Oct. 1999

Appendix

Code Attachments

- **Programming basics**

Now we'll discuss about the programming techniques of Arduino sketch in the Arduino IDE. There are two main parts every sketch will always have, they are:

• void setup () • void loop ()

1. Void setup();

This is the first routine that begins when the Arduino starts functioning. This function is executed only once throughout the entire program functioning. The setup function contains the initialization of every pin we intend use in our project for input or output. Here is an example of how it should be written:

```
void setup()
{
pinMode(pin, INPUT);
pinMode(pin, OUTPUT);
}
```

Here the pin is the no. of the pin that is to be defined. INPUT / OUTPUT correspond to the mode in which the pin is to be used.

```
Void setup()
{
Serial.begin(9600);
}
```

It also contains the initialization of the Serial Monitor. A serial monitor is used to know the data that are being sent serially to any peripheral device.

Before using any variables for programming it is necessary to define them above the function “void setup()”.

1. **void loop();**

   This function is the next important function in the Sketch. It consists of that part of the code that needs to be continuously executed unlike the part of the code written in the setup function.

```
#include <Servo.h>
#include <LiquidCrystal.h>

Servo hipL;
Servo hipR;
Servo kneeL;
Servo kneeR;
Servo ankleL;
Servo ankleR;

#define hipLOffset 105
#define kneeLOffset 155
#define ankleLOffset 85
#define hipROffset 82
#define kneeROffset 25
#define ankleROffset 85

#define l1 5
#define l2 5.7

#define stepClearance 1
#define stepHeight 10

const int rs =13, en=14, d4=12, d5=11, d6=10, d7=3;
LiquidCrystal lcd(rs,en,d4, d5, d6, d7);
```

```
void updateServoPos(int target1, int target2, int target3, char leg){
  if (leg == 'l'){
    hipL.write(hipLOffset - target1);
    kneeL.write(kneeLOffset - target2);
    ankleL.write(2*ankleLOffset - target3);
  }
  else if (leg == 'r'){
    hipR.write(hipROffset + target1);
    kneeR.write(kneeROffset + target2);
    ankleR.write(target3);
  }
}

void pos(float x, float z, char leg){
  float hipRad2 = atan(x/z);
  float hipDeg2 = hipRad2 * (180/PI);

  float z2 = z/cos(hipRad2);

  float hipRad1 = acos((sq(l1) + sq(z2) - sq(l2))/(2*l1*z2));
  float hipDeg1 = hipRad1 * (180/PI);

  float kneeRad = PI - acos((sq(l1) + sq(l2) - sq(z2))/(2*l1*l2));

  float ankleRad = PI/2 + hipRad2 - acos((sq(l2) + sq(z2) - sq(l1))/(2*l2*z2));
```

```
  float hipDeg = hipDeg1 + hipDeg2;
  float kneeDeg = kneeRad * (180/PI);
  float ankleDeg = ankleRad * (180/PI);

//  Serial.print(hipDeg);
//  Serial.print("\t");
//  Serial.print(kneeDeg);
//  Serial.print("\t");
//  Serial.println(ankleDeg);

  updateServoPos(hipDeg, kneeDeg, ankleDeg, leg);
}

void takeStep(float stepLength, int stepVelocity){
  for (float i = stepLength; i >= -stepLength; i-=0.5){
    pos(i, stepHeight, 'r');
    pos(-i, stepHeight - stepClearance, 'l');
    delay(stepVelocity);
  }

  for (float i = stepLength; i >= -stepLength; i-=0.5){
    pos(-i, stepHeight - stepClearance, 'r');
    pos(i, stepHeight, 'l');
    delay(stepVelocity);
  }
}
```

```
void initialize(){
  for (float i = 10.7; i >= stepHeight; i-=0.1){
    pos(0, i, 'l');
    pos(0, i, 'r');
  }
}

void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
  hipL.attach(9);
  hipR.attach(8);
  kneeL.attach(7);
  kneeR.attach(6);
  ankleL.attach(5);
  ankleR.attach(4);

  hipL.write(hipLOffset);
  kneeL.write(kneeLOffset);
  ankleL.write(ankleLOffset);

  hipR.write(hipROffset);
  kneeR.write(kneeROffset);
  ankleR.write(ankleROffset);
```

```
  lcd.begin(5,2);    //setup the lcd's number of column and rows:
  lcd.print("food");

  lcd.begin(10,1);
  lcd.print("medicine");

  lcd.begin(6,1);
  lcd.print("water");


  delay(5000);

  initialize();
}

void loop() {
  // put your main code here, to run repeatedly:
  takeStep(2, 0);

  lcd.noDisplay();
  delay(500);

  lcd.display();
  delay(500);
}
```