A group of men holding a robot

Description automatically generated

Abstract

Proyek ini berfokus pada pengembangan gerakan robot humanoid, dengan desain fisik  
robot yang telah tersedia. Tim teknisi kami bertugas merencanakan, mengintegrasikan   
komponen perangkat keras, serta mengembangkan kode pengendali gerakan. Dengan   
fokus pada pengembangan gerakan yang beragam, kami berhasil mengimplementasikan   
berbagai gerakan yang telah ditentukan pada robot humanoid yang sudah ada.   
Dokumentasi proyek ini mencakup langkah-langkah yang kami ambil dalam mencapai   
solusi desain gerakan robot yang sukses. Proyek ini merupakan kontribusi yang berharga   
dalam pengembangan teknologi robotika.

Humanoid Robot

Akmal Nurhidayat (4121600041)

akmalnurhidayat28@me.student.pens.ac.id

Artaka Sunu Adhi Prasetya (4121600057)

artakasunuap@me.student.pens.ac.id

Fahri Alia Murthadho (4121600059)

fahriali@me.student.pens.ac.id

Dalam proyek pengembangan robot humanoid dan antarmuka pengendali GUI kami, tim teknisi kami mulai dengan analisis awal untuk menentukan tujuan dan kebutuhan perangkat keras serta perangkat lunak. Kami merancang struktur robot, memilih komponen yang sesuai, dan mengembangkan perangkat lunak untuk mengendalikan gerakan robot melalui GUI. Setelah integrasi perangkat keras dan lunak, kami melakukan pengujian dan debugging, mengoptimalkan kinerja robot dan antarmuka GUI. Hasilnya adalah solusi desain akhir yang berhasil menghasilkan robot humanoid yang dapat dikendalikan dengan sukses melalui GUI, dengan dokumentasi yang mencakup semua langkah yang diperlukan untuk membangun sistem ini.

Table of Contents

[1 Introduction and Initial Analysis 3](#_Toc149728741)

[1.1 Project Context 3](#_Toc149728742)

[1.2 Initial Thought Process 3](#_Toc149728743)

[2 Requirement Analysis and Specification 4](#_Toc149728744)

[2.1 User Requirements 4](#_Toc149728745)

[2.2 System Requirements 4](#_Toc149728746)

[2.3 Tools and Technologies 4](#_Toc149728747)

[3 Conceptual Design 6](#_Toc149728748)

[3.1 System Architecture 6](#_Toc149728749)

[3.2 Interface Design 6](#_Toc149728750)

[3.3 Control Algorithm Design 7](#_Toc149728751)

[4 Detailed Design and Development 8](#_Toc149728752)

[4.1 Component Design 8](#_Toc149728753)

[4.2 Coding and Implementation 12](#_Toc149728754)

[4.3 Integration 16](#_Toc149728755)

[4.4 Unique Features 22](#_Toc149728756)

[5 Testing, Evaluation, and Optimization 22](#_Toc149728757)

[5.1 Testing Strategy 22](#_Toc149728758)

[5.2 Performance Evaluation 22](#_Toc149728759)

[5.3 Optimization 22](#_Toc149728760)

[6 Collaboration and Project Management 23](#_Toc149728761)

[6.1 Teamwork Dynamics 23](#_Toc149728762)

[6.2 Project Management 24](#_Toc149728763)

[7 Conclusion and Reflection 24](#_Toc149728764)

[7.1 Project Summary 24](#_Toc149728765)

[7.2 Future Work 24](#_Toc149728766)

[7.3 Personal and Group Reflections 24](#_Toc149728767)

[8 Appendices 25](#_Toc149728768)

[8.1 Bill of Materials 25](#_Toc149728769)

[8.2 Electrical Wiring and System Layout 25](#_Toc149728770)

[8.3 Code Repository 26](#_Toc149728771)

[8.4 Additional Documentation 27](#_Toc149728772)

[9 References 27](#_Toc149728773)

# 1 Introduction and Initial Analysis

## 1.1 Pendahuluan

Robot humanoid adalah robot dengan tampilan keseluruhannya menyerupai dengan tubuh manusia yang mampu berinteraksi secara sosial. Robot humanoid memiliki tingkat kesulitan yang lebih tinggi dibanding robot lainnya dikarenakan setiap pergerakan robot humanoid ditentukan dari pergerakan sudut motor servo yang di programkan. Untuk berjalan robot humanoid tidak bisa menggerakan kaki secara spontan seperti manusia tetapi apabila robot ingin menggerakkan kaki kanan maka salah satu motor servo di sebelah kaki kiri harus terlebih dahulu dimiringkan begitu juga sebaliknya jika ingin menggerakkan kaki sebelah kiri maka salah satu motor servo sebelah kanan harus terlebih dahulu dimiringkan untuk mendapatkan keseimbangan. Robot humanoid saat ini banyak dibuat untuk robot pelayan atau peniru manusia, misalnya robot menyambut tamu di restoran.

Graphical user interface (GUI) adalah user interface yang menjadi gerbang komunikasi pengguna dengan perangkat. Pasalnya tipe ini merupakan standar dari pengembangan teknologi di masa kini. GUI merupakan sistem komponen yang di develop untuk berbagai perangkat komputer berbasis visual interaktif. Karena kehadirannya lah user bisa mengetahui apa saja yang telah diinput tidak hanya itu saja yang bisa diperoleh oleh user. Akan terlihat responnya secara visual. Antara lain dari berbagai perubahan warna, visibilitas, ukuran, dan yang lainnya. Penemuan teknologi ini merupakan solusi yang jitu bagi pengguna yang sebagian besar bukan developer program. Sebab dapat memberikan kemudahan penggunaan perangkat teknologi yang digunakan sehari-hari.

Adapun proyek ini di fokuskan dalam mempelajari pembuatan Graphical user interface (GUI) yang akan disambungkan dengan robot humanoid untuk menampilkan sebuah gerakan yang disajikan pada layar monitor.

* + 1. Rumusan Masalah

1. Bagaimana robot humanoid dapat berjalan dengan baik
2. Bagaimana robot humanoid dapat menampilkan gerakan yang diatur dengan *Graphical User Interface (GUI)*
3. Bagaimana servo robot humanoid dapat digerakkan menggunakan *Graphical User Interface (GUI)*
   * 1. Tujuan
4. Untuk mengetahui cara penggunaan *Graphical User Interface (GUI)*
5. Untuk membuat gerakan pada robot humanoid

## 1.2 Initial Thought Process

Pada proses brainstorming awal mencari referensi terlebih dahulu melalui internet terkait gerakan-gerakan robot humanoid, yang dilanjut dengan mencoba menggunakan aplikasi Open CM untuk set up ID servo pada robot. Setelah set up ID robot dilanjut dengan melanjutkan mencoba program untuk berjalan pada robot, dilanjut untuk membuat gerakan menarik pada robot. Yang akhirnya kami membuat keputusan untuk gerakan berjalan, gerakan bebas menarik, dan gerakan bebas yang nantinya akan disambungkan dengan GUI yang akan ditampilkan pada layar monitor.

Adapun tantangan dalam proyek ini ialah:

1. Baru pertama kali menggunakan robot humanoid
2. Baru pertama kali menggunakan Graphical User Interface (GUI)

Peluang dalam proyek ini ialah:

1. Salah satu dari anggota team sudah berpengalaman dalam pemrograman robot
2. Terdapat device yang mumpuni dalam proyek kali ini

# 2 Requirement Analysis and Specification

## 2.1 User Requirements

Robot ini akan digunakan dan dioperasikan oleh mahasiswa sebagai modul pembelajaran dalam mata kuliah pemrograman sistem mekatronik. GUI nantinya akan dimasukkan data terlebih dahulu oleh pengguna, GUI pun akan menampilkan informasi yang dibutuhkan oleh pengguna, serta nantinya akan mengontrol servo dari robot humanoid.

## 2.2 System Requirements

Spesifikasi robot humanoid:

* Sensor IMU
* OpenCM robotis
* Dynamixel Ax 12 A

Koneksivitas robot dengan GUI akan menggunakan sambungan koneksi kabel usb type B dan didukung dengan power 220 V melalui power adaptor.

Untuk pergerakan robot humanoid ini nantinya dapat menampilkan berjalan, menampilkan gerakan bebas, menampilkan opsi untuk gerakan menarik yang dibuat. Yang nantinya gerakan-gerakan ini ditampilkan pada GUI untuk mendapatkan nilai servo, tombol untuk menentukan gerakan menarik, dan slider untuk menggerakan servo-servo dari robot humanoid.

## 2.3 Tools and Technologies

* Robot humanoid dengan spesifikasi:

1. Sensor IMU
2. OpenCM robotis
3. Dynamixel Ax 12A

* Aplikasi Qt Designer
* Aplikasi OpenCM
* Aplikasi Arduino IDE

Tabel 1. Aplikasi yang Dibutuhkan.

|  |  |  |
| --- | --- | --- |
| A cartoon of a grey cat  Description automatically generated  Aplikasi OpenCM | Aplikasi Arduino IDE | Aplikasi Qt Designer |
| Dynamixel Ax 12A | OpenCM | Sensor IMU |

## 2.4 Target specification

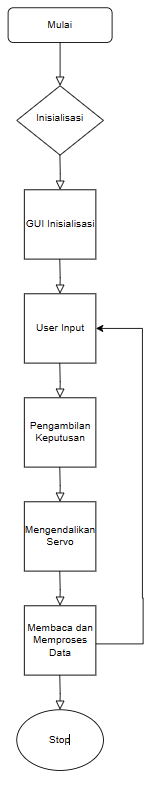
Tabel 2. Target Spesification

|  |  |  |  |
| --- | --- | --- | --- |
| Feature | Description | Measurement Metric | Target Value |
| Example | Detail of the feature | Units/Scale/Methodology | Specified Value |
| Gerakan Pose Dadah | Robot humanoid bergerak untuk dadah dengan tangannya | Sudut rotasi tangan dari posisi awal | Menampilkan gerakan dadah 5 kali dadah |
| Gerakan Pose Slebew | Robot humanoid bergerak slebew dengan tangan dan kaki sedikit miiring | Sudut rotasi tangan dan kaki dari posisi awal | Menampilkan gerakan slebew dengan kaki yang sedikit miring, badan miring dengan 1 kali gerakan |
| Gerakan Pose Menonjok Kanan dan Kiri | Robot humanoid bergerak untuk menonjok dengan lengan kanan dan kirinya | Sudut rotasi tangan dari posisi awal | Menampilkan gerakan menonjok 1 kali |

# 

# 3 Conceptual Design

## 3.1 System Architecture



## Interface Design

Tabel 3. Tampilan GUI.

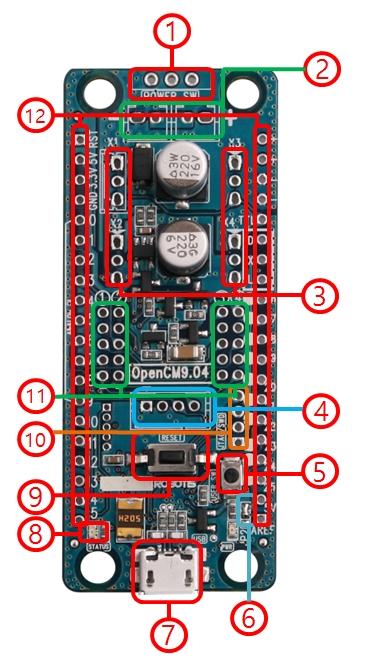
|  |  |
| --- | --- |
| Tampilan GUI | Keterangan |
|  | Gambar disamping merupakan gambar tampilan dari default pada GUI kami |
|  | Untuk gambar disamping merupakan tampilan dari GUI ketika ditekan “BYE” |
|  | Untuk gambar disamping merupakan tampilan dari GUI ketika ditekan “SLEBEW” |
|  | Untuk gambar disamping merupakan tampilan dari GUI ketika ditekan “PUNCH LEFT” |
|  | Untuk gambar disamping merupakan tampilan dari GUI ketika ditekan “PUNCH RIGHT” |

## 3.3 Control Algorithm Design

Pada robot humanoid ini dilengkapi dengan sistem GUI untuk mengontrol dan monitoring dari robot tersebut. Untuk robot ini dimasukkan untuk fitur pose. Dimana nantinya tampilan GUI seperti pada tampilan sub bab 3.2 pada tampilan pose dapat ditekan pada layar untuk menampilkan dari gerakannya. Semua tampilan tersebut terhubung dengan robot dengan mengoneksikan menggunakan *serial communication* yang dapat secara real time terhubung.

# 4 Detailed Design and Development

## 4.1 Component Design

****Robot humanoid ini terdapat komponen antara lain:

Gambar 1. Open CM Board

1. **Open CM**

Berikut penjelasan dari gambar diatas:

* 1. **Power Switch**

Digunakan untuk mengontrol daya yang disuplai ke papan dan menghidupkan/mematikan DYNAMIXEL.

* 1. **Serial Battery Socket**

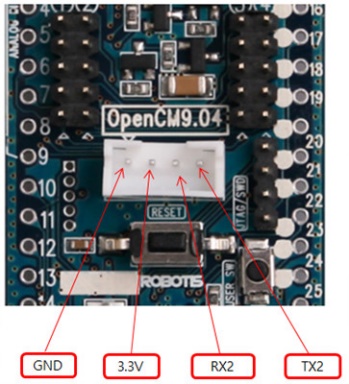
Untuk menghubungkan baterai lithium

* 1. **Dynamixel TL 3 PIN**

Port untuk rangkaia daisy DYNAMIXEL, yang menggunkan kabel 3 pin

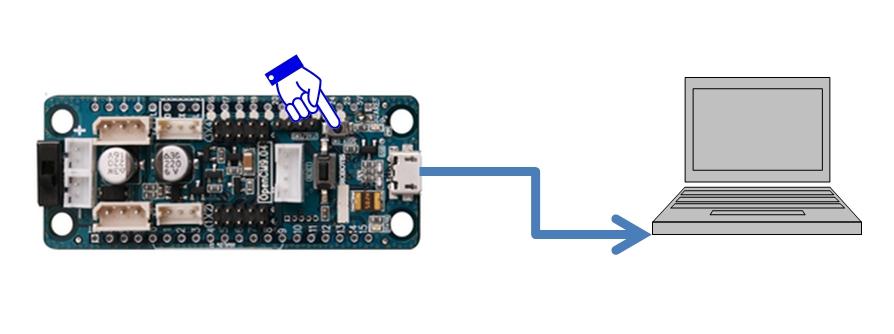
* 1. **Communication Port**

Digunakan dari komunikasi kabel/nirkabel menggunkan perangkat periferal. Port komunikasi 4 pin Open CM9.04 menggunakan serial 2



Gambar 2. Communication Part

* 1. **User Button**

Tombol yang dapat dikontrol / diprogram oleh pengguna. Dapat digunakan sebagai pin 23 atau BOARD\_BUTTON\_PIN. Dapat diinisialisasi di setup() sebagai pinMode(23, INPUT\_PULLDOWN). Jika OpenCM9.04 tidak mendownload program, sambungkan kabel USB sambil menahan "Tombol Pengguna". LED status akan tetap menyala dan pengunduhan akan dimulai.

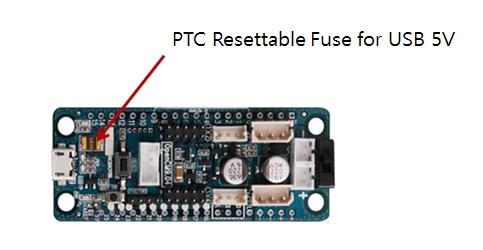
Gambar 3. User Button

* 1. **EXT.ADC Ref Jumper**

Tegangan Referensi Analog dapat dimodifikasi.

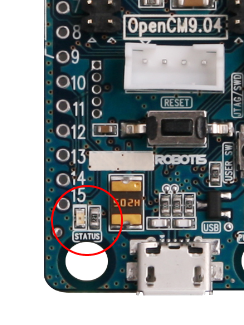
* 1. **Micro-B USB**

Digunakan untuk mengunduh program ke OpenCM9.04 dan digunakan untuk berkomunikasi dengan perangkat lain melalui USB sekaligus menyuplai 5V ke board. Setelah menghubungkan baterai, daya 5V dari USB secara otomatis terputus dan daya disuplai dari baterai. Jika arus yang diambil berlebihan, sekring internal memutus arus yang diambil dari koneksi USB 5V untuk melindungi PC pengguna dari kerusakan.



Gambar 4. Micro USB Port

* 1. **Status LED**

LED digunakan untuk menguji program OpenCM9.04. LED menyala saat Pin 14 atau BOARD\_LED\_PIN TINGGI dan MATI saat LOW. Kontrol PWM dimungkinkan.

Gambar 5. Status LED

* 1. **Reset Button**

Tombol ini digunakan untuk mereset Open CM 9.04

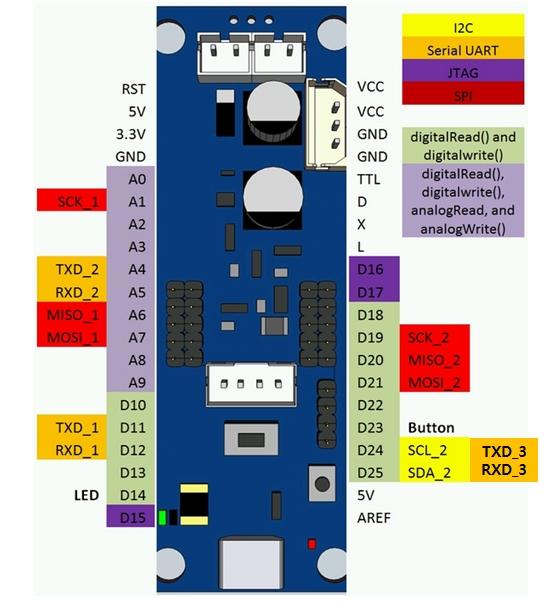
* 1. **JTAG/SWD 4 PIN**

Digunakan untuk menghubungkan ST-LINK atau Debugger/Programmer Dalam sirkuit lainnya.  
Port 4-pin JTAG/SWD dapat digunakan untuk berbagai pengembangan oleh pengguna tingkat lanjut.

* 1. **Robotis 5-Pin Port**

Digunakan untuk menghubungkan perangkat ROBOTIS 5-pin

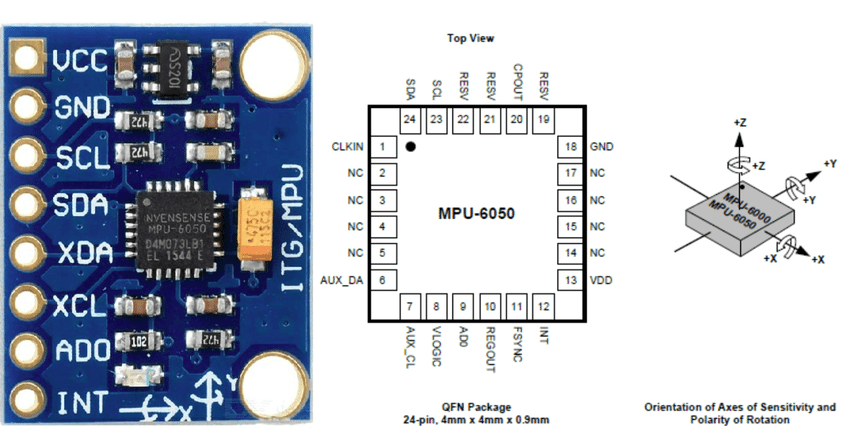
* 1. **I/O Header**

CPU STM32F103CB OpenCM9.04 dapat diakses melalui pin GPIO. 26 pin GPIO (0~25) adalah input/output digital dan beroperasi pada 3.3V. Input digital TINGGI(3.3V) atau RENDAH(0V). Kecepatan sakelar maksimum input digital adalah 18Mhz. (Kecuali pin 22 & 23, yang memiliki kecepatan sakelar maksimum 2Mhz). Pin digital hanya dapat mentolerir hingga 3.3V; tegangan apa pun masukan lebih besar dari 3.3V dapat merusak OpenCM9.04. Pin Toleransi 5V : 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21 Arus maksimum yang dapat ditoleransi oleh pin ini adalah 25mA. (pengecualian: pin 22 & 23 hanya dapat mentolerir 3mA.) Semua pin GPIO OpenCM9.04 dapat “pull-up” atau “pull-down” secara internal dan dapat dimodifikasi melalui perangkat lunak. Resistensi “Pull-up” dan “pull-up” adalah 40KΩ (Khas).

Gambar 6. I/O Header

1. **Sensor IMU**

Sensor MPU-6050 merupakan gabungan dari sensor Accelerometer dan Gyroscope, dimana dengan menggunakan kedua sensor ini, kita dapat menghasilkan output berupa orientasi. Accelerometer digunakan untuk mengetahui percepatan gravitasi, sedangkan gyroscope digunakan untuk mengetahui kecepatan sudut. Fusion dari output accelerometer dan gyroscope akan menghasilkan orientasi (kemiringan terhadap sumbu x dan y).



Gambar 7. Sensor IMU

1. **Dynamixel Ax 12A**

Dynamixel AX 12A merupakan motor servo cerdas yang memiliki torsi hingga 15kgf.cm dan dilengkapi kemampuan networking melalui antarmuka UART TTL half duplex multidrop. Dynamixel AX-12A terdiri dari reduction gear, precision DC Motor, dan rangkaian kontrol yang telah dilengkapi dengan kemampuan komunikasi (networking) dalam satu kemasan.AX-12A merupakan versi terbaru dari AX-12+ dengan kemampuan yang sama tetapi dengan penambahan desain eksternal. Saat ini jenis yang tersedia adalah AX-12A.

## 4.2 Coding and Implementation

|  |
| --- |
| #define DXL\_BUS\_SERIAL1 1 //Dynamixel on Serial1(USART1) <-OpenCM9.04  #define DXL\_BUS\_SERIAL2 2 //Dynamixel on Serial2(USART2) <-LN101,BT210  #define DXL\_BUS\_SERIAL3 3 //Dynamixel on Serial3(USART3) <-OpenCM 485EXP`  #define ID\_NUM 3  char Cmd;  //ID 1 bahu kanan  //ID 2 bahu kiri  //ID 3 lengan atas kanan  //ID 4 lengan atas kiri  //ID 5 lengan bawah kanan  //ID 6 lengan bawah kiri  //ID 9 pinggul kanan  //ID 10 pinggul kiri  //ID 11 paha kanan  //ID 12 paha kiri  //ID 13 lutut kanan  //ID 14 lutut kiri  //ID 15 tumit kanan  //ID 16 tumit kiri  //ID 17 telapak kanan  //ID 18 telapak kiri  Dynamixel Dxl(DXL\_BUS\_SERIAL3);  void setup() {  Dxl.begin(3);  for (int i = 1; i<=20; i++) {  Dxl.jointMode(i);  }  }  void initial(){  delay(500);  Dxl.setPosition(9, 532, 50); //ID 9 pinggul kanan (++ kanan)  Dxl.setPosition(10, 492, 50); //ID 10 pinggul kiri (++ kanan)  Dxl.setPosition(11, 672, 50); //ID 11 paha kanan (++ maju)  Dxl.setPosition(12, 352, 50); //ID 12 paha kiri (-- maju)  Dxl.setPosition(13, 752, 50); //ID 13 lutut kanan (++ mundur), tidak boleh <512  Dxl.setPosition(14, 272, 50); //ID 14 lutut kiri (tidak boleh >512)  Dxl.setPosition(15, 612, 50); //ID 15 tumit kanan (++ maju)  Dxl.setPosition(16, 412, 50); //ID 16 tumit kiri (-- maju)  //===========================================================  Dxl.setPosition(17, 532, 50); //ID 15 telapak kanan (++ miring kiri)  Dxl.setPosition(18, 492, 50); //ID 16 telapak kiri (++ miring kiri)  delay(500);  Dxl.setPosition(1, 512, 50); //ID 1 bahu kanan  Dxl.setPosition(2, 512, 50); //ID 2 bahu kiri  Dxl.setPosition(3, 542, 50); //ID 3 lengan atas kanan  Dxl.setPosition(4, 472, 50); //ID 4 lengan atas kiri  Dxl.setPosition(5, 512, 50); //ID 5 lengan bawah kanan (maks 212)  Dxl.setPosition(6, 512, 50); //ID 6 lengan bawah kiri (maks 812)  }  void slebew(){  delay(500);  Dxl.setPosition(15, 632, 50); //ID 15 tumit kanan (++ maju)  Dxl.setPosition(16, 512, 50); //ID 16 tumit kiri (-- maju)  Dxl.setPosition(9, 532, 50); //ID 9 pinggul kanan (++ kanan)  Dxl.setPosition(10, 512, 50); //ID 10 pinggul kiri (++ kanan)  Dxl.setPosition(11, 692, 50); //ID 11 paha kanan (++ maju)  Dxl.setPosition(12, 512, 50); //ID 12 paha kiri (-- maju)  Dxl.setPosition(13, 752, 50); //ID 13 lutut kanan (++ mundur), tidak boleh <512  Dxl.setPosition(14, 512, 50); //ID 14 lutut kiri (tidak boleh >512)    //===========================================================  Dxl.setPosition(17, 492, 50); //ID 15 telapak kanan (++ miring kiri)  Dxl.setPosition(18, 472, 50); //ID 16 telapak kiri (++ miring kiri)  delay(500);  Dxl.setPosition(1, 512, 50); //ID 1 bahu kanan  Dxl.setPosition(2, 512, 50); //ID 2 bahu kiri  Dxl.setPosition(3, 812, 50); //ID 3 lengan atas kanan  Dxl.setPosition(4, 212, 50); //ID 4 lengan atas kiri  Dxl.setPosition(5, 792, 50); //ID 5 lengan bawah kanan (maks 212)  Dxl.setPosition(6, 792, 50); //ID 6 lengan bawah kiri (maks 812)  }  void haloDek(){  //===================================================  Dxl.setPosition(9, 532, 50); //ID 9 pinggul kanan (++ kanan)  Dxl.setPosition(10, 492, 50); //ID 10 pinggul kiri (++ kanan)  Dxl.setPosition(11, 672, 50); //ID 11 paha kanan (++ maju)  Dxl.setPosition(12, 352, 50); //ID 12 paha kiri (-- maju)  Dxl.setPosition(13, 752, 50); //ID 13 lutut kanan (++ mundur), tidak boleh <512  Dxl.setPosition(14, 272, 50); //ID 14 lutut kiri (tidak boleh >512)  Dxl.setPosition(15, 612, 50); //ID 15 tumit kanan (++ maju)  Dxl.setPosition(16, 412, 50); //ID 16 tumit kiri (-- maju)  //===========================================================  Dxl.setPosition(17, 532, 50); //ID 15 telapak kanan (++ miring kiri)  Dxl.setPosition(18, 492, 50); //ID 16 telapak kiri (++ miring kiri)  delay(500);  //==================================================================  Dxl.setPosition(1, 812, 50); //ID 1 bahu kanan  Dxl.setPosition(2, 212, 50); //ID 2 bahu kiri  Dxl.setPosition(3, 812, 50); //ID 3 lengan atas kanan  Dxl.setPosition(4, 212, 50); //ID 4 lengan atas kiri  // Dxl.setPosition(5, 512, 150); //ID 5 lengan bawah kanan (maks 212)  // Dxl.setPosition(6, 512, 150); //ID 6 lengan bawah kiri (maks 812)  }  void haloDada2(){  for(int i=1; i<=5; i++){  Dxl.setPosition(5, 312, 150); //ID 5 lengan bawah kanan (maks 212)  Dxl.setPosition(6, 712, 150); //ID 6 lengan bawah kiri (maks 812)  delay(1000);  Dxl.setPosition(5, 712, 150); //ID 5 lengan bawah kanan (maks 212)  Dxl.setPosition(6, 312, 150); //ID 6 lengan bawah kiri (maks 812)  delay(1000);  }  }  void kuda2(){  delay(500);  Dxl.setPosition(9, 562, 50); //ID 9 pinggul kanan (++ kanan)  Dxl.setPosition(10, 462, 50); //ID 10 pinggul kiri (++ kanan)  //===========================================================  Dxl.setPosition(11, 672, 50); //ID 11 paha kanan (++ maju)  Dxl.setPosition(12, 352, 50); //ID 12 paha kiri (-- maju)  Dxl.setPosition(13, 752, 50); //ID 13 lutut kanan (++ mundur), tidak boleh <512  Dxl.setPosition(14, 272, 50); //ID 14 lutut kiri (tidak boleh >512)  Dxl.setPosition(15, 612, 50); //ID 15 tumit kanan (++ maju)  Dxl.setPosition(16, 412, 50); //ID 16 tumit kiri (-- maju)  //===========================================================  Dxl.setPosition(17, 562, 50); //ID 15 telapak kanan (++ miring kiri)  Dxl.setPosition(18, 462, 50); //ID 16 telapak kiri (++ miring kiri)  delay(500);  Dxl.setPosition(1, 412, 250); //ID 1 bahu kanan  Dxl.setPosition(2, 612, 50); //ID 2 bahu kiri  Dxl.setPosition(3, 542, 50); //ID 3 lengan atas kanan  Dxl.setPosition(4, 472, 50); //ID 4 lengan atas kiri  Dxl.setPosition(5, 612, 250); //ID 5 lengan bawah kanan (maks 212)  Dxl.setPosition(6, 412, 50); //ID 6 lengan bawah kiri (maks 812)  }  void pukulA(){  // Dxl.setPosition(1, 412, 250); //ID 1 bahu kanan  // Dxl.setPosition(5, 612, 250); //ID 5 lengan bawah kanan (maks 212)  // delay(100);  Dxl.setPosition(1, 812, 250); //ID 1 bahu kanan  Dxl.setPosition(3, 542, 250); //ID 3 lengan atas kanan  Dxl.setPosition(5, 312, 250); //ID 5 lengan bawah kanan (maks 212)  delay(1000);  Dxl.setPosition(1, 412, 250); //ID 1 bahu kanan  Dxl.setPosition(5, 612, 250); //ID 5 lengan bawah kanan (maks 212)  }  void pukulB(){  // Dxl.setPosition(1, 412, 250); //ID 1 bahu kanan  // Dxl.setPosition(5, 612, 250); //ID 5 lengan bawah kanan (maks 212)  // delay(100);  Dxl.setPosition(2, 212, 250); //ID 1 bahu kanan  Dxl.setPosition(4, 472, 50); //ID 4 lengan atas kiri  Dxl.setPosition(6, 812, 250); //ID 5 lengan bawah kanan (maks 212)  delay(1000);  Dxl.setPosition(2, 612, 250); //ID 1 bahu kanan  Dxl.setPosition(6, 412, 250); //ID 5 lengan bawah kanan (maks 212)  }  void loop() {  initial();  //=========================================================  //komunikasi serial//  //=========================================================  while(SerialUSB.available()){  Cmd = SerialUSB.read();  if(Cmd == 'w' || Cmd == 'W'){  haloDek();  delay(3000);  haloDada2();  delay(2000);  }else if(Cmd == 'd' || Cmd == 'D'){  kuda2();  delay(500);  pukulA();  delay(500);  kuda2();  }else if(Cmd == 'a' || Cmd == 'A'){  kuda2();  delay(500);  pukulB();  delay(500);  kuda2();  }else if(Cmd == 's' || Cmd == 'S')  slebew();  delay(2000);  }  //=========================================================  //komunikasi serial//  //=========================================================  } |

## 4.3 Integration

Koneksi GUI dengan Open CM Board

|  |
| --- |
| # main.py  import sys  from PyQt5.QtWidgets import QApplication, QMainWindow  from PyQt5.QtCore import QTimer  from humanoidrobot2 import Ui\_MainWindow # Import the UI class  import serial  class MainWindow(QMainWindow, Ui\_MainWindow):  def \_init\_(self):  super(MainWindow, self).\_init\_()  self.setupUi(self)  # Set up serial communication with Arduino  self.arduino = serial.Serial('COM7', 9600) # Change 'COM3' to the appropriate port  # Connect button signals to Arduino functions  self.byeButton.clicked.connect(self.send\_bye\_command)  self.punchKiriButton.clicked.connect(self.send\_punch\_command)  self.punchKananButton.clicked.connect(self.send\_punch2\_command)  self.slebewButton.clicked.connect(self.send\_custom\_command)  # Add connections for other buttons if needed  # Set up a timer to periodically check for Arduino messages  self.timer = QTimer(self)  self.timer.timeout.connect(self.read\_serial\_data)  self.timer.start(100) # Adjust the interval based on your needs  def send\_command(self, command):  self.arduino.write(command.encode('utf-8'))  def send\_bye\_command(self):  self.send\_command("w")  def send\_punch\_command(self):  self.send\_command("a")  def send\_punch2\_command(self):  self.send\_command("d")  def send\_custom\_command(self):  self.send\_command("s")    def read\_serial\_data(self):  if self.arduino.in\_waiting > 0:  arduino\_data = self.arduino.readline().decode('utf-8').strip()  # Process Arduino data as needed  print("Arduino:", arduino\_data)  if \_name\_ == "\_main\_":  app = QApplication(sys.argv)  window = MainWindow()  window.show()  sys.exit(app.exec\_()) |

Tampilan pada GUI

|  |
| --- |
| # -\*- coding: utf-8 -\*-  # Form implementation generated from reading ui file 'humanoidrobot.ui'  #  # Created by: PyQt5 UI code generator 5.15.10  #  # WARNING: Any manual changes made to this file will be lost when pyuic5 is  # run again. Do not edit this file unless you know what you are doing.  from PyQt5 import QtCore, QtGui, QtWidgets  class Ui\_MainWindow(object):  def setupUi(self, MainWindow):  MainWindow.setObjectName("MainWindow")  MainWindow.resize(1140, 674)  font = QtGui.QFont()  font.setFamily("Monotxt")  MainWindow.setFont(font)  MainWindow.setStyleSheet("background-color: rgb(99, 198, 200);")  self.centralwidget = QtWidgets.QWidget(MainWindow)  self.centralwidget.setObjectName("centralwidget")  self.frame\_2 = QtWidgets.QFrame(self.centralwidget)  self.frame\_2.setGeometry(QtCore.QRect(140, 240, 311, 291))  font = QtGui.QFont()  font.setFamily("Monotxt")  font.setPointSize(12)  self.frame\_2.setFont(font)  self.frame\_2.setFrameShape(QtWidgets.QFrame.StyledPanel)  self.frame\_2.setFrameShadow(QtWidgets.QFrame.Raised)  self.frame\_2.setObjectName("frame\_2")  self.byeButton = QtWidgets.QPushButton(self.frame\_2)  self.byeButton.setGeometry(QtCore.QRect(40, 10, 91, 91))  self.byeButton.setStyleSheet("background-color: rgb(0, 0, 255);")  icon = QtGui.QIcon()  icon.addPixmap(QtGui.QPixmap("wave.png"), QtGui.QIcon.Normal, QtGui.QIcon.Off)  icon.addPixmap(QtGui.QPixmap("wave.png"), QtGui.QIcon.Normal, QtGui.QIcon.On)  self.byeButton.setIcon(icon)  self.byeButton.setIconSize(QtCore.QSize(75, 75))  self.byeButton.setObjectName("byeButton")  self.slebewButton = QtWidgets.QPushButton(self.frame\_2)  self.slebewButton.setGeometry(QtCore.QRect(170, 10, 91, 91))  self.slebewButton.setStyleSheet("background-color: rgb(0, 255, 106);")  self.slebewButton.setText("")  icon1 = QtGui.QIcon()  icon1.addPixmap(QtGui.QPixmap("haluan.png"), QtGui.QIcon.Normal, QtGui.QIcon.Off)  self.slebewButton.setIcon(icon1)  self.slebewButton.setIconSize(QtCore.QSize(75, 75))  self.slebewButton.setObjectName("slebewButton")  self.punchKiriButton = QtWidgets.QPushButton(self.frame\_2)  self.punchKiriButton.setGeometry(QtCore.QRect(40, 116, 91, 91))  self.punchKiriButton.setStyleSheet("background-color: rgb(255, 0, 0);")  self.punchKiriButton.setText("")  icon2 = QtGui.QIcon()  icon2.addPixmap(QtGui.QPixmap("punch.png"), QtGui.QIcon.Normal, QtGui.QIcon.Off)  icon2.addPixmap(QtGui.QPixmap("punch.png"), QtGui.QIcon.Normal, QtGui.QIcon.On)  self.punchKiriButton.setIcon(icon2)  self.punchKiriButton.setIconSize(QtCore.QSize(75, 75))  self.punchKiriButton.setObjectName("punchKiriButton")  self.punchKananButton = QtWidgets.QPushButton(self.frame\_2)  self.punchKananButton.setGeometry(QtCore.QRect(170, 117, 91, 91))  self.punchKananButton.setStyleSheet("background-color: rgb(255, 255, 0);")  self.punchKananButton.setText("")  icon3 = QtGui.QIcon()  icon3.addPixmap(QtGui.QPixmap("punchKanan.png"), QtGui.QIcon.Normal, QtGui.QIcon.Off)  self.punchKananButton.setIcon(icon3)  self.punchKananButton.setIconSize(QtCore.QSize(75, 75))  self.punchKananButton.setObjectName("punchKananButton")  self.label\_8 = QtWidgets.QLabel(self.frame\_2)  self.label\_8.setGeometry(QtCore.QRect(40, 213, 90, 41))  font = QtGui.QFont()  font.setFamily("Monotxt\_IV50")  font.setPointSize(12)  self.label\_8.setFont(font)  self.label\_8.setObjectName("label\_8")  self.label\_9 = QtWidgets.QLabel(self.frame\_2)  self.label\_9.setGeometry(QtCore.QRect(160, 213, 110, 41))  font = QtGui.QFont()  font.setFamily("Monotxt\_IV50")  font.setPointSize(12)  self.label\_9.setFont(font)  self.label\_9.setObjectName("label\_9")  self.label = QtWidgets.QLabel(self.centralwidget)  self.label.setGeometry(QtCore.QRect(350, 3, 431, 61))  font = QtGui.QFont()  font.setFamily("Poor Richard")  font.setPointSize(18)  self.label.setFont(font)  self.label.setStyleSheet("background-color: qradialgradient(spread:pad, cx:0.5, cy:0.5, radius:0.5, fx:0.5, fy:0.5, stop:0 rgba(255, 255, 255, 255), stop:0.1 rgba(255, 255, 255, 255), stop:0.2 rgba(255, 176, 176, 167), stop:0.3 rgba(255, 151, 151, 92), stop:0.4 rgba(255, 125, 125, 51), stop:0.5 rgba(255, 76, 76, 205), stop:0.52 rgba(255, 76, 76, 205), stop:0.6 rgba(255, 180, 180, 84), stop:1 rgba(255, 255, 255, 0));")  self.label.setObjectName("label")  self.label\_4 = QtWidgets.QLabel(self.centralwidget)  self.label\_4.setGeometry(QtCore.QRect(168, 153, 240, 51))  font = QtGui.QFont()  font.setFamily("Monotxt\_IV50")  font.setPointSize(12)  self.label\_4.setFont(font)  self.label\_4.setStyleSheet("background-color: rgb(255, 255, 255);")  self.label\_4.setObjectName("label\_4")  self.label\_5 = QtWidgets.QLabel(self.centralwidget)  self.label\_5.setGeometry(QtCore.QRect(-50, 650, 261, 61))  font = QtGui.QFont()  font.setFamily("Monotxt\_IV50")  font.setPointSize(12)  self.label\_5.setFont(font)  self.label\_5.setObjectName("label\_5")  self.label\_7 = QtWidgets.QLabel(self.centralwidget)  self.label\_7.setGeometry(QtCore.QRect(310, 227, 91, 21))  font = QtGui.QFont()  font.setFamily("Monotxt\_IV50")  font.setPointSize(12)  self.label\_7.setFont(font)  self.label\_7.setObjectName("label\_7")  self.label\_6 = QtWidgets.QLabel(self.centralwidget)  self.label\_6.setGeometry(QtCore.QRect(180, 227, 90, 20))  font = QtGui.QFont()  font.setFamily("Monotxt\_IV50")  font.setPointSize(12)  self.label\_6.setFont(font)  self.label\_6.setObjectName("label\_6")  self.Gambar1 = QtWidgets.QLabel(self.centralwidget)  self.Gambar1.setGeometry(QtCore.QRect(640, 79, 460, 531))  self.Gambar1.setText("")  self.Gambar1.setPixmap(QtGui.QPixmap("humanoid.png"))  self.Gambar1.setScaledContents(True)  self.Gambar1.setObjectName("Gambar1")  self.label\_3 = QtWidgets.QLabel(self.centralwidget)  self.label\_3.setGeometry(QtCore.QRect(10, 20, 81, 71))  self.label\_3.setText("")  self.label\_3.setPixmap(QtGui.QPixmap("robot (2).png"))  self.label\_3.setScaledContents(True)  self.label\_3.setObjectName("label\_3")  self.label\_11 = QtWidgets.QLabel(self.centralwidget)  self.label\_11.setGeometry(QtCore.QRect(10, 590, 171, 61))  font = QtGui.QFont()  font.setFamily("Monotxt\_IV50")  font.setPointSize(12)  self.label\_11.setFont(font)  self.label\_11.setObjectName("label\_11")  self.frame\_2.raise\_()  self.label.raise\_()  self.label\_4.raise\_()  self.label\_5.raise\_()  self.label\_6.raise\_()  self.label\_7.raise\_()  self.Gambar1.raise\_()  self.label\_3.raise\_()  self.label\_11.raise\_()  MainWindow.setCentralWidget(self.centralwidget)  self.statusbar = QtWidgets.QStatusBar(MainWindow)  self.statusbar.setObjectName("statusbar")  MainWindow.setStatusBar(self.statusbar)  self.retranslateUi(MainWindow)  QtCore.QMetaObject.connectSlotsByName(MainWindow)  self.byeButton.clicked.connect(self.show\_bye)  self.slebewButton.clicked.connect(self.show\_slebew)  self.punchKiriButton.clicked.connect(self.show\_pukulkiri)  self.punchKananButton.clicked.connect(self.show\_pukulkanan)  def show\_humanoid(self):  self.Gambar1.setPixmap(QtGui.QPixmap("humanoid.png"))  def show\_bye(self):  self.Gambar1.setPixmap(QtGui.QPixmap("dada-dada.png"))  def show\_slebew(self):  self.Gambar1.setPixmap(QtGui.QPixmap("slebew.png"))  def show\_pukulkiri(self):  self.Gambar1.setPixmap(QtGui.QPixmap("pukul kiri.png"))  def show\_pukulkanan(self):  self.Gambar1.setPixmap(QtGui.QPixmap("pukul kanan.png"))  def retranslateUi(self, MainWindow):  \_translate = QtCore.QCoreApplication.translate  MainWindow.setWindowTitle(\_translate("MainWindow", "MainWindow"))  self.byeButton.setText(\_translate("MainWindow", " "))  self.label\_8.setText(\_translate("MainWindow", "<html><head/><body><p align=\"center\"><span style=\" font-size:14pt; font-weight:696;\">PUNCH<br/>LEFT</span></p></body></html>"))  self.label\_9.setText(\_translate("MainWindow", "<html><head/><body><p align=\"center\"><span style=\" font-size:14pt; font-weight:696;\">PUNCH<br/>RIGHT</span></p></body></html>"))  self.label.setText(\_translate("MainWindow", "<html><head/><body><p align=\"center\"><span style=\" font-weight:700; color:#ffffff;\">HUMANOID ROBOT</span></p><p align=\"center\"><span style=\" font-weight:700; color:#ffffff;\">GRAPHICAL USER INTERFACE</span></p></body></html>"))  self.label\_4.setText(\_translate("MainWindow", "<html><head/><body><p align=\"center\"><span style=\" font-size:14pt; font-weight:700;\">CONTROL<br/>HUMANOID ROBOT</span></p></body></html>"))  self.label\_5.setText(\_translate("MainWindow", "<html><head/><body><p align=\"center\"><span style=\" font-size:14pt; font-weight:700;\">YOU MOVE <br/>WE CTRL</span></p></body></html>"))  self.label\_7.setText(\_translate("MainWindow", "<html><head/><body><p align=\"center\"><span style=\" font-size:14pt; font-weight:696;\">SLEBEW</span></p></body></html>"))  self.label\_6.setText(\_translate("MainWindow", "<html><head/><body><p align=\"center\"><span style=\" font-size:14pt; font-weight:700;\">BYE</span></p></body></html>"))  self.label\_11.setText(\_translate("MainWindow", "<html><head/><body><p align=\"center\"><span style=\" font-size:16pt; font-weight:700;\">YOU MOVE<br/>WE CONTROL</span></p></body></html>"))  if \_name\_ == "\_main\_":  import sys  app = QtWidgets.QApplication(sys.argv)  MainWindow = QtWidgets.QMainWindow()  ui = Ui\_MainWindow()  ui.setupUi(MainWindow)  MainWindow.show()  sys.exit(app.exec\_()) |

## 4.4 Unique Features

Dalam robot humanoid ini terdapat beberapa teknologi baru yang dicoba diimplementasikan ialah pemasangan GUI pada robot humanoid bertujuan untuk memudahkan pengguna dalam monitoring dan mengontrol robot humanoid. Dengan beberapa fitur yang terpasang robot ini dapat di kontrol oleh GUI.

# 5 Testing, Evaluation, and Optimization

## 5.1 Testing Strategy

Karena pada robot humanoid menggunakan dynamixel AX 12A kami menggunakan metodelogi *trial and error*  dalam pengerjaan proyek ini. Karena dengan menggunakan metode tersebut dapat memahami kerja dari dynamixel secara efektif serta mengetahui perubahan parameter dan bagaimana parameter-parameter tersebut mempengaruhi gerakan atau kinerja motor dalam konteks aplikasi yang diinginkan.

Tetapi dengan menggunakan metode *trial and error* ini dalam pengerjaan menjadi lama karena harus satu per satu penyetingan dari dyanamixelnya. Jadi dalam pencapaian target yang terdapat pada tabel 2.4 emakan waktu lama dalam penentuan gerakan pose yang diinginkan. Dan seringkali pada robot humanoid terjadi sesuatu hal yang tidak dinginkan seperti dynamixel yang tidak bergerak yang dapat menghambat pengerjaan.

Proyek kali ini kami menggunakan aplikasi open CM dalam penentuan dari dynamixelnya dan menentukan sudut yang tepat pada saat pertama kali. Berikutnya kami menggunakan aplikasi arduino IDE dalam pemrograman robot dan menggunakan aplikasi pyqt dalam membuat GUI untuk membuat tombol serta slider yang diperlukan sesuai dengan target yang sudah ditentukan.

## 5.2 Performance Evaluation

Setelah menyelesaikan pemrograman gerakan pada robot humanoid kami, kami tidak puas begitu saja. Kami secara berkesinambungan melakukan pemeriksaan dan uji coba berulang untuk memastikan apakah program yang kami buat berjalan secara optimal dan konsisten. Tidak hanya memberikan hasil satu kali saja. Sebagai contoh, saat kami mengembangkan program untuk gerakan pertama, kami menciptakan program yang memungkinkan robot untuk berdiri tegak. Namun, kami menemukan kelemahan di mana robot tersebut tidak stabil dan rentan roboh saat menerima sentuhan dari luar dan pada saat berjalan. Oleh karena itu, kami melakukan perubahan dalam membuat program menjadi gerakan pose menarik seperti melambai, slebew dan menonjok dengan kuda-kuda saat berdiri yang tujuannya supaya robot tidak gampang rubuh saat terkena sentuhan dari luar. Tujuan kami adalah mengubah gerakan yang awalnya berjalan menjadi gerakan pose karena robot yang tidak memungkinkan dalam prakteknya dan kami membuat ide menampilkan gerakan lainnya. Proses modifikasi ini merupakan bagian dari upaya kami untuk terus meningkatkan kualitas gerakan robot, memastikan bahwa robot mampu bertahan dan beroperasi secara baik dan kokoh dalam berbagai situasi dan kondisi.

## 5.3 Optimization

* Kinerja Sistem: Memastikan bahwa seluruh sistem robot humanoid berjalan dengan lancar, meminimalkan bug atau kesalahan, dan memperbaiki respons sistem terhadap perintah-perintah yang diberikan.
* Kualitas Gerakan: Memperbaiki kualitas gerakan robot, termasuk kecepatan, ketepatan, dan keselarasan dalam melakukan gerakan yang diinginkan. Pengoptimalan dilakukan dengan melakukan modifikasi pada program, perangkat keras, atau algoritma kontrol untuk mencapai gerakan yang lebih realistis dan terkoordinasi.
* Stabilitas dan Keandalan: Mengatasi masalah kestabilan, khususnya dalam situasi dimana robot rentan roboh atau tidak dapat menjaga keseimbangan dengan baik Modifikasi dilakukan untuk meningkatkan kemampuan robot dalam beradaptasi dengan perubahan lingkungan atau interaksi dengan luar.

• Pengalaman Pengguna: Pengoptimalan tidak hanya terfokus pada kinerja teknis, tetapi juga pada pengalaman pengguna. Ini meliputi penyesuaian pada antarmuka pengguna, pemahaman perilaku robot, serta upaya untuk memastikan interaksi antara manusia dan robot berjalan dengan mulus dan intuitif.

Dengan melakukan pengoptimalan secara terus-menerus, pengembang robot humanoid bertujuan untuk meningkatkan kemampuan robot dalam menjalankan tugasnya dengan lebih baik, lebih aman, dan lebih responsif terhadap lingkungan sekitarnya. Tujuan akhirnya adalah memberikan pengalaman pengguna yang lebih baik dan mengoptimalkan manfaat dari penggunaan robot humanoid di berbagai bidang aplikasi

# 6 Collaboration and Project Management

## 6.1 Teamwork Dynamics

Dalam dinamika kerja tim mencakup antara lain kolaborasi, peran, serta kontribusi anggota tim dalam mencapai tujuan bersama. Dalam pengembangan robot humanoid, dinamika kerja tim yang baik sangat penting. Ini mencakup:

Kolaborasi: Dalam pengembangan robot humanoid, kerja tim yang dinamis mencakup kolaborasi, penentuan peran, dan kontribusi anggota tim untuk mencapai tujuan bersama. Kolaborasi dimulai dari berbagi ide, penyamaan ide, serta keterampilan di antara anggota tim dengan spesialisasi yang beragam, seperti perangkat lunak, perangkat keras, mekanika, dan pemrograman yang perlu terkoordinasi dalam bekerja.

Peran dan Tanggung Jawab: Setiap anggota tim memiliki peran dan tanggung jawab tertentu, seperti fokus pada pengembangan perangkat lunak atau terlibat dalam perancangan mekanika dan pengujian. Dalam kelompok ini adapun pembagian perannya antara lain:

* Fahri sebagai programmer untuk membuat gerakan dari robot humanoid
* Sunu sebagai pengembangan perangkat lunak yang dipakai seperti GUI yang menggunakan aplikasi pyqt
* Akmal sebagai non teknis dalam pembuatan laporan robot humanoid

Semua anggota tim memiliki porsi dan tanggung jawab masing-masing dalam menyelesaikan proyek ini hingga selesai

Kontribusi: Kontribusi dalam sebuah kelompok menjadi kunci dalam memecahkan masalah, berkolaborasi, dan menyajikan ide untuk meningkatkan kualitas serta kinerja robot humanoid secara holistik dan efektif.

## 6.2 Project Management

Manajemen proyek berfokus pada dokumentasi jadwal proyek, pencapaian, dan praktik manajemen untuk memastikan kelancaran dan pencapaian tujuan proyek. Dalam pengembangan robot humanoid, aspek manajemen proyek mencakup:

Dokumentasi: Penyusunan jadwal proyek dengan rincian tenggat waktu, tahapan pengembangan, dan target pencapaian. Dokumentasi ini memudahkan pemantauan kemajuan proyek serta membantu identifikasi masalah atau kendala selama proses pengembangan.

Pencapaian: Memantau pencapaian proyek sesuai dengan jadwal yang ditetapkan. Ini mencakup pencatatan progres, evaluasi setiap tahap pengembangan, dan memastikan pencapaian tujuan yang telah ditetapkan.

Praktik Manajemen: Implementasi praktik manajemen yang efektif, seperti identifikasi dan mitigasi risiko, alokasi sumber daya yang tepat, komunikasi efisien antar tim, dan adaptasi terhadap perubahan yang mungkin terjadi selama proses pengembangan.

Dengan dinamika kerja tim yang baik dan manajemen proyek yang efektif, pengembangan robot humanoid dapat dilakukan dengan lebih terstruktur, efisien, dan mampu mencapai tujuan sesuai dengan jadwal yang telah ditetapkan.

# 7 Conclusion and Reflection

## 7.1 Project Summary

Proyek pengembangan gerakan robot humanoid telah sukses mengatasi sejumlah tantangan utama dalam upaya meningkatkan kemampuan geraknya. Peningkatan kualitas gerakan berhasil dicapai melalui optimalisasi algoritma, penyesuaian perangkat keras, dan peningkatan responsivitas sistem. Melalui langkah-langkah pengujian sistematis, kami memastikan bahwa robot mampu konsisten dan optimal dalam melakukan gerakan di berbagai konteks. Hasil dari proyek ini menandai kemajuan penting dalam pengembangan robot humanoid yang lebih adaptif dan responsif

## 7.2 Future Work

Perkembangan di masa depan dapat difokuskan pada integrasi teknologi sensor terkini, pengembangan algoritma kontrol yang lebih pintar, dan pemahaman yang lebih mendalam terkait interaksi antara manusia dan robot. Penerapan robot humanoid juga dapat diperluas ke berbagai sektor penting dalam kehidupan sehari-hari, seperti perawatan kesehatan, edukasi, dan eksplorasi lingkungan berbahaya

## 7.3 Personal and Group Reflections

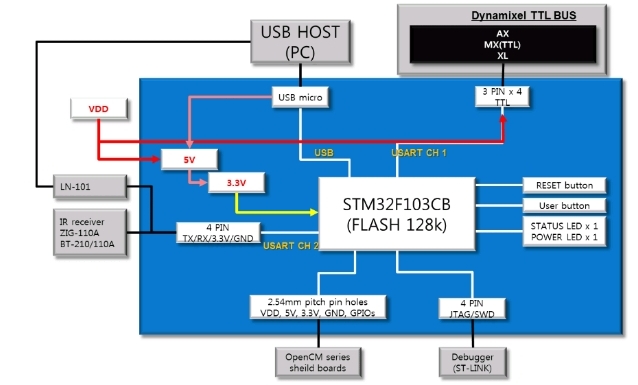
Selama perjalanan proyek ini, kami dihadapkan pada berbagai tantangan yang signifikan. Namun, melalui kerjasama tim yang solid dan manajemen proyek yang efektif, kami berhasil mengatasi setiap rintangan. Pengalaman ini mengajarkan kami banyak hal tentang kompleksitas pengembangan robot humanoid, baik dari segi teknis maupun interaksi manusia-robot. Dalam refleksi pribadi, kami menekankan betapa pentingnya kerjasama, kesabaran, dan dedikasi dalam mencapai tujuan bersama. Secara bersama-sama, proyek ini tidak hanya memperluas wawasan kami, tetapi juga mempersiapkan kami untuk menghadapi tantangan di masa depan dalam pengembangan teknologi robotika.

# 8 Appendices

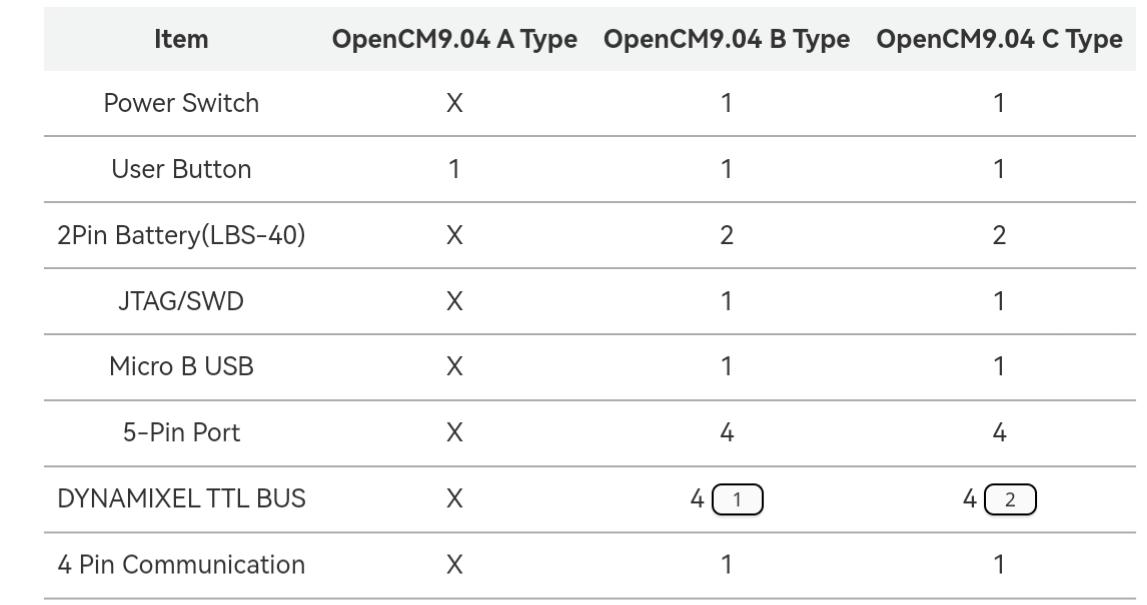
## 8.1 Bill of Materials

Rincian tampilan GUI untuk mengendalikan pergerakan robot humanoid menggunakan Python melibatkan beberapa elemen kunci. Bagian utama mencakup antarmuka grafis itu sendiri, tombol kontrol untuk berbagai gerakan dan fungsi, serta tampilan status yang memberikan informasi tentang kondisi robot. Dalam konteks biaya, komponen seperti sensor gerak, kamera untuk visualisasi, dan mikrofon untuk interaksi suara dapat mempengaruhi anggaran secara signifikan. Investasi tambahan mungkin diperlukan untuk sumber daya pembelajaran Python GUI development, termasuk buku, kursus online, dan dokumentasi resmi. Dengan mengintegrasikan semua elemen ini, tampilan GUI dapat memberikan pengendalian yang intuitif dan informatif atas pergerakan robot humanoid, sambil menjaga efisiensi biaya yang terkendali.

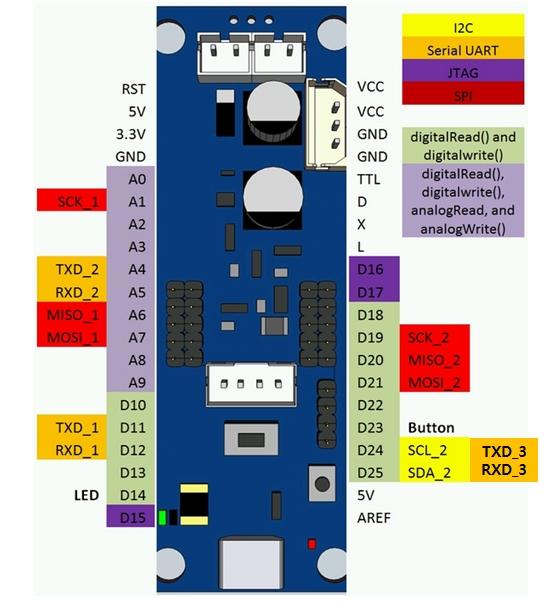
## 8.2 Electrical Wiring and System Layout

1. **Block Diagram**

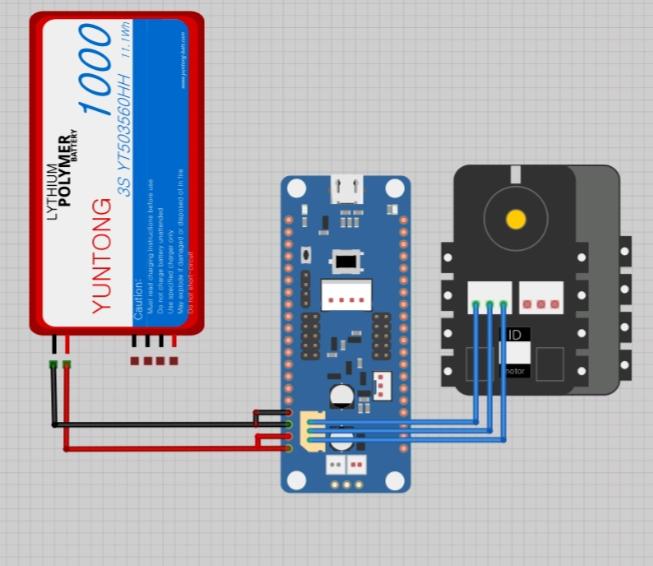
*Gambar 8. Block Diagram*

1. **Open CM9.04**

*Gambar 9. Open CM 9.04 Board*

1. **Open CM 9.04 PinMap**

*Gambar 10. Pin Map Open CM 9.04*

1. **Contoh: Pengoneksian Open CM dengan Dynamixel AX 12A**

*Gambar 11. Contoh Pengoneksian Open CM*

## 8.3 Code Repository

Include links to the code repository <https://github.com/anh0001/HumanoidRobot-GUI-Control/tree/kelasB_group2>

## 8.4 Additional Documentation

# 9 References

Azizah, N. 2023. Rancang Bangun Robot Humanoid Penari untuk Berjalan pada Lantai Berundak. Jurnal Teknik Elektro Indonesia. Vol 4 No 2

Farid, M. 2022. Apa Itu Robot Humanoid? Ini Pengertian dan Contoh Prototype yang Sudah Ada. <https://voi.id/teknologi/220567/apa-itu-robot-humanoid-ini-pengertian-dan-contoh-prototype-yang-sudah-ada>

Sari, P. R. 2023. Tahun 2025 China Akan Merilis Robot Humanoid Canggih. <https://www.cloudcomputing.id/berita/tahun-2025-china-akan-meriilis-robot-humanoid-canggih>

Soim, S. 2015. Perancangan Robot Humanoid Berbasis Mikrokontroller Atmega 32. <https://www.neliti.com/publications/174308/perancangan-robot-humanoid-berbasis-mikrokontroler-atmega-32>

Thai, C. N. 2017. Exploring Robotics with Robotis Systems. Switzerland: Springer International.