



PPNS POLITEKNIK
PERKAPALAN
NEGERI SURABAYA

PROPOSAL

OPTIMASI KOMUNIKASI PADA BASESTATION ROBOT SEPAKBOLA BERODA MENGGUNAKAN METODE MULTI- THREADING BERBASIS ROS

WISNU SUKMA WARDANI
NRP. 0920040080

Calon Dosen Pembimbing

1. AGUS KHUMAIDI , S.ST., M.T.
2. MOHAMMAD BASUKI RAHMAT , S.T., M.T.

PROGRAM STUDI D4 TEKNIK OTOMASI
JURUSAN TEKNIK KELISTRIKAN KAPAL
POLITEKNIK PERKAPALAN NEGERI SURABAYA
SURABAYA
2024



PPNS

POLITEKNIK
PERKAPALAN
NEGERI SURABAYA

PROPOSAL

OPTIMASI KOMUNIKASI PADA BASESTATION ROBOT SEPAKBOLA BERODA MENGGUNAKAN METODE MULTI- THREADING BERBASIS ROS

**WISNU SUKMA WARDANI
NRP. 0920040080**

Calon Dosen Pembimbing

1. AGUS KHUMAIDI , S.ST., M.T.
2. MOHAMMAD BASUKI RAHMAT , S.T., M.T.

**PROGRAM STUDI D4 TEKNIK OTOMASI
JURUSAN TEKNIK KELISTRIKAN KAPAL
POLITEKNIK PERKAPALAN NEGERI SURABAYA
SURABAYA
2024**

(Halaman ini sengaja dikosongkan)

LEMBAR PENGESAHAN

1. Judul Tugas Akhir : Optimasi Komunikasi Pada Basestation Robot Sepakbola Beroda Menggunakan Metode Multi Threading Berbasis ROS
2. Bidang Tugas Akhir : Sistem Industri Cerdas
3. Bidang Keahlian : Robotika
4. Pengusul
- a. Nama Lengkap : Wisnu Sukma Wardani
 - b. NRP : 0920040080
 - c. Program Studi : D4-Teknik Otomasi
 - d. Jurusan : Teknik Kelistrikan Kapal
 - e. Alamat Rumah : Jalan Ngadiluwih , Desa Segaran
 - f. No. Telp/HP : 081998633332
 - g. Alamat Email : wisnusukma@student.ppns.ac.id
5. Dosen Pembimbing
- Dosen Pembimbing I
- a. Nama Lengkap dan Gelar : Agus Khumaidi, S.ST., M.T
 - b. NIP : 199308172020121004
- Dosen Pembimbing II
- a. Nama Lengkap dan Gelar : Muhammad Basuki Rahmat, S.ST., M.T
 - b. NIP : 1997305222000031001
6. Jangka Waktu Pelaksanaan : 4 Bulan

Menyetujui,
Ketua Jurusan

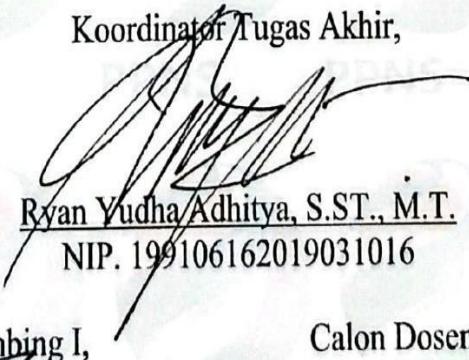
Isa Rachman, S.T., M.T.
NIP. 198008162008121001

Surabaya, 28 Desember 2023
Pengusul,

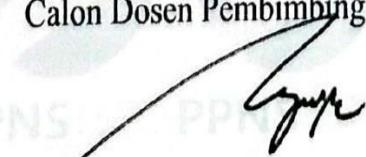

Wisnu Sukma Wardani

NRP. 0920040080

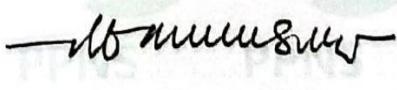
Koordinator Tugas Akhir,


Ryan Yudha Adhitya, S.ST., M.T.
NIP. 199106162019031016

Calon Dosen Pembimbing I,


Agus Khumaidi, S.ST., M.T.
NIP. 199308172020121004

Calon Dosen Pembimbing II,


Muhammad Basuki Rahmat, S.ST., M.T.
NIP. 1997305222000031001

(Halaman ini sengaja dikosongkan)

OPTIMASI KOMUNIKASI PADA BASESTATION ROBOT SEPAKBOLA BERODA MENGGUNAKAN METODE MULTI THREADING BERBASIS ROS

Wisnu Sukma Wardani

ABSTRAK

Pada KRSBI Beroda robot diharapkan dapat bergerak otomatis dengan menggunakan *Basestation* sebagai pusat komunikasi antar robot dalam tim yang sama. Pada lomba KRSB-B pada tahun 2023 tim robot Gerhana Dewaruci (Tim sepak bola beroda PPNS) mengalami kendala pada komunikasi antara robot penyerang dengan robot kiper yaitu robot kiper selalu keluar ke posisi base di sebelah gawang dan tidak dapat tetap berada di bawah gawang pada saat pengiriman data *ready* (Posisi robot sebelum peluit dibunyikan) maka penelitian ini bertujuan untuk meningkatkan efisiensi komunikasi antara basestation dan robot sepakbola beroda melalui penerapan metode *multithreading* berbasis Robot Operating System(ROS). Dalam upaya meningkatkan kinerja komunikasi, metode *multithreading* diimplementasikan untuk memungkinkan penanganan simultan dari beberapa tugas komunikasi.

Kata kunci: Basestation, Multithread, ROS (Robot Operation System), Robot Sepakbola Beroda

(Halaman ini sengaja dikosongkan)

DAFTAR ISI

ABSTRAK.....	i
DAFTAR ISI	iii
DAFTAR TABEL.....	vii
DAFTAR GAMBAR	viii
DAFTAR NOTASI.....	xi
BAB 1 PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Pertanyaan Penelitian	2
1.3 Batasan Penelitian.....	2
1.4 Tujuan Penelitian	2
1.5 Manfaat Penelitian	3
BAB 2 TINJAUAN PUSTAKA	5
2.1 Kajian Penelitian Terdahulu	5
2.2 Sensor dan Aktuator	6
2.2.1 Incremental Rotary Recorder	6
2.2.2 Motor DC PG-36.....	7
2.2.3 Motor DC PG-45.....	8
2.2.4 Sensor HWT101CT-TTL	9
2.2.5 Sensor Proximity	10
2.2.6 LCD 20x4.....	11
2.2.7 Camera Omni-directional	12
2.3 Kontroler.....	13
2.3.1 Arduino Mega 2560 Pro Mini.....	14
2.3.2 Laptop.....	15

2.4 Software	16
2.4.1 Arduino IDE.....	16
2.4.2 Keil uVision	16
2.4.3 Visual Studio Code	17
2.4.5 STM CubeMX.....	17
2.4.6 QT (Q Toolkit)	17
2.5 TCP/IP(Transmission Control Protocol/Internet Protocol)	18
2.6 UDP (User Datagram Protocol)	19
2.7 ROS (Robot Operation System).....	19
2.8 Metode Yang Digunakan	20
2.8.1 Multithreading.....	20
2.8.2 Odometry	22
2.8.3 Teory Gyroscope.....	24
2.8.4 Gyrodometry.....	24
2.8.5 You Only Look Once (YOLO)	25
2.9 Sistem Penggerak Omni-Directional Empat Roda.....	26
BAB 3 METODE PENELITIAN	29
3.1 Konsep Penelitian	29
3.1.1 State of The Art (Keterbaruan dari Penelitian).....	29
3.1.2 Prinsip Kerja Sistem	29
3.1.3 Diagram Blok Sistem.....	30
3.1.4 Diagram Many to Many Model.....	33
3.1.5 User Thread.....	33
3.1.6 Kernel Thread.....	34
3.1.7 Diagram Blok Jaringan ROS (Robot Operation System).....	34
3.1.8 Diagram Blok Peer to Peer Ros Architecture	36

3.1.9 Protokol Data Masuk dan Keluar UDP dan TCP	37
3.1.10 Flowchart dan Pengiriman Data Start dan Stop Robot.....	38
3.2 Tahapan Penelitian	40
3.2.1 Flowchart Pengiriman Strategi	41
3.2.2 Identifikasi Masalah dan Studi Literatur.....	43
3.2.3 Analisis Kebutuhan Sistem	43
3.3 Perencanaan dan Desain.....	44
3.3.1 Perancangan Software.....	45
3.3.2 Installasi ROS	46
3.3.3 Membuat Node.....	52
3.3.4 Membuat Thread.....	55
3.3.5 QTCreator	56
3.3.6 Setting QTCreator	58
3.3.7 Cara Pengujian	61
3.4 Rencana Anggaran Penelitian	62
BAB 4 HASIL DAN PEMBAHASAN	63
4.1 Pengujian Sensor dan Aktuator.....	63
4.1.1 Pengujian <i>Sensor Proximity</i>	63
4.1.2 Pengujian Rotation Angle.....	65
4.1.3 Pengujian Rotary Encoder	67
4.1.4 Pengujian Motor.....	68
4.2 Pengujian Kamera	70
4.2.1 Pengujian Koneksi Kamera dengan PC	70
4.3 Pembuatan Basestation.....	70
4.3.1 Pembuatan GUI.....	70
4.3.2 Menghubungkan Basestation dengan ROS	71

4.3.4 Pengujian Komunikasi Antar Node.....	72
4.4 Komunikasi Antar Device	73
DAFTAR PUSTAKA	77

DAFTAR TABEL

Tabel 2. 1 Perbandingan Jurnal.....	6
Tabel 2. 2 Spesifikasi Rotary Encoder LPD3806 400BM G5 24C.....	7
Tabel 2. 3 Spesifikasi Motor DC-36	8
Tabel 2. 4 Spesifikasi Motor DC-45	8
Tabel 2. 5 Spesifikasi HWT101CT-TTL	10
Tabel 2. 6 Spesifikasi Driver Motor IBT-2-H-Bridge	12
Tabel 2. 7 Spesifikasi STM32F4 DiyMore.....	14
Tabel 2. 8 Spesifikasi Arduino Mega 2560 Pro Mini.....	15
Tabel 3. 1 Analisa Kebutuhan Sistem.....	43
Tabel 3. 2 Jadwal Pengerjaan Penelitian.....	62
Tabel 3. 3 Rencana Anggaran Penelitian	62
Tabel 4. 1 Pengujian Jarak Proximity.....	64
Tabel 4. 2 pengujian sensor gyroscope	66
Tabel 4. 3 Data Pengujian Rotary Encoder	68
Tabel 4. 4 Pengujian Motor	69

DAFTAR GAMBAR

Gambar 1. 1 Pertandingan Robot Sepak Bola (KRI2023, 2023).....	1
Gambar 2. 1 Rotary Encoder	7
Gambar 2. 2 Motor DC PG-36.....	8
Gambar 2. 3 Motor DC PG-45.....	9
Gambar 2. 4 Sensor HWT101CT-TTL	9
Gambar 2. 5 Sensor <i>Proximity</i> Inframerah.....	10
Gambar 2. 6 LCD 20x4	11
Gambar 2. 7 Driver Motor IBT-2-H-Bridge.....	12
Gambar 2. 8 Kamera Omni-directional.....	13
Gambar 2. 9 STM32F4 DiyMore	13
Gambar 2. 10 Arduino Mega 2560 Pro Mini	15
Gambar 2. 11 Laptop	16
Gambar 2. 12 Logo Robot Operating System.....	20
Gambar 2. 13 Model Thread	21
Gambar 2. 14 Many to One	22
Gambar 2. 15 Representasi penempatan rotary encoder (Darmawan et al., 2023)	23
Gambar 2. 16 Koordinat robot dalam sumbu kartesian x=0.5 dan y=0.5 dan arah hadap robot $\theta=0^\circ$	25
Gambar 2. 17 Deteksi Object YOLO	26
Gambar 2. 18 Ilustrasi Desain Robot Omni-directional Empat Roda.....	27
Gambar 2. 19 Representasi Kinematika dari Sistem Pergerakan Omni-directional Empat Roda (Hidayat, 2023)	28
Gambar 3. 1 Diagram Alir Sistem pada Basestation.....	29
Gambar 3. 2 Diagram Blok Sistem Robot.....	30
Gambar 3. 3 Diagram Blok Sistem Komunikasi Basestation.....	31
Gambar 3. 4 Diagram Many to Many model	33
Gambar 3. 5 Diagram Blok Konfigurasi jaringan ROS	34
Gambar 3. 6 Diagram Blok Peer to Peer ROS Architecture	36
Gambar 3. 7 Protokol Komunikasi UDP dan TCP	37
Gambar 3. 8 Flowchart Pengiriman Data Start.	38

Gambar 3. 9 Pengiriman Data Stop.....	39
Gambar 3. 10 Flowchart Penelitian.....	40
Gambar 3. 11 Pengiriman Data Strategi 1.....	41
Gambar 3. 12 Perancangan Hardware.....	44
Gambar 3. 13 Desain Tampilan Basestation	45
Gambar 3. 14 Perancangan Software.....	46
Gambar 3. 15 Code Repository ROS.....	47
Gambar 3. 16 Install Perangkat Lunak Curl.....	47
Gambar 3. 17 Impor kunci GPG (GNU Privacy Guard).....	48
Gambar 3. 18 Update Ubuntu	49
Gambar 3. 19 Install ROS Neotic.....	49
Gambar 3. 20 Install Paket Ros.....	50
Gambar 3. 21 Paket Ros	50
Gambar 3. 22 Code Untuk Mengaktifkan Environment ROS.....	50
Gambar 3. 23 Otomatisasi Aktivasi Environment ROS	51
Gambar 3. 24 Menjalankan Perintah Ros	51
Gambar 3. 25 Membuat Workspace Ros	52
Gambar 3. 26 Cmakelist.txt.....	53
Gambar 3. 27 Build Proyek ROS.....	54
Gambar 3. 28 Running Node C++	55
Gambar 3. 29 Install Snapd.....	57
Gambar 3. 30 Install QTCreator-ros.....	57
Gambar 3. 31 Membuka QTCreator Pada Sistem Operasi ROS	58
Gambar 3. 32 Project Name and Location.....	58
Gambar 3. 33 Workspace QTCreator	59
Gambar 3. 34 Menambahkan File Pada Project	59
Gambar 3. 35 File Subscriber, Publiser, dan Node	60
Gambar 3. 36 Interface	60
Gambar 3. 37 Form Pembuatan Interface	61
Gambar 4. 1 Robot Satu, Robot Dua dan Robot Tiga.....	63
Gambar 4. 2 Bola Terdeteksi	64
Gambar 4. 3 Bola Tidak Terdeteksi	64

Gambar 4. 4 Pengujian Proximity	65
Gambar 4. 5 Gamba Pengujian Dengan Busur	66
Gambar 4. 6 Penempatgam Sensor HWT101CT	66
Gambar 4. 7 Pemasangan Rotary Encoder	68
Gambar 4. 8 Pengujian Rotary Encoder.....	68
Gambar 4. 9 Pengujian Menggunakan Tachometer.....	69
Gambar 4. 10 Hasil Tangakapan Kamera C922	70
Gambar 4. 11 QT Creator.....	71
Gambar 4. 12 Running tanpa Running ROS Master	71
Gambar 4. 13 ROS Master Berjalan	72
Gambar 4. 14 Pengiriman Data	72
Gambar 4. 15 Menerima Data	73

DAFTAR NOTASI

V_1 = Kecepatan roda 1

V_2 = Kecepatan roda 2

V_3 = Kecepatan roda 3

V_4 = Kecepatan roda 4

V_x = Kecepatan robot pada arah sumbu X

V_y = Kecepatan roda pada arah sumbu Y

V_θ = Kecepatan sudut robot

**ω = Kecepatan angular dari roda (rad/detik) R = Jari-jari
robot (cm) r = Jari-jari robot roda omni (cm)**

**Y_{pos} = posisi x robot dalam koordinat kartesian
(mm) X_{pos} = posisi y robot dalam koordinat
kartesian (mm)**

θ = arah hadap robot ($^{\circ}$)

$S_i (1,2,3)$ = jarak tempuh dari masing-masing roda

**θ = sudut roda omni terhadap sumbu
acuan S_x = Jarak tempuh robot
pada sumbu x**

(Halaman ini sengaja dikosongkan)

BAB 1

PENDAHULUAN

1.1 Latar Belakang

Base Station merupakan sebuah perangkat lunak yang dirancang sedemikian rupa untuk mengontrol, mengkoordinasi serta memonitoring kondisi robot selama bertanding. *Basestation* diharuskan dapat bekerja secara otomatis tanpa adanya interupsi selama pertandingan, oleh karena itu dibutuhkan koneksi yang handal serta kemampuan dalam pemrosesan data yang diterima secara cepat. Aplikasi Base Station terhubung dengan Referee Box melalui kabel Ethernet (LAN) sedangkan terhubung dengan robot melalui perantara wireless (WLAN)(Tjoanapessy et al., 2019).



Gambar 1. 1 Pertandingan Robot Sepak Bola (KRI2023, 2023)

Dalam pertandingan KRSBI-Beroda pada tahun 2023 KRSBI-Beroda masih banyak dari tim lain menggunakan metode komunikasi *Multicasting* yang dianggap tidak efektif jika di bandingkan dengan metode komunikasi pada banyak *basestation* lain, metode komunikasi ini dinilai tidak efektif karena mengurangi keefisien gerak robot karena data yang di kirim dari *basestation* ke robot menggunakan data yang sama sehingga pergerakan masing-masing robot menjadi terbatas dan robot tidak dapat bergerak secara mandiri.

Berdasarkan permasalahan di atas maka dilakukan penelitian pada robot sepak bola beroda untuk membuat *basestation* menggunakan metode

Multithreading dengan mempertimbangkan pengiriman data ke semua robot secara bersamaan dengan membagi beberapa tugas yang berbeda di mana sebuah program dibagi menjadi beberapa thread kecil atau jalur eksekusi independen dapat berjalan secara bersamaan dalam satu proses menggunakan sistem operasi ROS (Robot Operation System).

1.2 Pertanyaan Penelitian

Pertanyaan penelitian antara lain:

1. Bagaimana membuat GUI (Graphical User Interface) menggunakan software QT Creator?
2. Bagaimana basestation dapat berjalan didalam ROS?
3. Bagaimana *basestation* dapat mengirimkan data pada robot menggunakan metode Multithread sehingga pengiriman lebih efektif dan efisien.

1.3 Batasan Penelitian

Batasan Penelitian antara lain:

1. Masing-masing robot dapat mengirimkan posisi ke *basestation*.
2. Penelitian ini lebih fokus pada proses komunikasi antar robot.
3. Pada penelitian ini lebih fokus dimana robot dapat menjalankan beberapa tugas berbeda yang dikirim *basestation* secara bersamaan.
4. Pengujian pada penelitian ini menggunakan setting tengah lapangan pertandingan.
5. Pada penelitian ini lebih fokus *basestation* dapat berjalan dalam ROS

1.4 Tujuan Penelitian

Tujuan dari penelitian ini antara lain:

1. Masing-masing robot mampu mengirimkan data berupa posisi dari masing-masing robot ke *basestation*.
2. *Basestation* mampu mengirimkan data ke robot sepak bola beroda.
3. Masing-masing robot dapat menjalankan perintah yang berbeda dari *basestation*.
4. Basestation dapat dijalankan pada ROS

1.5 Manfaat Penelitian

Manfaat penelitian ini adalah

1. Meningkatkan komunikasi robot melalui *basestation* yang berbasis ROS.
2. Meningkatkan koordinasi strategi antar robot penyerang dan robot kiper.
3. Meningkatkan pembuatan strategi pada robot penyerang dan juga robot kiper.

(Halaman ini sengaja dikosongkan)

BAB 2

TINJAUAN PUSTAKA

Bab ini mencakup tinjauan pustaka yang komprehensif yang mengacu pada berbagai referensi, laporan penelitian, dan publikasi ilmiah untuk menjelaskan tujuan dan metodologi studi literatur. Dengan mencantumkan referensi atau sumber, laporan penelitian, dan publikasi ilmiah yang merinci objek, prosedur, dan perangkat keras atau perangkat lunak yang digunakan, komponen yang digunakan, teknik yang digunakan, dan perangkat keras atau perangkat lunak yang digunakan dalam pembuatan tugas akhir ini.

2.1 Kajian Penelitian Terdahulu

Penelitian sebelumnya membahas tentang penggunaan basestation untuk komunikasi robot sepak bola beroda menggunakan metode multicast yang berjudul “Singkronisasi Protokol UDP untuk Komunikasi Robot Sepakbola Bola Beroda”(Herman et al., 2019), pada penelitian ini hanya berfokus pada singkronisasi protokol udp dan menggunakan metode multicast yang hanya dapat mengirim tugas ke semua robot sekaligus, dan pada pengembangan penilitian ini basestation dapat mengirim banyak tugas secara bersamaan dalam satu proses menggunakan *multithreading* tanpa melibatkan *broadcast* data pada robot.

Pada penelitian yang berjudul ‘‘Pemetaan Posisi Robot Ssepak Bola Beroda Menggunakan Metode Gyrodometry untuk Memprediksi Sudut Tendangan Bola Terhadap Gawang Lawan Dengan Perhitungan Trigonometry’’(Nasikhin, 2019). Pada penelitian tersebut *interface* pada basestation dibuat menggunakan Visual Studio dan menggunakan bahasa C# sedangkan pada penelitian ini menggunakan QT untuk membuat *interface* dan c++ sebagai bahasa pemrograman agar dapat berjalan pada sistem operasi *linux*.

Tabel 2. 1 Perbandingan Jurnal

Penulis (tahun)	Judul	Fokus penelitian sebelumnya	Perbedaan/Pengembangan
(Edy Surya Prabowo et al., 2022)	Perancangan aplikasi <i>basestation</i> dalam sistem koordinasi robot sepak bola beroda dengan <i>multithread</i>	Pada penelitian sebelumnya berfokus untuk menampilkan posisi dua robot Penyerang yang ada pada lapangan	Pada penelitian ini basestation menampilkan posisi semua robot yang ada pada lapangan
(Herman et al., 2019)	Singkronisasi protokol UDP untuk komunikasi robot sepakbola bola beroda	Singkronisasi protokol UDP guna mengurangi <i>delay</i> tinggi	Penggunaan protokol UDP yang di jalankan pada ROS
(Fouk et al., 2022)	Implementation of <i>base station</i> as an intermediary referee box in the delivery of wheeled football robot movement commands	Pada penelitian sebelumnya berfokus pada efektifitas penggunaan TCP dan ROS antara <i>basestation</i> dan robot	Pada penelitian ini menggunakan protokol UDP, TCP dan ROS pada komunikasi antara <i>basestation</i> dan robot.

2.2 Sensor dan Aktuator

Dalam Tugas Akhir ini, para peneliti memanfaatkan beberapa sensor dan aktuator, khususnya:

2.2.1 Incremental Rotary Recorder

Rotary encoder adalah perangkat listrik yang digunakan untuk mengubah posisi sudut atau gerakan poros menjadi informasi analog atau digital. Rotary encoder dapat diklasifikasikan ke dalam dua kategori berdasarkan kapasitas outputnya: rotary encoder inkremental dan rotary

encoder absolut. Rotary encoder yang digunakan dalam investigasi ini adalah rotary encoder tambahan. Rotary encoder inkremental adalah perangkat yang menghasilkan sinyal gelombang persegi saat poros berputar. Untuk spesifikasi rotary encoder dapat di lihat pada Tabel 2.2.

Tabel 2. 2 Spesifikasi Rotary Encoder LPD3806 400BM G5 24C

<i>Attribute</i>	<i>Value</i>
<i>Performance</i>	400 ppr
<i>Operating Voltage</i>	5-24 VDC
<i>Maximum mechanical speed</i>	5000 rpm
<i>Electrical response frequency</i>	20K/sec
<i>Output Signal Type</i>	<i>NPN Open Collector</i>
<i>Shaft Type</i>	<i>Radial, Thrust</i>
<i>Operating Temperature</i>	-40°C to 85°C
<i>Draw Current</i>	30mA
<i>Weight</i>	118 g
<i>Mounting Holes</i>	M3



Gambar 2. 1 Rotary Encoder
(Sumber : Dokumentasi Pribadi)

Penggunaan rotary encoder tambahan, seperti yang ditunjukkan pada Gambar 2.1, adalah untuk menghitung jumlah rotasi pada roda robot omnidirectional yang terpasang pada poros rotary encoder.

2.2.2 Motor DC PG-36

Motor DC PG-36 seperti pada gambar 2.2 adalah motor yang digunakan sebagai penggiring robot pada penelitian ini dikarenakan mempunyai rpm yang tinggi. Untuk spesifikasi dari mototr DC PG-36 bisa dilihat pada Tabel

2.2.

Tabel 2. 3 Spesifikasi Motor DC-36

Parameter	Nilai
Arus	5A
Tegangan kerja	DC 24V
Torsi	10kgfcm
Speed	600 rpm



Gambar 2. 2 Motor DC PG-36
(Sumber : Dokumentasi Pribadi)

Motor DC PG-36 seperti pada gambar 2.2 memiliki torsi yang relatif lebih kecil daripada motor lain yang digunakan pada penelitian ini.

2.2.3 Motor DC PG-45

Selain Motor DC PG-36, pada penelitian ini juga menggunakan Motor DC PG-45 seperti pada Gambar 2.3. Untuk spesifikasi dari Motor DC PG-45 dapat dilihat pada Tabel 2.3

Tabel 2. 4 Spesifikasi Motor DC-45

Parameter	Nilai
Arus	4A
Tegangan	DC 24V
Torsi	25 kg/cm

<i>Speed</i>	500 rpm
<i>Performance</i>	7 ppr



Gambar 2. 3 Motor DC PG-45
(Sumber : Dokumentasi Pribadi)

Motor DC PG-45 adalah motor DC jenis planetary gearbox yang mempunyai daya dan torsi yang besar tetapi mempunyai rpm yang lebih kecil daripada Motor DC PG-36.

2.2.4 Sensor HWT101CT-TTL

Sensor HWT101CT-TTL adalah sensor kemiringan satu sumbu yang memanfaatkan teknologi MEMS (Microelectromechanical System). Sensor ini memiliki jangkauan pengukuran hingga 180 derajat dengan resolusi 0,01 derajat. Sensor ini juga dilengkapi dengan giroskop internal yang dapat digunakan untuk mengukur perubahan sudut secara real-time.Untuk spesifikasi Sensor HWT101CT-TTL dapat dilihat pada Tabel 2.4.



Gambar 2. 4 Sensor HWT101CT-TTL

Tabel 2. 5 Spesifikasi HWT101CT-TTL

Parameter	Nilai
Model	HWT101CT
Voltage	9-36V
Current	25mA
Output	Z axis angle, Z-axis angular velocity, chip time
Accuracy	0.1°

2.2.5 Sensor Proximity

Sensor *proximity* adalah sensor yang dapat mendeteksi object pada jarak dekat. Sensor *proximity* memiliki beberapa jenis, yaitu sensor *proximity* induktif, sensor *proximity* kapasitif , sensor *proximity* ultrasonic, dan sensor *proximity* optik dan dalam penelitian ini menggunakan sensor *proximity* optik.

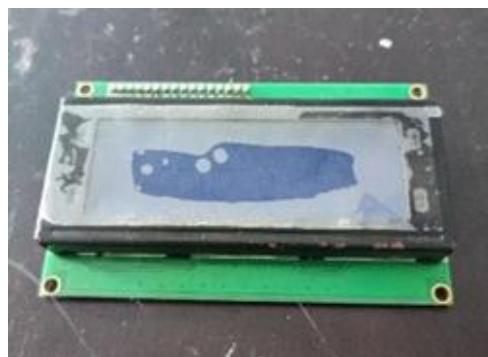


Gambar 2. 5 Sensor *Proximity* Inframerah
(Sumber : Dokumentasi Pribadi)

Pada penelitian ini menggunakan sensor *proximity* optik dengan inframerah seperti pada Gambar 2.5 yang di gunakan sebagai pendekksi bola apakah bola sudah masuk ke penggiring atau belum.

2.2.6 LCD 20x4

LCD 20x4 memiliki dua puluh karakter mendatar (*columns*) dan empat karakter menurun (*rows*). Sebagian besar LCD ini digunakan untuk menampilkan angka atau teks. Seperti pada Gambar 2.6.



Gambar 2. 6 LCD 20x4
(Sumber : Dokumentasi Pribadi)

Pada penelitian ini menggunakan LCD 20x4 sebagai penampil data robot. Seperti koordinat x, koordinat y, sudut robot, parameter jarak bola, jarak gawang, koordinat robot kawan, serta jarak robot lawan, dan lain sebagainya.

2.2.5 Driver Motor IBT-2-H-Bridge

Driver IBT-2-H-Bridge adalah driver motor DC yang menggunakan konfigurasi H-bridge untuk mengontrol arah dan kecepatan motor. Diver motor seperti pada Gambar 2.7, driver motor ini dapat mengontrol motor dalam rentang tegangan 6-27 volt VDC dengan menggunakan Pulse Width Modulation, atau PWM. Dalam penelitian ini, driver IBT-2-H-Bridge digunakan untuk mengontrol motor PG45 sebagai penggerak dan penendang robot dan motor PG36 sebagai penggiring robot.



Gambar 2. 7 Driver Motor IBT-2-H-Bridge
(Sumber Dokumentasi Pribadi)

Spesifikasi daripada Driver motor IBT-2-H-Bridge seperti pada tabel 2.6.

Tabel 2. 6 Spesifikasi Driver Motor IBT-2-H-Bridge

Parameter	Nilai
Input Level	3.3V-5V
Tegangan Masukan	6V-27V
Arus Masukan	43A
Control mode	PWM or Level

2.2.7 Camera Omni-directional

Kamera ini merupakan kamera utama pada robot sepak bola beroda. Tipe yang digunakan pada penelitian ini menggunakan produk ELP usb8mp02g-sfv digabungkan dengan cermin tele Vstone dengan tipe vs-c450mr seperti dalam Gambar 2.8. Kamera juga sudah dilengkapi dengan cermin hiperbolik yang mana mampu memberikan kemampuan pengambilan citra 360 derajat pada posisi horizontal. Memiliki bidang pandang vertikal sebesar 10 - 15 pada sisi atas dan 55 pada sisi bawah sumbu sejajar dengan cermin (Octavian et al., 2021)



Gambar 2. 8 Kamera Omni-directional
(Sumber : Dokumentasi Pribadi)

Beberapa bagian utama struktur omnidirectional kamera terdiri dari cermin omnidirectional, adaptor lensa tele, dan kamera CCD. Keunggulan kamera jenis ini adalah daya tangkap citra yang lebih luas dibandingkan kamera konvensional. karena cocok untuk sensor utama robot yang membutuhkan navigasi.

2.3 Kontroler

Berikut adalah kontroler yang di gunakan dalam penelitian ini.

2.1.1 STM32F4 *DiyMore*

Salah satu jenis prosesor ARM adalah mikrokontroler Diymore STM32F4. Mikrokontroler Diymore STM32F4 memiliki banyak pin input atau output dan kecepatan clock yang tinggi, sehingga dapat menampung banyak pin yang dibutuhkan sistem robot dan menjalankan program.



Gambar 2. 9 STM32F4 DiyMore
(STM32F407VGT6, 2020)

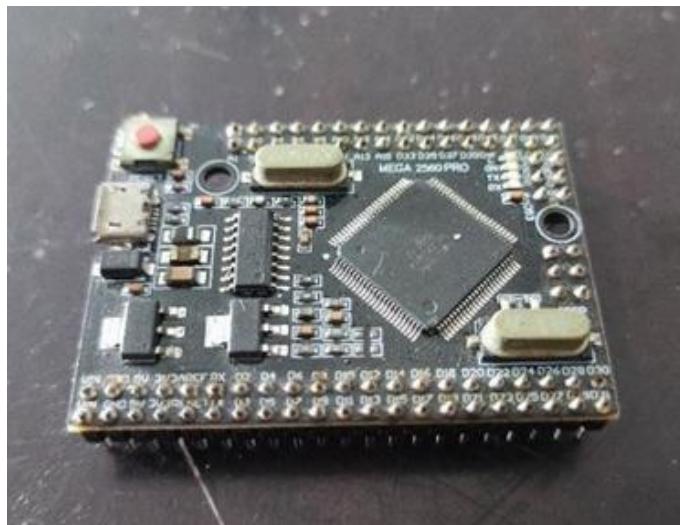
Dalam tugas akhir ini seperti pada Gambar 2.10 di gunakan untuk menerima hasil dari pengolahan citra dari pc dan digunakan untuk parameter menggerakan aktuator berupa motor serta di gunakan sebagai pembacaan sensor *proximity*, dan mengirim serta menerima data dari Laptop/PC.Untuk spesifikasi STM32F4 *DiyMore* dapat di lihat pada Tabel 2.10

Tabel 2. 7 Spesifikasi STM32F4 DiyMore

Fitur	Spesifikasi
Ukuran	45mm x 60mm
Kecepatan	168Mhz
Core	ARM Cortex-M4
Timer	2 x WDG, RTC, 24-bit down counter
RAM Internal	192Kb
Backup RAM	4Kb
Source	3.3V pin(3.3V)
Flash	1024Kb
Input	USB Connector 5V
Jumlah Pin	82 Pin I/O 16 Bit ADC 12 ADC

2.3.1Arduino Mega 2560 Pro Mini

Arduino Mega 2560 Pro Mini adalah sistem mini berbasis mikrokontroller ATmega 2560 dengan 54 pin input dan output. Arduino Mega 2560 Pro Mini juga memiliki kristal 16MHz seperti pada Gambar 2.11. Salah satu keuntungan dari Arduino ini adalah menggunakan kabel micro USB untuk disambungkan ke komputer.Arduino Mega 2560 Pro Mini digunakan untuk pembacaan gyroscope dan penggiring bola dalam penelitian ini. Spesifikasi Arduino Mega 2560 Pro Mini dapat dilihat pada Tabel 2.7



Gambar 2. 10 Arduino Mega 2560 Pro Mini
(Sumber : Dokumentasi Pribadi)

Tabel 2. 8 Spesifikasi Arduino Mega 2560 Pro Mini

Spesifikasi	Keterangan
<i>Microcontroller</i>	Atmega2560
<i>Operating Voltage</i>	5V
<i>Input Voltage (limits)</i>	5.5-16V
<i>Input Voltage (recommended)</i>	7-12V
<i>Digital I/O Pins</i>	54 (<i>of which 14 provide PWM output</i>)
<i>Analog Input Pins</i>	16 (Pin)
<i>DC Current for 3.3V Pin</i>	50 mA
<i>DC Current per I/O Pin</i>	40 mA
<i>CLOCK SPEED</i>	16 MH
<i>Flash Memory</i>	256 KB <i>of which 8 KB used by bootloader</i>
<i>SRAM</i>	8 KB
<i>EEPROM</i>	4 KB

2.3.2 Laptop

Dalam Penelitian ini, peneliti menggunakan laptop seperti Gambar 2.12 untuk pengolahan citra dari kamera dengan koneksi USB.



Gambar 2. 11 Laptop

Data yang dihasilkan dari proses pengolahan gambar dikirim ke mikrokontroller STM32F4 melalui komunikasi serial dengan modul USB TTL, yang diinstal pada laptop yang digunakan dengan OS Ubuntu

2.4 Software

Dalam tugas akhir ini, peneliti menggunakan software pendukung seperti Visual Studio Code, Arduino IDE, Keil uVision, dan STM32CubeMX yang akan dijelaskan sebagai berikut :

2.4.1 Arduino IDE

Arduino IDE atau Arduino Integrated Development Environment adalah perangkat lunak cross-platform yang digunakan untuk menulis dan mengunduh program ke platform Arduino dengan bahasa C/C++. Arduino memiliki kepopuleran yang sangat tinggi di dunia kontrol karena sistem open-source yang dibangun sehingga muncul komunitas-komunitas pengguna Arduino. Arduino IDE memiliki fungsi utama yaitu sebagai text editor untuk membuat program, memvalidasi program, dan untuk mengunduh program ke platform Arduino. Pada tugas akhir ini software Arduino IDE digunakan untuk membuat algoritma pemrograman untuk pembacaan sensor gyroscope dan juga penendang bola pada robot.(ArduinoIndonesia.id, 2023)

2.4.2 Keil uVision

Keil uVision atau biasa disebut Keil MicroVision adalah perangkat lunak yang digunakan untuk membuat, membangun, dan men-debug

mikrokontroller yang berbasis Arm®. Perangkat lunak ini mengkombinasikan pengelolaan proyek, lingkungan runtime, fasilitas pembangunan, pengeditan source code, dan debugging program. Bahasa pemrograman yang digunakan pada perangkat lunak ini adalah C/C++. Peneliti menggunakan Keil uVision untuk membuat algoritma pemrograman untuk kinematika gerak robot.

2.4.3 Visual Studio Code

Visual studio code merupakan kode editor yang ringan namun kuat yang berbasis Windows, macOS, dan Linux. Dalam penggunaannya perangkat lunak ini mendukung JavaScript, TypeScript, dan Node.js serta lebih banyak varian lingkungan tambahan untuk bahasa pemrograman lainnya seperti C++, C#, Java, Python, PHP, Go, dan runtime seperti .NET dan Unity. Kemudahan yang ditawarkan yaitu mampu bekerja dengan Git dan menyediakan SCM lainnya. Editor ini juga didukung Microsoft Azure dimana mampu menyebarkan dan hosting situs ataupun penyimpanan dengan dukungan React, Angular, Vue, Node, Python, dan lebih (Visual Studio, 2020). Pada Tugas Akhir ini Visual Studio Code digunakan sebagai text editor untuk membuat program image processing dengan menggunakan Bahasa pemrograman C++, dan Python.

2.4.5 STM CubeMX

Perangkat lunak berbasis grafis STM32CubeMX memudahkan pengaturan mikrokontroller dan mikroprocessor STM32. Pengguna dapat mengatur GPIO dan clock speed mikrokontroller yang diinginkan. Perangkat lunak ini memiliki kemampuan untuk mengatur mikrokontroller serta membangun sistem yang telah diatur yang dapat diteruskan ke perangkat lunak yang digunakan untuk membuat program yang sudah terintegrasi seperti Keil uVision. Tugas Akhir ini menggunakan program STM32CubeMX untuk mengubah semua pin STM32F4 DiyMore sesuai dengan fungsinya.

2.4.6 QT (Q Toolkit)

Qt, singkatan dari "Q Toolkit" atau dikenal sebagai "Qt Toolkit," adalah sebuah framework pengembangan perangkat lunak lintas platform yang lahir

pada tahun 1991 oleh Haavard Nord dan Eirik Chambe-Eng. Awalnya dikembangkan oleh Trolltech, perusahaan Norwegia, Qt kemudian diakuisisi oleh Nokia pada tahun 2008, sebelum akhirnya The Qt Company mengambil alih pengembangan. Salah satu keunggulan utama Qt adalah kemampuannya untuk membuat aplikasi yang dapat berjalan di berbagai sistem operasi tanpa modifikasi signifikan. Dengan dukungan untuk bahasa pemrograman seperti C++, Python, dan JavaScript, Qt menyediakan modul-modul modular yang dapat digunakan bersama-sama atau terpisah untuk berbagai kebutuhan pengembangan. Qt juga dikenal sebagai pilihan populer untuk pengembangan aplikasi desktop lintas platform dan aplikasi mobile, dengan dukungan untuk Android dan iOS. Qt memiliki model lisensi ganda, yakni GPL dan komersial, dan menawarkan dukungan aktif dari komunitas pengembang serta ekosistem yang kuat. Kelebihan lintas platform, modularitas, dan dukungan luas menjadikan Qt sebagai salah satu framework terkemuka dalam dunia pengembangan perangkat lunak. Dan penelitian ini qt digunakan sebagai *basestation* dan juga sebagai tempat program metode multi-threading (Kusumastutie & Alif Fiolana, 2020).

2.5 TCP/IP(Transmission Control Protocol/Internet Protocol)

TCP/IP (Transmission Control Protocol/Internet Protocol) merupakan standar komunikasi data yang digunakan dalam proses tukar-menukar data dari satu komputer ke komputer lain. TCP/IP merupakan jaringan terbuka yang bersifat independen terhadap mekanisme transport pada jaringan fisik yang digunakan, sehingga dapat digunakan di mana saja. Protokol ini menggunakan skema pengalamatan yang sederhana yang disebut sebagai alamat IP (IP Address) yang mengizinkan banyak komputer untuk dapat saling berhubungan satu sama lainnya di Internet (Pamungkas et al., 2018).

Alamat IP (Internet Protocol) sebuah jenis pengalamatan jaringan unik yang di alokasikan untuk mengidentifikasi sebuah perangkat atau host yang digunakan di dalam protokol jaringan TCP/IP (Transmission Control Protocol/Internet Protocol). Alamat IP terdiri dari kombinasi angka, huruf, dan titik yang ditentukan berdasarkan versi alamat IP berdasarkan standar RFC (Request for Comments).

Setiap jaringan dapat mengalokasikan beberapa alamat IP, alamat IP dapat dikonfigurasi secara otomatis atau manual(Tjoanapessy et al., 2019).

2.6 UDP (User Datagram Protocol)

Salah satu protokol lapisan transportasi dalam model referensi Open Systems Interconnection (OSI) adalah User Datagram Protocol (UDP), yang menyediakan layanan tanpa koneksi dan tidak menjamin pengiriman paket data. UDP dianggap ringan dan efektif karena tidak memerlukan pembuatan koneksi sebelum mentransfer data, sehingga biayanya lebih rendah daripada Protocol Control Transmission (TCP). Tetapi UDP memiliki kekurangan utama, yaitu tidak memiliki mekanisme pengontrol kesalahan dan tidak menjamin pengiriman paket. Akibatnya, aplikasi yang menggunakan UDP harus menggunakan mekanisme pengontrol kesalahan mereka sendiri jika diperlukan. Pada penelitian ini UDP digunakan untuk komunikasi antara pc/laptop dengan mikrokontroler(Kusumastutie & Alif Fiolana, 2020).

2.7 ROS (Robot Operation System)

Robot Operating System (ROS) adalah sistem operasi berbasis Linux yang dibuat khusus untuk robot. ROS memungkinkan pengembang robot untuk membuat, mengembangkan, dan menjaga sistem robotik yang kompleks. ROS terdiri dari sekumpulan node yang masing-masing adalah sebuah program yang bertanggung jawab atas suatu fungsi tertentu. ROS Master adalah bus pesan yang memungkinkan node-node berkomunikasi satu sama lain. Dengan demikian, ROS memungkinkan pengembang untuk mengembangkan sistem robotik yang terdiri dari berbagai komponen yang saling terhubung(Jalil, 2018).



Gambar 2. 12 Logo Robot Operating System

ROS (Robot Operating System) adalah sebuah perangkat lunak open source yang memiliki kemampuan untuk melakukan simulasi robot. Karena ROS membuat membuat perangkat lunak untuk robot yang kompleks sangat sulit, perangkat lunak ini menyediakan berbagai tools dan library untuk pengembangan robot, mulai dari perancangan hingga pembuatan perilaku robot. Dari sudut pandang robot, masalah yang sebenarnya sangat sederhana bagi manusia sering menjadi sangat kompleks ketika dilakukan oleh robot. Oleh karena itu, ROS memberikan perangkat lunak ini secara gratis (open source) untuk menangani kesulitan perancangan robot. Disebabkan kebutuhan industri dan akademis, teknologi robot berkembang dengan cepat. Perkembangan ini mendorong banyak industri robot untuk membuat komponen seperti mikrokontroler, sensor, dan IC (Integrated Circuit) untuk perangkat keras robot. Selain perangkat keras, juga dikembangkan platform perangkat lunak yang mensimulasikan operasi robot, yang disebut ROS(Pratikto et al., 2021).

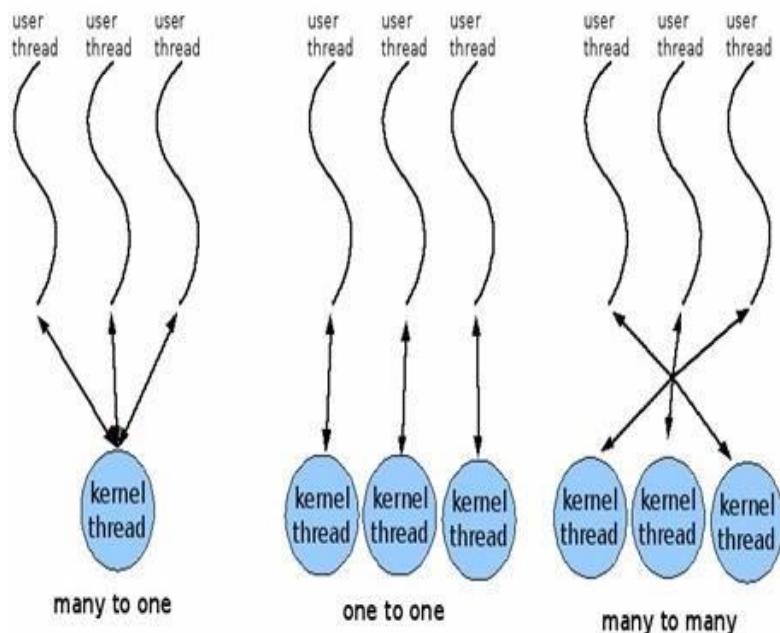
2.8 Metode Yang Digunakan

Dalam tugas akhir ini, peneliti menggunakan metode Multithreading untuk membuat thread pada masing-masing robot agar lebih mudah dalam mengirim data pada robot. *Thread* adalah sebuah alur kontrol dari sebuah proses. Kontrol single thread hanya memungkinkan proses untuk menjalankan satu tugas pada satu waktu. Banyak sistem operasi modern telah memiliki konsep yang dikembangkan agar memungkinkan sebuah proses untuk memiliki eksekusi multithreads.

2.8.1 Multithreading

Komputer dapat melakukan beberapa tugas sekaligus dengan multithreading. Bayangkan sebuah program sebagai pekerjaan besar.

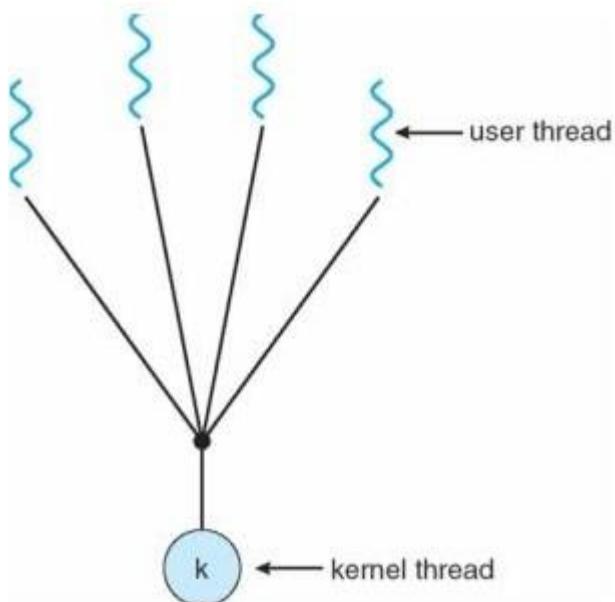
Multithreading dapat membagi program tersebut menjadi beberapa tugas kecil yang dilakukan oleh berbagai *thread* (Edy Surya Prabowo et al., 2022). Thread adalah sebuah proses yang berukuran kecil yang dibuat oleh sebuah program untuk dijalankan bersamaan dengan thread – thread lainnya. Tujuan thread ini adalah agar proses – proses yang dapat dikerjakan bersamaan dapat dijalankan secara bersamaan tanpa memerlukan waktu tunggu untuk proses berikutnya (Onggrono et al., 2017).



Gambar 2. 13 Model Thread

Dalam desain hubungan dalam suatu sistem atau basis data, terdapat tiga konsep utama: Many-to-Many, Many-to-One, dan One-to-One. Many-to-Many menggambarkan hubungan di mana setiap entitas pada satu sisi hubungan dapat terhubung dengan banyak entitas pada sisi lainnya, dan sebaliknya, menciptakan jaringan keterkaitan yang tidak terbatas. Di sisi lain, Many-to-One mencerminkan hubungan di mana banyak entitas dapat terhubung dengan satu entitas tertentu, mengindikasikan adanya satu entitas yang mungkin menjadi tujuan bagi banyak entitas lain. Sementara itu, One-to-One menciptakan hubungan yang bersifat unik, di mana setiap entitas pada satu sisi hanya terhubung dengan satu entitas pada sisi lainnya, dan sebaliknya. Konsep-konsep ini membantu menggambarkan pola keterkaitan

antara entitas dalam suatu lingkungan data atau sistem seperti pada gambar 2.14.



Gambar 2. 14 Many to One

Many-to-One adalah konsep hubungan dalam basis data di mana banyak entitas dari satu kelompok dapat terkait dengan satu entitas tunggal dari kelompok lainnya. Artinya, banyak entitas dapat memiliki keterkaitan atau hubungan dengan satu entitas tertentu. Dalam banyak kasus, Many-to-One digunakan ketika beberapa elemen atau anggota dari satu kategori dapat terhubung dengan satu elemen atau anggota tunggal dari kategori lainnya. Konsep ini memfasilitasi pengorganisasian data dengan cara yang sederhana dan mudah dimengerti, memungkinkan keterhubungan yang efisien antara entitas dalam suatu sistem atau basis data. Pada penelitian ini peneliti menggunakan model Many to One seperti pada gambar 2.14.

2.8.2 Odometry

Odometry merupakan metode yang menggunakan sensor berbasis posisi untuk memperkirakan perubahan posisi dari waktu ke waktu. Odometry ini akan memetakan posisi sumbu x dan sumbu y dalam sistem koordinat kartesian. Odometri derakan translasi robot sudah diketahui pasti pada robot beroda (Studi et al., 2023).

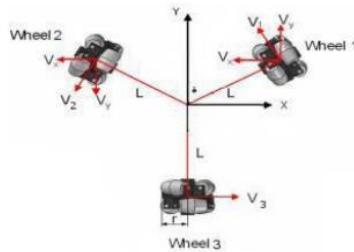
kartesian pada lapangan diperlukan matriks transformasi. Matriks tersebut berguna untuk mentransformasikan koordinat pada persamaan rotasi 2.15

$$K_{roda} = 2\pi r$$

$$Pulse_{per_{mm}} = resolusi_{inc} / K_{roda} \quad (2.15)$$

Dimana :

- K_{roda} = Keliling Roda Omni
- $Pulse_{per_{mm}}$ = Konversi jumlah pulsa ke milimeter



Gambar 2. 15 Representasi penempatan rotary encoder (Darmawan et al., 2023)

Koordinat X dan koordinat Y pada robot omni-directional tiga roda dapat diperoleh dengan cara mengganti kecepatan setiap roda dari sistem kinematik robot *omni-directional* tiga roda dengan jarak yang telah ditempuh masing-masing roda. Berdasarkan hasil perhitungan tersebut diperoleh persamaan

$$Sx = S3 - S1\cos(\theta1) - S2\cos(\theta2) \quad (2.16)$$

$$Sy = S1\cos(\theta1) - S2\cos(\theta2) \quad (2.17)$$

Dimana :

$S_i (1,2,3)$ = jarak tempuh dari masing-masing roda

θ = sudut roda omni terhadap sumbu acuan

Sx = Jarak tempuh robot pada sumbu x

Sy = Jarak tempuh robot pada sumbu y

Persamaan koordinat 2.16 dan 2.17 merupakan koordinat yang dihasilkan oleh robot. Apabila ingin mengaplikasikan pada koordinat kartesain pada lapangan diperlukan matriks transformasi. Matriks tersebut berguna untuk mentransformasikan koordinat pada persamaan rotasi 2.18

$$\begin{bmatrix} Y pos \\ X pos \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} Sy \\ Sx \end{bmatrix} \quad (2.18)$$

Dengan :

Ypos = posisi x robot dalam koordinat kartesian (mm)

Xpos = posisi y robot dalam koordinat kartesian (mm)

θ = arah hadap robot ($^{\circ}$)

Odometry memiliki kesalahan sistematis seperti diameter roda yang tidak sama, roda tidak berbentuk lingkaran sempurna, ketidakpastian titik gesekan pada roda, rotasi sensor encoder yang terbatas, dan juga sampling rate yang terbatas. Adapula kesalahan non-sistematis meliputi lantai yang tidak rata, selip pada roda, dan juga terganjal benda yang tidak terduga.

2.8.3 Teory Gyroscope

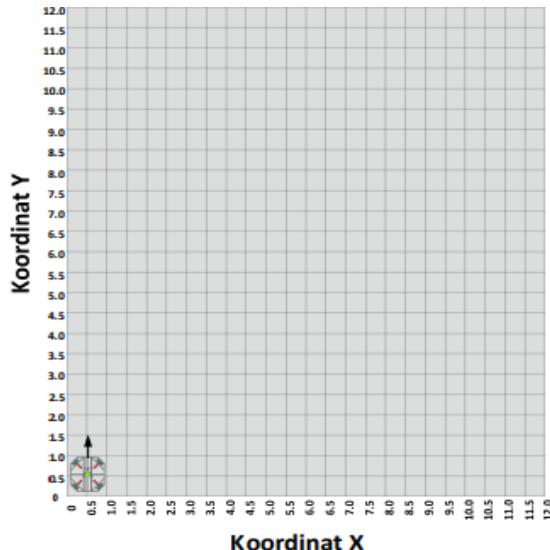
Gyroscope merupakan roda yang berputar pada titik pusat atau porosnya yang berguna untuk mengukur atau mempertahankan orientasi dari suatu objek. Poros roda terhubung dengan lingkaran di sekeliling roda atau disebut dengan gimbal. Gimbal satu dengan lainnya saling terhubung pada dasar lempengan. Hal tersebut menyebabkan gyroscope berputar secara stabil. Kemampuan gyroscope mempertahankan kedudukan karena tidak adanya gangguan gaya dari luar. Hal tersebut sesuai dengan hukum newton pertama yang dipaparkan ole Isaac Newton, dimana gaya total benda sama dengan nol jika benda tersebut diam atau pada keadaan setimbang. Prinsip dasar gyroscope dalam mengukur orientasi berupa momentum sudut yang memiliki kepekaan tinggi terhadap kecepatan sudut dari arah sumbu x, sumbu y, dan sumbu z. Setiap sumbu tersebut akan membentuk perubahan sudut yaitu sumbu x sebagai sudut phi (roll), sumbu y sebagai sudut theta (pitch), dan sumbu z sebagai sudut psi (yaw)(Zhang et al., 2017).

2.8.4 Gyrodometry

Gyrodometry merupakan metode yang mengkombinasikan data dari sensor gyroscope untuk penentuan arah hadap robot dan data posisi yang didapatkan dari metode odometry.

Gyrodometry mudah untuk diimplementasikan dan juga sangat efektif untuk mengurangi kesalahan non-systematic seperti lantai yang tidak rata

ketika menggunakan metode odometry. Pada penelitian ini, metode gyrodometry digunakan untuk menentukan posisi gerak robot dalam sumbu



kartesian x dan y menggunakan sensor rotary encoder dan penentuan arah hadap dengan sensor gyroscope.

Pada Gambar 2.16 Sensor gyroscope ditempatkan di tengah robot agar pembacaan arah hadap robot lebih presisi. Pada posisi awal (Gambar 2.16) robot diinisiasi terlebih dahulu sesuai koordinat dengan rotary encoder untuk penentuan posisi robot terhadap sumbu kartesian x dan y dengan koordinat $x = 0.5$ dan koordinat $y=0.5$. Pada lintasan lapangan dan kalibrasi sensor gyroscope untuk penentuan arah hadap robot sesuai posisi arah hadap robot $\theta= 0^\circ$ kemudian menentukan koordinat yang akan dituju oleh robo 10.0 , $y =$

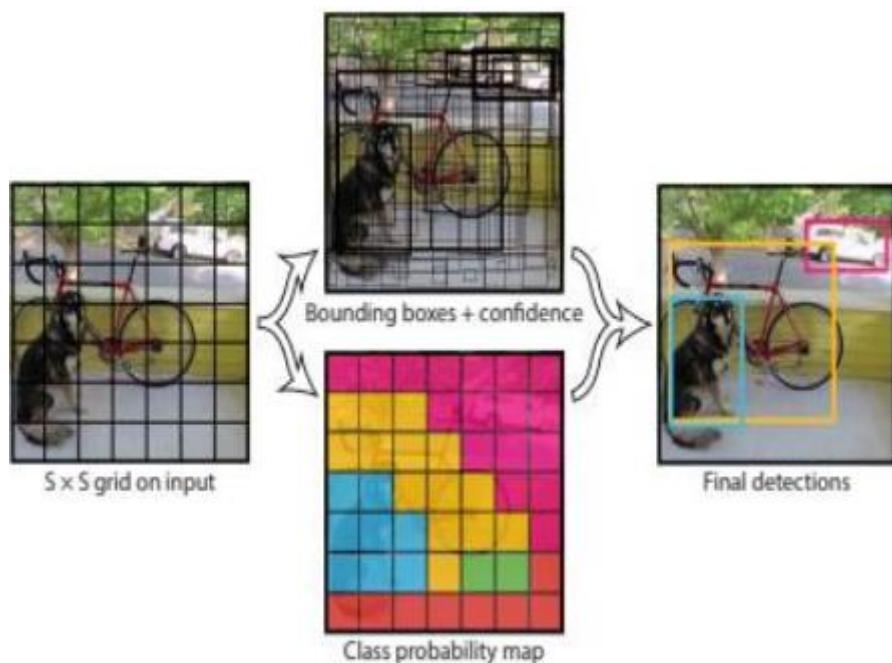
Gambar 2. 16 Koordinat robot dalam sumbu kartesian $x=0.5$ dan $y=0.5$ dan arah hadap robot $\theta=0^\circ$ (Safatain et al., 2022)

0.5, dan $\theta = 0^\circ$ t (Safatain et al., 2022).

2.8.5 You Only Look Once (YOLO)

You Only Look Once (YOLO) adalah sebuah algoritma pengembangan dari algoritma sebelumnya yaitu CNN (Convolutional Neural Network). Algoritma ini sengaja dikembangkan untuk melakukan pengenalan objek secara real-time, dimana YOLO dapat memproses gambar secara real-time pada kecepatan 45 frame per second (FPS). Sistem deteksi yang dilakukan adalah dengan menggunakan repurpose classifier atau localizer.

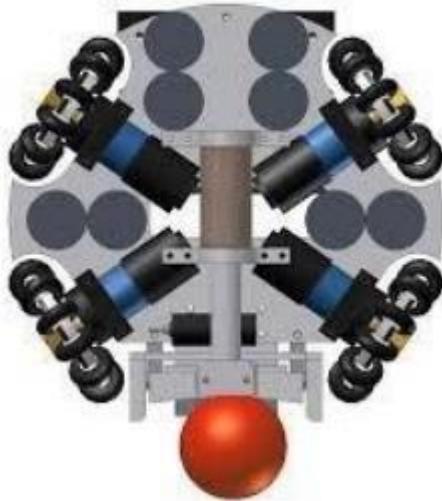
Sebuah model diterapkan pada citra di beberapa lokasi dan skala. Daerah dengan citra yang diberi score paling tinggi akan dianggap sebagai sebuah objek yang dideteksi (J. Redmon, S. Divvala, R. Girshick, 2016). Pendekatan pada metode YOLO sangat berbeda dengan metode sebelumnya. Yaitu dengan menerapkan jaringan syaraf tunggal pada keseluruhan gambar. Jaringan ini akan membagi gambar menjadi beberapa wilayah yang kemudian memprediksi kotak pembatas dan probabilitas pada setiap wilayah. Citra dibagi menjadi NxN grid. Selanjutnya memprediksi bounding box dan confidence pada 9 bounding box tersebut dan kelas probabilitas secara bersamaan. Sistem deteksi objek YOLO dapat dilihat pada gambar 2.1 (TAUFIQ dkk., 2023).



Gambar 2. 17 Deteksi Object YOLO
(FAIZ, 2023)

2.9 Sistem Penggerak Omni-Directional Empat Roda

Robot omni-directional empat roda adalah robot berbentuk persegi dengan roda omni yang diletakkan pada tiap ujungnya. Pada penelitian ini sistem penggerak utama akan menggunakan sistem penggerak omni-directional empat roda. Ilustrasi desain robot dapat dilihat dalam Gambar 2.19.



Gambar 2. 18 Ilustrasi Desain Robot Omni-directional Empat Roda

Dari Gambar 2.2 akan diperoleh persamaan kinematik yang digunakan pada sistem kontrol robot adalah:

$$V_x = -V_1\cos(\theta_1) - V_2\cos(\theta_2) + V_3\cos(\theta_3) + V_4\cos(\theta_4) \quad (2.21)$$

$$V_y = V_1\cos(\theta_1) - V_2\cos(\theta_2) - V_3\cos(\theta_3) + V_4\cos(\theta_4) \quad (2.22)$$

$$V_\theta = \frac{V_1}{R} + \frac{V_2}{R} + \frac{V_3}{R} + \frac{V_4}{R} \quad (2.23)$$

$$V_{i(1,2,3,4)} = \omega \cdot r \quad (2.24)$$

Dengan :

$V1$ = Kecepatan roda 1

$V2$ = Kecepatan roda 2

$V3$ = Kecepatan roda 3

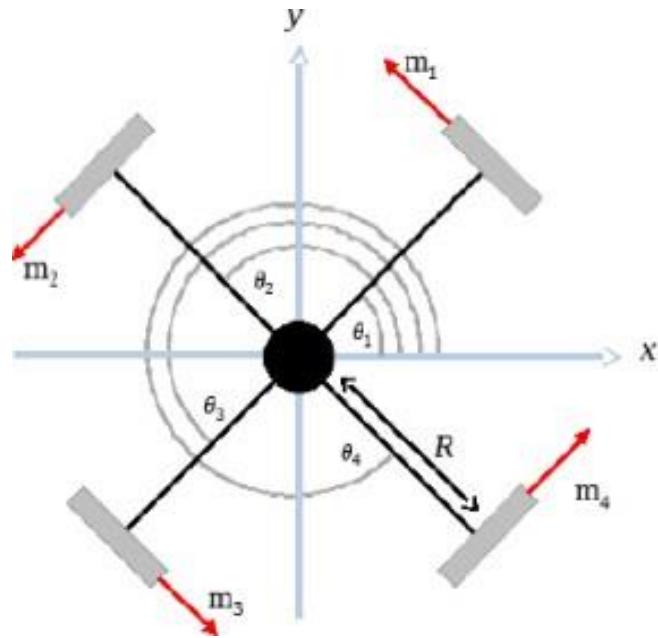
$V4$ = Kecepatan roda 4

Vx = Kecepatan robot pada arah sumbu X

Vy = Kecepatan roda pada arah sumbu Y

$V\theta$ = Kecepatan sudut robot

ω = Kecepatan angular dari roda (rad/detik) R = Jari-jari robot (cm) r = Jari-jari robot roda omni (cm)



Gambar 2. 19 Representasi Kinematika dari Sistem Pergerakan Omni-directional Empat Roda
 (Hidayat, 2023)

Dari Gambar 2.19 diketahui setiap roda disusun secara simetris dengan perbedaan sudut setiap roda (90°). Kecepatan tiap roda dapat dihitung pada (2.25). (Hidayat, 2023)

$$V_{i(1,2,3,4)} = -V_x \sin(\theta_i) + V_y \cos(\theta_i) + R\omega \quad (2.25)$$

BAB 3

METODE PENELITIAN

3.1 Konsep Penelitian

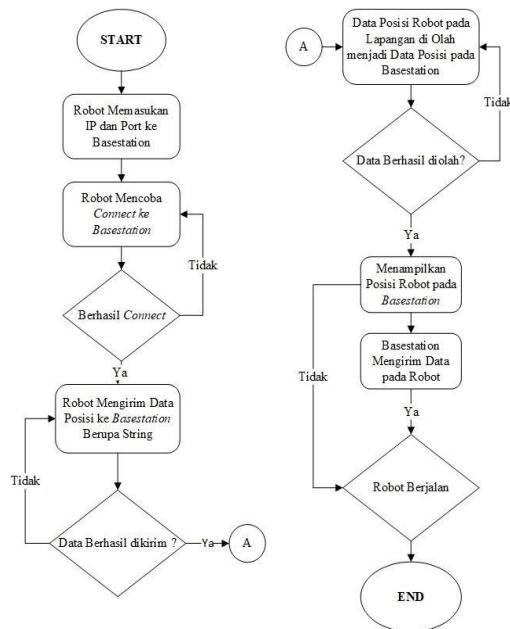
Konsep penelitian digunakan sebagai landasan oleh penulis untuk menyelesaikan penelitian untuk tugas akhir ini. Fokus pada penelitian ini yaitu komunikasi basestation pada robot sepakbola beroda.

3.1.1 State of The Art (Keterbaruan dari Penelitian)

Berdasarkan penelitian sebelumnya berjudul “Singkronisasi Protokol UDP Untuk Komunikasi Robot Sepakbola Indonesia” (Maulidina, 2019) keterbaruan dari penelitian sebelumnya adalah membuat basestation pada sistem operasi linux menggunakan metode multithread agar robot dapat menjalankan perintah dan dapat mengolah strategi lebih baik.

3.1.2 Prinsip Kerja Sistem

Pada prinsip kerja sistem ini akan mencerminkan gambaran keseluruhan sistem yang akan dikembangkan. Gambar 3.1 merupakan alur diagram sistem basestation.

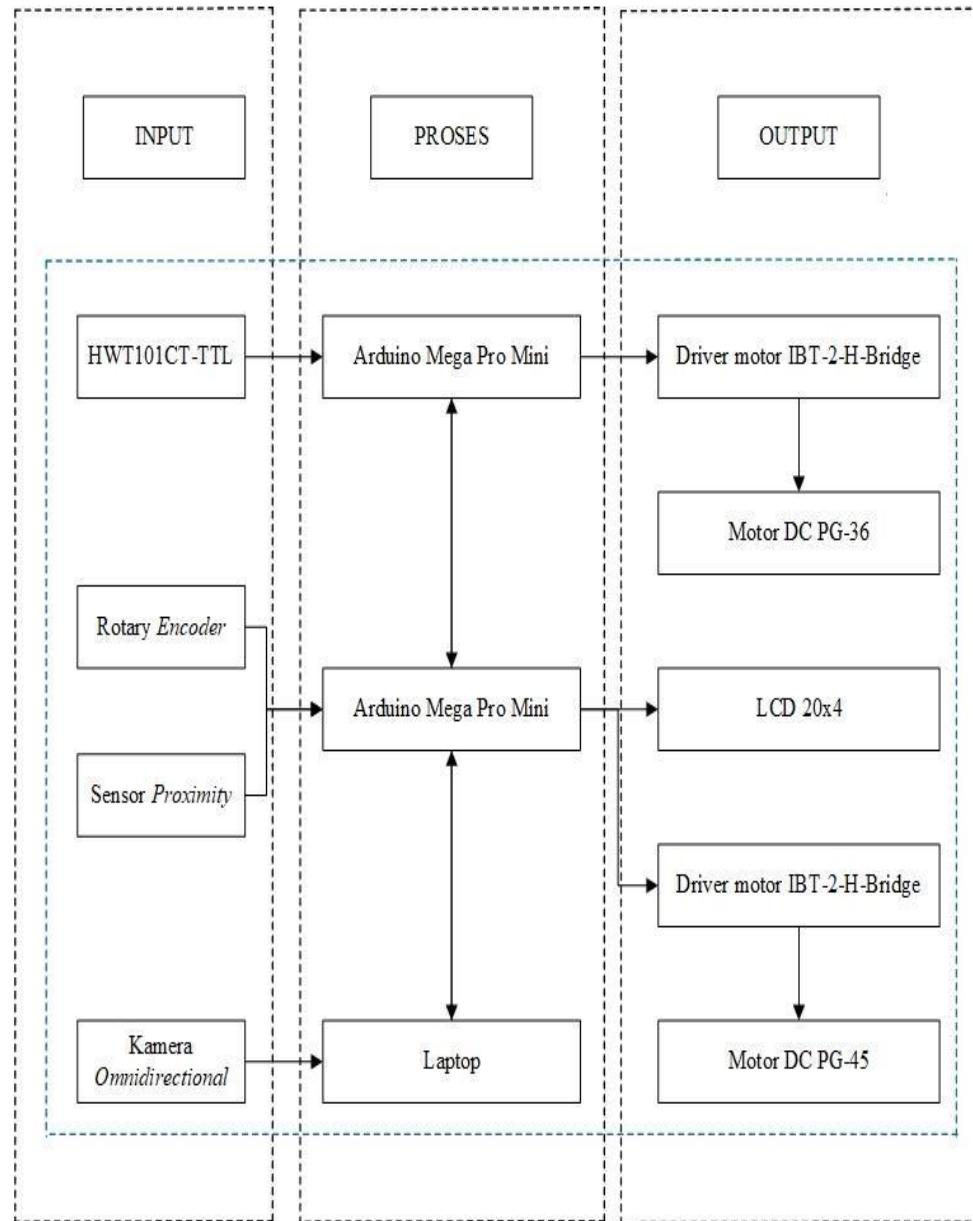


Gambar 3. 1 Diagram Alir Sistem pada Basestation

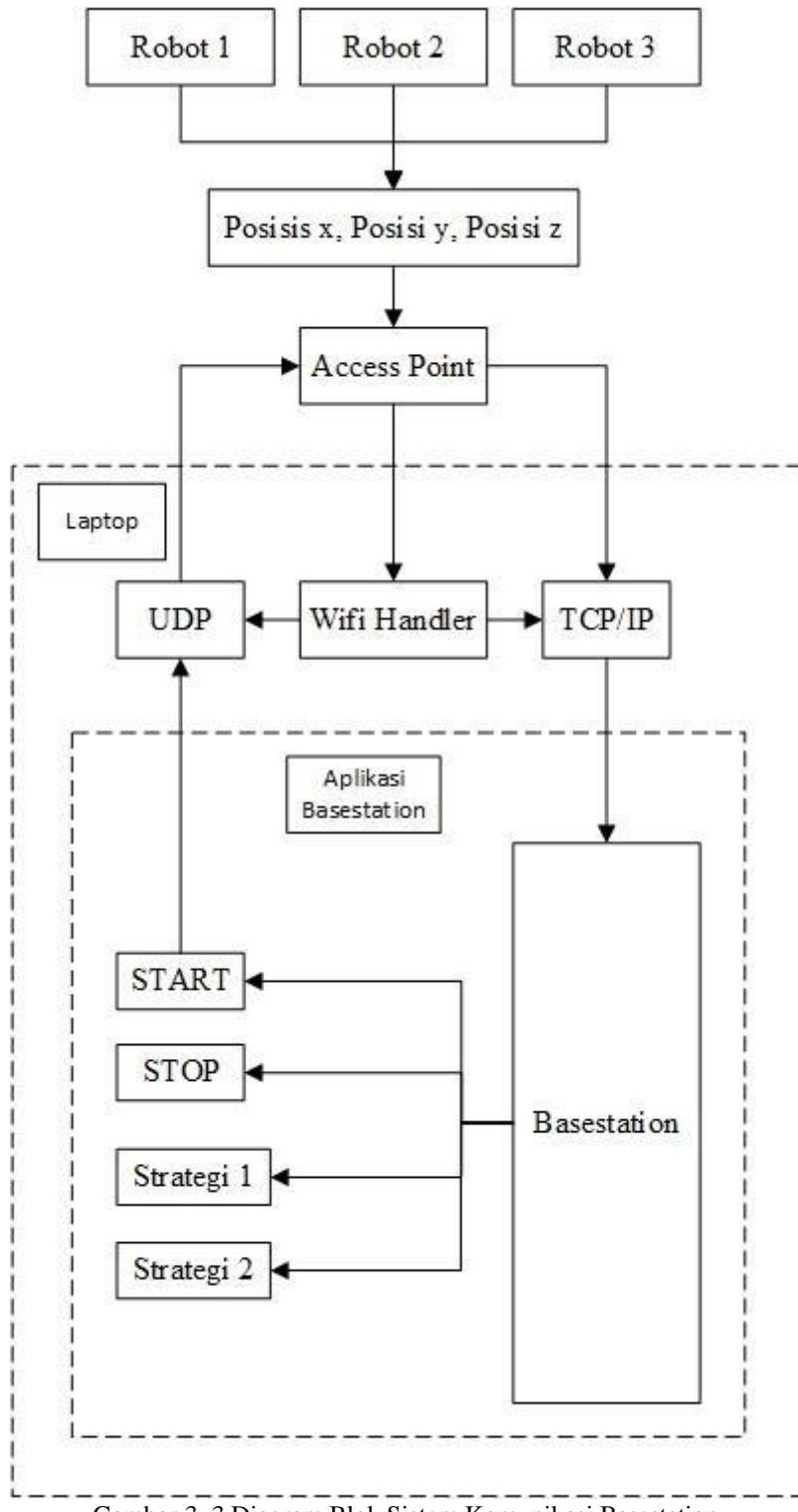
Diagram alir sistem pada *basestation* seperti pada Gambar 3.1 menjelaskan dimulai dari memasukan ip dan juga port robot pada basestation

agar basestation dan juga robot dapat terhubung. Selanjutnya data robot akan di terima oleh basestation berupa posisi robot pada lapangan yang akan di olah pada simulasi.

3.1.3 Diagram Blok Sistem



Gambar 3. 2 Diagram Blok Sistem Robot

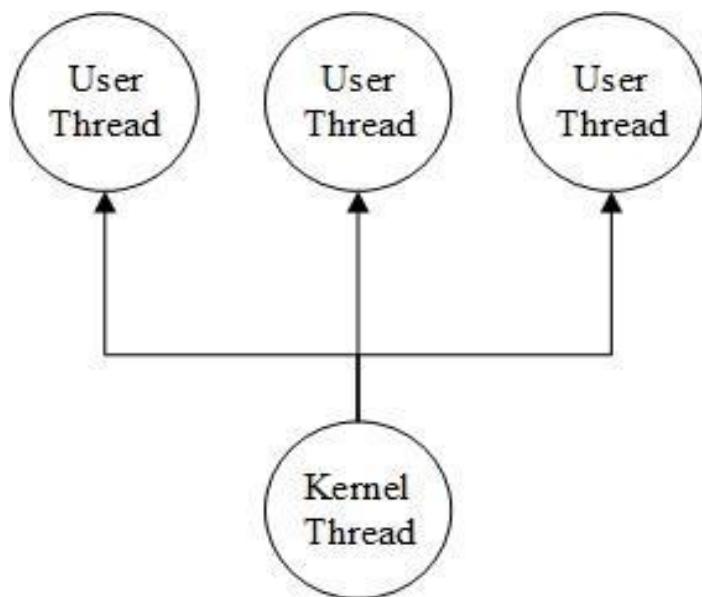


Gambar 3. 3 Diagram Blok Sistem Komunikasi Basestation.

Sistem terdiri dari tiga robot sepak bola beroda yang dapat berfungsi sebagai robot kawan, seperti yang ditunjukkan pada Gambar 3.2. Masukan, proses, dan keluaran adalah komponen sistem yang ditunjukkan dalam diagram tersebut. Sebagai sensor utama pada robot untuk penelitian ini,

terdapat rotary encoder HWT101CT, kamera omnidirectional, kamera proximity, dan kamera webcam C922. Rotary encoder berfungsi sebagai pengukur perpindahan robot dengan memanfaatkan jumlah putaran rota omni yang terpasang pada shaft sensor. Sensor HWT101CT berfungsi sebagai pengukur sudut orientasi robot, sensor dekat berfungsi sebagai pendekripsi bola pada robot, dan kamera omnidirectional yang mengambil gambar diteruskan ke laptop untuk diproses gambar, yang memungkinkan untuk mendekripsi lokasi bola dan robot lawan. Proses pengolahan data HWT101CT dilakukan dengan cara yang sama. STM32F4 akan menangani sensor encoder rotasi dan menggunakan fitur timer pada setiap channel yang tersedia. Selanjutnya, metode gyrodometry akan diterapkan untuk mengotorisasi data sensor ke STM32F4. Namun, sensor jarak dekat digunakan untuk menemukan bola pada penggiring bola ketika robot mendapatkan bola. Selain berfungsi sebagai pengolahan gambar, laptop juga dapat melakukan pengiriman dan penerimaan data dari STM32F4 ke basestation melalui router. Robot berkomunikasi dengan base station melalui protokol komunikasi IP multithread, dan perhitungan trigonometri digunakan base station untuk mengetahui sudut robot dengan gawang lawan. Hasil perhitungan trigonometri ini dikirim kembali ke laptop. Sistem komunikasi seperti yang ditunjukkan pada Gambar 3.3. Dari tiga robot yaitu terdiri dari dua robot penyerang dan satu robot kiper mengirimkan data yang akan di terima oleh basestation melalui komunikasi UDP dan TCP yang kemudian data akan di teruskan untuk mengirim data start, stop, strategi 1, dan strategi 2 yang akan dikirim melalui protokol komunikasi UDP ke semua robot. Pada penelitian ini proses ini yang akan menggunakan metode multithread dimana basestation akan mengirim beberapa tugas dalam proses yang bersamaan.

3.1.4 Diagram Many to Many Model



Gambar 3. 4 Diagram Many to Many model

3.1.5 User Thread

User thread, atau yang disebut juga sebagai user-level thread, adalah thread yang sepenuhnya diimplementasikan dan dikelola di ruang pengguna (user space) suatu program atau aplikasi, tanpa campur tangan langsung dari kernel sistem operasi. Thread pengguna ini diinisiasi, diatur, dan dihancurkan oleh program atau pustaka threading di tingkat pengguna, dan tidak memerlukan dukungan langsung dari kernel. Manajemen thread pengguna terjadi di lapisan perangkat lunak di atas kernel.

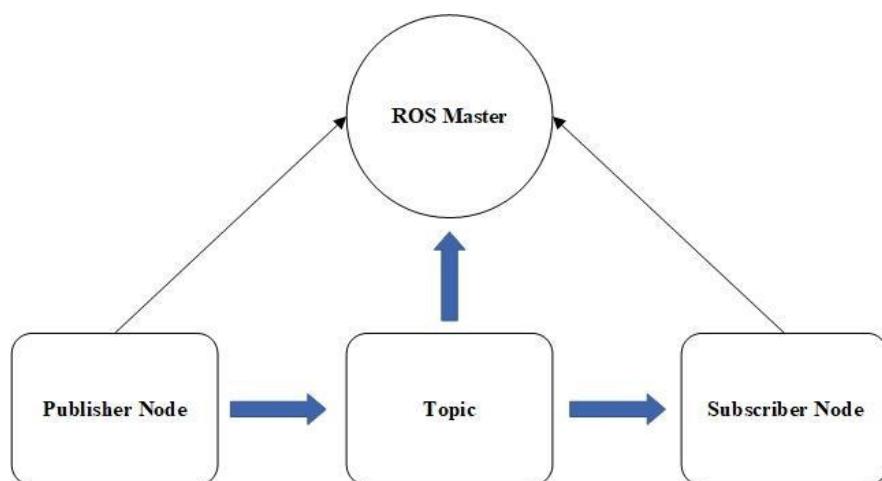
Pada model many-to-one dalam konteks threading, yang umumnya mengacu pada banyak thread pengguna yang diatur oleh satu thread kernel yang digunakan sebagai *basestation*, user thread pada robot dapat diimplementasikan menggunakan thread pengguna (user-level thread). Dalam hal ini, beberapa thread pengguna robot dihubungkan atau diatur oleh satu thread kernel atau *basestation*. Ini memiliki dampak tertentu pada manajemen thread dan eksekusi pada robot.

3.1.6 Kernel Thread

Kernel thread adalah jenis thread yang dikelola secara langsung oleh kernel sistem operasi. Salah satu keunggulan utama kernel thread adalah kemampuannya untuk melakukan operasi yang memblokir tanpa menghentikan thread lain dalam proses yang sama, memungkinkan tingkat konkurensi yang lebih tinggi. Kernel thread sering digunakan untuk menangani operasi I/O yang memakan waktu lama atau kejadian kernel tertentu, meningkatkan responsivitas sistem. Karena dikelola langsung oleh kernel, mereka dapat memberikan tingkat keamanan dan stabilitas yang tinggi.

Dalam penelitian ini komunikasi robot menggunakan model many-to-one, kernel thread mengacu pada implementasi di mana satu thread kernel mengendalikan dan mengatur beberapa thread user pengguna pada robot. Model many-to-one memberikan efisiensi dalam pengelolaan thread oleh kernel, karena satu thread kernel memiliki kontrol langsung terhadap sejumlah thread user.

3.1.7 Diagram Blok Jaringan ROS (Robot Operation System)



Gambar 3. 5 Diagram Blok Konfigurasi jaringan ROS

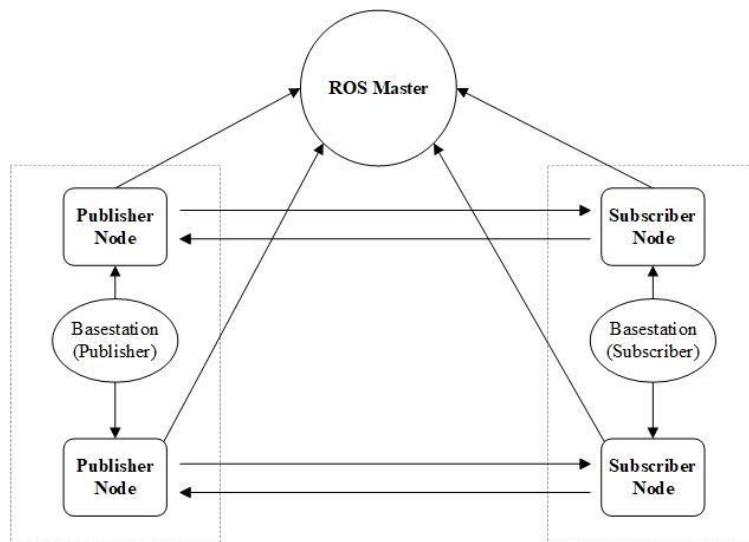
Seperti pada Gambar 3.5 bahwa komunikasi jaringan pada ros diolah oleh ros master dimana ros master adalah sebuah entitas dalam arsitektur komunikasi ROS yang bertanggung jawab untuk mengoordinasikan komunikasi antar node di dalam sistem robotika. Dimana Publisher adalah

entitas yang bertanggung jawab untuk mempublikasikan informasi ke dalam sistem. Informasi ini biasanya disebut sebagai "pesan" dan dapat berupa data sensor, perintah, atau informasi lainnya yang dibutuhkan oleh sistem dan subscriber adalah entitas yang berlangganan untuk menerima informasi atau pesan yang dipublikasikan oleh publisher. Subscriber menyatakan minatnya pada topik atau jenis informasi tertentu dan kemudian menerima pesan yang sesuai saat diterbitkan. Sedangkan topic adalah saluran komunikasi yang digunakan oleh publisher untuk mempublikasikan pesan dan oleh subscriber untuk berlangganan pesan. Topik memiliki nama unik dan menyediakan cara untuk mengorganisir dan mengarahkan aliran informasi di dalam sistem.

Dalam model publish-subscribe di ROS, proses umum dimulai dengan penerbit (publisher) yang mendaftarkan dirinya ke ROS Master dan menyatakan niatnya untuk mempublikasikan pesan pada suatu topik tertentu. Seiring itu, penyeimbang (subscriber) juga mendaftarkan dirinya ke ROS Master, mengindikasikan minatnya pada topik atau jenis pesan tertentu. Proses ini memungkinkan ROS Master memahami ketersediaan dan minat komponen-komponen dalam sistem.

Ketika penerbit mengirimkan pesan ke topik yang telah didaftarkan, ROS Master memainkan peran sentral dalam mengoordinasikan proses ini. ROS Master membantu menyampaikan pesan tersebut kepada semua penyeimbang yang telah berlangganan ke topik tersebut, memastikan bahwa informasi yang dipublikasikan dapat diterima oleh komponen-komponen yang berminat. Dengan demikian, model publish-subscribe memfasilitasi komunikasi terdesentralisasi dan fleksibel di dalam sistem ROS, memungkinkan komponen-komponen untuk berinteraksi secara efisien tanpa ketergantungan langsung satu sama lain.

3.1.8 Diagram Blok Peer to Peer Ros Architecture



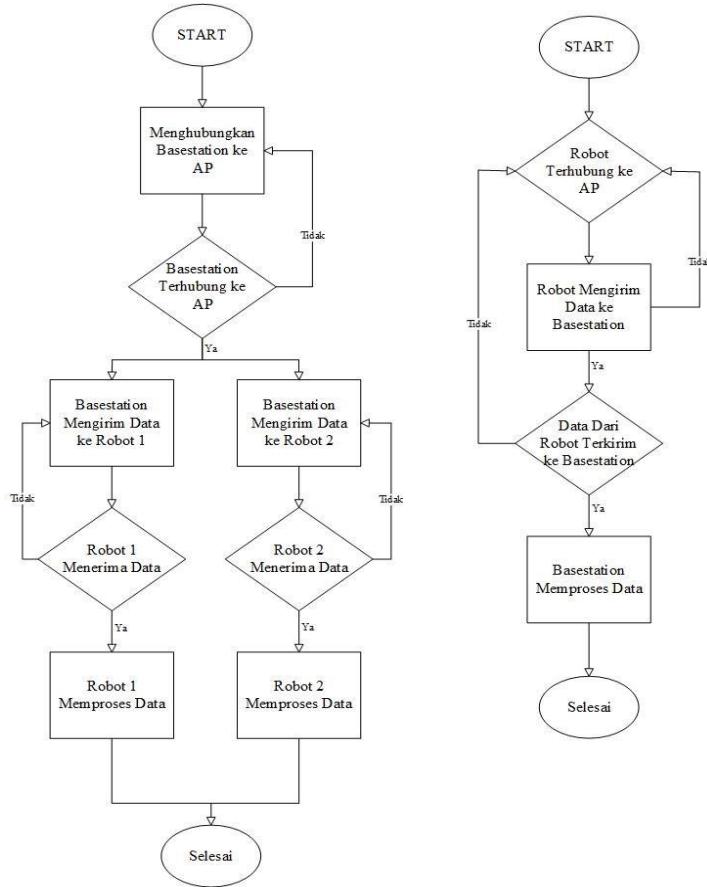
Gambar 3. 6 Diagram Blok Peer to Peer ROS Architecture

Peer-to-peer (P2P) merupakan salah satu teknologi komputasi terdistribusi yang memberi kemudahan sebuah peer untuk melakukan koneksi dengan peer lain secara langsung dan melakukan pertukaran data antara dua komputer atau peer dalam jaringan tanpa perantara. Sebelumnya, peer-to-peer dikenal sebagai konsep host-to-host saat pertama kali digunakan untuk komunikasi setara antara dua peer. Pada konsep ini, setiap peer dapat bertindak sebagai server dan client secara bersamaan(Putri, 2022).

Jaringan peer to peer biasanya diterapkan pada jaringan dengan skala kecil, 2 hingga 10 komputer. Tujuan penggunaan jaringan peer to peer yang paling utama adalah penggunaan program, data secara bersama-sama. Untuk keamanan, setiap user komputer bertanggung jawab terhadap keamanan komputernya masing-masing(Saputra, 2018).

Seperti pada gambar 3.6 dimana *basestation* pada penelitian ini berperan sebagai publisher dan juga subscriber dimana basestation sebagai *publisher* sebagai pengirim data seperti start, stop, strategi dan lain sebagainya pada robot, sedangkan *basestation* yang berperan sebagai *subscriber* berfungsi sebagai penerima data seperti x,y robot kawan dan juga sudut robot kawan dan juga x,y dari robot lawan.

3.1.9 Protokol Data Masuk dan Keluar UDP dan TCP



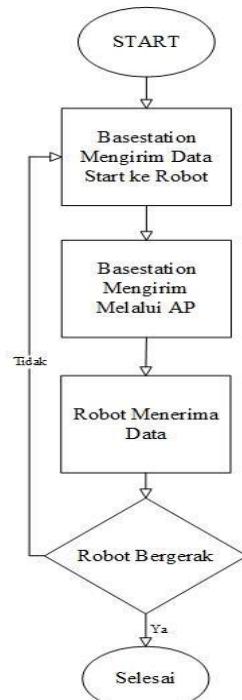
Gambar 3. 7 Protokol Komunikasi UDP dan TCP

Pada penelitian ini data yang masuk pada basestation dari robot menggunakan protokol komunikasi UDP (User Datagram Protocol) seperti pada Gambar 3.7 sebelah kiri dimana banyaknya jumlah data yang masuk seperti sensor yang ada pada tiga robot, dan karena data yang dikirim bersifat realtime penggunaan protokol UDP digunakan karena dalam UDP client atau robot tidak membangun koneksi dengan server atau basestation, melainkan client hanya mengirim paket data ke server tanpa mengecek apakah server tersebut telah siap atau tidak. Sama halnya dengan server tidak menerima koneksi dengan fungsi accept, namun server hanya menjalankan perintah untuk menerima data, server akan terus menunggu sampai data diterima.

Sedangkan pada pengiriman data dari basestation ke robot menggunakan protokol TCP karena TCP menyediakan mekanisme untuk memastikan pengiriman data yang baik dan jika data yang dikirim tidak diterima dengan benar, TCP akan mencoba mengirim ulang data tersebut. Dikarenakan pengiriman data dari basestation ke robot mempengaruhi efisiensi dari pergerakan robot dan juga kontrol yang bagus maka pada penelitian ini peneliti menggunakan protokol komunikasi TCP pada pengiriman data dari basestation ke robot seperti pada Gambar 3.7 sebelah kanan.

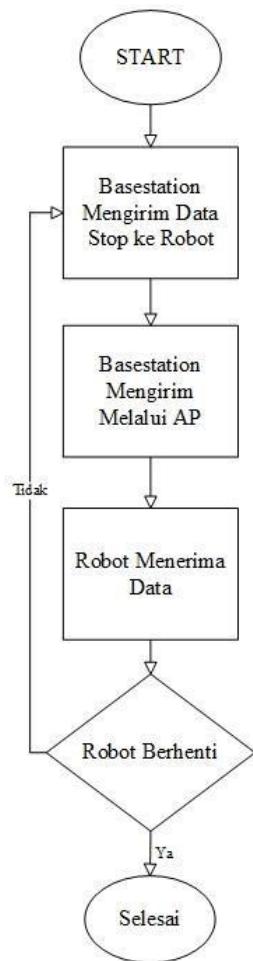
3.1.10 Flowchart dan Pengiriman Data Start dan Stop Robot

Alur pemrograman seperti pada Gambar 3.8 dijelaskan bagaimana basestation mengirim data start sampai dengan stop pada robot sepak bola beroda untuk penelitian ini. Seperti pada Gambar 3.8 pengiriman data ke robot menggunakan protokol komunikasi UDP dimana seperti pada Gambar 3.8 data start akan di kirim dari basestation ke robot dan ketika perintah dari basestation di terima oleh robot melalui access point maka robot akan bergerak sesuai perintah dari basestation.



Gambar 3. 8 Flowchart Pengiriman Data Start.

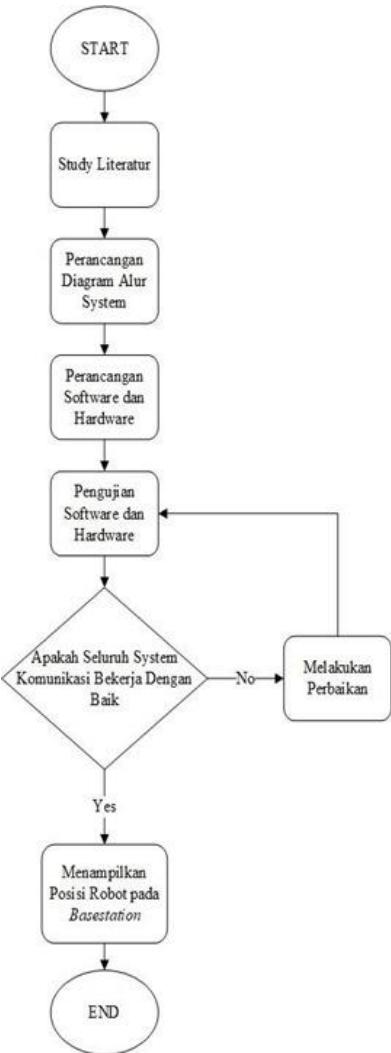
Seperti pada Gambar 3.9 pengiriman data ke robot menggunakan



Gambar 3. 9 Pengiriman Data Stop

protokol komunikasi UDP dimana seperti pada Gambar 3.9 data stop akan di kirim dari basestation ke robot dan ketika perintah dari basestation di terima oleh robot melalui access point maka robot akan berhenti sesuai perintah dari basestation.

3.2 Tahapan Penelitian

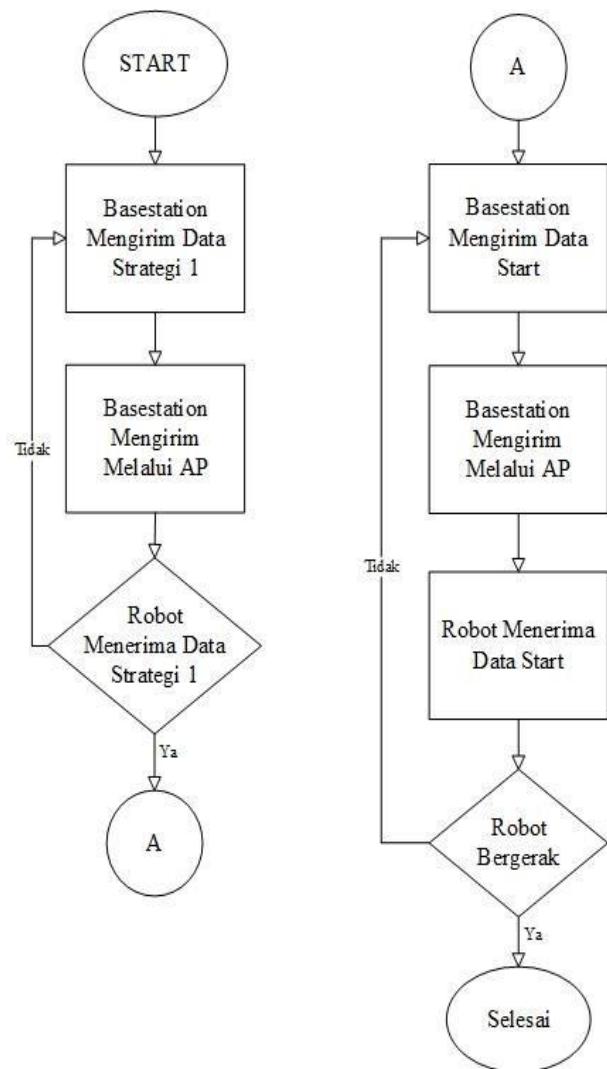


Gambar 3. 10 Flowchart Penelitian

Gambar 3.10 secara umum menunjukkan diagram alur penelitian yang dimulai dari identifikasi masalah komunikasi yang digunakan, kemudian menganalisa kebutuhan sistem untuk mengetahui kebutuhan seluruh sistem yang digunakan pada tugas akhir, kemudian dilanjutkan untuk perancangan robot sepak bola beroda yang digunakan. Setelah itu akan dilakukan pembuatan dan perancangan software dan hardware, lalu software dan hardware diuji coba, jika tidak berhasil, maka alur penelitian akan kembali ke pembuatan software dan hardware untuk diadakan perbaikan. Jika pada uji coba sudah berhasil, maka akan ditarik sebuah analisa dan kesimpulan mengenai Tugas Akhir “Optimasi Komunikasi pada Basestation Robot Sepakbola Beroda Menggunakan Metode

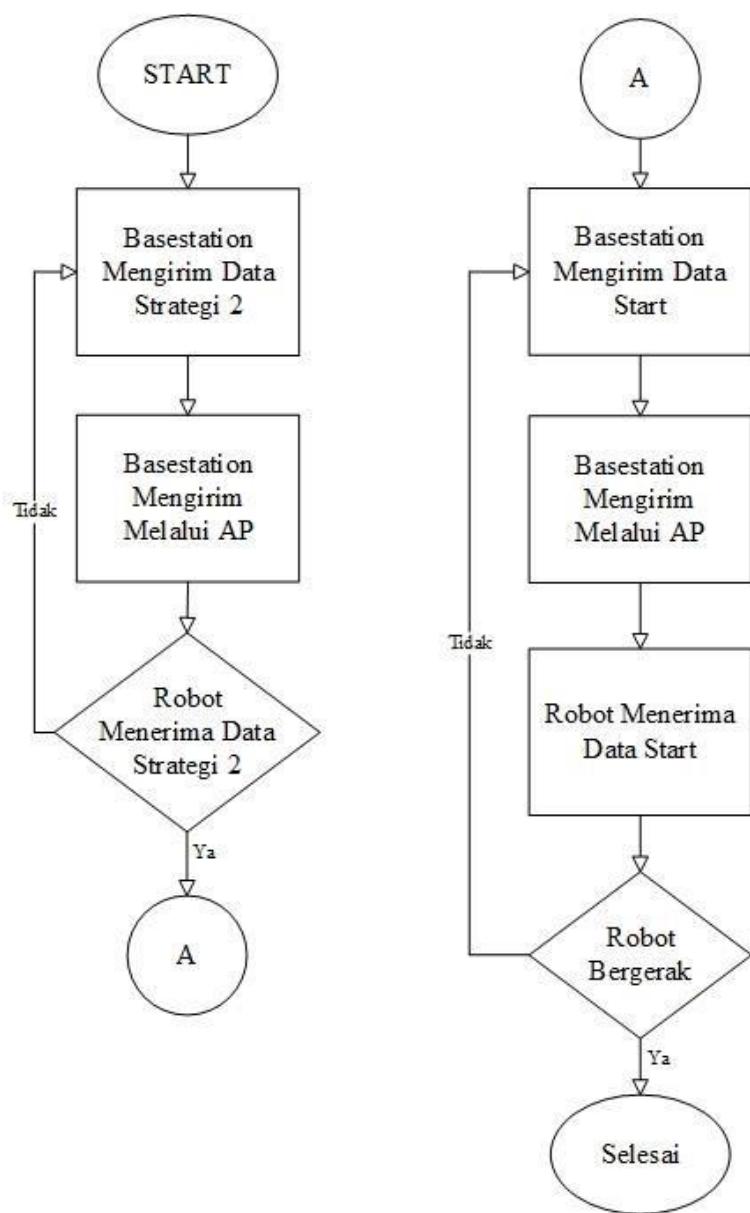
Multithreading Berbasis Ros”.

3.2.1 Flowchart Pengiriman Strategi



Gambar 3. 11 Pengiriman Data Strategi 1

Seperti pada Gambar 3.10 dimana strategi di kirimkan sebelum pengiriman data start pada robot sepak bola beroda, stelah pengiriman data strategi 1 menggunakan protokol komunikasi UDP dikirimkan dan robot menerima data strategi 1 maka basestation akan mengirimkan data start sebagai perintah robot untuk bergerak.



Gambar 3. 11 Pengiriman Data Strategi 2

Seperti pada Gambar 3.11 dimana strategi di kirimkan sebelum pengiriman data start pada robot sepak bola beroda, stelah pengiriman data strategi 2 menggunakan protokol komunikasi UDP dikirimkan dan robot menerima data strategi 2 maka basestation akan mengirimkan data start sebagai perintah robot untuk bergerak.

3.2.2 Identifikasi Masalah dan Studi Literatur

Sebelum melakukan tahapan perancangan pada penelitian ini, dilakukan studi literatur untuk mencari cara pemecahan masalah yang bersumber pada penelitian yang sudah dilakukan sebelumnya. Pada tahap ini, materi-materi yang menunjang penelitian dikumpulkan serta dipelajari dari berbagai sumber yang relevan, terpercaya dan dapat dipertanggung jawabkan. Literatur yang dikumpulkan akan difokuskan ke arah metode komunikasi *Multithreading* menggunakan sistem operasi ROS.

3.2.3 Analisis Kebutuhan Sistem

Pada tahap ini dilakukan analisis kebutuhan semua sistem tujuannya untuk mempermudah dalam melakukan perancangan, pembuatan, dan pengujian sistem. Kebutuhan sistem juga meliputi hardware dan software sebagai berikut:

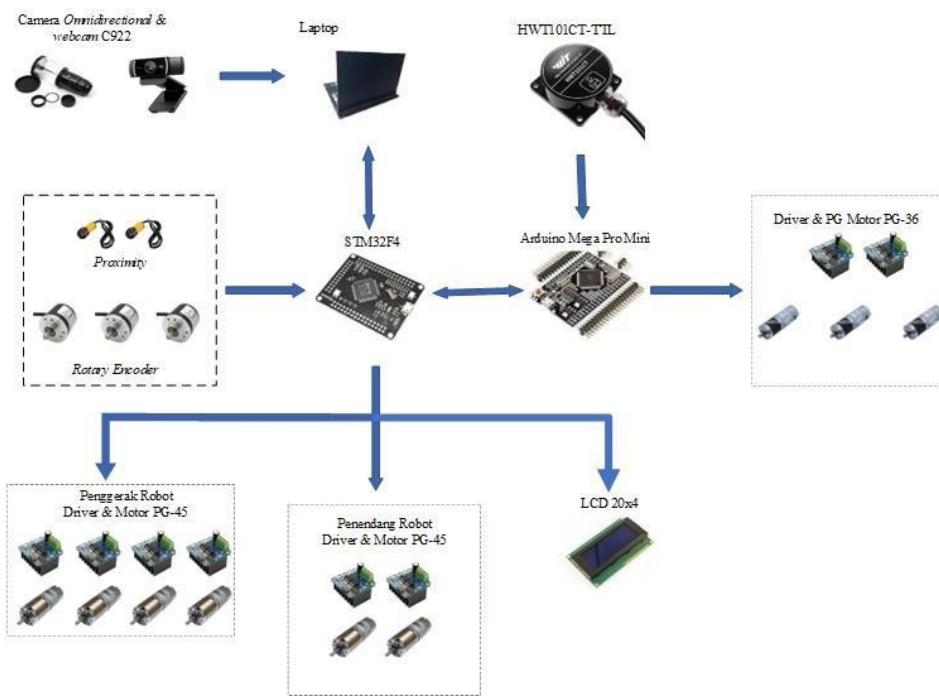
Tabel 3. 1 Analisa Kebutuhan Sistem

<i>Hardware</i>	<i>Software</i>	<i>Framework</i>
Kamera <i>Omnidirectional</i>	Ardiono IDE	ROS
Router Mercusys MR30G	Visual Studio Code	
Laptop	Keil uVision	
STM32F4	STM CubeMX	
Arduino Mega Pro Mini	QT(Q Toolkit)	
Sensor HWT101CT-TTL		
Sensor <i>Rotary Encoder</i>		
Sensor Proximity		
Motor DC PG-45		
Motor DC PG-36		
Driver IBT-2		
LCD 20x4		

3.3 Perencanaan dan Desain

Pada tahap ini dilakukan perancangan desain yang baik secara mekanik maupun elektrik dan juga software untuk alat yang akan dibuat. Selain pembuatan desain, pemodelan terhadap sistem diperlukan sebagai dasar pembuatan alat serta pembuatan *basestation* integrasi antar sistem maupun integrasi alat dengan sistem.

3.1.1 Perancangan *Hardware*



Gambar 3. 12 Perancangan Hardware

Perancangan robot sepak bola beroda pada penelitian ini akan menggunakan hardware seperti Gambar 3.12. Kamera omnidirectional sebagai input sensor utama yang diproses oleh laptop. Terdapat juga rotary encoder dan juga sensor HWT101CT sebagai input sensor yang akan diproses oleh mikrokontroller. Data dari rotary encoder akan diproses oleh STM32F4 sedangkan HWT101CT akan diproses oleh Arduino Mega Pro Mini. Lalu sensor proximity dan data dari Arduino Mega Pro Mini juga akan diproses oleh STM32F4. Dalam penelitian ini setiap robot akan memiliki satu buah STM32F4 DiyMore sebagai mikrokontroller utama, satu buah Arduino Mega Pro Mini, dan satu buah laptop sebagai perangkat yang melakukan

pemrosesan data. Dibagian output terdapat empat buah motor PG-45 beserta driver motor IBT-2-H-Bridge sebagai penggerak, dua buah motor PG-45 beserta driver motor IBT-2-H-Bridge sebagai penendang, dua buah motor PG-36 beserta driver motor IBT 2-H-Bridge sebagai penggiring serta satu buah LCD 20x4 sebagai tampilan data data robot.

3.1.2 Perancangan Basestation

Pada penelitian ini menggunakan QT(QTools) sebagai rancangan pembuatan desain tampilan basestation seperti pada Gambar 3.9.



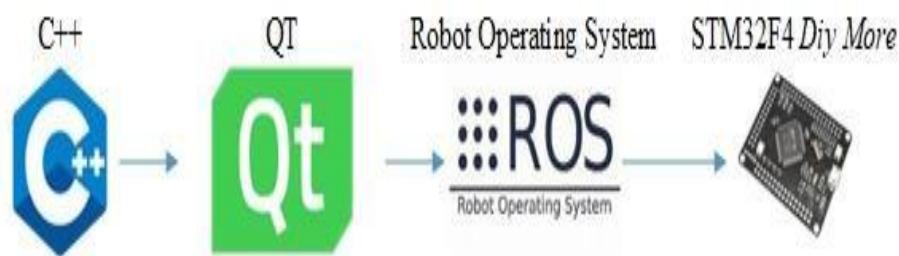
Gambar 3. 13 Desain Tampilan Basestation

Seperti pada Gambar 3.13 dimana ada tiga robot yaitu dua robot penyerang berwarna merah dan satu robot kiper berwarna biru. *Button* start dan stop pada bawah lapangan berfungsi sebagai instruksi pada robot untuk memulai dan menghentikan program. Pada tampilan bagian bawah terdapat data (XR1,YR1,TR1) sebagai monitoring posisi dan sudut dari robot penyerang satu, (XR2,YR2,TR2) sebagai monitoring posisi dan sudut robot penyerang dua dan (XR3,YR3,TR3) sebagai monitoring posisi dan sudut pada robot kiper. Sedangkan (S1,S2,S3) merupakan *button* yang di gunakan untuk mengirim strategi mana yang akan di gunakan.

3.3.1 Perancangan Software

Perancangan software bertujuan untuk merencanakan sistem perangkat

lunak sesuai dengan kebutuhan penelitian. Selain itu, perancangan sistem juga perancangan juga bertujuan agar penelitian mempunyai arsitektur sistem yang tepat dan efisien. Oleh karena itu, tahap perancangan mutlak dilakukan guna perencanaan sistem perangkat lunak yang komprehensif sebelum proses pengkodean program dimulai sehingga selanjutnya pengembangan sistem dapat terstruktur dan sesuai dengan rencana.

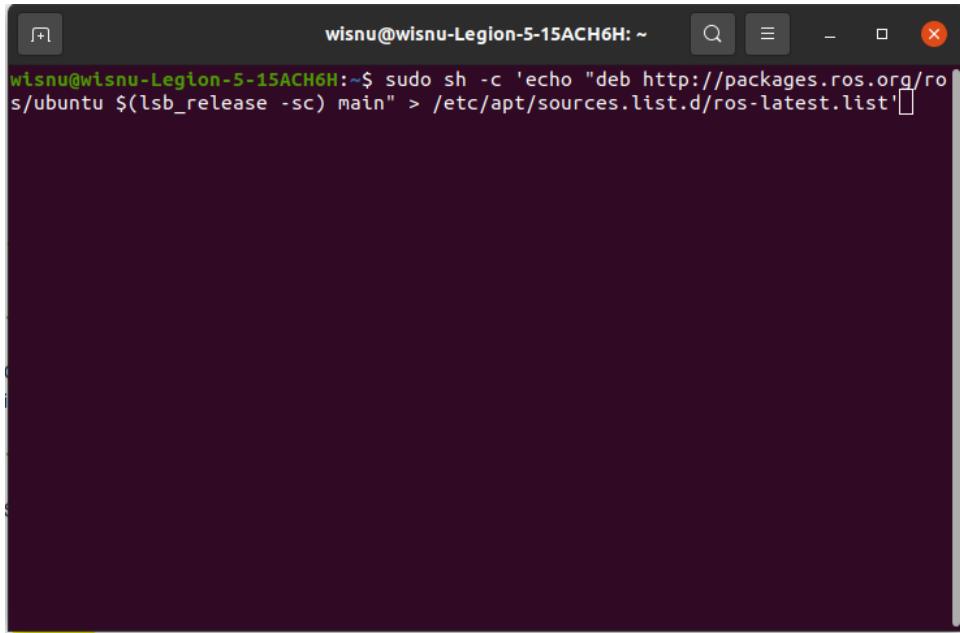


Gambar 3. 14 Perancangan Software

Pada penelitian ini pembuatan *interface* untuk basestation menggunakan aplikasi QT menggunakan bahasa pemrograman C++ yang kemudian komunikasi data di kelola oleh ROS (Robot Operating System). Data yang kirim pada ros akan di teruskan ke mikrokontroler melalui komunikasi ROS.

3.3.2 Installasi ROS

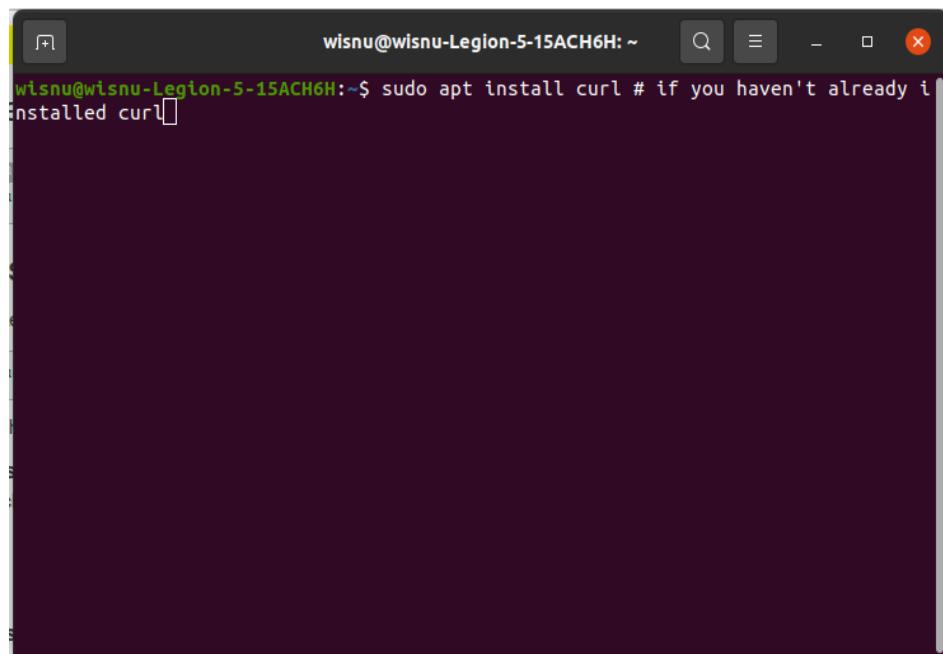
Pada penelitian ini peneliti menggunakan ROS1 neotic sebagai sistem operasi untuk menjalankan basestation. Dan pada bab kali ini akan di jelaskan bagaimana cara install ROS (Robot Operating System) sebagai berikut:



```
wisnu@wisnu-Legion-5-15ACH6H:~$ sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu $(lsb_release -sc) main" > /etc/apt/sources.list.d/ROS-Latest.list'
```

Gambar 3. 15 Code Repository ROS

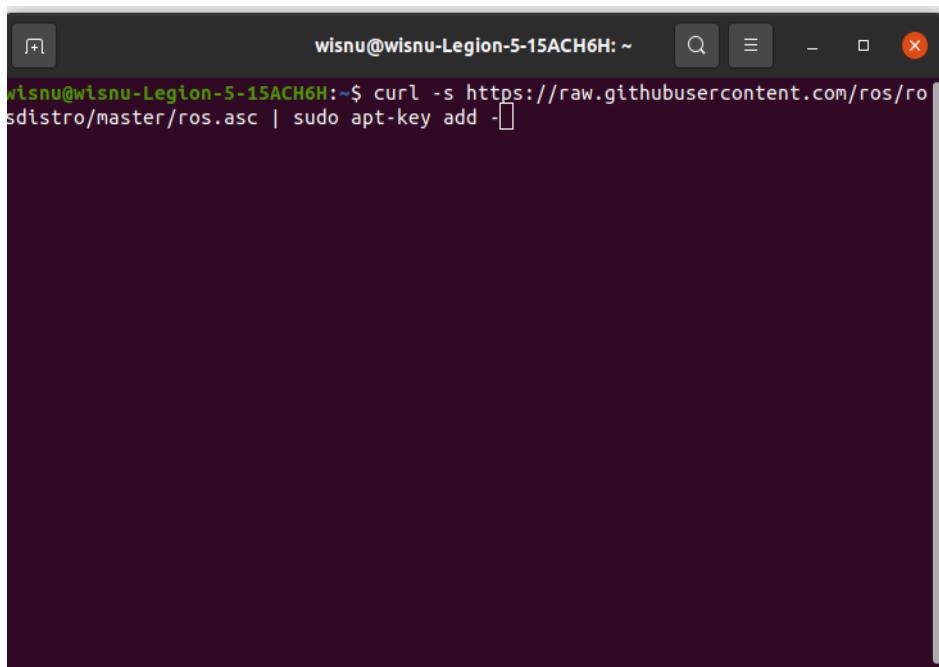
Pertama perintah untuk menambahkan repository ROS (Robot Operating System) ke daftar sumber paket pada sistem operasi berbasis Ubuntu. Seperti pada gambar 3.15 adalah code untuk menambahkan repository ROS pada ubuntu.



```
wisnu@wisnu-Legion-5-15ACH6H:~$ sudo apt install curl # if you haven't already installed curl
```

Gambar 3. 16 Install Perangkat Lunak Curl

Perintah sudo apt install curl yang seperti pada gambar 3.16 digunakan untuk menginstal perangkat lunak curl jika belum terinstal pada sistem. Curl adalah utilitas baris perintah yang digunakan untuk mentransfer data dengan menggunakan berbagai protokol, termasuk HTTP, HTTPS, FTP, FTPS, SCP, SFTP, LDAP, dan banyak lagi. Ini dapat digunakan untuk mengunduh atau mengunggah data melalui berbagai protokol jaringan.

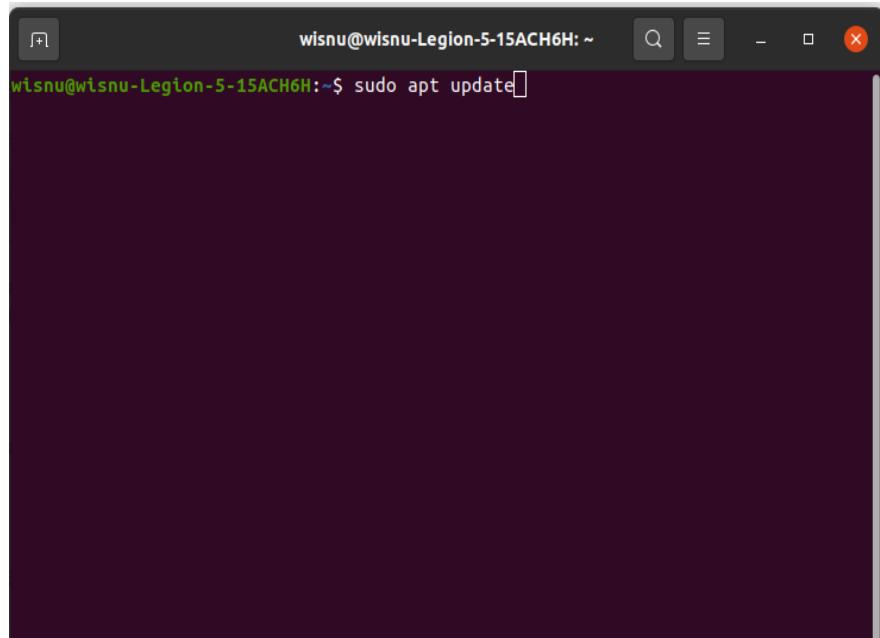
A screenshot of a terminal window titled "Terminal". The title bar shows the user "wisnu@wisnu-Legion-5-15ACH6H: ~". The main area of the terminal contains the following command:

```
wisnu@wisnu-Legion-5-15ACH6H:~$ curl -s https://raw.githubusercontent.com/ros/rosdistro/master/ros.asc | sudo apt-key add -
```

The terminal window has a dark background and light-colored text. It includes standard terminal icons at the top right: a magnifying glass for search, a list icon, a minus sign, a square, and a red X.

Gambar 3. 17 Impor kunci GPG (GNU Privacy Guard)

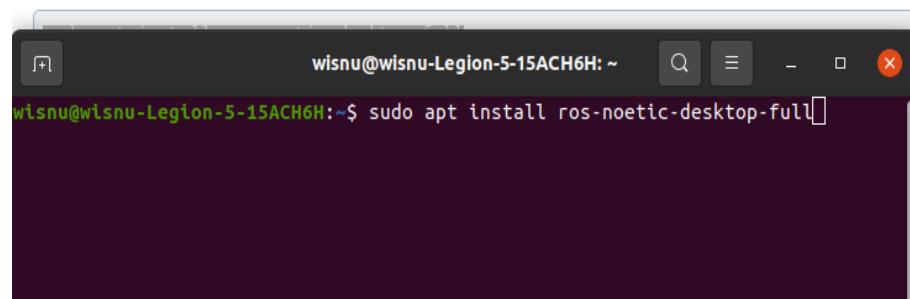
Perintah seperti Gambar 3.17 digunakan untuk mengimpor kunci GPG (GNU Privacy Guard) yang diperlukan untuk memverifikasi keaslian paket-paket yang akan diunduh dan diinstal dari repository ROS. Perintah ini mengunduh kunci GPG dari repository ROS di GitHub dan menambahkannya ke daftar kunci yang digunakan oleh sistem untuk memverifikasi integritas paket-paket yang diunduh dari repository ROS. Ini merupakan langkah yang umum dalam proses instalasi dan pembaruan paket pada sistem yang menggunakan manajer paket APT di lingkungan berbasis Debian, seperti Ubuntu.

A screenshot of a terminal window titled "wisnu@wisnu-Legion-5-15ACH6H: ~". The window has standard Linux-style window controls at the top. In the terminal, the command "sudo apt update" is typed and highlighted in green, indicating it is the current input. The rest of the screen is dark and mostly empty.

Gambar 3. 18 Update Ubuntu

Setelah menjalankan perintah untuk mengimpor kunci GPG dengan sudo apt-key add -, disarankan untuk menjalankan sudo apt update. Ini karena proses apt-key add untuk menambahkan kunci GPG hanya memperbarui daftar kunci pada sistem. Namun, daftar sumber paket (package sources) pada sistem belum diperbarui dengan informasi terbaru dari repository, termasuk informasi mengenai paket-paket yang dapat diunduh.

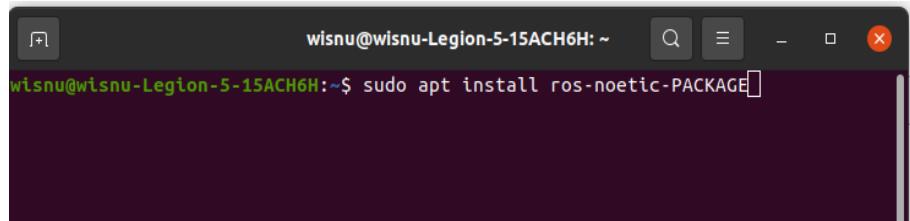
Sudo apt update akan mengambil informasi terbaru dari semua sumber paket yang terdaftar pada sistem, termasuk repository ROS yang baru saja Anda tambahkan. Dengan melakukan apt update, sistem akan mengetahui paket-paket terbaru yang tersedia di repository tersebut dan menyinkronkannya dengan daftar paket lokal di sistem.

A screenshot of a terminal window titled "wisnu@wisnu-Legion-5-15ACH6H: ~". The window has standard Linux-style window controls at the top. In the terminal, the command "sudo apt install ros-noetic-desktop-full" is typed and highlighted in green, indicating it is the current input. The rest of the screen is dark and mostly empty.

Gambar 3. 19 Install ROS Neotic

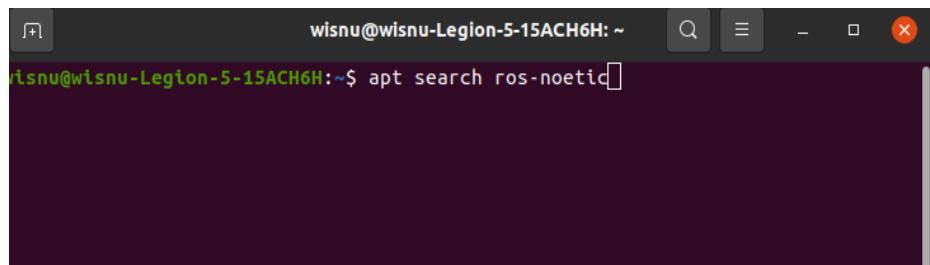
Perintah sudo apt install ros-noetic-desktop-full seperti pada Gambar 3.19 digunakan untuk menginstal versi lengkap dari ROS (Robot Operating

System) yang disebut "Noetic" pada sistem operasi Ubuntu.

A screenshot of a terminal window titled 'wisnu@wisnu-Legion-5-15ACH6H: ~'. The user has typed the command 'sudo apt install ros-noetic-PACKAGE' into the terminal. The terminal interface includes standard window controls (minimize, maximize, close) and a search bar.

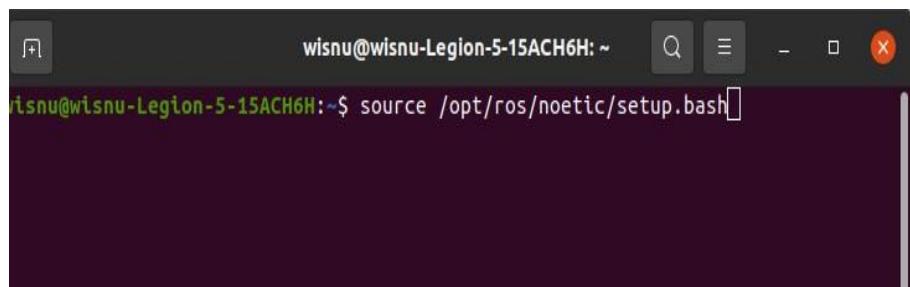
Gambar 3. 20 Install Paket Ros

Perintah sudo apt install ros-noetic-PACKAGE seperti pada Gambar 3.21 digunakan untuk menginstal paket ROS individu pada distribusi ROS Noetic di sistem operasi Ubuntu.

A screenshot of a terminal window titled 'wisnu@wisnu-Legion-5-15ACH6H: ~'. The user has typed the command 'apt search ros-noetic' into the terminal. The terminal interface includes standard window controls (minimize, maximize, close) and a search bar.

Gambar 3. 21 Paket Ros

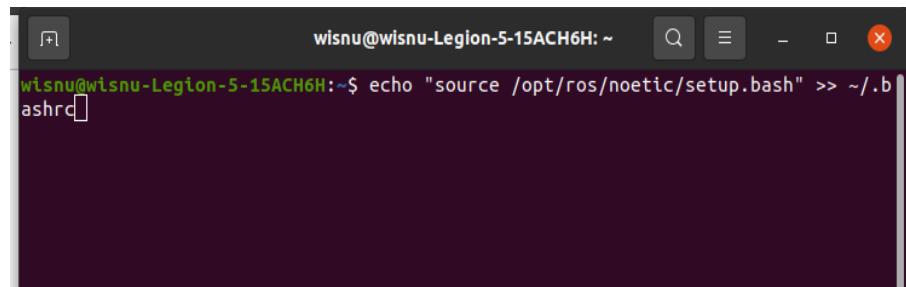
Perintah apt search ros-noetic seperti pada Gambar 3.21 digunakan untuk mencari dan menampilkan paket-paket ROS Noetic yang tersedia untuk diinstal pada sistem menggunakan manajer paket APT (Advanced Package Tool).

A screenshot of a terminal window titled 'wisnu@wisnu-Legion-5-15ACH6H: ~'. The user has typed the command 'source /opt/ros/noetic/setup.bash' into the terminal. The terminal interface includes standard window controls (minimize, maximize, close) and a search bar.

Gambar 3. 22 Code Untuk Mengaktifkan Environment ROS

Perintah source /opt/ros/noetic/setup.bash digunakan untuk mengaktifkan environment ROS (Robot Operating System) pada terminal yang sedang peneliti gunakan. Dengan menjalankan perintah ini, peneliti "memasukkan" path dan variabel-variabel lingkungan yang dibutuhkan oleh ROS Noetic ke dalam sesi terminal .

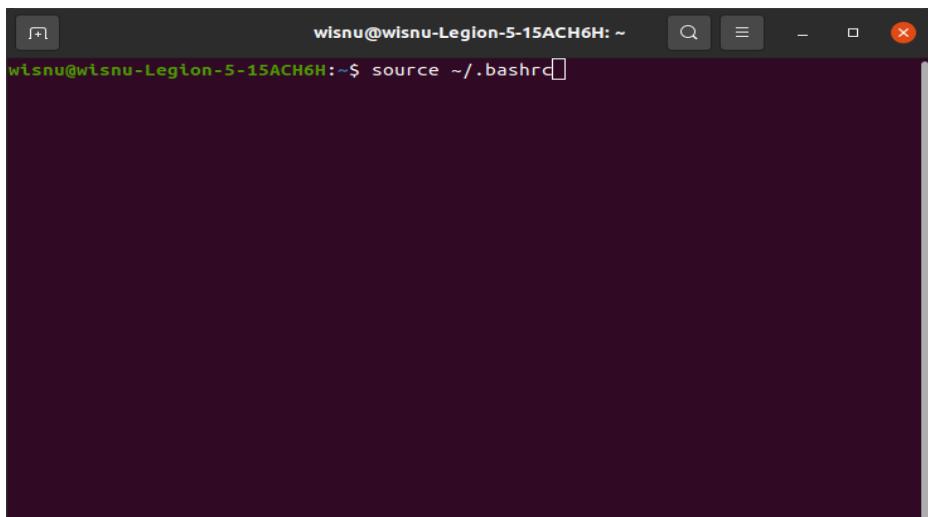
Setelah menjalankan perintah ini, peneliti akan memiliki akses ke berbagai perintah dan alat yang terkait dengan ROS Noetic pada sesi terminal tersebut. Ini termasuk akses ke perintah seperti rosdep, roslaunch, roscore, dan banyak lagi, serta variabel-variabel lingkungan yang diperlukan oleh ROS dan perlu menjalankan perintah ini setiap kali Anda memulai sesi terminal baru jika ingin menggunakan alat atau perintah ROS pada sesi tersebut. Untuk otomatisasi, dapat memasukkan perintah ini ke dalam file .bashrc atau .zshrc (tergantung pada shell yang Anda gunakan) agar dijalankan secara otomatis setiap kali membuka terminal.



```
wisnu@wisnu-Legion-5-15ACH6H:~$ echo "source /opt/ros/noetic/setup.bash" >> ~/.bashrc
```

Gambar 3. 23 Otomatisasi Aktivasi Environment ROS

Dengan menambahkan perintah source /opt/ros/noetic/setup.bash ke dalam .bashrc, mencapai otomatisasi aktivasi environment ROS setiap kali membuka terminal baru. Dengan demikian, setiap kali membuka terminal, perintah ini secara otomatis dijalankan, dan langsung dapat menggunakan alat dan perintah ROS tanpa harus secara manual menjalankan source/opt/ros/noetic/setup.bash setiap kali.

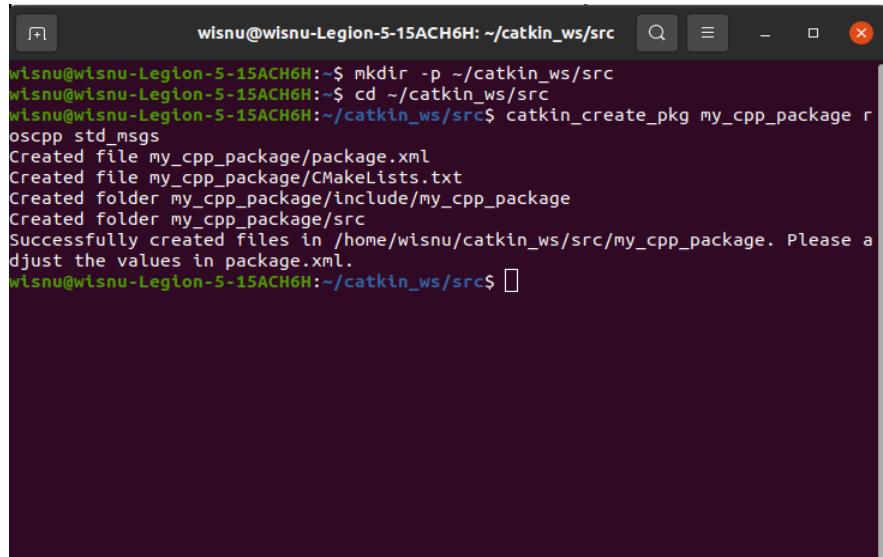


```
wisnu@wisnu-Legion-5-15ACH6H:~$ source ~/.bashrc
```

Gambar 3. 24 Menjalankan Perintah Ros

Dengan menjalankan source `~/.bashrc`, adalah memastikan bahwa perubahan yang baru saja di buat pada file `.bashrc`, seperti menambahkan perintah source `/opt/ros/noetic/setup.bash`, segera berlaku. Ini memungkinkan Anda untuk langsung menggunakan perintah-perintah dan alat ROS tanpa harus membuka terminal baru.

3.3.3 Membuat Node



```
wisnu@wisnu-Legion-5-1SACH6H:~/catkin_ws/src
wisnu@wisnu-Legion-5-1SACH6H:~/catkin_ws/src
wisnu@wisnu-Legion-5-1SACH6H:~/catkin_ws/src$ catkin_create_pkg my_cpp_package roscpp std_msgs
Created file my_cpp_package/package.xml
Created file my_cpp_package/CMakeLists.txt
Created folder my_cpp_package/include/my_cpp_package
Created folder my_cpp_package/src
Successfully created files in /home/wisnu/catkin_ws/src/my_cpp_package. Please adjust the values in package.xml.
wisnu@wisnu-Legion-5-1SACH6H:~/catkin_ws/src$
```

Gambar 3. 25 Membuat Workspace Ros

Code di atas adalah code untuk membuat sebuah workspace ROS dan juga membuat paket ROS di dalam workspace, dalam direktori src digunakan untuk menjalankan perintah-perintah terkait pembuatan paket dan build ROS.

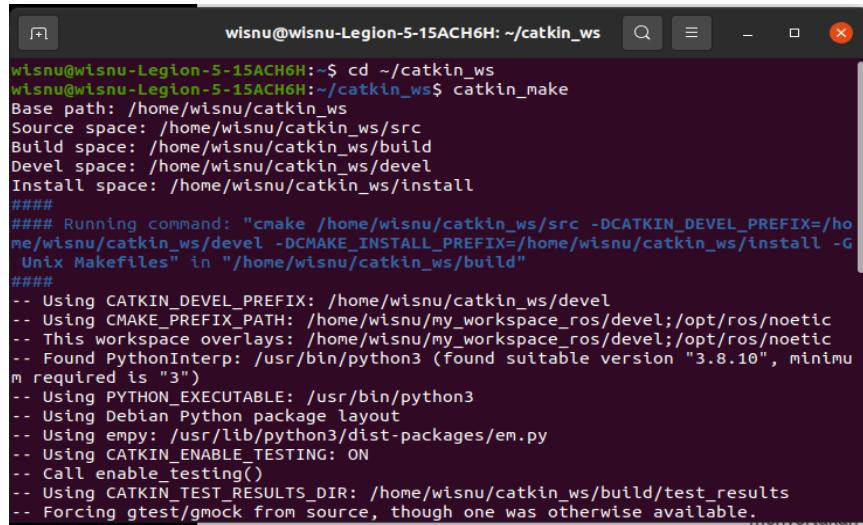
```

155
156 # all install targets should use catkin DESTINATION variables
157 # See http://ros.org/doc/api/catkin/html/adv\_user\_guide/variables.html
158
159 ## Mark executable scripts (Python etc.) for installation
160 ## In contrast to setup.py, you can choose the destination
161 # catkin_install_python(PROGRAMS
162 #   scripts/my_python_script
163 #   DESTINATION ${CATKIN_PACKAGE_BIN_DESTINATION}
164 # )
165
166 ## Mark executables for installation
167 ## See http://docs.ros.org/melodic/api/catkin/html/howto/format/building\_executables.html
168 # install(TARGETS ${PROJECT_NAME}_node
169 #   RUNTIME DESTINATION ${CATKIN_PACKAGE_BIN_DESTINATION}
170 # )
171
172 ## Mark libraries for installation
173 ## See http://docs.ros.org/melodic/api/catkin/html/howto/format/building\_libraries.html
174 # install(TARGETS ${PROJECT_NAME})
175 #   ARCHIVE DESTINATION ${CATKIN_PACKAGE_LIB_DESTINATION}
176 #   LIBRARY DESTINATION ${CATKIN_PACKAGE_LIB_DESTINATION}
177 #   RUNTIME DESTINATION ${CATKIN_GLOBAL_BIN_DESTINATION}
178 # )
179
180 ## Mark cpp header files for installation
181 # install(DIRECTORY include/${PROJECT_NAME}/
182 #   DESTINATION ${CATKIN_PACKAGE_INCLUDE_DESTINATION}
183 #   FILES_MATCHING PATTERN "*.h"
184 #   PATTERN ".svn" EXCLUDE
185 # )
186
187 ## Mark other files for installation (e.g. launch and bag files, etc.)
188 # install(FILES
189 #   # myFile1
190 #   # myFile2
191 #   DESTINATION ${CATKIN_PACKAGE_SHARE_DESTINATION}
192 # )
193
194 #####
195 ## Testing ##
196 #####
197
198 ## Add gtest based cpp test target and link libraries
199 # catkin_add_gtest(${PROJECT_NAME}_test test/test_my_cpp_package.cpp)
200 # if(TARGET ${PROJECT_NAME}_test)
201 #   target_link_libraries(${PROJECT_NAME}_test ${PROJECT_NAME})
202 # endif()
203
204 ## Add folders to be run by python nosetests
205 # catkin_add_nosetests(test)
206
207 add_executable(my_cpp_node src/my_cpp_node.cpp)
208 target_link_libraries(my_cpp_node ${catkin_LIBRARIES})
209

```

Gambar 3. 26 Cmakelist.txt

Dengan menambahkan perintah pada CMakeLists.txt,dengan memberikan petunjuk kepada CMake tentang cara membangun dan menghubungkan program C++ dengan pustaka ROS yang diperlukan. Tanpa konfigurasi ini, CMake mungkin tidak tahu bagaimana cara membangun program dengan benar atau bagaimana menghubungkannya dengan pustaka-pustaka ROS yang diperlukan. Secara keseluruhan, konfigurasi ini memastikan bahwa program C++ dapat dibangun dan dijalankan dengan benar sebagai bagian dari proyek ROS.



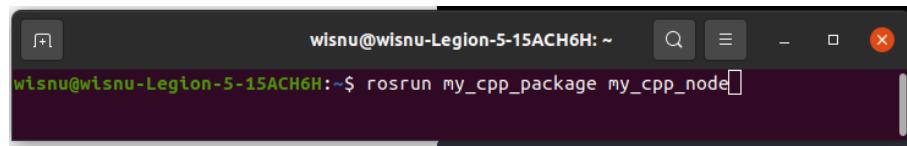
```
wisnu@wisnu-Legion-5-15ACH6H:~/catkin_ws$ cd ~/catkin_ws
wisnu@wisnu-Legion-5-15ACH6H:~/catkin_ws$ catkin_make
Base path: /home/wisnu/catkin_ws
Source space: /home/wisnu/catkin_ws/src
Build space: /home/wisnu/catkin_ws/build
Devel space: /home/wisnu/catkin_ws/devel
Install space: /home/wisnu/catkin_ws/install
#####
##### Running command: "cmake /home/wisnu/catkin_ws/src -DCATKIN_DEVEL_PREFIX=/home/wisnu/catkin_ws/devel -DCMAKE_INSTALL_PREFIX=/home/wisnu/catkin_ws/install -G Unix Makefiles" in "/home/wisnu/catkin_ws/build"
#####
-- Using CATKIN_DEVEL_PREFIX: /home/wisnu/catkin_ws/devel
-- Using CMAKE_PREFIX_PATH: /home/wisnu/my_workspace_ros/devel;/opt/ros/noetic
-- This workspace overlays: /home/wisnu/my_workspace_ros/devel;/opt/ros/noetic
-- Found PythonInterp: /usr/bin/python3 (Found suitable version "3.8.10", minimum required is "3")
-- Using PYTHON_EXECUTABLE: /usr/bin/python3
-- Using Debian Python package layout
-- Using empy: /usr/lib/python3/dist-packages/empy
-- Using CATKIN_ENABLE_TESTING: ON
-- Call enable_testing()
-- Using CATKIN_TEST_RESULTS_DIR: /home/wisnu/catkin_ws/build/test_results
-- Forcing gtest/gmock from source, though one was otherwise available.
```

Gambar 3. 27 Build Projek ROS

Perintah `catkin_make` digunakan untuk melakukan proses build pada proyek ROS yang menggunakan Catkin sebagai sistem build. Dengan `catkin_make` menginisiasi proses build untuk seluruh proyek ROS di dalam workspace Catkin serta proses ini mencakup kompilasi kode sumber, link pustaka, dan pembuatan eksekutabel dan setiap paket di dalam workspace yang memiliki file `CMakeLists.txt` akan dibangun. Selanjutnya adalah menghasilkan direktori `build` dan `devel`, `catkin_make` membuat atau mengelola direktori `build` di dalam setiap paket untuk menyimpan file sementara selama proses build dan hasil dari build, seperti eksekutabel dan pustaka, akan ditempatkan di direktori `devel`.

Selanjutnya adalah menyusun file konfigurasi lingkungan yaitu `catkin_make` menghasilkan file konfigurasi lingkungan (`setup.bash` atau `setup.zsh` tergantung pada shell yang digunakan) di dalam direktori `devel` yang di dalam file ini memuat variabel-variabel lingkungan yang diperlukan agar ROS dapat menemukan dan menjalankan program dan pustaka yang telah dibangun. Yang terakhir adalah memfasilitasi penggunaan ROS, yaitu setelah menjalankan `catkin_make`, pengguna biasanya menjalankan skrip `setup` yang dihasilkan untuk mengatur variabel lingkungan saat menggunakan terminal. Ini memungkinkan pengguna untuk menggunakan perintah ROS seperti `rosrun`, `roslaunch`, dan lainnya dengan benar. Secara keseluruhan, `catkin_make` adalah perintah yang penting dalam pengembangan proyek ROS karena menyederhanakan proses build, menyusun lingkungan, dan

mempersiapkan proyek untuk dijalankan dalam lingkungan ROS.

A screenshot of a terminal window titled "wisnu@wisnu-Legion-5-15ACH6H: ~". The window contains the command "rosrun my_cpp_package my_cpp_node" in green text, indicating it has been successfully executed.

Gambar 3. 28 Running Node C++

Perintah `rosrun my_cpp_package my_cpp_node` seperti pada Gambar 3.28 digunakan untuk menjalankan node C++ yang disebut `my_cpp_node` dari paket ROS `my_cpp_package`.

3.3.4 Membuat Thread

Thread adalah sebuah proses yang berukuran kecil yang dibuat oleh sebuah program untuk dijalankan bersamaan dengan thread – thread lainnya. Tujuan thread ini adalah agar proses – proses yang dapat dikerjakan bersamaan dapat dijalankan secara bersamaan tanpa memerlukan waktu tunggu untuk proses berikutnya (Onggrono et al., 2017). Pada penelitian ini di jelaskan cara pembuatan thread adalah sebagai berikut :

1. Include Header File

Sertakan header file `<thread>` pada bahasa pemrograman C++ berfungsi untuk mengimpor atau menyertakan deklarasi dan definisi dari fungsi-fungsi dan kelas-kelas yang terkait dengan penanganan thread. Dalam hal ini, `<thread>` adalah header file yang menyediakan fasilitas untuk bekerja dengan thread dalam program C++.

2. Tentukan Fungsi yang Akan Dijalankan dalam Thread

Pada pembuatan fungsi mengacu pada fungsi-fungsi yang akan dieksekusi secara paralel dalam thread. Fungsi-fungsi ini akan berisi tugas-tugas atau operasi-operasi yang ingin Anda jalankan secara bersamaan dalam thread terpisah.

3. Buat Objek Thread

Menggunakan `std::thread` untuk membuat objek thread, dan memberikan fungsi yang akan dijalankan dalam thread sebagai parameter sedangkan `std::thread` adalah kelas yang disediakan oleh pustaka `<thread>` dalam C++. Ini digunakan untuk membuat objek thread. Objek thread ini akan mewakili

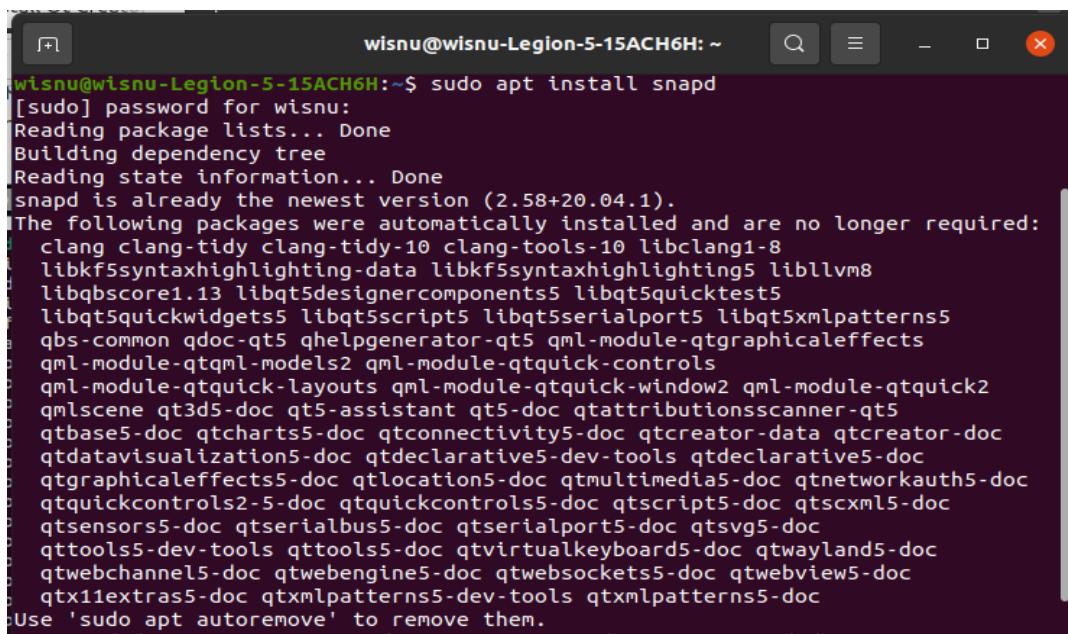
eksekusi dari fungsi yang akan dijalankan secara paralel.

4. Join Thread

Gunakan metode “join()” pada objek thread untuk menunggu hingga thread selesai sebelum melanjutkan eksekusi program utama. Dengan memanggil “join()” pada objek thread, program utama akan menunggu hingga thread yang bersangkutan selesai dieksekusi sebelum melanjutkan ke instruksi selanjutnya. Ini memastikan bahwa eksekusi program utama tidak berlanjut sebelum pekerjaan dalam thread selesai.

3.3.5 QTCreator

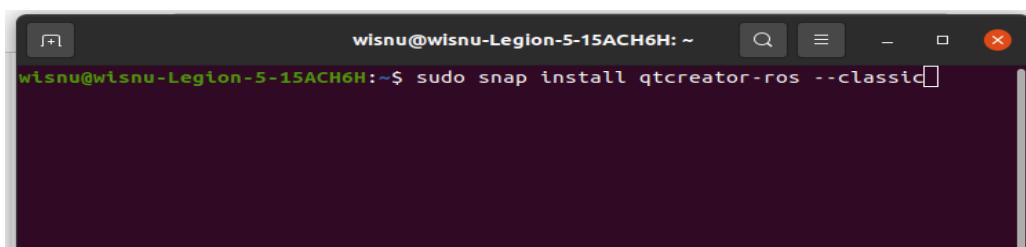
Qt Creator ROS (Robot Operating System) adalah integrasi antara lingkungan pengembangan Qt Creator dan ROS, yang memungkinkan pengembang untuk bekerja dengan proyek-proyek yang menggunakan ROS dalam lingkungan pengembangan Qt Creator. Qt Creator sendiri adalah Integrated Development Environment (IDE) yang digunakan untuk pengembangan perangkat lunak berbasis Qt, sebuah kerangka kerja pengembangan perangkat lunak yang populer untuk aplikasi GUI dan non-GUI, dan qt dapat di install sebagai berikut :



```
wisnu@wisnu-Legion-5-15ACH6H:~$ sudo apt install snapd
[sudo] password for wisnu:
Reading package lists... Done
Building dependency tree
Reading state information... Done
snapd is already the newest version (2.58+20.04.1).
The following packages were automatically installed and are no longer required:
  clang clang-tidy clang-tidy-10 clang-tools-10 libclang1-8
  libkf5syntaxhighlighting-data libkf5syntaxhighlighting5 libllvm8
  libqbscore1.13 libqt5designercomponents5 libqt5quicktest5
  libqt5quickwidgets5 libqt5script5 libqt5serialport5 libqt5xmlpatterns5
  qbs-common qdoc-qts5 qhelpgenerator-qts5 qml-module-qtgraphicaleffects
  qml-module-qtqml-models2 qml-module-qtquick-controls
  qml-module-qtquick-layouts qml-module-qtquick-window2 qml-module-qtquick2
  qmlscene qt3d5-doc qt5-assistant qt5-doc qtattritionsscanner-qt5
  qtbase5-doc qtcharts5-doc qtconnectivity5-doc qtcreator-data qtcreator-doc
  qtdatavisualization5-doc qtdeclarative5-dev-tools qtdeclarative5-doc
  qtgraphicaleffects5-doc qtlocation5-doc qtmultimedia5-doc qtnetworkauth5-doc
  qtquickcontrols2-5-doc qtquickcontrols5-doc qtscript5-doc qtscxml5-doc
  qtsensors5-doc qtserialbus5-doc qtserialport5-doc qtsvg5-doc
  qttools5-dev-tools qttools5-doc qtvirtualkeyboards5-doc qtwaylands5-doc
  qtwebchannel5-doc qtwebengine5-doc qtwebsocket5-doc qtwebview5-doc
  qtx11extras5-doc qtxmlpatterns5-dev-tools qtxmlpatterns5-doc
Use 'sudo apt autoremove' to remove them.
```

Gambar 3. 29 Install Snap

Seperti pada Gambar 3.29 dimana perintah “sudo apt install snapd” digunakan untuk menginstal perangkat lunak Snap Daemon (snapd) di sistem berbasis Ubuntu atau distro Linux yang menggunakan manajer paket APT (Advanced Package Tool). Snap adalah sistem pengemasan perangkat lunak yang dirancang untuk membuat pengalaman instalasi dan manajemen perangkat lunak lebih mudah di berbagai distribusi Linux. Snapd adalah daemon yang menjalankan dan mengelola paket Snap di sistem. Dengan menginstal Snap Daemon menggunakan perintah di atas akan memungkinkan sistem untuk mendukung penggunaan paket Snap. Setelah proses instalasi selesai dan dapat menggunakan perintah snap untuk menginstal, menghapus, dan mengelola paket Snap pada sistem.



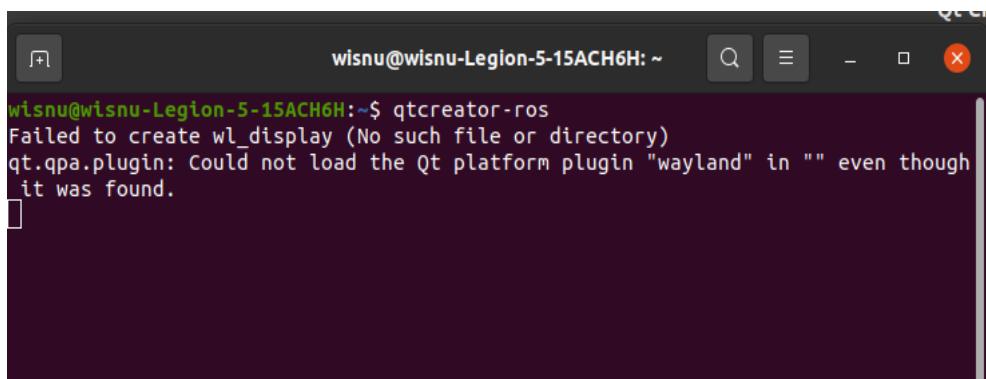
Gambar 3. 30 Install QTCreator-ros.

Selanjutnya adalah dimana seperti pada Gambar 3.31 perintah sudo snap install qtcreator-ros --classic digunakan untuk menginstal Qt Creator yang telah diintegrasikan dengan ROS (Robot Operating System) menggunakan sistem pengelola paket Snap di sistem berbasis Ubuntu atau distro Linux yang mendukung Snap. Setelah perintah ini dieksekusi, Qt Creator yang telah diintegrasikan dengan ROS akan diunduh, diinstal, dan

dapat diakses dari lingkungan Snap pada sistem dan dapat menggunakan Qt Creator untuk mengembangkan perangkat lunak dengan dukungan ROS.

3.3.6 Setting QTCreator

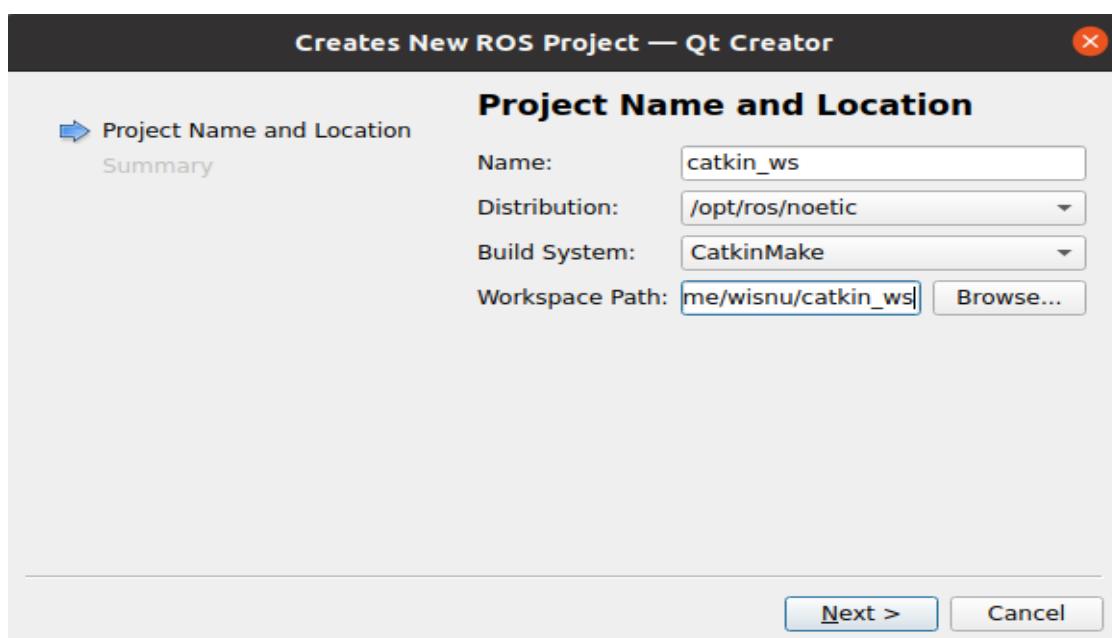
Pada penelitian ini menjelaskan bagaimana qtcreator dapat berjalan pada sistem operasi ROS(Robot Operation System), adalah sebagai berikut :



```
wisnu@wisnu-Legion-5-15ACH6H:~$ qtcreator-ros
Failed to create wl_display (No such file or directory)
qt.qpa.plugin: Could not load the Qt platform plugin "wayland" in "" even though
it was found.
```

Gambar 3. 31 Membuka QTCreator Pada Sistem Operasi ROS

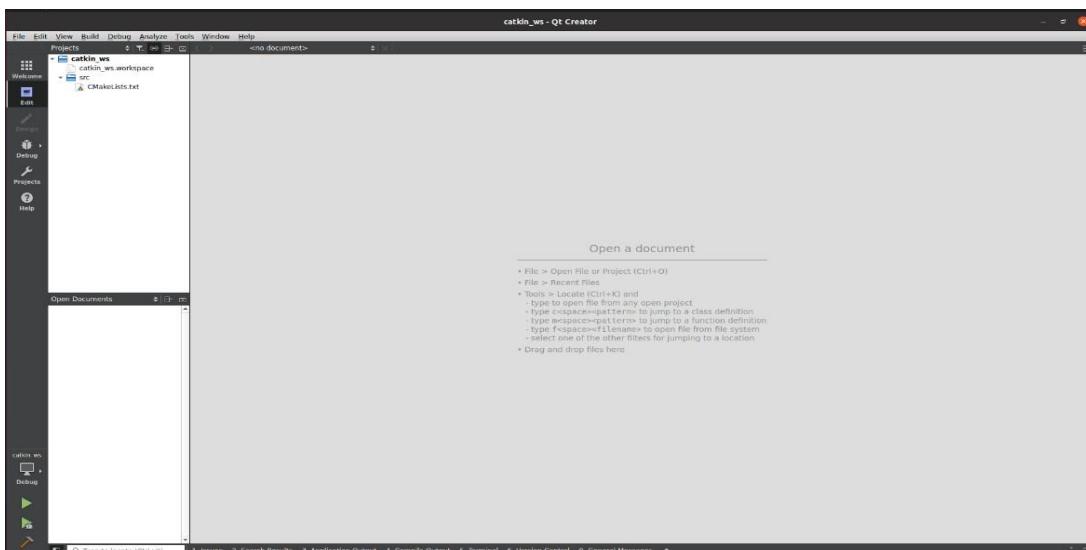
Seperti pada Gambar 3.31 dimana penulis menggunakan code “qtcreator-ros” pada terminal untuk membuka aplikasi QTCreator yang sudah terinstall pada linux. Seperti pada Gambar 3.33 dimana penulis membuat *workspace* untuk membuat *interface* dan juga program pada template di QTCreator dengan memilih ROS *Workspace*.



Gambar 3. 32 Project Name and Location

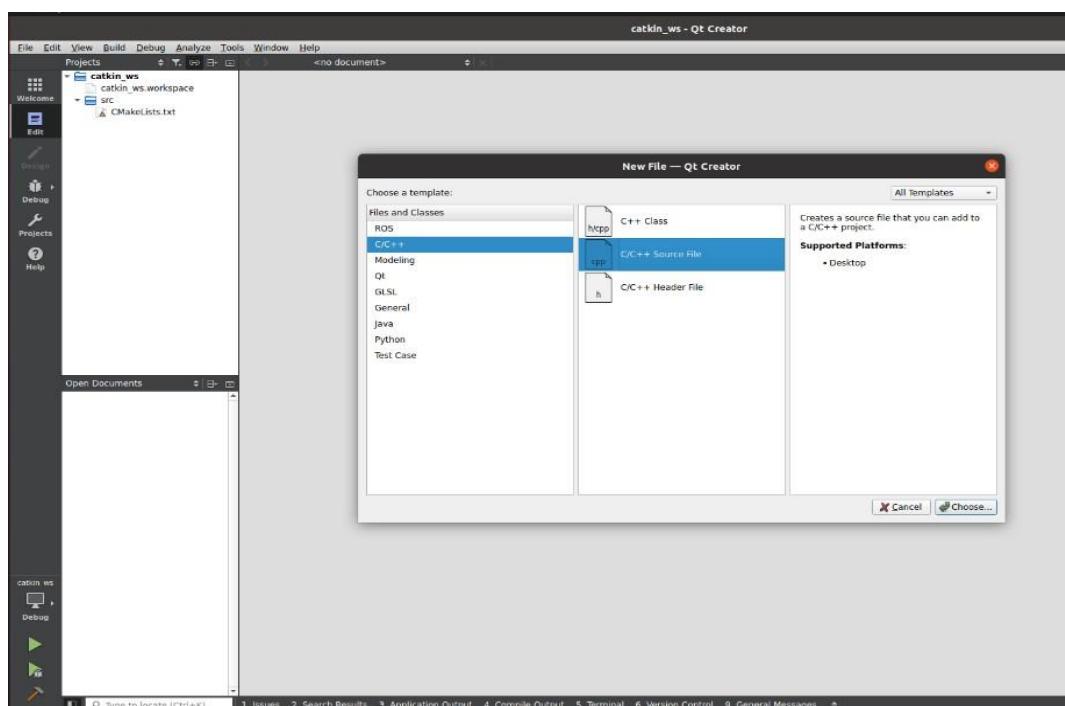
Pada tahap selanjutnya adalah dimana penulis memilih lokasi *workspace* QTCreator akan di tempatkan, dimana seperti pada Gambar 3.33 menempatkan QTCreator pada “catkin_ws” dimana lokasi tersebut adalah tempat build sistem operasi

ROS (Robot Operation System).



Gambar 3. 33 Workspace QTCreator

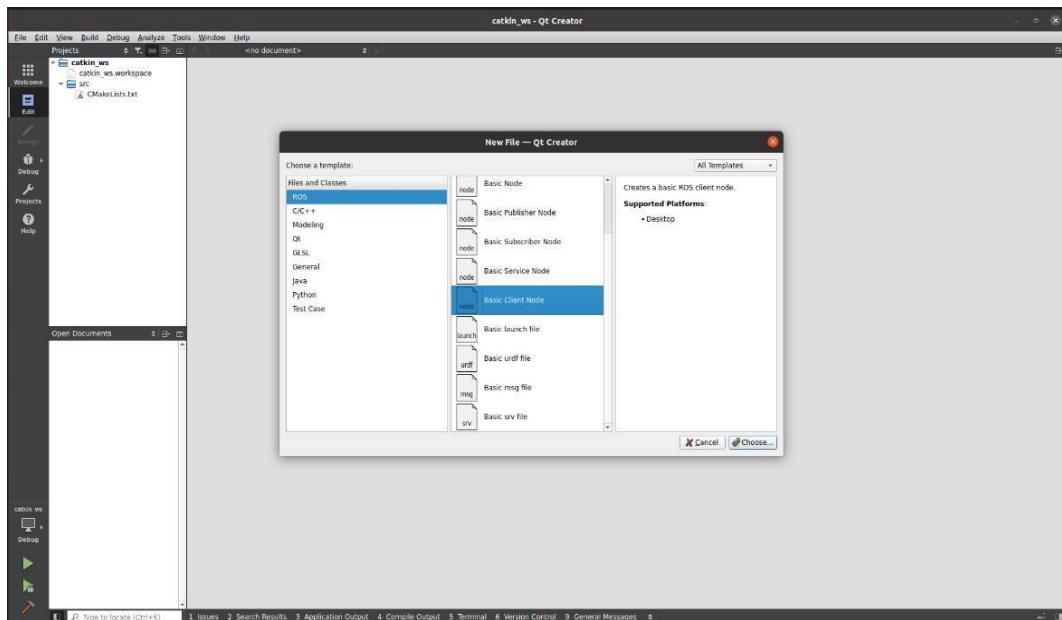
Seperti pada Gambar 3.35 adalah tampilan *workspace* dari QTCreator setelah dibuat, pada *workspace* inilah penulis membuat *interface* dan juga program komunikasi untuk menghubungkan antara *basestation* dengan robot sepakbola beroda.



Gambar 3. 34 Menambahkan File Pada Project

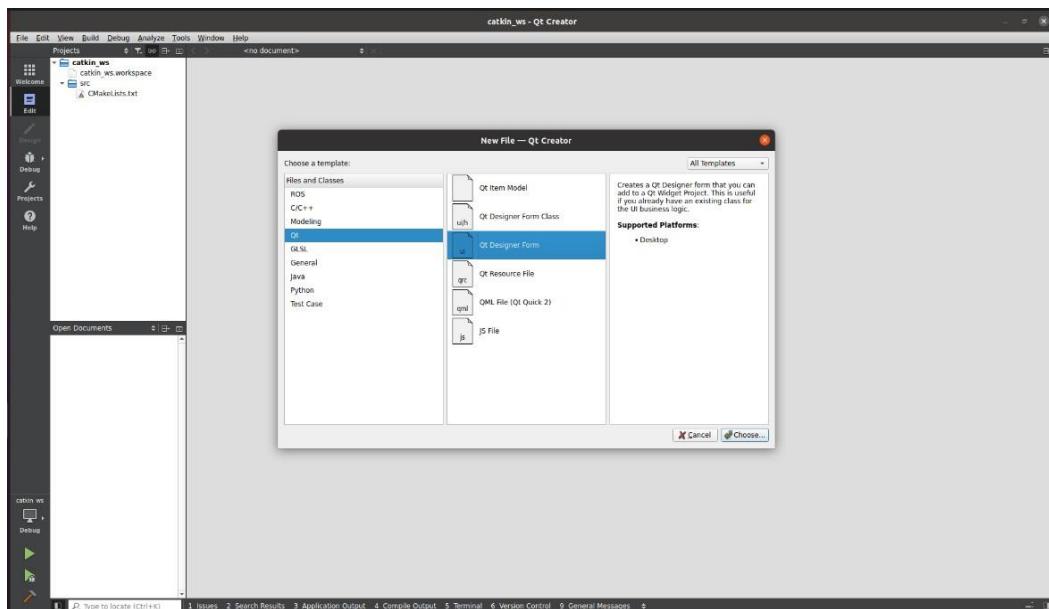
Selanjutnya penulis menambahkan file dengan bahas C++ pada *project* catkin_ws untuk program komunikasi antara *basestation* dan robot sepakbola beroda dimana program tersebut berisi program TCP dan UDP sebagai protokol komunikasi pada pengiriman data robot sepakbola beroda dan file daripada program tersebut haruslah pada

lokasi src dimana src adalah tempat semua program yang akan di jalankan pada ROS (Robot Operation System).



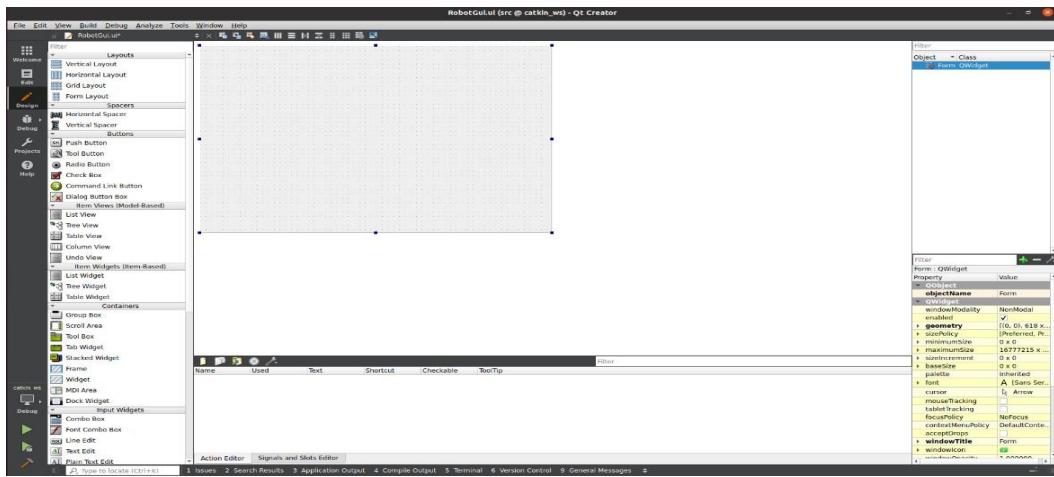
Gambar 3. 35 File Subscriber, Publiser, dan Node

Selanjutnya pada Gambar 3.35 juga penulis dapat menambahkan file *subscriber*, *publisher* yaitu adalah sebagai komunikasi pada ros dan juga node yang berisi program agar *basetation* dapat menerima semua sensor robot.



Gambar 3. 36 Interface

Selanjutnya penulis menambahkan file desain pada QTCreator seperti pada Gambar 3.36 sebagai tempat pembuatan *interface* robot sepakbola beroda yang berfungsi sebagai monitoring robot dan juga komunikasi antar robot sepakbola beroda.



Gambar 3. 37 Form Pembuatan Interface

Selanjutnya seperti pada Gambar 3.37 adalah tampilan dimana penulis membuat desain tampilan *basestation* robot sepakbola beroda dengan *tools* yang sudah di sediakan oleh QTCreator.

3.3.7 Cara Pengujian

Pada penelitian kali ini dimana pembuatan *basestation* berbasis ROS (Robot Operation Sistem) dapat menjalankan robot dengan beberapa pengujian sebagai berikut:

1. *Basestation* mengirimkan data start tanpa mengirimkan data strategi dan robot tetap dalam posisi diam.
2. *Basestation* mengirimkan data strategi 1 lalu mengirimkan data start dan robot bergerak ke posisi yang sudah di tentukan.
3. *Basestation* mengirimkan data strategi 2 lalu mengirimkan data start dan robot bergerak ke posisi yang sudah di tentukan.
4. Setelah robot bergerak *basestation* mengirimkan data stop agar robot berhenti.

3.2 Jadwal (*Timeline*)

Tabel 3. 2 Jadwal Pengerjaan Penelitian

No	Jenis Kegiatan	Minggu Ke-														
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	Studi Literatur															
2	Analisa Kebutuhan															
3	Desain dan Perencanaan System															
4	Pembuatan Hardware															
5	Pembuatan Software															
6	Implementasi dan pengujian															
7	Analisa Hasil															
8	Pembuatan Laporan															

3.4 Rencana Anggaran Penelitian

Tabel 3. 3 Rencana Anggaran Penelitian

No	Komponen	Jumlah	Harga Satuan	Total
1	STM32F407VGT6	1	1.050.000	1.050.000
2	Arduino Mega Pro Mini	1	175.000	175.000
3	Sensor HWT901B	1	990.912	990.912
4	Sensor <i>Proximity Infrared</i>	2	22.000	44.000
5	Driver Motor IBT-2-H-Bridge	10	75.000	750.000
6	Motor DC PG-45	6	900.000	5.400.000
7	Motor DC PG-36	2	800.000	1.600.000
8	LCD 20x4	1	45.000	45.000
9	Incremental Rotary Recorder	3	250.000	750.000
10	Camera Omni-directional	1	2.300.000	2.300.000
11	Router Mercusys MR30G	1	339.000	339.000
Total				13.443.912

BAB 4

HASIL DAN PEMBAHASAN

Bab ini berisi tentang hasil pengujian dan pembahasan seluruh komponen hardware maupun software yang digunakan pada Tugas Akhir ini. Pengujian akan dilakukan pada dua buah robot seperti pada Gambar 4.1. Robot akan diberi nama yaitu robot satu dan robot dua. Masing masing robot tersebut akan digunakan sebagai robot penyerang dalam satu tim.



Gambar 4. 1 Robot Satu, Robot Dua dan Robot Tiga.

4.1 Pengujian Sensor dan Aktuator

Pengujian sensor pada pengujian ini adalah sebagai berikut :

4.1.1 Pengujian Sensor Proximity

Pada pengujian sensor proximity, sensor ini akan dihubungkan ke program mikrokontroller. Sensor ini digunakan sebagai parameter dimana robot satu atau robot dua membawa bola, dimana jika robot satu membawa bola basestation akan mengirim data 1 pada robot dua dan jika robot dua membawa bola maka akan mengirimkan data 2 pada robot satu.



Gambar 4. 2 Bola Terdeteksi



Gambar 4. 3 Bola Tidak Terdeteksi

Pada Gambar 4.2 adalah hasil pengujian sensor dimana *proximity* mendeteksi bola. Pada Gambar 4.3 adalah hasil pengujian sensor dimana bola tidak terdeteksi dengan *proximity*. Sensor *proximity* ini digunakan sebagai parameter dimana robot mendapatkan bola atau bola berada pada penggiring robot. Pengujian ini dilakukan sebanyak 5 kali dengan menempatkan bola pada jarak yang berbeda dan mendapatkan hasil seperti pada tabel 4.1.

Tabel 4. 1 Pengujian Jarak Proximity

No	Jarak	Keterangan
1	5mm	Terdeteksi
2	10mm	Terdeteksi
3	15mm	Terdeteksi
4	20mm	Terdeteksi
5	25mm	Tidak Terdeteksi

Berdasarkan Tabel 4.1 Hasil Pengujian Jarak *Proximity* didapatkan kesimpulan

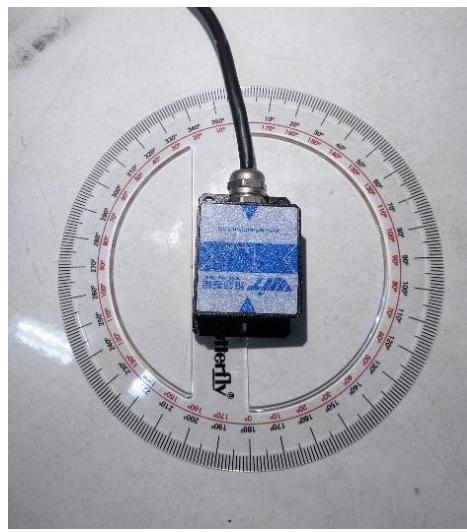
bahwa jarak maksimal *proximity* mendekksi bola adalah 20mm. proses pengujian sensor dilakukan seperti pada Gambar 4.4.



Gambar 4. 4 Pengujian Proximity

4.1.2 Pengujian Rotation Angle

Pada penelitian Tugas akhir ini sensor Rotation Angle yang digunakan adalah sensor HWT101CT-TTL. Sensor tersebut digunakan untuk menentukan arah hadap robot. Pembacaan sensor Rotation Angle juga digunakan dalam metode gyrodometry. Pengujian akan dilakukan dengan membandingkan hasil pengukuran sensor Rotation Angle dengan pengukuran busur derajat sebagai alat bantu pengukuran seperti pada gambar 4.5 dan pada gambar 4.6 adalah posisi penempatan sensor HWT101CT pada robot. Perbandingan hasil pengukuran sensor Rotation Angle dengan pengukuran busur derajat terdapat pada Tabel 4.2.



Gambar 4. 5 Gamba Pengujian Dengan Busur



Gambar 4. 6 Penempatam Sensor HWT101CT

Tabel 4. 2 pengujian sensor gyroscope

No	Data Aktual	Data Sensor	Persentasi Error
1	0	0	0,00%
2	20	20	0,00%
3	40	40	0,00%
4	60	60	0,00%
5	80	80	0,00%
6	100	100	0,00%
7	120	120	0,00%
8	140	139	0,71%

9	160	160	0,00%
10	180	181	0,56%
11	200	200	0,00%
12	220	220	0,00%
13	240	240	0,00%
14	260	260	0,00%
15	280	280	0,00%
16	300	300	0,00%
17	320	320	0,00%
18	340	340	0,00%
<i>Rata-rata Error</i>			0,07 %

Berdasarkan 18 data pengujian yang telah dilakukan. Perbandingan nilai sensor dan nilai aktual pengukuran memiliki rata rata presentase error sebesar 0,07% pada robot. Eror tersebut disebabkan oleh pembacaan sensor yang kurang baik, namun error yang dihasilkan tidak terlalu besar dan masih dapat ditoleransi karena tidak sampai merubah arah hadap robot yang signifikan. Berdasarkan data tersebut dapat disimpulkan bahwa sensor HWT101CT bekerja dengan baik pada pengujian ini.

4.1.3 Pengujian Rotary Encoder

Dalam penelitian ini sensor rotary encoder akan digunakan untuk mengetahui posisi atau perpindahan robot dari titik yang ditentukan sebagai titik awal atau titik 0. Sensor ini merupakan sensor yang membaca jumlah putaran. Prinsip rotary encoder pada penelitian ini akan membaca jumlah putaran roda, kemudian data tersebut dikonversi menjadi jarak. Pada penelitian ini akan menggunakan tiga rotary encoder yang dipasang dengan membentuk sudut 120° Gambar 4.7 merupakan hardware peletakan sensor rotary encoder yang di-couple dengan roda omni. Dan Gambar 4.8 merupakan pengujian dari masing-masing rotary encoder.



Gambar 4. 7 Pemasangan Rotary Encoder



Gambar 4. 8 Pengujian Rotary Encoder

Tabel 4. 3 Data Pengujian Rotary Encoder

No	Data Aktual	Data Sensor	Error
1	2000 mm	2000	0,00%
2	3000 mm	3029	0,97%
3	4000 mm	4035	0,88%
4	5000 mm	5017	0,34%
5	6000 mm	6023	0,38%
Rata-rata Persentase Error			0,514%

4.1.4 Pengujian Motor

Dalam penelitian ini aktuator yang di gunakan adalah motor PG-45 sebagai penggerak dari robot dimana uji kelayakan motor PG-45 ini menggunakan tachometer, uji kelayakan dilakukan sebanyak 10 kali seperti berikut :

Tabel 4. 4 Pengujian Motor

NO	Tegangan Input (V)	Datasheet (RPM)	Tachometer (RPM)	Error (%)
1	24	500	514,6	2,92 %
2	24	500	512,9	2,58 %
3	24	500	510,0	2 %
4	24	500	514,2	2,84 %
5	24	500	513,4	2,68 %
6	24	500	513,8	2,76 %
7	24	500	514,5	2,9 %
8	24	500	514,6	2,94 %
9	24	500	514,7	2,98 %
10	24	500	514,9	2,9 %
Rata-rata Error				2,76 %

Dari hasil pengujian aktuator yang dilakukan 10 kali persentase keakuratan cukup akurat dimana error tidak lebih dari 3%. Gambar 4.9 adalah pengujian menggunakan tachometer.



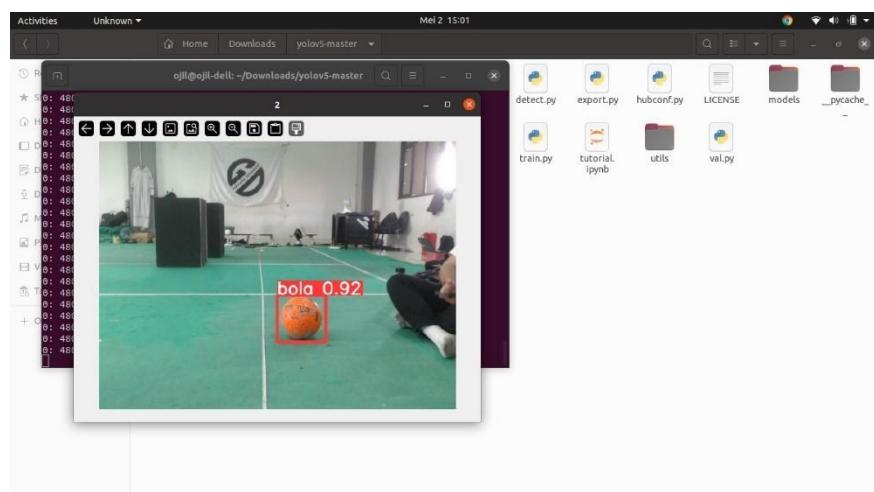
Gambar 4. 9 Pengujian Menggunakan Tachometer

4.2 Pengujian Kamera

Pengujian Kamera pada penelitian ini dilakukan beberapa pengujian sebagai berikut :

4.2.1 Pengujian Koneksi Kamera dengan PC

Pada penelitian ini, kamera yang digunakan adalah kamera webcam C922. Dalam pengujian koneksi kamera dengan PC ini menggunakan kamera omnidirectional. Tahap awal pengujian kamera adalah pengecekan koneksi kamera terhadap PC. PC akan di program untuk membaca dan menampilkan video secara real-time. Apabila program berjalan dan dapat membuka kamera, maka akan muncul tampilan frame yang berisi video secara real-time. Apabila program tidak dapat membuka kamera maka program akan selesai atau keluar otomatis dan frame tidak akan muncul. Kegagalan membuka kamera bisa disebabkan kesalahan program atau port kamera yang tidak terpasang dengan benar sehingga kamera tidak dapat dibuka. Gambar 4.10 merupakan hasil pengujian koneksi kamera dengan PC saat berhasil dan frame kamera dapat dibuka.



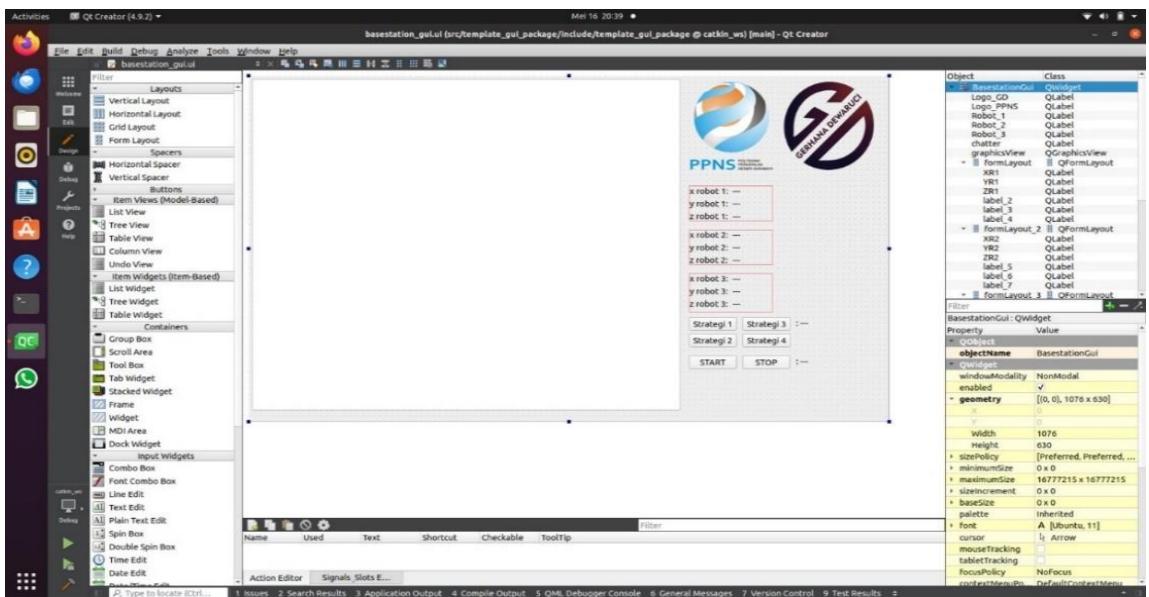
Gambar 4. 10 Hasil Tangkapan Kamera C922

4.3 Pembuatan Basestation

Pada penelitian ini menggunakan aplikasi QT Creator untuk membuat *interface* basestation sebagai berikut :

4.3.1 Pembuatan GUI

Pembuatan GUI (Graphics User Interface) pada penelitian menggunakan software QT Creator bawaan ROS yang hanya dapat menggunakan bahasa pemrograman C++ sebagai berikut:

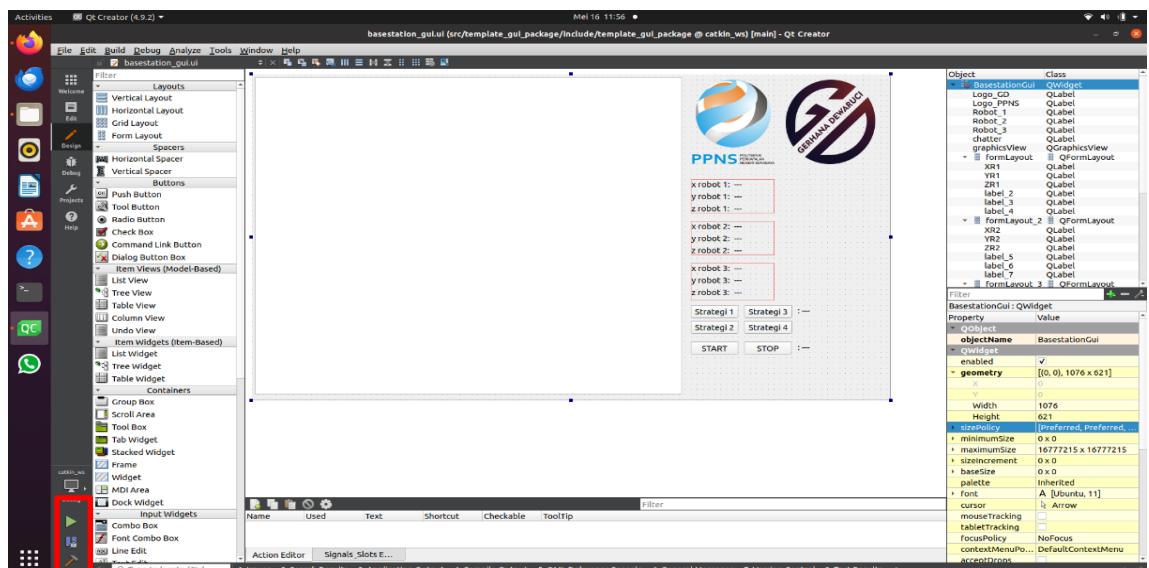


Gambar 4. 11 QT Creator

Pada penelitian ini peneliti menggunakan aplikasi QT Creator seperti pada gambar 4.10 dimana banyksk tool yang bisa di gunakan untuk membuat *interface* dari basestation sendiri.

4.3.2 Menghubungkan Basestation dengan ROS

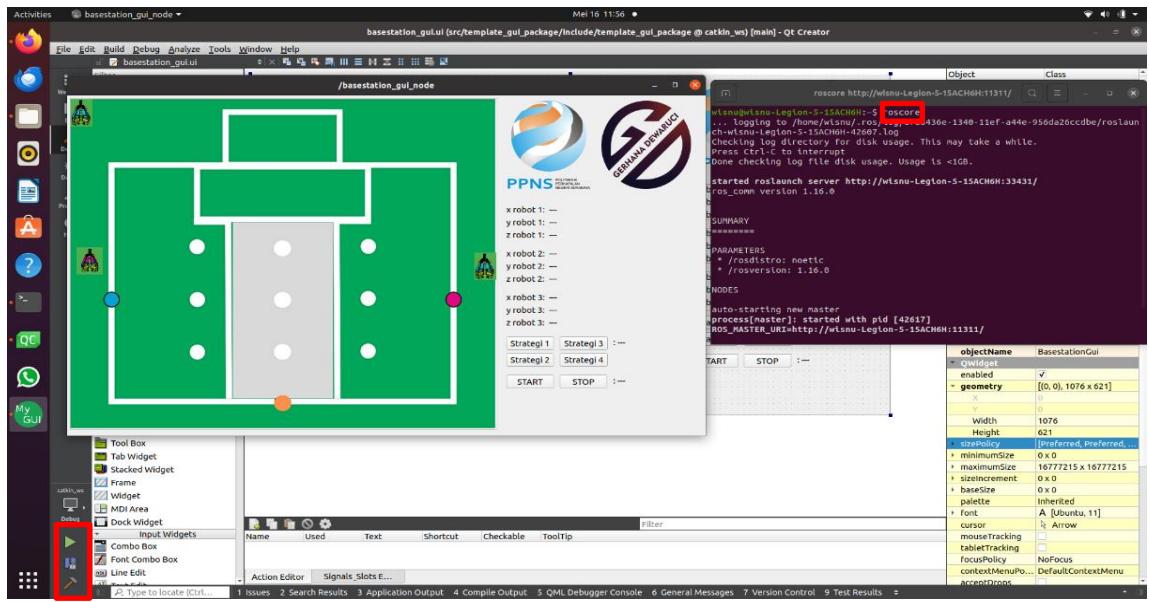
Yang dilakukan adalah mengkonfigurasi pada cmake seperti pada Gambar 2.37 yang berguna untuk memastikan bahwa program C++ dapat dibangun dan dijalankan dengan benar sebagai bagian dari proyek ROS.



Gambar 4. 12 Running tanpa Running ROS Master

Seperti pada Gambar 4.12 dimana QT Creator dapat di *running* tetapi tidak dapat

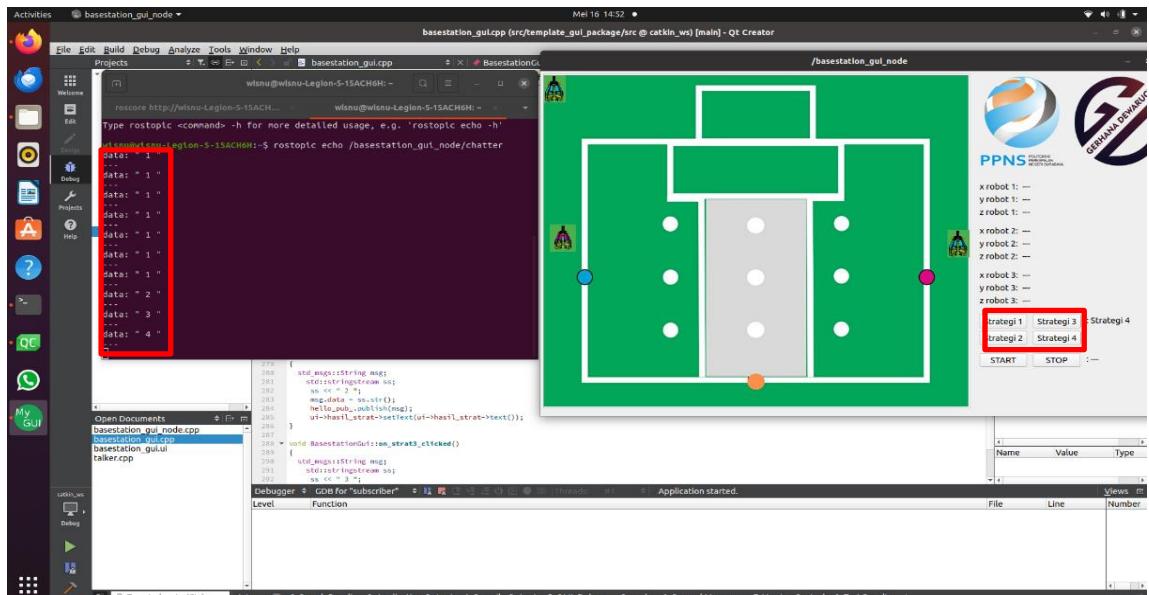
menampilkan basestation dikarenakan ROS Master masih belum di jalankan, dan pada Gambar 4.13 dimana ROS Master sudah di jalankan maka basestation juga akan dapat berjalan dan menampilkan *interface*.



Gambar 4. 13 ROS Master Berjalan

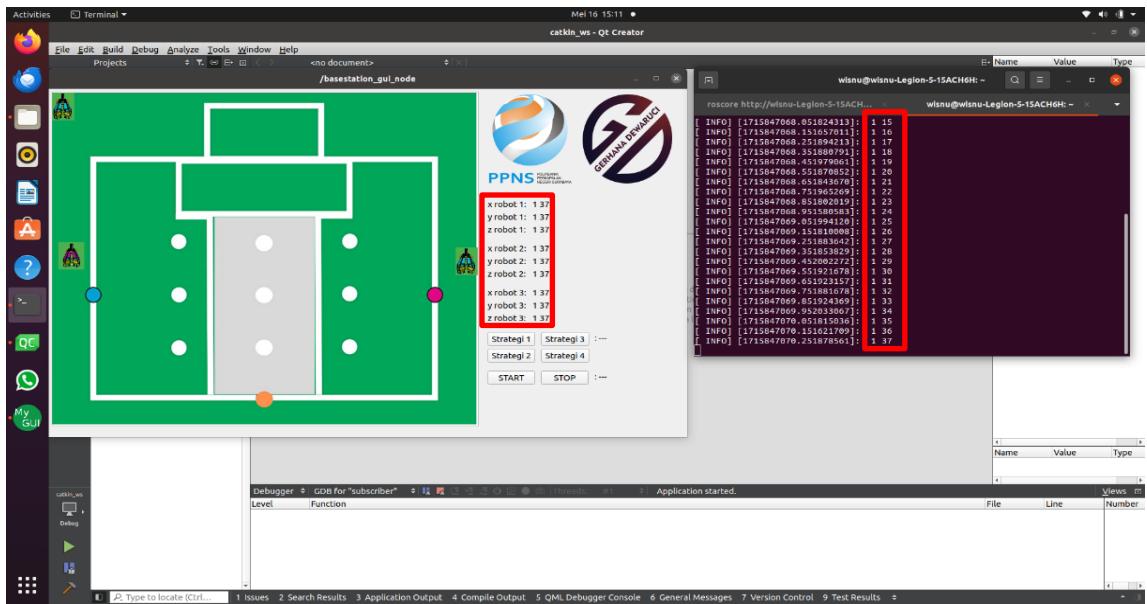
4.3.4 Pengujian Komunikasi Antar Node

Pada penelitian ini pengujian antar node menggunakan satu device dengan publisher dan subscriber menggunakan bahasa C++ seperti berikut :



Gambar 4. 14 Pengiriman Data

Pada Gambar 4.14 pengiriman data, dimana publisher dari basestation mengirimkan data 1 sampai 4 pada subscriber dengan topic yang sama.

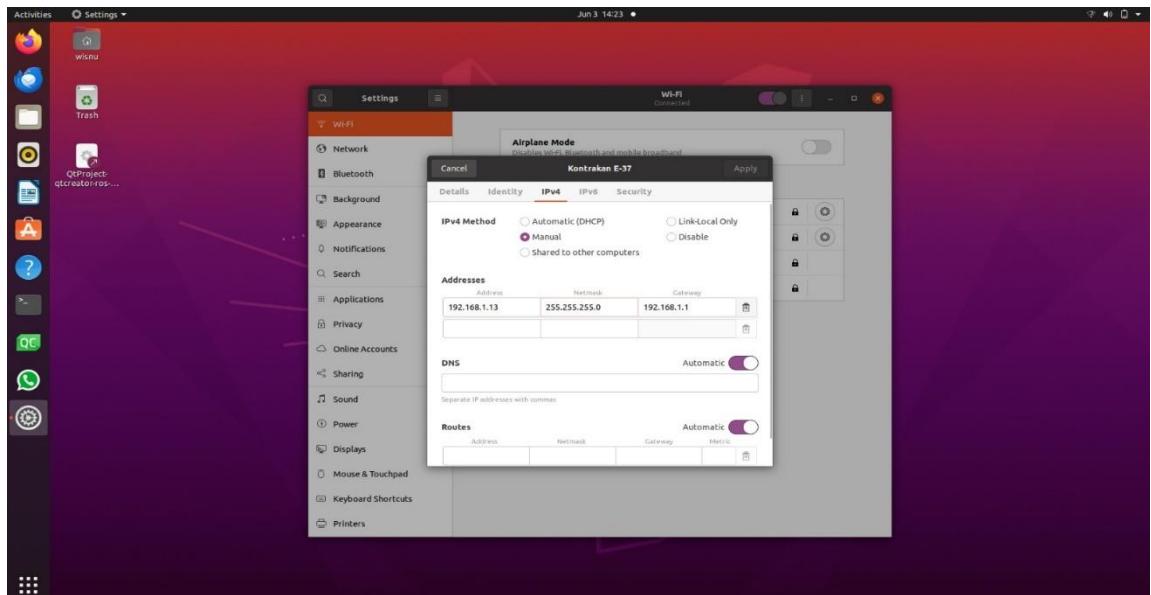


Gambar 4. 15 Menerima Data

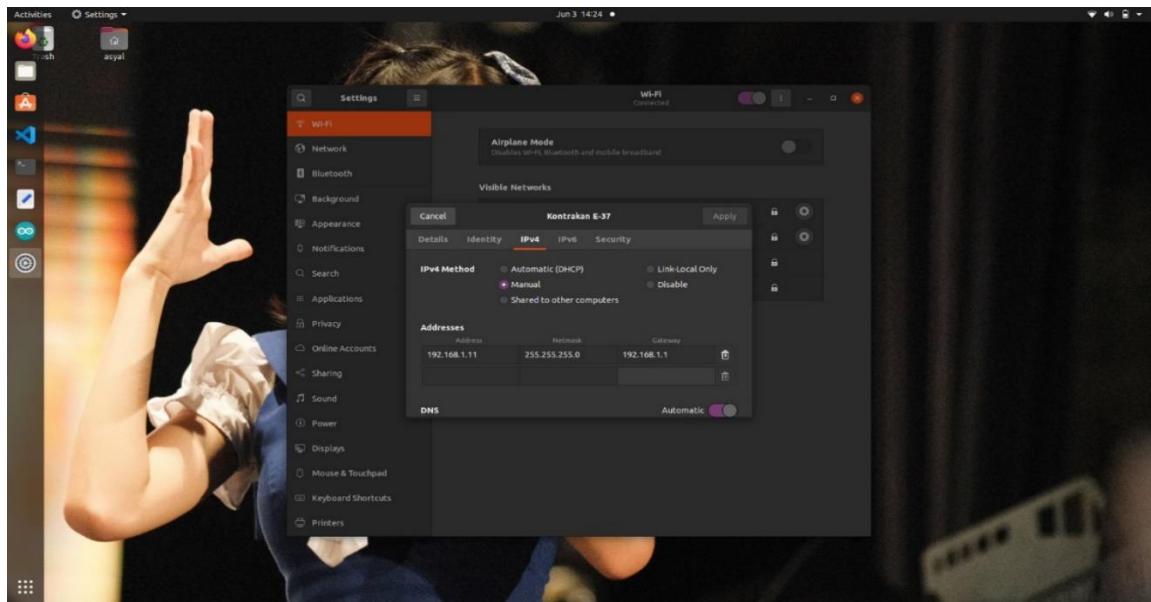
Pada Gambar 4.15 menerima data dimana, subscriber dari basestation menerima data dari publisher dengan topic yang sama.

4.4 Komunikasi Antar Device

Pada bab ini peneliti menguji apakah antara device basestation dengan device robot dapat terhubung menggunakan TCP/IP.

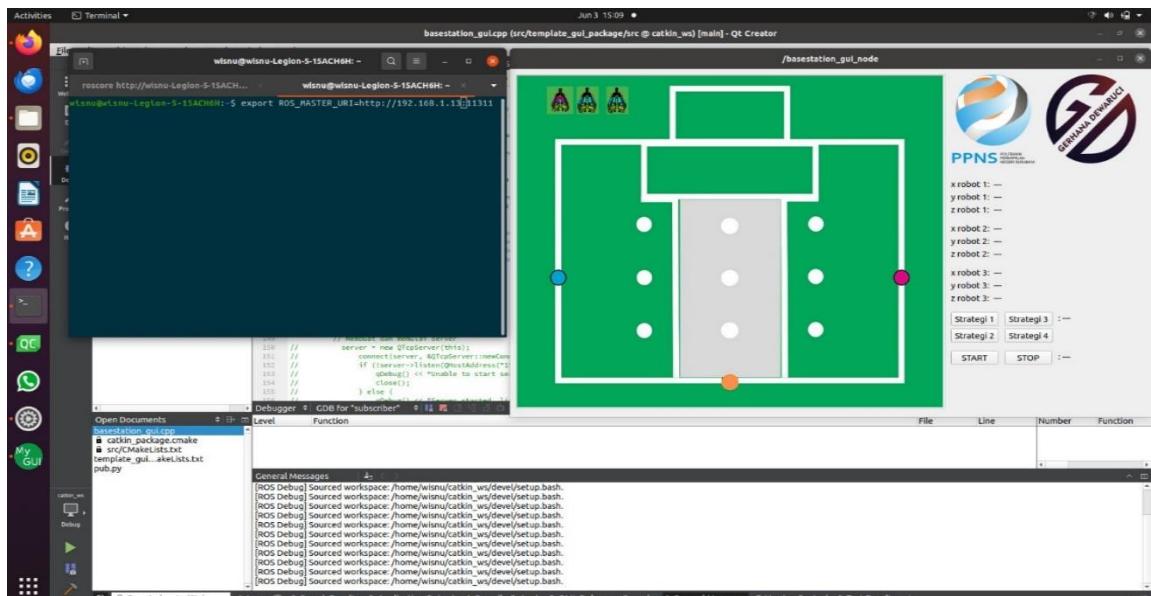


Gambar 4. 16 Ip Address Basestation



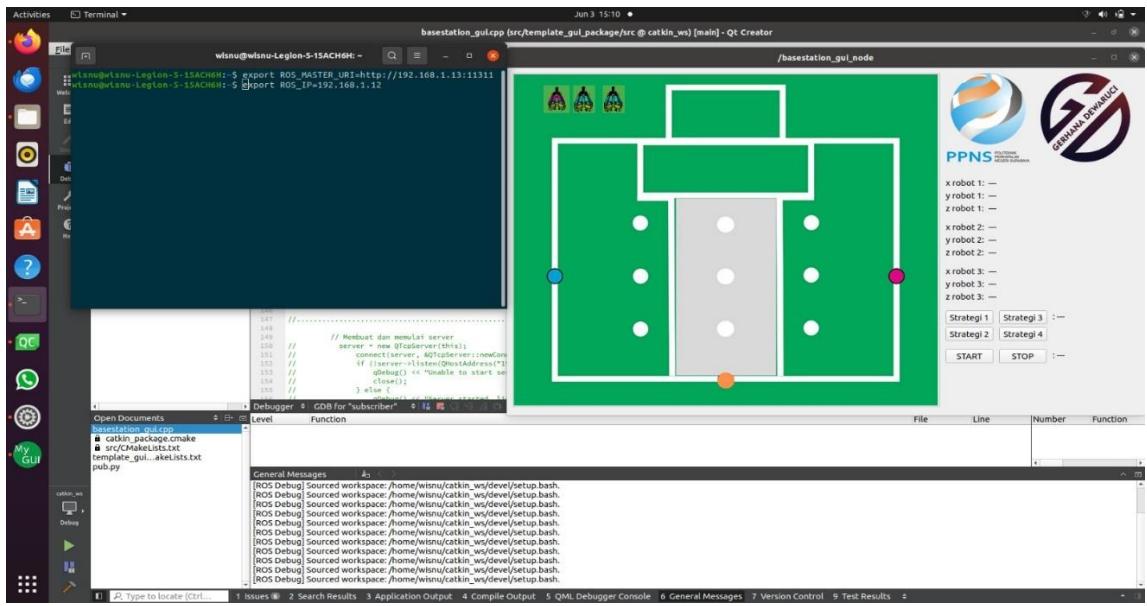
Gambar 4. 17 Ip Address Device Robot

Pada Gambar 4.16 dan Gambar 4.17 adalah membuat ip address yang di gunakan ROS untuk berkomunikasi.



Gambar 4. 18 Export Address Basestation Menjadi ROS Master

Seperti yang ada pada Gambar 4.18 dimana format tersebut membuat ip 192.168.1.13 menjadi ROS Master setelah itu mendaftarkan device dengan ip address yang sama agar device daripada basestation dapat berkomunikasi dengan device lainya seperti pada Gambar 4.19.



Gambar 4. 19 Export Address Basestation ke ROS Master

```
Activities Terminal Jun 3 14:31
asyl@asyl:~$ export ROS_MASTER_URI=http://192.168.1.13:11311
asyl@asyl:~$ export ROS_IP=192.168.1.12
asyl@asyl:~$ rostopic /lstat
rostopic is a command-line tool for printing information about ROS Topics.

Commands:
  rostopic bw      display bandwidth used by topic
  rostopic delay   display delay of topic from timestamp in header
  rostopic echo    print messages to screen
  rostopic end     find topics of type
  rostopic hz      display publishing rate of topic
  rostopic info    print information about active topic
  rostopic list    list active topics
  rostopic pub    publish data to topic
  rostopic type   print topic or field type

Type rostopic <command> -h for more detailed usage, e.g. 'rostopic echo -h'

asyl@asyl:~$ rostopic list
ERROR: Unable to communicate with master!
asyl@asyl:~$ rostopic list
/basestation_gui_node/chatter
/robot_1
/robot_2
/robot_3
/asyl/asyl:~$
```

Gambar 4. 20 Pembacaan Topic dari ROS Master

Seperti yang ada pada Gambar 2.20 dimana device daripada robot juga melakukan export ros master dengan ip 192.168.1.13, dengan melakukan export ip basestation maka device ROS dari device robot akan membaca bahwa ip 192.168.1.13 adalah ROS Master. Pada Gambar 4.20 di perlihatkan bahwa device robot telah dapat membaca ROS Topic dari basestation, jadi dipastikan bahwa komunikasi sudah dapat berjalan karena robot dapat mengakses ROS Topic basestation.

(Halaman ini sengaja dikosongkan)

DAFTAR PUSTAKA

- Darmawan, W., Basuki Rahmat, M., Khumaidi, A., Yudha Adhitya, R., & Pristovani Riananda, D. (2023). Perancangan Strategi Keputusan Robot Sepak Bola Beroda menggunakan Metode Decision Tree. In *Jurnal Elektronika dan Otomasi Industri* (Vol. 10, Issue 2). <https://doi.org/10.33795/elkolind.v10i2.3020>
- Edy Surya Prabowo, K., Divayana, Y., & Rahardjo, P. (2022). Perancangan Aplikasi Base Station Dalam Sistem Koordinasi Robot Sepak Bola Beroda Dengan Multi Thread. *Jurnal SPEKTRUM*, 9(4), 17. <https://doi.org/10.24843/spektrum.2022.v09.i04.p3>
- Fouk, M. A., Hudiono, H., & Hariyadi, A. (2022). Implementation of Base Station as An Intermediary Referee Box in the delivery of Wheeled Football Robot Movement Commands. *Jartel*, 12(3), 114–120. <https://doi.org/10.33795/jartel.v12i3.314>
- Herman, S., Studi, P., Mesin, T., Mesin, J. T., Teknik, F., Sriwijaya, U., Saputra,
R. A., IRLANE MAIA DE OLIVEIRA, Rahmat, A. Y., Syahbanu, I.,
Rudiyansyah, R., Sri Aprilia and Nasrul Arahman, Aprilia, S.,
Rosnelly, C. M., Ramadhani, S., Novarina, L., Arahman, N., Aprilia,
S., Maimun, T., ... Jihannisa, R. (2019). No 主観的健康感を中心とした在宅高齢者における健康関連指標に関する共分散構造分析
Title. *Jurusan Teknik Kimia USU*, 3(1), 18–23.
- Jalil, A. (2018). Robot Operating System (Ros) Dan Gazebo Sebagai Media Pembelajaran Robot Interaktif. *ILKOM Jurnal Ilmiah*, 10(3), 284–289. <https://doi.org/10.33096/ilkom.v10i3.365.284-289>
- Kusumastutie, D. A. W., & Alif Fiolana, F. (2020). Design of Wheeled Football Robot Coordination System at Base Station Using TCP / IP. *JEEE-U (Journal of Electrical and Electronic Engineering-UMSIDA)*, 4(1), 1–17. <https://doi.org/10.21070/jeeeu.v4i1.341>
- Maulidina, H. (2019). No Title. ペインクリニック学会治療指針 2, 2, 1–13.
- Nasikhin, K. (2019). *Pemetaan Posisi Robot Sepak Bola Beroda Menggunakan Metode Gyrodometry Untuk Memprediksi Sudut Tendangan Bola Terhadap Gawang Lawan Dengan Perhitungan Trigonometri*.
- Octavian, Y., Widodo, H. A., Khumaidi, A., Teknik, J., Kapal, K., Perkapalan, P., & Surabaya, N. (2021). *Optimasi Deteksi Bola Pada Robot Sepak Bola*.
- Onggrono, K., Tulus, T., & Nababan, E. B. (2017). Analisis Penggunaan Parallel

- Processing Multithreading Pada Resilient Backpropagation. *InfoTekJar (Jurnal Nasional Informatika Dan Teknologi Jaringan)*, 2(1), 33–40. <https://doi.org/10.30743/infotekjar.v2i1.146>
- Pamungkas, R. A., Maulana, A., Sya'ban, D. P., Ramdani, R., Musafa, A., & Riyanto, I. (2018). WiFi Data Communication System Design for Wheeled Soccer Robot Controller. *Advanced Science Letters*, 24(11), 8782–8786. <https://doi.org/10.1166/asl.2018.12345>
- Pratikto, R. B., Setiawan, E., & Syauqy, D. (2021). Rancang Bangun Simulasi Robot Beroda untuk Pengiriman Barang di dalam Gedung berbasis Metode Particle Filter. ... *Teknologi Informasi Dan Ilmu* ..., 5(8), 3229–3236. <https://j-ptiik.ub.ac.id/index.php/j-ptiik/article/view/9528%0Ahttp://j-ptiik.ub.ac.id/index.php/j-ptiik/article/download/9528/4300>
- Putri, I. D. (2022). Rancang Bangun Jaringan Peer-To-Peer Webcam. *Jurnal Media Infotama*, 1, 7.
- Safatain, R., Khumaidi, A., Sukoco, D., Studi, P., Otomasi, T., Teknik, J., Kapal, K., Perkapalan, P., & Surabaya, N. (2022). *Tugas akhir (ae43250)*.
- Saputra, R. D. (2018). Implementasi Jaringan Peer To Peer Dalam Proses Transfer Data Dua Personal Computer Menggunakan Kabel Utp Bertype Cross. *Universitas Mitra Indonesia*, 02(02), 11–16. https://d1wqtxts1xzle7.cloudfront.net/63186773/Modul_9_FTP_Server20200503-117812-c1vgwy-libre.pdf?1588658685=&response-content-disposition=attachment%3B+filename%3DModul_9_File_Transfer_Protocol_FTP_Serve.pdf&Expires=1695878269&Signature=NSyEouVnHPfekQsCh
- Studi, P., Otomasi, T., Teknik, J., Kapal, K., Perkapalan, P., & Surabaya, N. (2023). *PERBAIKAN NILAI ERROR POSISI ROBOT KRSBI-BEROUDA DENGAN MENGGUNAKAN ROTARY ENCODER BUILT-IN DAN EKSTERNAL DENGAN ALGORITMA SENSOR FUSION*.
- Tjoanapessy, F., Poekoel, V. C., Salmon, A., Lumenta, M., Robot, R. F., Elektro, T., Sam, U., Manado, R., & Manado, J. K. B. (2019). Aplikasi Base Station Untuk Robot Sepak Bola Beroda. *Jurnal Teknik Informatika*, 14(3), 285–290.
- Zhang, Q., Liu, Z., Zhao, J., Zhang, S., Abdelghany, M. A., Elnady, A. O., Ibrahim, S. O., Apriaskar, E., Fahmizal, F., Salim, N. A., Prastiyanto, D., Krista, I., M, E. D., Suryadi, D., Qomarudin, M. N., Surabaya, M., Prayogo, R. A., Prihatama Kunto Wicaksono, Rachmawan, A., & Hudati, I. (2017). Penentuan Posisi Robot Sepak Bola Beroda Menggunakan Rotary Encoder dan Kamera. *Tugas Akhir ITS*, 4(November), 172–176. <http://repository.its.ac.id/44447%0Ahttps://repository.its.ac.id/41995%0Ahttps://repository.its.ac.id/41995/1/2213106021-Undergraduate-Thesis.pdf>
- Visual Studio. (2020). *Getting Started*. Retrieved from <https://code.visualstudio.com/docs>
- ArduinoIndonesia.id. (2023).

