# SISTEM NAVIGASI PADA ROBOT SEPAK BOLA BERODA KRSBI MENGGUNAKAN METODE ODOMETRY

**PROYEK AKHIR**

Laporan akhir ini dibuat dan diajukan untuk memenuhi salah satu syarat kelulusan Diploma III Politeknik Manufaktur Negeri Bangka Belitung


Disusun oleh :

| | | |
|---|---|---|
| Eko Okta Pratama | NIRM | 0031638 |
| Chiki Leamongga | NIRM | 0031634 |

**POLITEKNIK MANUFAKTUR NEGERI**

**BANGKA BELITUNG**


**2019**

# LEMBAR PENGESAHAN

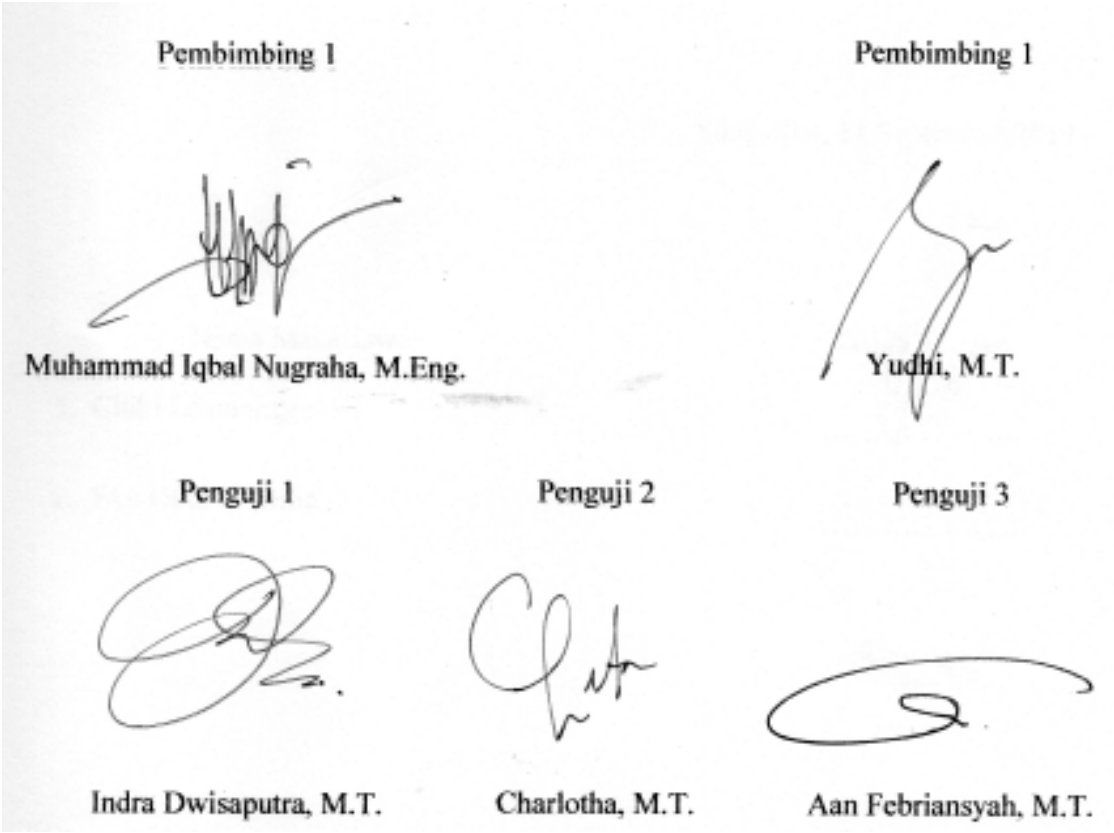
**SISTEM NAVIGASI PADA ROBOT SEPAK BOLA BERODA KRSBI MENGGUNAKAN METODE *ODOMETRY***

Oleh:


| | | |
|---|---|---|
| Chiki Leamongga | NIRM | 0031634 |
| Eko Okta Pratama | NIRM | 0031638 |


Laporan akhir ini telah disetujui dan disahkan sebagai salah satu syarat kelulusan Program Diploma III Politeknik Manufaktur Negeri Bangka Belitung

Menyetujui,




## PERNYATAAN BUKAN PLAGIAT

Yang bertanda tangan dibawah ini:

Nama Mahasiswa 1 : Chiki Leamongga NIRM : 0031634
Nama Mahasiswa 2 : Eko Okta Pratama NIRM : 0031638

Dengan judul : Sistem Navigasi Pada Robot Sepak Bola Beroda KRSBI
Menggunakan Metode *Odometry*

Menyatakan bahwa laporan akhir ini adalah hasil kerja kami sendiri dan bukan merupakan plagiat. Pernyataan ini kami buat dengan sebenarnya dan bila ternyata dikemudian hari ternyata melanggar

pernyataan ini, kami bersedia menerima sanksi yang berlaku.

Sungailiat, 11 September 2019

| Nama Mahasiswa | Tanda Tangan |
| --- | --- |
| 1. Chiki Leamongga | ……. |
| 2. Eko Okta Pratama | ……. |

# ABSTRAK

*Pada KRSBI, setiap tim diwajibkan membuat 3 robot otomatis yang dapat bergerak berdasarkan strategi yang diinginkan peserta seperti bergerak menuju bola, menendang bola ke gawang lawan, dan menghalau bola masuk ke gawang sendiri. Untuk dapat menerapkan strategi yang diinginkan, tentunya diperlukan komponen dan fitur-fitur yang mendukung. Pembacaan posisi adalah salah satu fitur penting dalam robot sepak bola beroda. Apabila robot tidak mengetahui posisinya di lapangan, maka strategi yang diterapkan sangat terbatas. Salah satu cara untuk mengetahui posisi robot adalah dengan membuat sistem navigasi menggunakan metode odometry. Metode odometry ini berfungsi untuk membaca pergerakan atau perpindahan robot dalam vektor posisi dan orientasi (x,y,θ). Metode pelaksanaan proyek akhir ini dimulai dari studi literatur, observasi robot, perancangan, pembuatan, analisis hasil, dan pembuatan laporan. Sistem navigasi dirancang menggunakan tiga buah sensor rotary encoder, tiga roda omni, sensor kompas, dan modul komunikasi nirkabel bluetooth. Sensor odometri didisain menggunakan tiga roda omni, dimana beda sudut antar masing-masing poros roda adalah 120º. Mekanisme odometrinya adalah diawali dengan pembacaan data rotary encoder dari masing-masing roda, lalu seluruh nilai tersebut dikombinasikan menggunakan kinematika maju untuk mendapatkan data pergerakan robot. Sensor kompas berfungsi untuk membaca orientasi robot. Hasil pergerakan robot ditampilkan secara real time melalui modul Bluetooth pada PC, yang meliputi posisi robot dalam koordinat, jarak, kecepatan, dan orientasi robot. Setelah dilakukan beberapa pengujian terhadap gerakan translasi robot diperoleh presentase error rata-rata secara keseluruhan adalah sebesar 0.34%.*

*Kata Kunci : Sistem navigasi, rotary encoder, three omni-directional.*

# *ABSTRACT*

*In KRSBI, every team is required to make three automatic robots which are able to move based on the strategy applied by the user such as move towards the ball, make a goal, against the opponent, and dispel the ball. To implement the desired strategy, supported components and features for the robots are needed. Recognition the robot position is one of important feature in wheeled soccer robot. If the robot is not able to recognize its position in the field, the strategy that can be applied will be limited. One way to localize the robot is that by building a navigation system using odometry method. Odometry method aims to read the robot movement or displacement in the form of position and orientation vector (x,y, θ). This final project is carried out with literature study, observation, designing, building, analyzing, and making the report respectively. The navigation system is designed by using three rotary encoder sensors, three omniwheels, a compass sensor, and a Bluetooth module. Odometry sensor is designed using three omniwheels, which the angle difference among omniwheels is 120º. The odometry mechanism is begun with reading the data of every rotaray encoder, then all the values are combined using forward kinematic to get the data of robot displacement. Compass is used to read the robot orientation. The result of robot movement is displayed on a PC via Bluetooth containing robot position in coordinate form, distance, velocity, and orientation. After conducting several experiments of robot linear movement, it was obtained that the overall average of error percentage is 0.34%.*

*Keywords: Navigation system, rotary encoder, three omni-directional*

# KATA PENGANTAR

Assalamu'alaikum Warahmatulahi Wabarakatuh

Alhamdulillah, Puji dan syukur kami panjatkan kehadirat Allah SWT, karena berkat, rahmat dan karunia-Nyalah kami dapat menyelesaikan Proyek Akhir yang berjudul "Sistem Navigas Pada Robot Sepak Bola Beroda KRSBI Menggunakan Metode *Odometry*" tepat pada waktunya. Proyek akhir ini disusun untuk memenuhi salah satu persyaratan atau kewajiban mahasiswa dalam menyelesaikan kurikulum program Diploma III Teknik Elektronika di Politeknik Manufaktur Negeri Bangka Belitung.

Dalam penyusunan laporan dan pelaksanaan proyek akhir ini telah banyak pihak yang membantu penulis. Untuk itu, penulis ingin mengucapkan terima kasih yang sebesar-besarnya kepada :

1. Kedua orang tua dan keluarga yang terus memotivasi, memberikan dukungan dan terus mendoakan kami;
2. Bapak Muhammad Iqbal Nugraha, M.Eng. selaku pembimbing I dan Bapak Yudhi, M.T. selaku pembimbing II yang telah meluangkan banyak waktu dan memberikan ilmu akademik dan non-akademik dalam menyelesaikan proyek akhir ini;
3. Seluruh dosen-dosen Politeknik Manufaktur Negeri Bangka Belitung, yang telah mengajarkan banyak hal sehingga kami menjadi seorang yang mempunyai wawasan dan ilmu akademik, pendidikan moral, dan cara pikir yang luas terhadap dunia; dan
4. Seluruh kawan-kawan "Markas" yang telah banyak meluangkan waktu untuk membantu menyelesaikan Proyek Akhir ini. Kalian luar biasa!

Demikian Laporan Proyek Akhir ini kami buat. Semoga dapat bermanfaat dan berguna sebagaimana yang diharapkan.

Sungailiat, 17 Agustus 2019



Penulis

# DAFTAR ISI

# DAFTAR TABEL

# DAFTAR GAMBAR

# DAFTAR LAMPIRAN

# BAB I
# PENDAHULUAN

## 1.1 Latar Belakang

Kontes Robot Sepak Bola Beroda Indonesia (KRSBI) adalah kontes robotik tingkat perguruan tinggi di Indonesia. KRSBI menuntut mahasiswa untuk bisa mengembangkan kemampuan dalam bidang robotika, *manufaktur*, elektronika, pemrograman, komonikasi digital, *image processing*, *article intelligent*, strategi, kemampuan meneliti dan menulis artikel ilmiah, sekaligus diperlukan pengembangan kearah disiplin, toleransi, sportifitas, kerjasama, saling menghargai, kontrol emosi dan kemampuan *softskill* lainnya. Dalam KRSBI setiap tim diwajibkan membuat 3 robot otomatis. Masing-masing robot bergerak berdasarkan strategi yang digunakan peserta, untuk dapat bergerak menuju bola, mencetak gol ke gawang lawan dan menghalau bola masuk gawang sendiri agar tidak kemasukan gol. Untuk dapat menerapkan strategi yang diinginkan tentunya diperlukan komponen dan fitur-fitur yang mendukung agar dapat dijalankan. Sistem navigasi adalah salah satu fitur penting dalam menjalankan fungsi robot.

Penentuan posisi adalah salah satu masalah utama dalam pembuatan robot sepak bola beroda. Apabila robot tidak mengetahui posisinya pada lapangan, maka koordinasi antar robot dan tindakan selanjutnya akan sulit ditentukan. Salah satu cara untuk mengetahui posisi robot adalah dengan membuat sistem navigasi menggunakan metode *odometry*. Metode *odometry* ini dapat membaca pergerakan robot menuju target berdasarkan titik(x,y,θ) target dari titik *start* robot (x=0, y=0, θ=0).


Gambar 1.1 Robot Sepak Bola Beroda Polman Babel

## 1.2 Perumusan Masalah

Berdasarkan pembahasan pada latar belakang rumusan masalah pada Proyek Akhir ini adalah sebagai berikut:

1. Bagaimana merancang kinematik untuk metode odometry.
2. Bagaimana mengkombinasikan 3 sensor *rotary encoder*.
3. Bagaimana membuat sistem navigasi yang dapat memberikan informasi perpindahan posisi, jarak, kecepatan, dan orientasi robot.

## 1.3 Batasan Masalah

Supaya penelitian pada Proyek Akhir ini lebih terarah dan memudahkan dalam pembahasan, maka perlu adanya batasan masalah adalah sebagai berikut:

1. Pengujian tidak diterapkan langsung pada robot sepak bola, tetapi digerakkan secara manual.
2. Komunikasi menggunakan *bluetooth.*

## 1.4 Tujuan Proyek Akhir

Mengacu pada perumusan masalah diatas, maka tujuan dari Proyek Akhir ini adalah sebagai berikut:

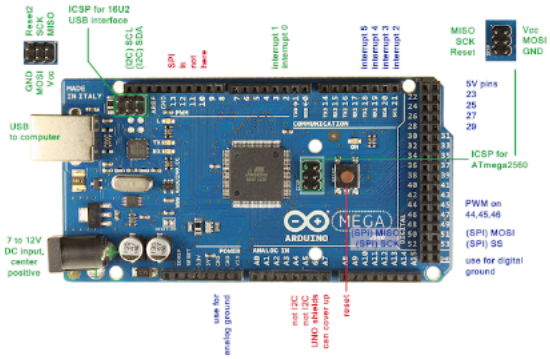
1. Merancang dan membuat *kinematic three omni-directional.*
2. Merancang dan membuat sistem navigasi dengan metode *odometry* yang memberikan informasi posisi, kecepatan, jarak, dan orientasi robot secara *real time*.
3. Membuat sistem yang dapat membantu *main controller* pada robot sepak bola.

# BAB II
# LANDASAN TEORI

## 2.1 Arduino Mega 2560

Arduino adalah *board* berbasis mikrokontroler atau papan rangkaian elektronik *open source* yang di dalamnya terdapat komponen utama yaitu sebuah chip mikrokontroler dengan jenis AVR dari perusahaan Atmel. Mikrokontroler itu sendiri adalah chip atau IC (*integrated circuit*) yang bisa diprogram menggunakan komputer. Tujuan menanamkan program pada mikrokontroler adalah agar rangkaian elektronik dapat membaca input, memproses input tersebut dan kemudian menghasilkan *output* sesuai yang diinginkan. Jadi mikrokontroler bertugas sebagai otak yang mengendalikan proses input, dan output sebuah rangkaian elektronik.

Pada Gambar 2.1 merupakan jenis Arduino Mega *type* 2560, Arduino Mega 2560 adalah papan pengembangan mikrokontroller yang berbasis Arduino denganmenggunakan chip AT Mega 2560. Board ini memiliki pin I/O yang cukup banyak,sejumlah 54 buah digital I/O pin (15 pin diantaranya adalah PWM), 16 pin analoginput, 4 pin UART (*serial port hardware*). Arduino Mega 2560 dilengkapi dengansebuah *oscillator* 16 Mhz, sebuah port USB, *power jack* DC, ICSP *header*, dantombol reset. Board ini sudah sangat lengkap, sudah memiliki segala sesuatu yangdibutuhkan untuk sebuah mikrokontroller (1).

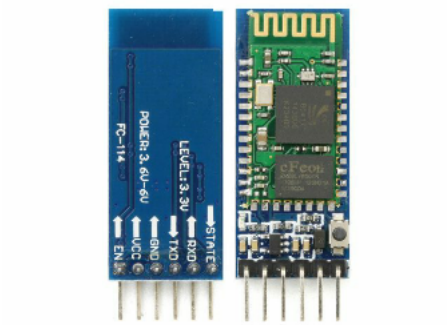



Gambar 2.1 Arduino Mega 2560 (2)

Dengan penggunaan yang cukup sederhana, anda tinggal menghubungkan power dari USB ke PC anda atau melalui adaptor AC/DC ke *jack* DC.

## 2.2 *Bluetooth* HC 05

Modul *Bluetooth* HC-05 terdiri dari 6 pin konektor, yang setiap pin konektor memiliki fungsi yang berbeda-beda. Modul *Bluetooth* HC-05 dengan *supply* tegangan sebesar 3,3 V ke pin 12 modul *Bluetooth* sebagai VCC. Pin 1 pada modul *Bluetooth* sebagai *transmitter*, kemudian pin 2 pada *Bluetooth* sebagai *receiver*. Untuk gambar modul *bluetooth* dapat dilihat pada Gambar 2.2 dibawah ini (3):




Gambar 2.2 Modul *Bluetooth* HC 05 (4)

## *2.3 Rotary Encoder*

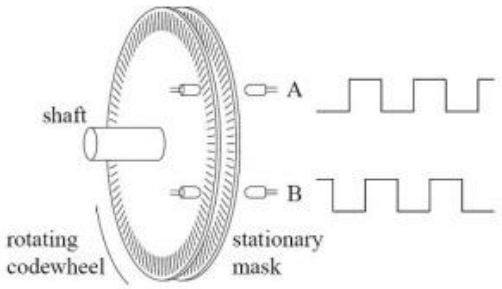
*Rotary encoder* tersusun dari suatu piringan tipis yang memiliki lubang-lubang pada bagian lingkaran piringan. LED ditempatkan pada salah satu sisi piringan sehingga cahaya akan menuju ke piringan. Di sisi yang lain suatu *photo*-transistor diletakkan sehingga *photo*-transistor ini dapat mendeteksi cahaya dari LED yang berseberangan. Piringan tipis tadi dikopel dengan poros motor, atau divais berputar lainnya yang ingin kita ketahui posisinya, sehingga ketika motor berputar piringan juga akan ikut berputar. Apabila posisi piringan mengakibatkan cahaya dari LED dapat mencapai *photo*-transistor melalui lubang-lubang yang ada, maka *photo*-transistor akan mengalami saturasi dan akan menghasilkan suatu pulsa gelombang persegi. Semakin banyak deretan pulsa yang dihasilkan

pada satu putaran menentukan akurasi *rotary encoder* tersebut, akibatnya semakin banyak jumlah lubang yang dapat dibuat pada piringan menentukan akurasi *rotary encoder* tersebut (5).


Gambar 2.3 *Rotary Encoder* (6)

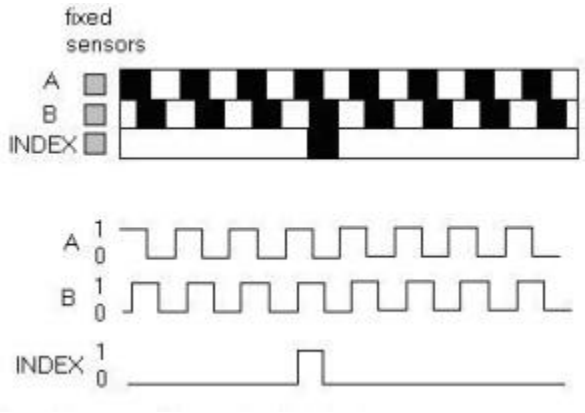
### 2.3.1 Incremental *Rotary Encoder*

*Incremental encoder* terdiri dari dua *track* atau *singletrack* dan dua sensor yang disebut *channel* A dan B (Gambar 2.5). Ketika poros berputar, deretan pulsa akan muncul di masing-masing *channel* pada frekuensi yang proporsional dengan kecepatan putar sedangkan hubungan fasa antara *channel* A dan B menghasilkan arah putaran. Dengan menghitung jumlah pulsa yang terjadi terhadap resolusi piringan maka putaran dapat diukur. Untuk mengetahui arah putaran, dengan mengetahui *channel* mana yang *leading* terhadap *channel* satunya dapat kita tentukan arah putaran yang terjadi karena kedua *channel* tersebut akan selalu berbeda fasa seperempat putaran (*quadrate resignal*). Seringkali terdapat output *channel* ketiga, disebut INDEX, yang menghasilkan satu pulsa per putaran berguna untuk menghitung jumlah putaran yang terjadi (5).

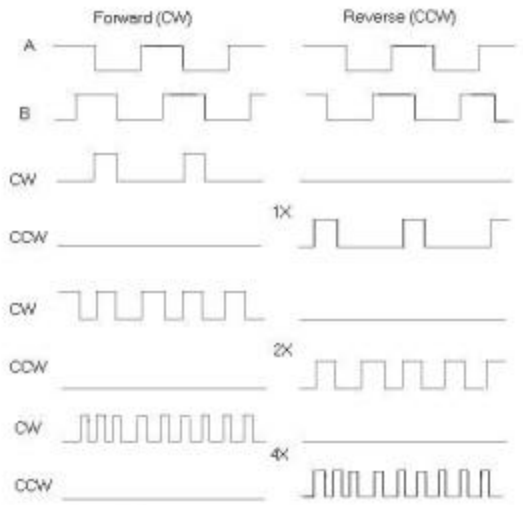



Gambar 2.4 Susunan Piringan *Incremental Encoder* (5)

Contoh pola diagram keluaran dari suatu *incremental Encoder* ditunjukkan pada Gambar 2.6 Resolusi keluaran dari sinyal *quadrature*A dan B dapat dibuat beberapa macam, yaitu 1X, 2X dan 4X. Resolusi 1X hanya memberikan pulsa tunggal untuk setiap siklus salah satu sinya A atau B, sedangkan resolusi 4X memberikan pulsa setiap transisi pada kedua sinyal A dan B menjadi empat kali resolusi 1X. Arah putaran dapat ditentukan melalui level salah satu sinyal selama transisi terhadap sinyal yang kedua. Pada contoh resolusi 1X, A=arah bawah dengan B=1 menunjukkan arah putaran searah jarum jam, sebaliknya B=arah bawah dengan A=1 menunjukkan arah berlawanan jarum jam.




Gambar 2.5 Pola Keluaran *Incremental Encoder* (5)




Gambar 2.6 *Output* dan Arah Putaran Pada Resolusi Yang Berbeda-beda (5)

## 2.4 Kinematik *Omni direction* 3 Roda

Desain robot holonomis yang terdiri dari tiga roda *omni directional* Swedia 90□90□ yang ditempatkan terpisah 120 derajat. Diagram geometrik robot holonomis yang digunakan untuk menemukan model kinematiknya. Pusat massa robot, P oP didefinisikan sebagai vektor yang menghubungkan O ke titik asal dan D iD i adalah vektor arah penggerak setiap roda. Matriks rotasi satu kesatuan R (θ) didefinisikan sebagai:

$$R(\theta)=\begin{pmatrix}\cos\theta & -\sin\theta\\ \sin\theta & \cos\theta\end{pmatrix} \quad (2.1)$$

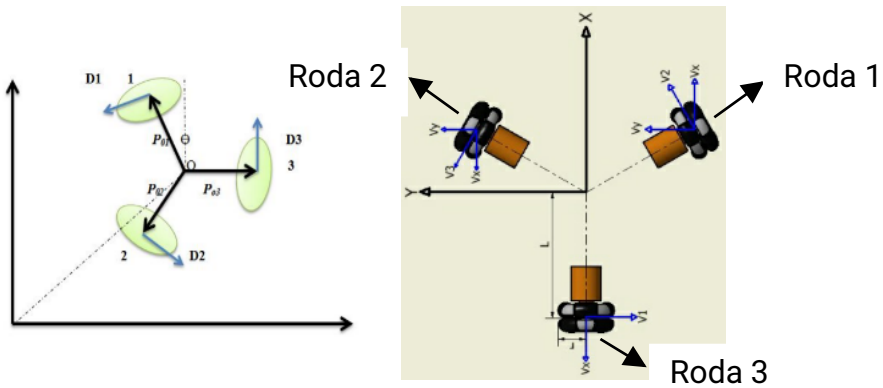
di mana, θ adalah sudut rotasi berlawanan arah jarum jam.

$P_0$ menunjukkan posisi pusat massa sehubungan dengan *frame universal,* mis.

$P_0$ = [x y] TPo = [x y] T. Posisi [xi yi] T [xi yi] T dari setiap roda dapat diberikan sehubungan dengan pusat massa robot, yaitu, untuk i = 1, 2, 3.

$$P_{01}=\begin{bmatrix}xi\\ yi\end{bmatrix}=R(\theta)\,L\begin{bmatrix}0\\ 1\end{bmatrix} \quad (2.2)$$

dimana L adalah jarak roda dari pusat massa robot (O).




Gambar 2.7 Kinematik *Omnidirectional*

$P_{01}$, $P_{02}$ dan $P_{03}$ sehubungan dengan koordinat lokal yang berpusat di pusat massa robot diberikan sebagai:

$$P_{01} = L\begin{bmatrix}0\\1\end{bmatrix}$$

$$P_{02} = R\left(\frac{2\pi}{3}\right)* P_{01} = \frac{L}{2}\begin{bmatrix}-1\\-\sqrt{3}\end{bmatrix}$$

$$P_{03} = \left(\frac{4\pi}{3}\right)* P_{01} = \frac{L}{2}\begin{bmatrix}-\sqrt{3}\\-1\end{bmatrix}$$

Petunjuk arah Di dari setiap roda dapat diperoleh

$$Di = \frac{1}{L}R\left(\frac{\pi}{2}\right) P_{0i},\ i = 1,2,3 \qquad (2.3)$$

Yang menghasilkan

$$D_1 = \begin{bmatrix}-1\\0\end{bmatrix} \qquad D_2 = \frac{1}{2}\begin{bmatrix}\sqrt{3}\\-1\end{bmatrix} \qquad D_3 = \frac{1}{2}\begin{bmatrix}1\\\sqrt{3}\end{bmatrix}$$

Posisi dan kecepatan setiap roda sehubungan dengan bingkai dunia kemudian dinyatakan oleh, untuk i=1, 2, 3

$$Ri = P_o + R\left(\theta + \frac{2\pi}{3}(i-1)\right) P_{oi} \qquad vi = Po + R\left(\theta\ \frac{2\pi}{3}(i1)\right) P_{oi} \qquad (2.4)$$

Kecepatan translasi setiap roda diperoleh sebagai:

$$Vi = vi^T\left(R\left(\theta + \frac{2\pi}{3}(i-1)\right)Di\right) \qquad (2.5)$$

Ini memberi

$$V_1 = -\cos(\theta)\, x - \sin(\theta)\, y + L\,\theta$$

$$V_2 = \cos\left(\frac{\pi}{3} - \theta\right) x - \sin\left(\frac{\pi}{3} - \theta\right) y + L\,\theta$$

$$V_3 = \cos\left(\frac{\pi}{3} + \theta\right) x + \sin\left(\frac{\pi}{3} + \theta\right) y + L\,\theta$$

Yang dapat ditulis dalam bentuk matriks sebagai:

$$\begin{bmatrix} V1 \\ V2 \\ V3 \end{bmatrix} = P(\theta) \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} \tag{2.6}$$

Dimana

$$P(\theta) = \begin{bmatrix} -\cos\theta & -\sin\theta & L \\ \cos\left(\frac{\pi}{3} - \theta\right) & \sin\left(\frac{\pi}{3} - \theta\right) & L \\ \cos\left(\frac{\pi}{3} + \theta\right) & \sin\left(\frac{\pi}{3} + \theta\right) & L \end{bmatrix}$$

Adalah matriks kinematika terbalik. Matriks P (θ) selalu tunggal untuk nilai θ. Jadi,

$$P^{-1}(\theta) = \begin{bmatrix} -\frac{2}{3}\cos\theta & \frac{2}{3}\cos\left(\frac{x}{3} - \theta\right) & \frac{2}{3}\cos\left(\frac{x}{3} + \theta\right) \\ \frac{-2}{3}\sin\theta & \frac{-2}{3}\sin\left(\frac{x}{3} - \theta\right) & \frac{2}{3}\sin\left(\frac{x}{3} + \theta\right) \\ \frac{1}{3L} & \frac{1}{3L} & \frac{1}{3L} \end{bmatrix} \tag{2.7}$$

## *2.5* Aplikasi *Processing*

### 2.5.1. Pengertian

*Processing* adalah bahasa pemrograman dan lingkungan pemrograman (*development environment*) *open source* untuk memprogram gambar, animasi dan interaksi. Digunakan oleh pelajar, seniman, desainer, peneliti, dan hobbyist untuk belajar, membuat prototipe,

dan produksi. *Processing* digunakan untuk mengajarkan dasar-dasar pemrograman komputer dalam konteks rupa dan berfungsi sebagai buku sketsa perangkat lunak (*software*) dan *tool* produksi profesional. *Processing* mengaitkan konsep *software* pada prinsip-prinsip bentuk rupa, gerak, dan interaksi. *Processing* mengintegrasikan suatu bahasa pemrograman, lingkungan pemrograman, dan metodologi pengajaran ke dalam sistem terpadu (7).

### 2.5.2. Cara menggunakan Processing

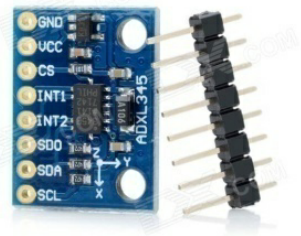
Jika kalian sedang mengambil matakuliah menggunakan aplikasi ini mungkin sudah mengetahui caranya tapi sebagian pasti tidak tau jadi untuk dapat menjalankan program processing silahkan kalian buka aplikasinya yang sudah kalian buka tadi maka akan tampil seperti ini (8):

1. *Button Play* : berguna untuk menjalankan sript yang kalian masukan pada aplikasi *processing* tersebut.
2. *Button Stop* : berguna untuk menghentikan program yang sedang berjalan dari program yang kalian jalankan sebelumnya.
3. *New* : untuk membuat file baru.
4. *Open* : untuk membuka script lama yang pernah kalian buat atau *download*.
5. *Export* : untuk mengexport aplikasi ke platform os lainnya sepeti Linux, Windows dan lainnya.

## 2.6 Sensor MPU 6050

Sensor MPU 6050 adalah sensor yang mampu membaca kemiringan sudut berdasarkan data dari sensor *accelerometer* dan sensor *gyroscope*. *Accelerometer* adalah perangkat yang tidak menggunakan gravitasi bumi yang berfungsi untuk menentukan orientasi (arah), contonya adalah mengganti wallpaper hanya dengan gerakan pada penggunaan pada *smartphone*. *Gyroscope* adalah perangkat yang menggunakan gravitasi bumi yang berfungsi untuk menentukan rotasi (perputaran), contohnya adalah kompas pada penggunaan pada

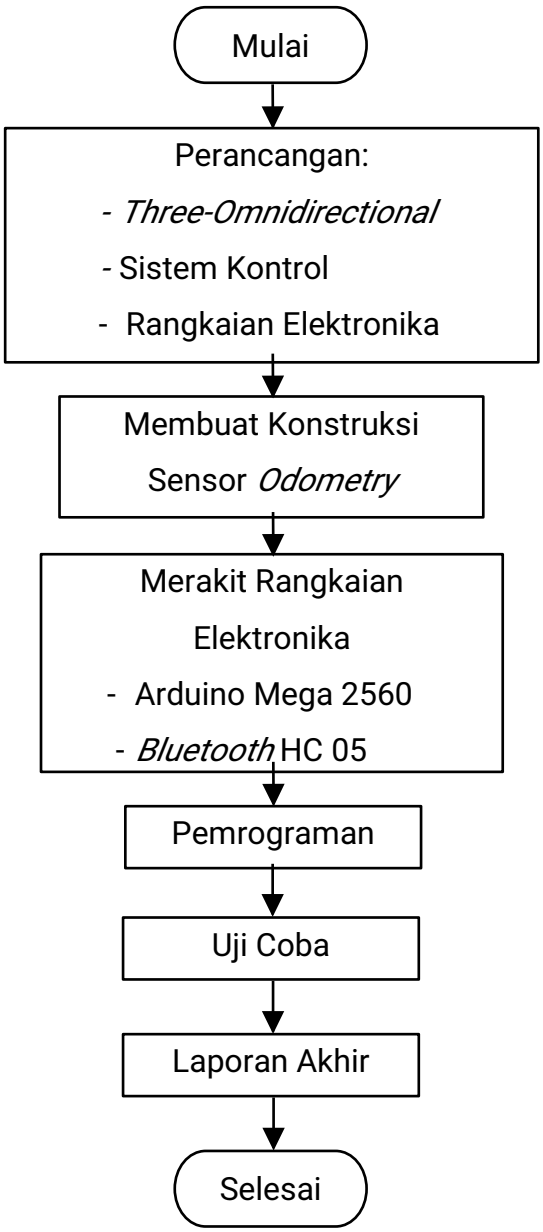
*smartphone.* Sensor ini juga dilengkapi oleh sensor suhu yang dapat digunakan untuk mengukur sushu dikeadaan sekitar. Jalur data yang digunakan pada sensor ini adalah jalur data 12C. Berikut ini merupakan gambar sensor MPU 6050. (9)




Gambar 2. 8 Sensor MPU 6050 (9)

# BAB III
# METODE PELAKSANAAN

Pada bab ini kami akan menjelaskan tentang metode pelaksanaan dari Proyek Akhir kami, yaitu seperti gambar dibawah ini.




Gambar 3.1 Alur Metode Pelaksanaan

## 3.1 Studi Pusaka dan Observasi Robot

Metode ini dilakukan untuk mencari data dan materi melalui jurnal, *browsing*, buku, konsultasi, dan laporan akhir alumni. Pada tahap ini penulis mencari referensi tentang *kinematic holonomic omni-directional*, metode *odometry*, sensor *rotary encoder,* sensor MPU 6050, komunikasi menggunakan *bluetooth*, dan aplikasi processing. Setelah melakukan menggumpulan data didapat daftar kebutuhan untuk membuat tugas akhir.Data-data tersebut kemudian menjadi acuan dalam membuat "Sistem Navigasi Pada Robot Sepak Bola Beroda Menggunakan Metode *Odometry*". Berikut daftar kebutuhannya:

Tabel 3.1 Daftar Kebutuhan

| Uraian |
|---|
| Sensor *rotay encoder* dapat mendeteksi kecepatan, jarak, dan posisi robot. |
| Sensor MPU 6050 dapat membaca orientasi robot. |
| Arduino Mega 2560 membaca dan mengolah nilai sensor lalu mengirimkan data ke PC melalui modul *bluetooth.* |
| pat menampilkan hasil pergerakan robot secara *real time.* |

## 3.2 Perancangan

Perancangan ini meliputi pembuatan *hardware* dan *software*. Pembuatan *hardware* seperti pemilihan bahan dan penentuan ukuran sesuai dengan yang ada di robot sepak bola beroda Polmanbabel. *Hardware* yang dibuat antara lain:

1. Konstruksi sensor
2. Dudukan *rotary encoder*
3. Modifikasi *shaft* roda

Pembutan *software* meliputi pemrograman arduino mega 2560 dan aplikasi processing, yang berfungsi untuk membuat sistem navigasi

sesuai daftar kebutuhan yang sudah dibuat sebelumnya. Pemrograman dilakukan menggunakan *software* arduino dan processing, menggunakan bahasa C. Pemrograman yang akan dibuat antara lain:

1. Pemrograman untuk membaca nilai 3 buah sensor *rotary encoder* dan MPU 6050 lalu mengolahnya jadi informasi jarak, kecepatan, posisi, dan orientasi, Pemrograman untuk mengolah data dari mikrokontroler ke PC menggunakan modul *bluetooth* HC 05.
2. Program untuk menampilkan hasil pergerakan robot.

## 3.3 Uji Coba

Metode ini digunakan untuk menguji fungsi dari kerja alat, sehingga jika ada yang tidak sesuai dapat dilakukan analisa. Uji coba alat dilakukan dibagi menjadi beberapa tahapan, yaitu:

1. Uji coba *hardware*

   Uji coba *hardware* meliputi kontruksi dan elektrik robot. Uji coba ini dilakukan untuk memastikan apakah *hardware* yang dibuat sudah bekerja dengan baik sesuai dengan yang diinginkan. Uji coba *hardware* yang dilakukan antara lain:

   - Uji coba *base* robot
   - Uji coba arduino, *rotary encoder*, dan modul *bluetooth*

2. Uji coba *software*

   Uji coba ini dilakukan terhadap sistem kontrol yang digunakan seperti uji coba program mikrokontroler dan aplikasi processing. Uji coba ini berfungsi untuk melihat apakah formula yang digunakan sudah benar atau belum.

3. Uji Coba Alat

   Uji coba ini bertujuan untuk mengetahui hasil dari keseluruhan alat yang sudah dibuat. Uji coba yang dilakukan adalah:

   - Membaca nilai sensor *rotary encoder* dan mengolahnya menjadi nilai rpm, kecepatan sudut, kecepatan linier, dan jarak.
   - Mengkalkulasikan 3 nilai sensor menjadi informasi kecepatan, jarak, dan posisi.

- Membaca nilai sensor MPU 6050 dan mengolahnya menjadi informasi orientasi robot.
- Mengirim data dari mikrokontroler ke PC menggunakan modul *bluetooth*.

## 3.4 Laporan Akhir

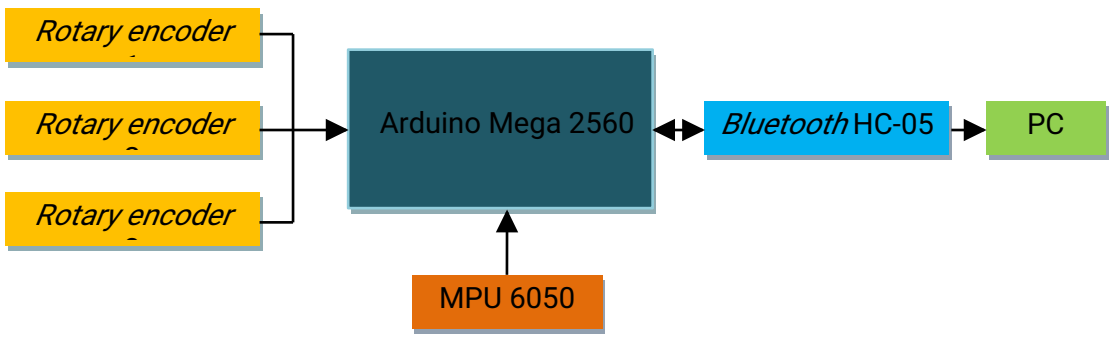
Pada langkah ini dilakukan penulisan laporan akhir sebagai syarat penilaian tugas akhir. Laporan dibuat sesuai pedoman yang sudah ditetapkan. Laporan ini juga bertujuan untuk merangkum keseluruhan tugas akhir yang dibuat, mulai dari latar belakang, rumusan masalah, landasan teori, metode penulisan, pembahasan, sampai dengan kesimpulan. Sehingga dapat mengetahui kekurangan, kelebihan, fungsi, dan dampak alat yang sudah dibuat. Juga bisa digunakan sebagai referensi bagi mahasiswa yang akan melaksanakan tugas akhir.

# BAB IV
# PEMBAHASAN

Pada bab ini akan membahas mengenai proses, metode, dan hasil tugas akhir dengan judul "Sistem Navigasi Pada Robot Sepak Bola Beroda KRSBI Menggunakan Metode *Odometry*".

## 4.1 Diagram Blok

Diagram blok Sistem Navigasi Pada Robot Sepak Bola Beroda KRSBI Menggunakan Metode *Odometry* ditunjukan pada gambar 4.1 berikut.

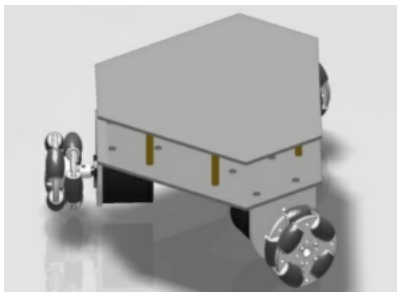



Gambar 4.1 Blok Diagram Alat

Keluaran *rotary encoder* berupa pulsa yang dapat membaca CW-CCW. Kemudian dikonversikan menggunakan mikrokontroler menjadi nilai rpm, kecepatan, dan jarak masing-masing roda. Lalu nilai 3 buah *rotary encoder* tadi dikalkulasikan menjadi kecepatan, jarak, dan sensor MPU 6050 digunakan untuk membaca orientasi robot. Informasi tersebut akan ditampilkan di PC dalam bentuk nilai dan gambar 2 dimensi pada aplikasi processing. Penerima dan pengirim data dari mikrokontroler ke PC menggunakan modul *bluetooth* HC-05.

## 4.2 Perancangan Kontruksi

Proses pembuatan desain dibuat secara bertahap, mulai dari *base*, dudukan sensor, *rotary encoder*, sampai dengan kinematik *omniwheels.* Adapun ukuran dari sensor yang akan dibuat yaitu 16x16x10 cm dan diameter roda 48 mm. Setelah part-part diatas dibuat, dilakukan proses *assembly* untuk menjadi satu kesatuan seperti pada gambar 4.2 dibawah ini.



Gambar 4.2 Desain *base* sensor

## 4.3 Pembuatan *Hardware* Sensor

Pembuatan *hardware* ini dilakukan dalam beberapa tahapan, antara lain :

1. *Base* Sensor

   Bahan material *base* yang digunakan adalah alumunium dengan ketebalan 2,5 mm. Selanjutnya dilakukan proses *marking,* pemotongan menggunakan grinda dan pengeboran dengan diameter 5 mm. *Base* dirancang agar roda bisa bergerak *holonomic* (bergerak kesegala arah), dengan beda sudut antar roda 120 derajat.

Gambar 4.3 *Base* Sensor

2. Dudukan *Rotary Encoder*

Bahan material dudukan yang digunakan adalah alumunium bentuk siku 90 derajat dengan ukuran 4x3 cm dan ketebalan 3 mm. Proses dilakukan mulai dari *marking,* bor dengan diameter 5, 3, 20 mm, dan pemotongan menggunakan gerinda. Dudukan *rotary encoder* dapat dilihat pada gambar 4.4 berikut.

Gambar 4.4 Dudukan *Rotary Encoder*

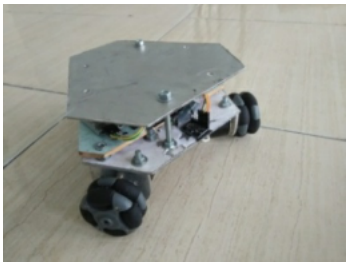
3. *Omniwheel*

   Roda omni ini dipilih karena dapat bergerak *holonomic* sehingga bisa menerapkan metode *odometry*, yang dimana dibutuhkan disain *three omni-directional*. Bahan roda yang dipilih adalah *ruber*, karena tidak licin dan cocok pada arena. Roda berdiameter 48 mm, yang dapat dilihat pada Gambar 4.5.

Gambar 4.5 *Omniwheel*

4. *Assembly* Konstruksi

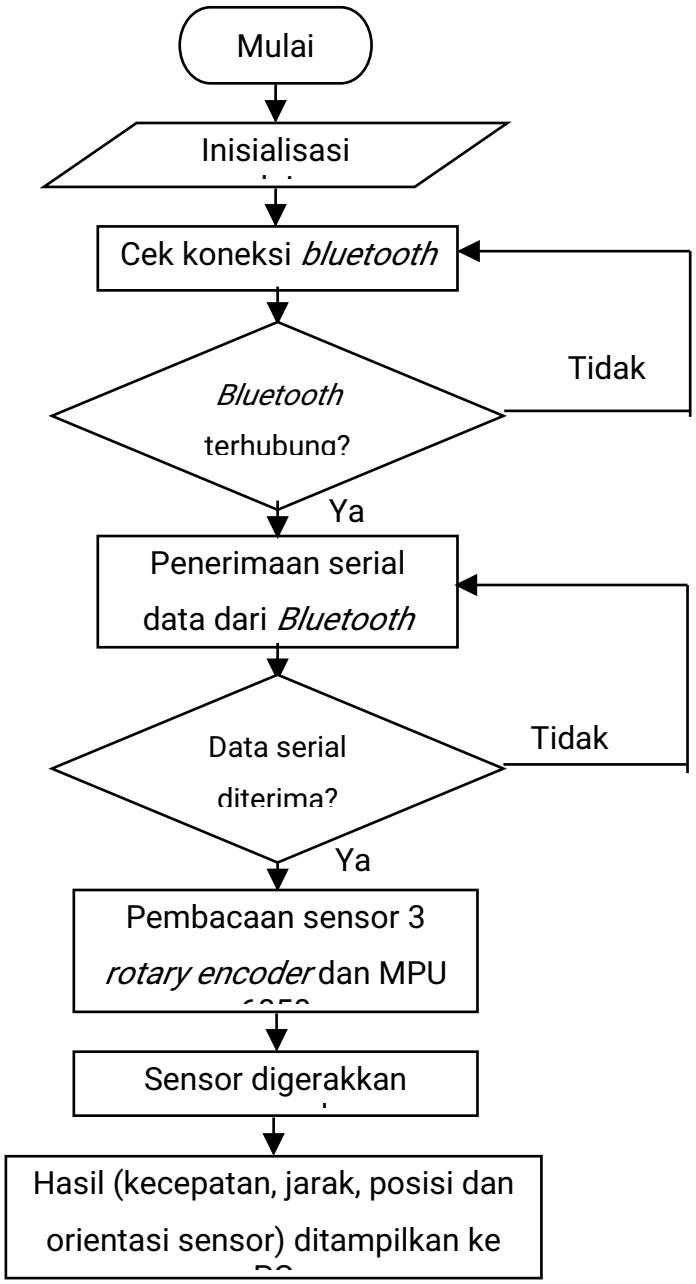
   Setelah part-part sebelumnya selesai dibuat, selanjutnya dilakukan *assembly* menggunakan komponen baut, mur, ring, dan *speser* untuk menyatukan keseluruhan bagian kontruksi. Berikut adalah kontruksi yang sudah di *assembly*.

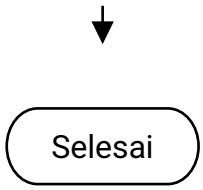


Gambar 4.6 Konstruksi Mekanik Sensor

## 4.4 Perancangan Sistem Navigasi

Perancangan program adalah untuk menentukan sistem sensor bekerja. Alur pemrograman dapat dilihat pada Gambar 4.7 berikut.






Gambar 4.7 Perancangan Sistem Navigasi

## 4.5 Pembuatan Program

Pembuatan program "Sistem Navigasi Pada Robot Sepak Bola Beroda Menggunakan Metode *Odometry*" dilakukan menggunakan aplikasi arduino pada laptop, yang meliputi:

1. Pemrograman sensor *rotary encoder* dengan fungsi intrupsi metode *rising* dan fungsi millis dengan *sampling time* pembacaan 200ms. Lalu dikonversikan untuk membaca nilai rpm, kecepatan linier, dan jarak, dengan menggunakan roda omni dengan diameter 48mm.
2. Pemrograman Sensor MPU 6050 untuk mendapatkan nilai sudut, sehingga mendapatkan orientasi robot.
3. Penggunaan metode *odometry* untuk menggabungkan 3 nilai sensor *rotary encoder* menjadi 1, yang kemudian menjadi informasi pergerakan robot seperti kecepatan, jarak, dan posisi.
4. Pemrograman modul *bluetooth* HC-05 untuk mengirim dan menerima data dari arduino ke PC.
5. Pemrograman sistem navigasi secara keseluruhan.

   Berikut pin yang digunakan pada Arduino Mega 2560.

Tabel 4.1 Pin Pada Arduino Mega 2560

| No. | Komponen | Pin Arduino |
|---|---|---|
| 1. | *Rotary Encoder* 1 | |
| | - VCC | 5 V |
| | - GND | GND |
| | - CH 1 | 2 Intrupt |
| | - CH 2 | 4 |
| 2. | *Rotary Encoder* 2 | |
| | - VCC | 5 V |
| | - GND | GND |

- CH 1 — 3 Intrupt
- CH 2 — 5

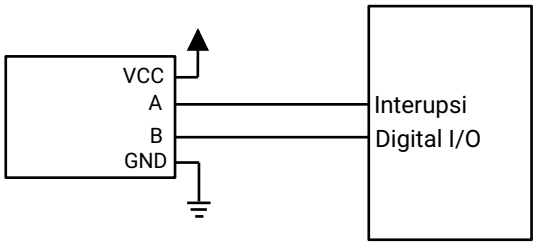
3. *Rotary Encoder* 3
   - VCC — 5 V
   - GND — GND
   - CH 1 — 18 Intrupt
   - CH 2 — 19 Intrupt
4. MPU 6050
   - VCC — 5 V
   - GND — GND
   - SCL — 21 SCL
   - SDA — 20 SDA
5. *Bluetoth* HC-05
   - VCC — 5 V
   - GND — GND
   - Rx — 1 Tx
   - Tx — 0 Rx

---

## 4.6 Pengujian *Rotary Encoder*

Pengujian *rotary encoder* ini bertujuan untuk mendapatkan cara dan metode yang tepat dalam mengkonversikan data luaran sensor yang berupa pulsa menjadi data atau informasi kecepatan, jarak, dan arah putaran roda. Pembacaan sensor *rotary encoder* dilakukan menggunakan metode interupsi dengan mode *rising edge*, dimana salah satu kanal dari

dua kanal yang ada pada sensor dihubungkan ke pin interupsi eksternal pada Arduino dan yang satunya lagi ke pin digital I/O. Sistem pewaktu yang digunakan adalah fungsi millis(), dengan *sampling time* sebesar 200ms. Spesifikasi *rotary encoder* yang digunakan adalah 400 ppr(*pulse per revolusion). Interfacing* antara *rotary encoder* dan arduino dapat dilihat pada Gambar 4.8 berikut.

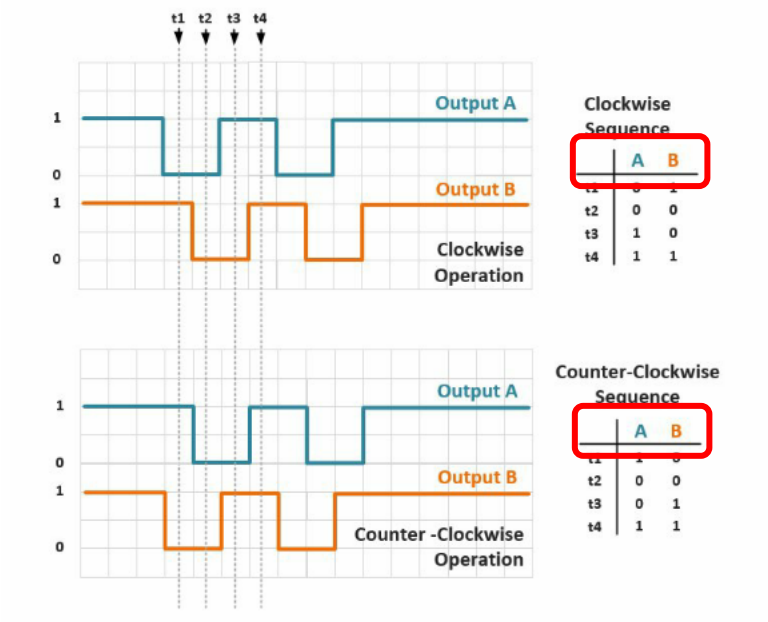



Gambar 4. 8 *Interfacing* Antara *Rotary Encoder* dan Arduino

## 4.7 Deteksi CW – CCW

Sensor *rotary encoder* pada dasarnya hanya mengeluarkan data digital atau pulsa saja. Pulsa tersebut dikeluarkan oleh dua buah kanal yang diberi nama kanal A dan kanal B. Perbedaan dari kedua kanal ini adalah pada *phase* pulsa yang berjarak ±90°. Detil bentuk pulsa dari kedua kanal tersebut dapat dilihat pada Gambar 4.9. Berdasarkan spesifikasi dari sensor ini, tidak ada indikator khusus yang menyatakan tentang arah putaran sensor, sehingga diperlukan pengolahan data pulsa tersebut lebih lanjut untuk mendeteksi arah putaran sensor.

Pada proyek akhir ini, pendeteksian arah putaran sensor (*cw-ccw)* dilakukan dengan algoritma sederhana yang diprogram pada Arduino. Konsep sederhananya adalah dengan memeriksa status atau nilai pada pin digital I/O ketika interupsi terjadi. Apabila pin digital I/O berlogika nol, maka berdasarkan Gambar 4.9 arah putaran sensor pada saat itu adalah searah jarum jam atau *clockwise (CW).* Sebaliknya, ketika interupsi terjadi dan nilai pada pin digital I/O berlogika satu, maka berdasarkan Gambar 4.9 arah putaran sensor pada saat itu adalah berlawanan jarum jam atau *counter clockwise (CCW)*.

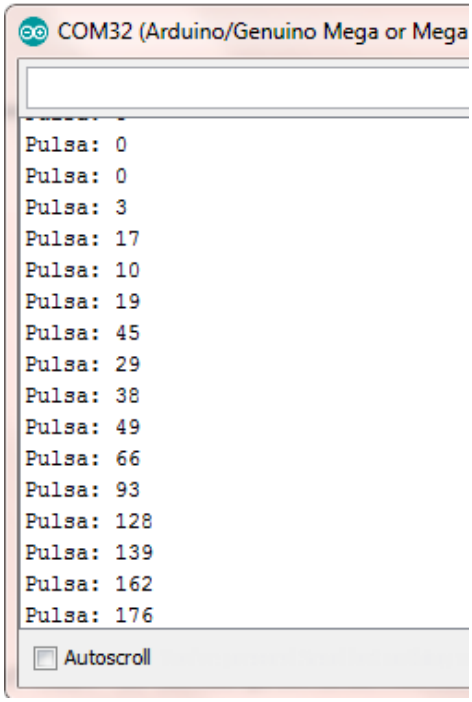



Gambar 4. 9 *Output Rotary Encoder*

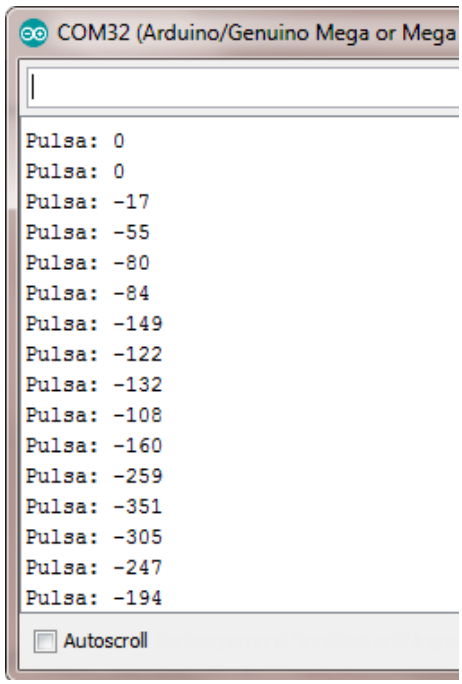
Algoritma yang sudah dijelaskan diatas tentang pendeteksian arah putaran sensor selanjutnya diprogram di Arduino. Berikut adalah *list* programnya.

```
const int ch_A1=2;  //pin interupsi
const int ch_B1=5;  //pin digital I/O
void setup()
{
  pinMode(ch_A1,INPUT_PULLUP); //input, internal pull up
  pinMode(ch_B1,INPUT_PULLUP); //input, internal pull up
  attachInterrupt(digitalPinToInterrupt(2),bacaencoder,
  RISING);                  //interupsi mode rising
}
void bacaencoder1()          // subrutin interupsi
{
  if(digitalRead(ch_B1) ){encoder1--;}  //CCW
  else {encoder1++;}                    //CW
}
```

Setelah melakukan pemrograman, langkah selanjutnya adalah melakukan pengujian sensor. Pengujian dilakukan dua kali, dengan memutar poros sensor searah jarum jam dan berlawanan arah jarum jam. Berikut hasil pengujiannya.




Gambar 4.10 Hasil Nilai Sensor *CW* Bernilai Positif




Gambar 4.11 Hasil Nilai Sensor *CCW* Diberi tanda Negatif

## 4.8 Pembacaan Jarak Dengan Roda

Pengujian ini dilakukan dengan memasang roda omni diameter 48 mm pada poros sensor. Langkah pertama adalah mengkonversi nilai pulsa

keluaran *rotary encoder* menjadi informasi kecepatan putaran roda. Untuk mendapat nilai kecepatan roda menggunakan perhitungan sebagai berikut.

- Jumlah pulsa dalam 1 putaran penuh adalah 400 ppr.
- Mencari jumlah putaran:

$$\text{Putaran} = \text{pulsa/ppr}$$
$$\text{Putaran} = \text{pulsa}/400$$

- Maka bisa mendapatkan nilai rpm:

$$\text{Rpm} = 60 \times \text{putaran}$$

- Dan nilai kecepatan sudut:

$$\omega = 2\pi/60 \times \text{rpm}$$

Langkah kedua adalah mengkonversi kecepatan sudut menjadi kecepatan linier, lalu bisa mendapatkan nilai jarak. Untuk mendapat nilai jarak, diperoleh dari perkalian antara kecepatan roda dengan waktu. Namun, pada kondisi sesungguhnya, kecepatan putaran roda bisa berubah-ubah yang diakibatkan oleh berbagai faktor seperti gesekan permukaan, perubahan dinamik robot, atau lainnya sehingga perhitungan jarak harus dilakukan berdasarkan teknik *sampling time*. Pada proyek akhir ini, *sampling time* yang digunakan adalah 200ms. Dengan interval waktu 200ms tersebut, dapat diartikan bahwa perhitungan nilai jarak dilakukan setiap 200ms sekali, lalu nilai-nilai tersebut diakumulasi menjadi jarak total dengan menggunakan rumus sebagai berikut

- Kecepatan linier:

$$V = r/1000 \times \omega$$
$$V = 24/1000 \times \omega$$

- Jarak roda:

$$S = \text{Sampling Time} \times V$$
$$S = 0.2 \times V$$

- Jadi jarak total roda:

$$St \mathrel{+}= s$$

Keterangan:

ω = Kecepatan Sudut

V = Kecepatan Linier

S = Jarak

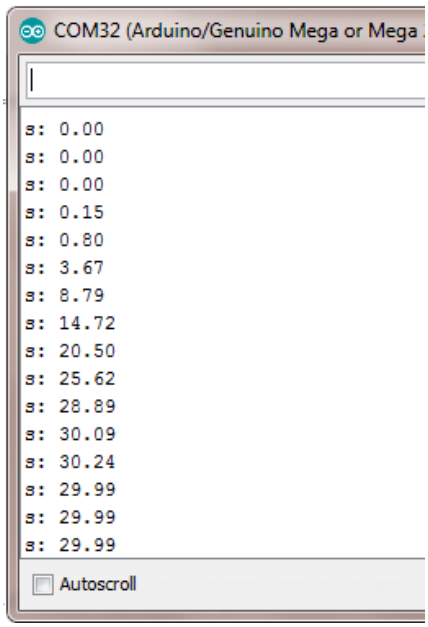
St = Jarak Total

Setelah mendapat rumus tersebut, rumus ditulis dalam program pada Arduino sebagai berikut.

```
void loop() {
if (millis()-waktu>=200) //sampling time
  {
   putaran=(float)encoder/400;
   rpm=(60/0.15)*putaran; //Membaca nilai RPM


   w=10.466667*rpm;        // Membaca kecepatan sudut
   v=0.024*w;          //Membaca kecepatan linier
   s=0.20*v;           //Membaca jarak per 200ms
   st=st1+s;           //Membaca jarak total
   encoder1=0;                 //Untuk mereset encoder
}
```

Selanjutnya dilakukan pengujian dengan menggerakkan roda dengan jarak 30cm. Berikut hasil pengujiannya.




Gambar 4.12 Data Hasil Nilai Jarak Roda Robot

Tabel 4.2 Data Hasil Jarak Roda Robot

| Percobaan ke- | Hasil(cm) | Eror(cm) |
|---|---|---|
| 1 | 30.14 | 0.14 |
| 2 | 30.09 | 0.09 |
| 3 | 30.29 | 0.29 |
| 4 | 29.99 | 0.01 |
| 5 | 29.94 | 0.06 |
| 6 | 29.89 | 0.11 |
| 7 | 29.99 | 0.01 |
| 8 | 29.99 | 0.01 |
| 9 | 30.19 | 0.19 |
| 10 | 29.94 | 0.06 |
| Rata-rata eror | | 0.097 |

## 4.9 Konfigurasi Awal Modul *Bluetooth* HC 05

Langkah awal dalam pengujian modul *bluetooth* HC-05 yaitu *bluetooth* di *set up* terlebih dahulu menggunakan perintah AT *Command* lewat *Serial Monitor* Arduino. Langkah yang dilakukun yaitu hubungkan pin *bluetooth* seperti pada pengkoneksian yang dibuat, hubungkan pin *key* ke VCC/3V, tekan tombol yang ada pada *bluetooth* kemudian pasang catu daya tunggu selama 5 detik hingga LED pada *bluetooth* HC-05 5 detik hidup 5 detik mati. Selanjutnya masuk *Serial Monitor* pada Arduino, perintah yang ditulis yaitu:

- AT digunakan untuk melakukan test *Bluetooth*. Untuk mengetahui jika *Bluetooth* dapat berfungsi atau tidak.
- AT+RESET digunakan untuk mengatur ulang *Bluetooth*.
- AT+VERSION digunakan untuk melihat versi *Bluetooth*.
- AT+NAME digunakan untuk melihat nama modul *Bluetooth.*
- AT+NAME=<Param> digunakan untuk mengganti name modul *Bluetooth.* (<Param> merupakan name baru yang ingin kita ganti)
- AT+PSWD digunakan untuk mengetahui nama *Bluetooth.*
- AT+PSWD=<Param> digunakan untuk mengganti nama *Bluetooth.*

Berikut adalah *list* program Arduino untuk konfigurasi awal *bluetooth* HC-05.

```
#include<SoftwareSerial.h>
SoftwareSerial (belutut(10, 11);  //RX | TX
Void setup() {
   Serial.begin(9600);
   Belutut.begin(9600);  //Baudrate
Void loop() {
//Membaca dari HC05 dari serial monitor
   If (belutut.available())
   Serial.write(write(blutut.read());
//Membaca dari serial monitor untuk dikirim ke HC 05
   If (Serial.available())
   Belutut.write(Serial.read()):
}
```

Setelah melakukan konfigurasi awal, dilakukan pengujian untuk menggetahui jarak maksimal koneksi *Bluetooth.* Hasilnya dapat dilihat pada Tabel 4.3 berikut.

Tabel 4.3 Hasil Pengujian Jarak Koneksi *Bluetooth* HC-05

| Jarak | Data Diterima |
|---|---|
| 1 Meter | Ya |
| 2 Meter | Ya |
| 3 Meter | Ya |

| | |
|---|---|
| 4 Meter | Ya |
| 5 Meter | Ya |
| 6 Meter | Ya |
| 7 Meter | Ya |
| 8 Meter | Ya |
| 9 Meter | Ya |
| 10 Meter | Ya |
| 11 Meter | Ya |
| 12 Meter | Tidak |

## 4.10 Pengujian Gerak Rotasi

Untuk mendapatkan hasil gerak rotasi, digunakan sensor MPU 6050. Pada dasarnya *output* sensor ini adalah pulsa yang berubah Jika robot rotasi searah jarum jam, maka data yang dierima adalah positif, juga sebaliknya. Satu putaran penuh dibatai 360º, dengan menggunakan fungsi "%" pada pemrograman.

Gambar 4. 13 Sensor MPU 6050 (9)

Berikut adalah *list* program yang digunakan untuk pengujian pembacaan data sensor MPU 6050.

```
int stepp,rotate,rotat;
#include <MPU6050_tockn.h>
#include <Wire.h>
MPU6050 mpu6050(Wire); //Library
void setup() {
  Serial.begin(9600);
  Wire.begin();
  mpu6050.begin();
  mpu6050.calcGyroOffsets(true);
}
void loop()
{
  mpu6050.update();
  rotate=mpu6050.getAngleZ();
  rotat=rotate%360; //Untuk membatasi pulsa satu
                                    putaran penuh menjadi 360
}
```

Berikut adalah hasil pembacaan data sensor MPU 6050:


Gambar 4. 14 Hasil Pembacaan Data Sensor MPU 6050

Percobaan sensor MPU 6050 dilakukan dengan menggerakan rotasi robot. Untuk mengetahui nilai rotasi, digunakan aplikasi kompas pada *smartphone* sebagai pembanding hasil.

- Pergerakan Rotasi

Tabel 4.11 Data Hasil Pergerakan Rotasi

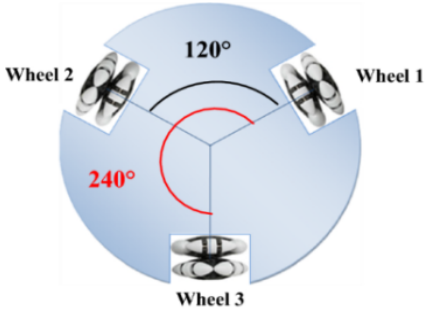
| Rotasi | Hasil |
|---|---|
| 90 º | 90 º |

| | |
|---|---|
| -90 º | -90 º |
| 180 º | 180 º |
| -180 | -180 º |

Tabel diatas menunjukan hasil pembacaan nilai rotasi robot pada sensor MPU 6050. Hasil pembacaan aplikasi kompas pada *smartphone* sama dengan hasil pembacaan sensor. Artinya sensor bekerja dengan baik.

## 4.11 Pembuatan Sistem Navigasi Menggunakan Metode Odometry

Setelah selesai melalui proses pembuatan *hardware,* dan pemrograman, selanjutnya dilakukan pengujian "Sistem Navigasi Menggunakan Metode *Odometry"*. Langkah pertama yang dilakukan adalah mengetahui ukuran dan disain *three omni-directional* robot yang digunakan, hal ini untuk menentukan rumus atau kalkulasi yang akan digunakan. Langkah kedua, membuat formulasi untuk membaca kecepatan linier masing-masing roda untuk mengetahui rasio perbandingan putaran roda jika robot bergerak. Ketiga, Menggabungkan ketiga nilai sensor *rotary encoder* tersebut menjadi informasi kecepatan dan jarak robot. Kecepatan yang diketahui adalah kecepatan pada sumbu x dan sumbu y, dengan mengetahui kecepatan robot, maka bisa mendapatkan informasi pergerakan atau translasi robot. Sedangkan untuk mengetahui orientasi robot digunakan sensor MPU-6050, sensor tersebut diolah untuk membaca orientasi robot dengan nilai maksimum satu putaran penuh adalah 360.




Gambar 4.15 Kinematika Sensor *Rotary Encoder*

## 4.12 Pengujian Kecepatan Robot

Untuk mendapatkan kecepatan robot harus dilakukan pengkalkulasian 3 nilai sensor menjadi 1, dengan perbedaan sudut antar roda 120º. Maka roda 1 dan roda 2 memiliki sudut θ sebesar 60º, sehingga didapatkan persamaan:

$$Vx = (v3 - v1\ cos30 - v2\ sin30) \times 3/2$$

$$Vy = (v1/cos30 - v2/cos30)\ /2$$

Keterangan:

Vx = Kecepatan sumbu X

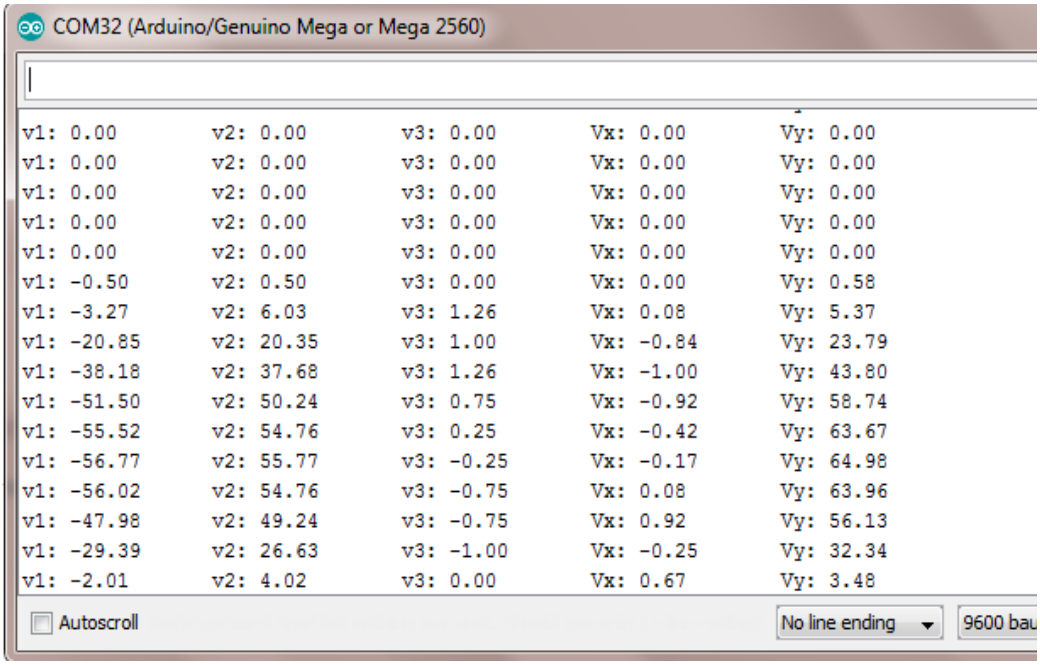
Vy = Kecepatan sumbu Y

v1 = Kecepatan 1

v2 = Kecepatan 2

v3 = Kecepatan 3

Berikut hasil pengujiannya:




Gambar 4.16 Hasil Pengujian Kecepatan Robot

Berdasarkan hasil pengujian diatas didapat pergerakan robot dengan rasio perbandingan kecepatan seperti tabel 4.4.

Tabel 4.4 Rasio Perbandingan Roda

| **No.** | **Rasio perbandingan roda** | | | **Pergerakan** |
|---|---|---|---|---|
| | Roda 1 | Roda 2 | Roda 3 | |
| 1. | -1 | 1 | 0 | Maju |
| 2. | 1 | -1 | 0 | Mundur |
| 3. | ½ | ½ | -1 | Geser kanan |
| 4. | -½ | -½ | 1 | Geser kiri |
| 5. | -½ | 1 | -1 | Serong depan kanan |
| 6. | 1 | ½ | 1 | Serong depan kiri |
| 7. | 1 | -½ | -1 | Serong belakang kanan |
| 8. | ½ | -1 | 1 | Serong belakang kiri |

| | | | | |
|---|---|---|---|---|
| 9. | 1 | 1 | 1 | Rotasi kanan |
| 10. | -1 | -1 | -1 | Rotasi kiri |

Diatas adalah nilai rasio perbandingan roda. Jika nilainya minus (-) artinya **satu putaran penuh roda berlawanan arah jarum jam dan sebaliknya.**

### 4.13 Jarak

Untuk mendapatkan nilai jarak, digunakan metode *sampling time*. Dengan rumus sebagai berikut:

$$Sx = Vx \times 0.2$$

$$Sy = Vy \times 0.2$$

$$Sx\ total \mathrel{+}= Sx$$

$$Sy\ total \mathrel{+}= Sy$$

Keterangan:

Sx = Jarak sumbu x

Vx = Kecepatan sumbu x

Sx total = Jarak total sumbu x

Sx total = Jarak total sumbu x

Pada pengujian ini robot digerakan maju, mundur, geser kiri, geser kanan, serong kiri dan serong kanan. Berikut adalah hasil dari pergerakan maju robot:

- Pergerakan Maju

Tabel 4.5 Data Hasil Jarak Pergerakan Maju

| Jarak(cm) | | Hasil(cm) | | Error(cm) | |
|---|---|---|---|---|---|
| X | Y | Sx | Sy | Sx | Sy |
| 0 | 30 | -0.56 | 29.80 | 0.56 | 0.20 |
| 0 | 30 | -0.48 | 29.88 | 0.48 | 0.12 |

| | | | | | |
|---|---|---|---|---|---|
| 0 | 30 | 0.08 | 29.80 | 0.08 | 0.20 |
| 0 | 30 | 0.10 | 30.64 | 0.10 | 0.64 |
| 0 | 30 | -0.07 | 30.12 | 0.07 | 0.12 |
| 0 | 30 | 0.33 | 30.21 | 0.33 | 0.21 |
| 0 | 30 | 0.20 | 29.94 | 0.20 | 0.06 |
| 0 | 30 | -0.51 | 30.00 | 0.51 | 0 |
| 0 | 30 | -0.24 | 30.18 | 0.24 | 0.18 |
| 0 | 30 | -0.19 | 30.15 | 0.19 | 0.15 |
| | Rata-rata eror | | | 0.276 | 0.188 |

Robot bergerak maju dengan jarak 30cm, didapat nilai rata-rata *error* pada jarak sumbu x sebesar 0.276cm dan pada sumbu y sebesar 0.188cm.

- Pergerakan Mundur

Tabel 4.6 Data Hasil Jarak Pergerakan Mundur

| **Jarak(cm)** | | **Hasil(cm)** | | **Error(cm)** | |
|---|---|---|---|---|---|
| **X** | **Y** | **Sx** | **Sy** | **Sx** | **Sy** |
| 0 | -20 | 0.22 | -19.99 | 0.22 | 0.01 |
| 0 | -20 | -0.12 | -19.98 | 0.12 | 0.02 |
| 0 | -20 | 0.09 | -20.37 | 0.09 | 0.37 |
| 0 | -20 | 0.09 | -20.21 | 0.09 | 0.21 |
| 0 | -20 | 0.51 | -20.00 | 0.51 | 0 |
| 0 | -20 | 0.32 | -20.03 | 0.32 | 0.03 |
| 0 | -20 | -0.19 | -19.91 | 0.19 | 0.09 |

| | | | | | |
|---|---|---|---|---|---|
| 0 | -20 | -0.60 | -20.66 | 0.60 | 0.66 |
| 0 | -20 | -0.10 | -20.16 | 0.10 | 0.16 |
| 0 | -20 | 0.33 | 19.87 | 0.33 | 0.13 |
| | | Rata-rata eror | | 0.257 | 0.168 |

Robot bergerak mundur dengan jarak 20cm, didapat nilai rata-rata *error* pada jarak sumbu x sebesar 0.257cm dan pada sumbu y sebesar 0.168cm.

- Pergerakan Geser Kiri

Tabel 4.7 Data Hasil Jarak Pergerakan Geser Kiri

| **Jarak(cm)** | | **Hasil(cm)** | | **Error(cm)** | |
|---|---|---|---|---|---|
| **X** | **Y** | **Sx** | **Sy** | **Sx** | **Sy** |
| -30 | 0 | -30.06 | 0.44 | 0.06 | 0.44 |
| -30 | 0 | -29.84 | -0.03 | 0.16 | 0.03 |
| -30 | 0 | -30.11 | -0.14 | 0.11 | 0.14 |
| -30 | 0 | -30.24 | 0.29 | 0.24 | 0.29 |
| -30 | 0 | -30.14 | 0.23 | 0.14 | 0.23 |
| -30 | 0 | -30.01 | 0.19 | 0.01 | 0.19 |
| -30 | 0 | -29.90 | 0.29 | 0.10 | 0.29 |
| -30 | 0 | -30.07 | -0.18 | 0.07 | 0.18 |
| -30 | 0 | -29.90 | 0.13 | 0.10 | 0.13 |
| -30 | 0 | -30.11 | 0.27 | 0.11 | 0.27 |
| | | Rata-rata eror | | 0.11 | 0.219 |

Robot bergerak geser kiri dengan jarak 30cm, didapat nilai rata-rata *error* pada jarak sumbu x sebesar 0.11cm dan pada sumbu y sebesar 0.219cm.

- Pergerakan Geser Kanan

Tabel 4.8 Data Hasil Jarak Pergerakan Geser Kanan

| Jarak(cm) | | Hasil(cm) | | Error(cm) | |
|---|---|---|---|---|---|
| X | Y | Sx | Sy | Sx | Sy |
| 10 | 0 | 9.89 | -0.09 | 0.11 | 0.09 |
| 10 | 0 | 9.99 | -0.19 | 0.31 | 0.19 |
| 10 | 0 | 10.13 | 0.24 | 0.13 | 0.24 |
| 10 | 0 | 10.21 | 0.41 | 0.21 | 0.41 |
| 10 | 0 | 10.00 | -0.32 | 0 | 0.32 |
| 10 | 0 | 10.16 | -0.01 | 0.16 | 0.01 |
| 10 | 0 | 9.87 | -0.21 | 0.23 | 0.21 |
| 10 | 0 | 9.91 | -0.37 | 0.09 | 0.37 |
| 10 | 0 | 10.01 | 0.06 | 0.21 | 0.06 |
| 10 | 0 | 10.32 | 0.14 | 0.32 | 0.14 |
| | | Rata-rata eror | | 0.177 | 0.204 |

Robot bergerak geser kanan dengan jarak 10cm, didapat nilai rata-rata *error* pada jarak sumbu x sebesar 0.177cm dan pada sumbu y sebesar 0.204cm.

- Pergerakan Serong Kanan

Tabel 4.9 Data Hasil Jarak Pergerakan Serong Kanan

| Jarak(cm) | | Hasil(cm) | | Error(cm) | |
|---|---|---|---|---|---|
| X | Y | Sx | Sy | Sx | Sy |

| | | | | | |
|---|---|---|---|---|---|
| 30 | 30 | 30.73 | 30.26 | 0.73 | 0.76 |
| 30 | 30 | 30.86 | 30.50 | 0.86 | 0.50 |
| 30 | 30 | 30.99 | 30.00 | 0.99 | 0 |
| 30 | 30 | 30.70 | 29.27 | 0.70 | 0.73 |
| 30 | 30 | 30.22 | 30.68 | 0.78 | 0.68 |
| 30 | 30 | 30.54 | 30.23 | 0.46 | 0.77 |
| 30 | 30 | 30.94 | 30.53 | 0.94 | 0.53 |
| 30 | 30 | 30.55 | 30.44 | 0.55 | 0.44 |
| 30 | 30 | 30.99 | 30.41 | 0.99 | 0.41 |
| 30 | 30 | 29.69 | 29.06 | 0.31 | 0.94 |
| | Rata-rata eror | | | 0.731 | 0.576 |

Robot bergerak serong kanan dengan jarak x = 30cm dan y = 30cm, didapat nilai rata-rata *error* pada jarak sumbu x sebesar 0.731cm dan pada sumbu y sebesar 0.576cm.

- Pergerakan Serong Kiri

Tabel 4.10 Data Hasil Jarak Pergerakan Serong Kiri

| **Jarak(cm)** | | **Hasil(cm)** | | **Error(cm)** | |
|---|---|---|---|---|---|
| **X** | **Y** | **Sx** | **Sy** | **Sx** | **Sy** |
| -20 | -20 | -19.19 | -19.01 | 0.91 | 0.99 |
| -20 | -20 | -19.86 | -20.62 | 0.44 | 0.38 |
| -20 | -20 | -20.21 | -20.51 | 0.79 | 0.51 |
| -20 | -20 | -20.38 | -20.77 | 0.38 | 0.77 |

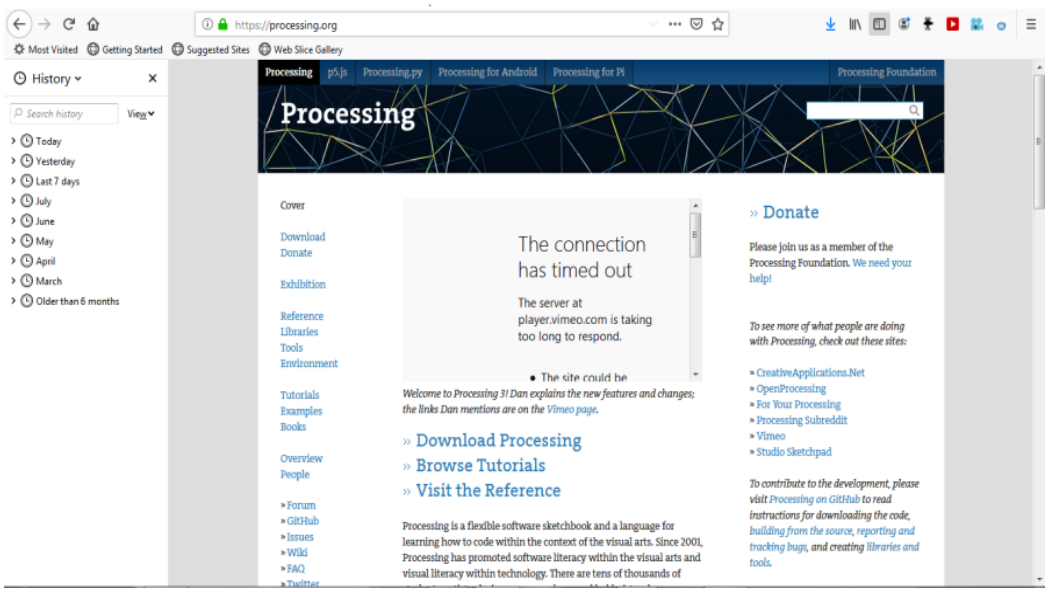
| | | | | | |
|---|---|---|---|---|---|
| -20 | -20 | -19.00 | -20.44 | 1.00 | 0.44 |
| -20 | -20 | -19.69 | -19.27 | 0.31 | 0.72 |
| -20 | -20 | -19.23 | -19.93 | 0.77 | 0.47 |
| -20 | -20 | -20.19 | -20.53 | 0.81 | 0.53 |
| -20 | -20 | -20.29 | -20.78 | 0.29 | 0.78 |
| -20 | -20 | -20.71 | -20.96 | 0.641 | 0.96 |
| | Rata-rata eror | | | 0.641 | 0.655 |

Robot bergerak serong kiri dengan jarak x = 20cm dan y = 20cm, didapat nilai rata-rata *error* pada jarak sumbu x sebesar 0.641cm dan pada sumbu y sebesar 0.655cm.

## 4.14 Pembuatan Aplikasi *Processing*

*Software* ini dibuat untuk membaca dan menampilkan nilai sensor dengan komunikasi menggunakan modul *bluetooth* HC-05. Data yang diterima diproses melalui pencocokan data *base* yang ada. Pembuatan aplikasi ini dilakukan langkah-langkah sebagai berikut:

1. Buka website processing.org, lalu download sesuai spesifikasi PC, dan install



Gambar 4.17 Download aplikasi *processing*

2. Langkah pertama dalam pemrograman adalah mendeklarasikan tipe data yang ingin digunakan dan membuat set up untuk ukuran tampilan, fungsi tombol, jenis font, menampilkan foto dan COM yang digunakan PC berkomunikasi dengan *bluetooth.*

```
void setup()
{
 background(200);
 size(1300,650);
 port = new Serial(this, "COM32", 9600);
 cp5 = new ControlP5(this);
 font1 = createFont("Arial",20);
 font2 = createFont("Times New Roman",30);

 cp5.addButton("START")
 .setPosition(410,500)
 .setSize(80,40)
 .setFont(font1)
 ;
  cp5.addButton("SIMPAN")
 .setPosition(410,550)
 .setSize(90,40)
 .setFont(font1)
 ;
}
```

3. Selanjutnya membuat gambar yang menampilkan hasil pergerakan robot. Letak posisi, ukuran, dan karakter disisain sesuai keinginan.

```
void draw()
{
 fill(#040824); // Memberi warna
 rect(0, 0, 1300, 60); // Membuat kotak
 fill(0);
 for (int i = 727; i < 1277; i += 50) {
 line(i, 120, i, 620); // Membau grid
```

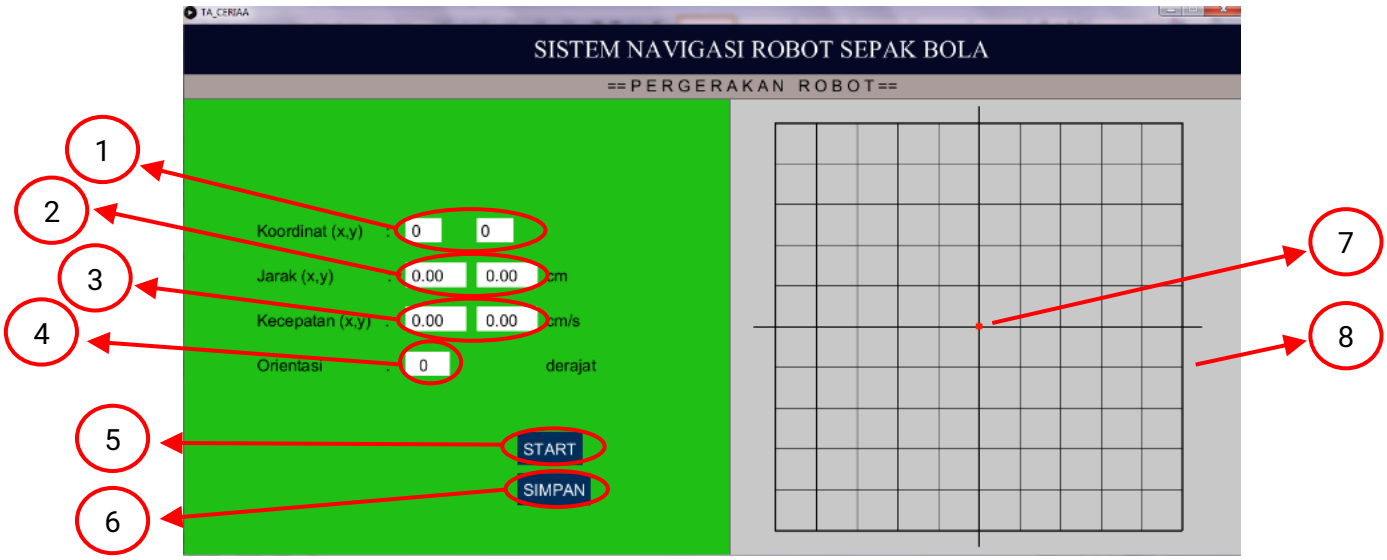
```
}
  fill(#FF1C03);
  noStroke();
  smooth();
  ellipse(977+koorX,370+koorY,6,6); //Membuat dot yang
  bergerak mengikuti pergerakan robot
  text("Koordinat (x,y)  :",90,260); // Membuat text
  text(nf(c,1,0),280,260);
  text(nf(d,1,0),365,260); // Menampilkan nilai yang
  dikirim arduino
}
```

4. Selanjutnya membuat fungsi baca data dari arduino. Data yang dibaca *processing* adalah data yang berurut dari arduino. Setiap data di arduino harus dipisahkan dengan karakter ";" agar dapat dibaca di *processing*. Juga membuat fungsi tombol "*START*" dan "*STOP*" berfungsi sebagai mulai jalannya pembacaan data di arduino dan *reset*. Jika tombol "*START*", maka arduino akan membaca nilai sensor, jika tombol "*STOP*" ditekan, maka arduino akan mereset nilai sensor dan menyimpan data yang sudah dibaca ke MS Excel.

```
if (port.available()>0)
  {
    val = port.readStringUntil('\n');
    if (val !=null)
  {
    String [] list = split(val, ';');  // Untuk
    menerima data masuk dari arduino
      vx = float(list[0]);
      vy = float(list[1]);
      sx = float(list[2]);
      sy = float(list[3]);
      c = float(list[4]);
      d = float(list[5]);
      koorX = float(list[6]);
      koorY = float(list[7]);
      sudut = float(list[8]); //Data yang diterima
      berurutan sesuai dengan yang di arduino
```

```
}
void START()
{
  port.write('y'); //Sebagai tombol start dan pemberi
  intruksi di arduino
}
void SIMPAN()
{
  port.write('a');
  saveTable(table, filename); // Menyimpan file
  exit(); //Close otomatis
}
```

5. Klik "*run*" untuk untuk memulai aplikasi




Gambar 4.18 Tampilan awal aplikasi

Keterangan :

1. Nilai titik koordinat pada sumbu X(kiri) dan sumbu Y(kanan).
2. Nilai jarak tempuh pada sumbu X(kiri) dan sumbu Y(kanan).
3. Nilai kecepatan sumbu X(kiri) dan sumbu Y(kanan).
4. Nilai orientasi robot.
5. Untuk memulai membaca pergerakan robot.

6. Untuk menyimpan data pergerakan robot dan sekaligus menutup otomatis aplikasi.

7. Titik berwarna merah, titik ini akan bertambah jumlahnya mengikuti gerak robot.

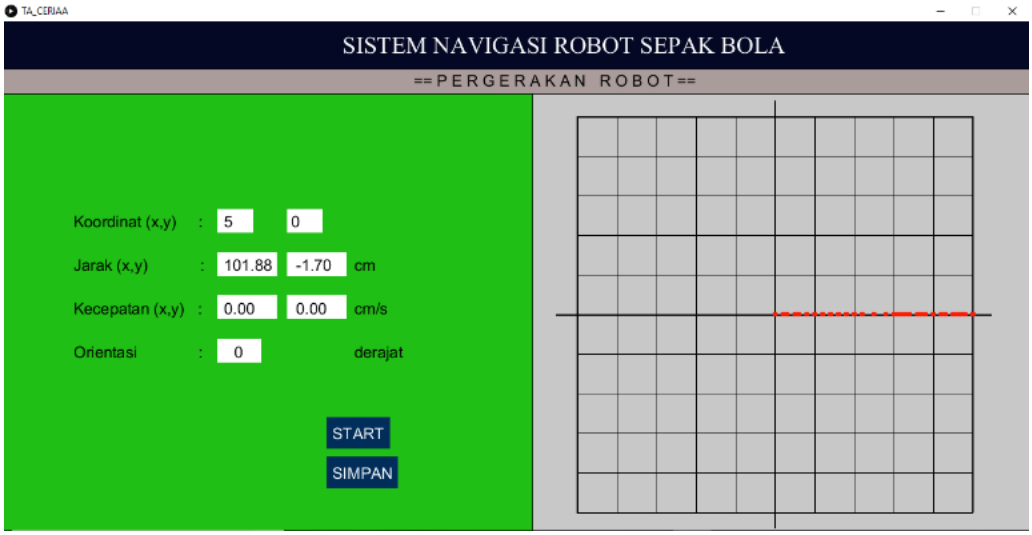
8. *Grid,* berbentuk persegi, sebagai representasi kondisi di lapangan dengan panjang sisi adalah 20cm. Garis vertikal memberi nilai sumbu Y dan garis horizontal memberi nilai sumbu X. Masing-masing garis bernilai satu koordinat.

## 4.15 Pengujian Aplikasi Navigasi Robot

Pengujian dilakukan menggunakan laptop 64 bit dengan jarak maksimal 11 meter. Pengujian aplikasi dilakukan sebanyak 3 kali, untuk melihat pergerakan robot. Adapun langkah-langkah yang dilakukan dalam pengujiannya sebagai berikut:

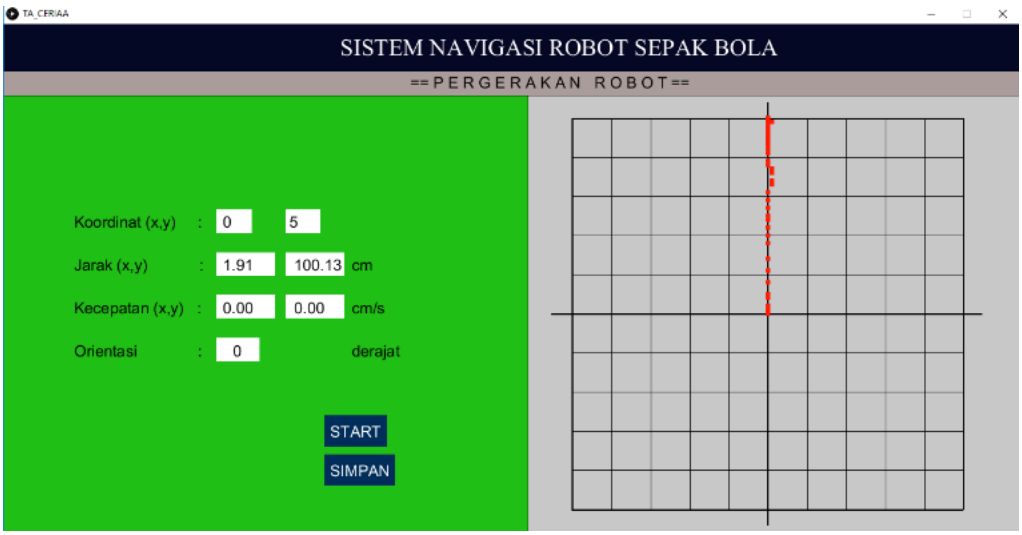
1. Sambungkan *bluetooth* pada laptop.
2. Buka aplikasi Navigasi Robot, lalu klik "*run*" untuk memulai.
3. Lalu klik "*start*" dan gerakan robot sesuai target, pergerakan robot bisa dilihat.
4. Setelah selesai, klik "simpan" untuk menyimpan data pergerakan dan aplikasi akan tertutup otomatis.

   Berikut adalah hasi pengujian aplikasi sistem navigasi:



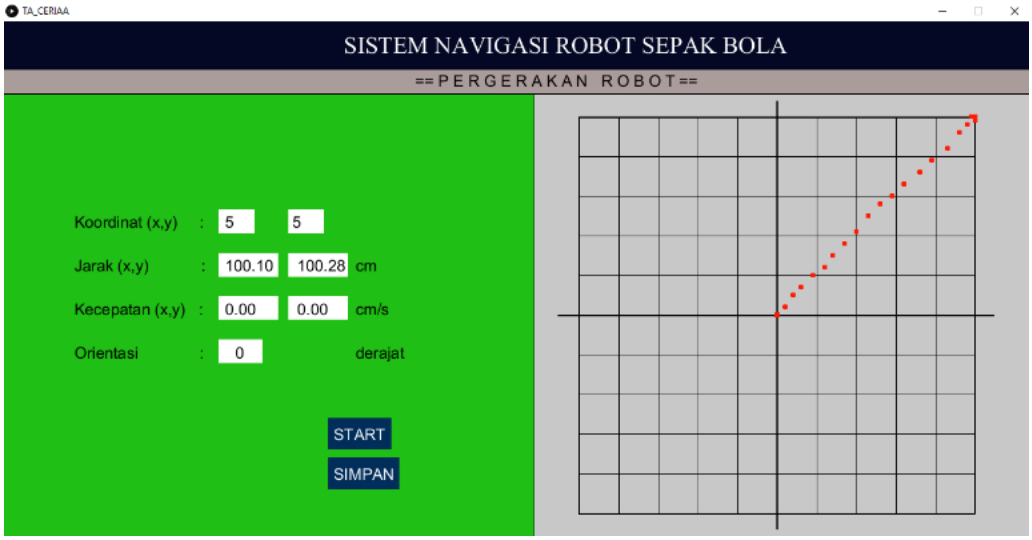

Gambar 4.19 Gerak Geser Kanan

Sensor digerakan manual menggunakan tangan kearah geser kanan dengan jarak 100cm atau koordinat(x=5, y=0). *Error* pada sumbu x=1.7cm dan sumbu y=1.88cm.



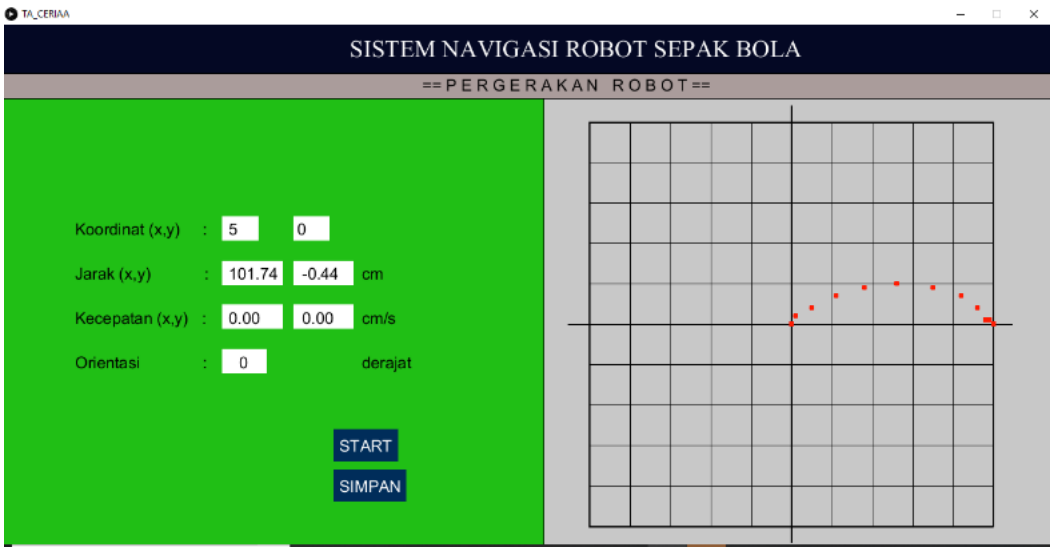

Gambar 4.20 Gerak Maju

Sensor digerakan manual menggunakan tangan kearah maju dengan jarak 100cm atau koordinat(x=0, y=5). *Error* pada sumbu x=1.91cm dan sumbu y=0.13cm.




Gambar 4.21 Gerak Serong

Sensor digerakan manual menggunakan tangan kearah serong kanan dengan koordinat(x=5, y=5). *Error* pada sumbu x=0.1cm dan sumbu y=0.28cm.




Gambar 4.22 Gerak Revolusi

Sensor digerakan manual menggunakan tangan kekoordinat(x=5, y=0) dengan gerak revolusi. *Error* pada sumbu x=1.74cm dan sumbu y=0.44cm. Dari hasil pengujian dapat dilihat pergerakan *real* robot, sehingga bisa diketahui erornya. Pengujian dilakukan dengan cara manual mempengaruhi hasil pengujian, dimana jika menggerakan robot tidak sesuai *track* target, maka hasil pergerakan akan menunjukan *error*.

# BAB V
# PENUTUP

## 5.1 Kesimpulan

Berdasarkan hasil pengujian dari proyek akhir ini, diperoleh kesimpulan sebagai berikut:

1. Teknik kinematika yang digunakan adalah kinematika maju untuk tiga buah roda *omni* segitiga sama sisi, dengan posisi dua buah roda *omni* didepan dan satu buah roda *omni* dibelakang.
2. Dari 10 kali percobaan membaca nilai jarak masing-masing roda didapat nilai *error* rata-rata 0.097%.
3. Dari 40 kali pengujian gerak lurus (maju, mundur, geser) robot didapatkan persentase *error* sebesar 0.27%.
4. Dari 20 kali pengujian gerak serong robot didapatkan rata-rata persentase *error* sebesar 0.41%.
5. Proses pengujian gerak lurus dilakukan dengan menggunakan alat bantu pengarah untuk mendapatkan gerak yang lurus, sedangkan gerak serong dilakukan tanpa pengarah. Hal ini yang membuat nilai gerak serong memperoleh presentase *error* lebih besar dari gerak lurus.
6. Dengan menggunakan modul *bluetooth HC-05* untuk berkomunikasi dengan PC, diperoleh jarak maksimal 11 meter.
7. Data pergerakan robot yang ditampilkan pada aplikasi Processing memiliki *delay* sebesar 300ms. Ini dikarenakan untuk menghindari terjadinya *error* akibat pembacaan data yang terlalu cepat.

## 5.2 Saran

Untuk memaksimalkan fungsi alat dibutuhkan perbaikan sebagai berikut :

1. Tambahan fitur "*Reset*" pada aplikasi, agar aplikasi tidak perlu di *close* untuk menjalankan kembali sensor.
2. Pengujian dilakukan langsung pada robot agar mendapat data yang

konkrit.

# DAFTAR PUSTAKA


1. *Pengertian dan kelebihan Arduino.* **Efendi, Ilham.** s.l. : IT-Jurnal.com, 2018.

2. *Indiamart.* [Online] India, 2019. [Dikutip: 14 Juni 2019.] https://www.indiamart.com/proddetail/arduino-uno-r3-smd-18668487973.html.

3. **Setiawan, Vendik.** s.l. : Academia.edu, 2019.

4. *RAM Electronic.* [Online] 2019. [Dikutip: 20 Juni 2019.] https://ram-e-shop.com/product/kit-bluetooth-hc05/.

5. *Speed Measurement Using Rotary Encoder for High Performance ac Drives.* **Briz, Fernado.** s.l. : researchgate.net, 1994.

6. [Online] Robu.in. [Dikutip: 20 Juni 2019.] https://robu.in/product/incremental-optical-rotary-encoder-400-pulse-400-ppr/.

7. **Nugroho, Widianto.** Processing. *Software Culture.* [Online] [Dikutip: 20 Juni 2019.] https://blogs.itb.ac.id/wnugroho/processing/.

8. **Old, Tekno.** Membuat Program Sederhana Dengan Aplikasi Processing Latihan 1. [Online] 21 Frebruary 2019. [Dikutip: 20 Juni Juni.] https://www.teknoowl.com/2019/02/membuat-program-sederhana-dengan.html.

9. **Prasetyo, Abdi.** Sensor MPU 6050 pada Mikrokontroller ATmega8535/16/32/64. *Coretan Kecil.* [Online] blogger, Sabtu 6 Desember 2014. [Dikutip: Selasa 8 September 2019.] http://prasetyoabdi.blogspot.com/2014/12/menggunakan-sensor-mpu6050-pada.html?m=1.

10. **.Taufiqqurohma, M.** *Odometry Method and Rotary Encoder for Wheeled Soccer Robot .* s.l. : researchgate, 2018.

# LAMPIRAN 1

## Daftar Riwayat Hidup

DAFTAR RIWAYAT HIDUP

1. Data Pribadi

| | |
|---|---|
| Nama Lengkap | : Eko Okta Pratama |
| Tempat & Tanggal Lahir | : Sungailiat, 19 Oktober 1998 |
| Alamat | : Jl. Imam Bonjol No. 20, Sungailiat Bangka |
| Hp | : 081271091345 |
| E-mail | : ekokta191098@gmail.com |
| Jenis Kelamin | : Laki-laki |
| Agama | : Islam |

2. Riwayat Pendidikan

| Institusi | Program Studi | Tahun Lulus |
|---|---|---|
| SD Harapan Sungailiat | | 2010 |
| SMP Harapan Sungailiat | | 2013 |
| SMK Negeri 1 Sungailiat | Teknik Komputer dan Jaringan | 2016 |
| Polman BABEL | DIII Teknik Elektronika | 2016-Sekarang |

3. Pendidikan Non Formal : -

Sungailiat, 27 Agustus 2019

Penulis

DAFTAR RIWAYAT HIDUP

1. Data Pribadi

| | |
|---|---|
| Nama Lengkap | : Chiki Leamongga |
| Tempat & Tanggal Lahir | : Toboali, 15 Marer 1998 |
| Alamat | : jl. Komplek Bukit Permai<br>Ds. Toboali<br>Kec. Toboali, Kab. Bangka Selatan,<br>Prov. Kep. Bangka Belitung |
| Hp | : 081949014626 |
| E-mail | : chiki15333@gmail.com |
| Jenis Kelamin | : Perempuan |
| Agama | : Islam |

2. Riwayat Pendidikan

| Institusi | Program Studi | Tahun Lulus |
|---|---|---|
| SD N 14 Toboali | | 2010 |
| SMP N 1 Toboali | | 2013 |
| SMK N 1 Toboali | Teknik Komputer dan Jaringan | 2016 |

Pendidikan Non Formal : -

Sungailiat, 27 Agustus 2019

Penulis

# LAMPIRAN 2

## Program Arduino Mega 2560

```
#include <MPU6050_tockn.h>
#include <Wire.h>
MPU6050 mpu6050(Wire);
const int ch_A1=2;
const int ch_B1=5;
const int ch_A2=18;
const int ch_B2=19;
const int ch_A3=3;
const int ch_B3=4;
int
a,b,c,d,e,f,encoder1,encoder2,encoder3,koorX,koorTX,koorTY,koorY,TX,TY,r
otasi,rotat,rpm1,rpm2,rpm3,t,rot,sudut,stepp,val;
float
putaran1,putaran2,putaran3,v1,v2,v3,w1,w2,w3,s1,s2,s3,st1,st2,st3,vx,vy,sx,
sy,sty,stx,r,SX,Vx;
unsigned long waktu;
uint8_t h; //int 0-25


void setup()
{
  // put your setup code here, to run once:
  pinMode(ch_A1,INPUT_PULLUP);
  pinMode(ch_B1,INPUT_PULLUP);
  attachInterrupt (digitalPinToInterrupt(2),bacaencoder1,RISING);

  pinMode(ch_A2,INPUT_PULLUP);
  pinMode(ch_B2,INPUT_PULLUP);
  attachInterrupt (digitalPinToInterrupt(18),bacaencoder2,RISING);

  pinMode(ch_A3,INPUT_PULLUP);
  pinMode(ch_B3,INPUT_PULLUP);
  attachInterrupt (digitalPinToInterrupt(3),bacaencoder3,RISING);
  waktu=millis();
  Serial.begin(9600);

  Wire.begin();
  mpu6050.begin();
  mpu6050.calcGyroOffsets(true);
}
void bacaencoder1()
{
  if(digitalRead(ch_B1) != digitalRead (ch_A1)){encoder1--;}
  else {encoder1++;}
}
void bacaencoder2()
{
```

```
  if(digitalRead(ch_B2) != digitalRead (ch_A2)){encoder2--;}
  else {encoder2++;}
}
void bacaencoder3()
{
  if(digitalRead(ch_B3) != digitalRead (ch_A3)){encoder3--;}
  else {encoder3++;}
}

void loop() {
  // put your main code here, to run repeatedly:
    mpu6050.update();
    rot=(mpu6050.getAngleZ());
    sudut=rot%360;
    if (Serial.available()){
    char val=Serial.read();
    if (val=='y')
    {
      stepp=1;
    }
    while(stepp==1)
   {
    Serial.print(c);Serial.print(";");
    Serial.print(d);Serial.print(";");
    Serial.print(stx);Serial.print(";");
    Serial.print(sty);Serial.print(";");
    Serial.print(vx);Serial.print(";");
    Serial.print(vy);Serial.print(";");
    Serial.print(koorX);Serial.print(";");
    Serial.print(koorY);Serial.println(";");
 Serial.print(sudut);Serial.println(";");

  if (millis()-waktu>=150)
   {
    putaran1=(float)encoder1/400;
    putaran2=(float)encoder2/400;
    putaran3=(float)encoder3/400;
    rpm1=(60/0.15)*putaran1; //Membaca nilai 1/4 detik
    rpm2=(60/0.15)*putaran2;
    rpm3=(60/0.15)*putaran3;
    w1=10.466667*rpm1;
    w2=10.466667*rpm2;
    w3=10.466667*rpm3;
    v1=0.024*w1;
    v2=0.024*w2;
    v3=0.024*w3;
    s1=0.20*v1; //Membaca nilai 1/4 detik
```

```
    s2=0.20*v2;
    s3=0.20*v3;
    st1=st1+s1;
    st2=st2+s2;
    st3=st3+s3;
    vy=((v1/0.866)-(v2/0.866))/-2;
    vx=(v3-(v1*0.5)-(v2*0.5))*-2/3;
    sx=0.152*vx;
    sy=0.152*vy;
    stx=stx+sx;
    sty=sty+sy;

    if(sudut>=-5&&sudut<=5)
    {
    koorX=stx/2; //Tanpa teta
    koorY=sty/2;

    c=stx/20;
    d=sty/20;
    a=koorX*5;//untuk titik2
    b=koorY*-5;
    }
    if(sudut>5||sudut<-5)
    {
    r=stx/2;
    SX=r*sin(sudut*0.01743295199433); //Lurus kk serong pake teta
    SY=r*cos(sudut*0.01743295199433);
    c=SX;
    d=SY;
    }
    encoder1=0;
    encoder2=0;
    encoder3=0;
   waktu=millis();
char val=Serial.read();
  if(val=='a')
{
  stepp=2;
  koorX=0;
  koorY=0;
  stx=0;
  sty=0;
  vx=0;
  vy=0;
  sudut=0;
  c=0;
  d=0;
```

```
    Serial.print(c);Serial.print(";");
    Serial.print(d);Serial.print(";");
    Serial.print(stx);Serial.print(";");
    Serial.print(sty);Serial.print(";");
    Serial.print(vx);Serial.print(";");
    Serial.print(vy);Serial.print(";");
    Serial.print(koorX);Serial.print(";");
    Serial.print(koorY);Serial.println(";");
  Serial.print(sudut);Serial.println(";");
   }
}
   }
```

# LAMPIRAN 3

## Program Aplikasi Processing

PROGRAM APLIKASI PROCESSING

```
import controlP5.*;
import processing.serial.*;
Serial port;
ControlP5 cp5;
Table table;
PFont font1,font2;
String val, filename;
float koorX,koorY,sx,sy,vx,vy,tx,ty;
PImage gambar;

void setup()
{
  background(200);
  size(1300,650);
  port = new Serial(this, "COM32", 9600);
  cp5 = new ControlP5(this);
  font1 = createFont("Arial",20);
  font2 = createFont("Times New Roman",30);

cp5.addButton("START")
.setPosition(410,500)
.setSize(80,40)
.setFont(font1)
;
   cp5.addButton("SIMPAN")
  .setPosition(410,550)
  .setSize(90,40)
  .setFont(font1)
  ;

table = new Table();
table.addColumn("tx");
table.addColumn("ty");
table.addColumn("sx");
table.addColumn("sy");
table.addColumn("vx");
table.addColumn("vy");
table.addColumn("koorX");
table.addColumn("koorY");
}

void draw()
{
  fill(#040824);
```

```
 rect(0, 0, 1300, 60);
 fill(#AA9C9B);
 rect(0,61, 1300, 30);
 fill(#20BF15);
 rect(0,92,670,700);
 //fill(255);
 //rect(672,92,670,700);
 fill(255);
 rect(272,237,45,30);
 rect(360,237,45,30);
 rect(272,292,75,30);
 rect(360,292,75,30);
 rect(272,346,75,30);
 rect(360,346,75,30);
 rect(272,401,55,30);

//strokeWeight(1);
fill(0);
for (int i = 727; i < 1277; i += 50) {
   line(i, 120, i, 620);
}
 line(977,100,977,640);
 line(976,100,976,640); //y titik 0
 line(1226,120,1226,620);//1 meter
 line(726,120,726,620);//1 meter

for (int i = 120; i < 670; i += 50) {
   line(726, i, 1226, i);
}
 line(700,371,1250,371);//x 0
 line(700,370,1250,370);
 line(726,621,1226,621);// 1 meter
 line(726,121,1226,121);//1 meter

 noStroke();
 smooth();
 fill(#FF1C03);
 ellipse(977+koorX,370+koorY,6,6);
 //line(976,371,976+koorX,371+koorY);

  fill(255);
  textFont(font2);
  text("SISTEM NAVIGASI ROBOT SEPAK BOLA",430,40);
  fill(0);
  textFont(font1);
  text("== P E R G E R A K A N   R O B O T ==",520,82);
```

```
  text("Koordinat (x,y)     :      ",90,260);
  text(nf(tx,1,0),280,260);
  text(nf(ty,1,0),365,260);

  text("Jarak (x,y)          :      ",90,315);
  text(nf(sx,1,2),280,315);
  text(nf(sy,1,2),371,315);
  text("cm",445,315);

  text("Kecepatan (x,y)   :      ",90,370);
  text(nf(vx,1,2),280,370);
  text(nf(vy,1,2),371,370);
  text("cm/s",445,370);

  text("Orientasi            :      ",90,425);
  //text(nf(g,1,0),92,425);
  text("derajat",445,425);

  if (port.available()>0)
  {
    val = port.readStringUntil('\n');
    if (val !=null)
    {
      String [] list = split(val, ';');
      tx = float(list[0]);
      ty = float(list[1]);
      sx = float(list[2]);
      sy = float(list[3]);
      vx = float(list[4]);
      vy = float(list[5]);
      koorX = float(list[6])*5;
      koorY = float(list[7])*-5;
}
int dd = day();
int m = month();
int y = year();
int h = hour();
int min = minute();
int s = second();
TableRow newRow = table.addRow();
newRow.setFloat("tx", tx);
newRow.setFloat("ty", ty);
newRow.setFloat("sx", sx);
newRow.setFloat("sy", sy);
newRow.setFloat("vx", vx);
newRow.setFloat("vy", vy);
newRow.setFloat("KoorX", koorX);
```

```
newRow.setFloat("KoorY", koorY);
newRow.setString("Jam", str(h) + ":" + str(min) + ":" + str(s));
}
}

void START()
{
  port.write('y');
}

void SIMPAN()
{
port.write('a');
int dd = day();
int m = month();
int y = year();
int h = hour();
int min = minute();
int s = second();
filename = "Data Robot/" + "Data Robot-" + str(h) + ":" + str(min) + ":" +
str(sec) + ".csv";
saveTable(table, filename);
exit();
}
```