



**PENERAPAN KINEMATIKA UNTUK LOKALISASI PADA
ROBOT SEPAK BOLA BERODA**



Oleh:

ANAN PEPE ABSENO

15410200045

PENERAPAN KINEMATIKA UNTUK LOKALISASI PADA ROBOT

SEPAK BOLA BERODA

TUGAS AKHIR



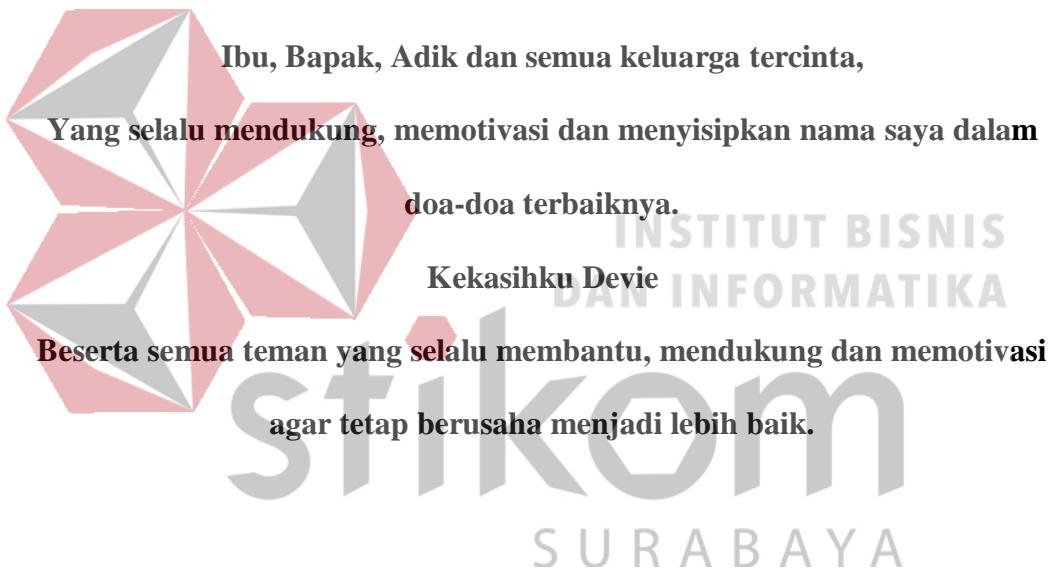
FAKULTAS TEKNOLOGI DAN INFORMATIKA

INSTITUT BISNIS DAN INFORMATIKA STIKOM SURABAYA

2019



Kupersembahkan Kepada ALLAH SWT



TUGAS AKHIR
PENERAPAN KINEMATIKA UNTUK LOKALISASI PADA
ROBOT SEPAK BOLA BERODA

Dipersiapkan dan disusun oleh
ANAN PEPE ABSENO
NIM : 15410200045

Telah diperiksa, diuji dan disetujui oleh Dewan Pembahasan
Pada : Januari 2019

Susunan Dewan Pembimbing dan Pengaji



Tugas Akhir ini telah diterima sebagai salah satu persyaratan

Untuk memperoleh gelar Sarjana



Dekan Fakultas Teknologi dan Informatika

INSTITUT BISNIS DAN INFORMATIKA STIKOM SURABAYA

SURAT PERNYATAAN
PERSETUJUAN PUBLIKASI DAN KEASLIAN KARYA ILMIAH

Sebagai mahasiswa Institut Bisnis dan Informatika Stikom Surabaya, saya :

Nama : Anan Pepe Abseno
NIM : 15410200045
Program Studi : S1 Sistem Komputer
Fakultas : Fakultas Teknologi dan Informatika
Jenis Karya : Tugas Akhir
Judul Karya : **PENERAPAN KINEMATIKA UNTUK LOKALISASI
PADA ROBOT SEPAK BOLA BERODA**

Menyatakan dengan sesungguhnya bahwa:

1. Demi pengembangan Ilmu Pengetahuan, Teknologi dan Seni, saya menyetujui memberikan kepada Institut Bisnis dan Informatika Stikom Surabaya Hak Bebas Royalti Non-Eksklusif (*Non-Exclusive Royalty Free Right*) atas seluruh isi/ sebagian karya ilmiah saya tersebut di atas untuk disimpan, dialihmediakan dan dikelola dalam bentuk pangkalan data (*database*) untuk selanjutnya didistribusikan atau dipublikasikan demi kepentingan akademis dengan tetap mencantumkan nama saya sebagai penulis atau pencipta dan sebagai pemilik Hak Cipta
2. Karya tersebut di atas adalah karya asli saya, bukan plagiat baik sebagian maupun keseluruhan. Kutipan, karya atau pendapat orang lain yang ada dalam karya ilmiah ini adalah semata hanya rujukan yang dicantumkan dalam Daftar Pustaka saya
3. Apabila dikemudian hari ditemukan dan terbukti terdapat tindakan plagiat pada karya ilmiah ini, maka saya bersedia untuk menerima pencabutan terhadap gelar kesarjanaan yang telah diberikan kepada saya.

Demikian surat pernyataan ini saya buat dengan sebenarnya.

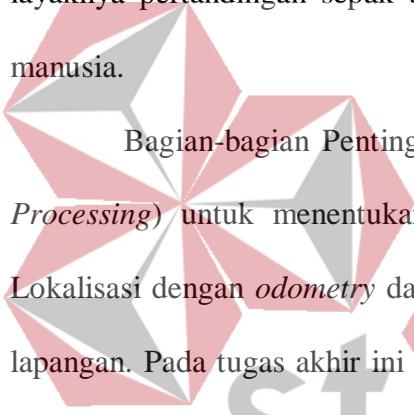
Surabaya, Januari 2019

menyatakan


Pepe Abseno
Nim : 15410200045

ABSTRAK

Sepak bola adalah salah satu cabang olahraga yang mempunyai banyak penggemar di setiap pertandingannya dimulai dari kalangan dewasa hingga anak-anak, namun dengan pesatnya perkembangan zaman saat ini pertandingan cabang olahraga tersebut kini dapat dinikmati dalam bentuk pertandingan antar robot. Robot sepak bola ialah suatu bidang pengembangan robotika yang menggunakan kecerdasan buatan sehingga robot ini dapat melakukan pertandingan sepak bola layaknya pertandingan sepak bola pada umumnya seperti yang dilakukan oleh manusia.



Bagian-bagian Penting pada bidang ini adalah pengolahan citra (*Image Processing*) untuk menentukan objek sekitar yang dideteksi melalui kamera, Lokalisasi dengan *odometry* dan kinematika untuk menentukan posisi robot pada lapangan. Pada tugas akhir ini telah dilakukan pembahasan mengenai pembuatan sistem untuk menentukan posisi robot berdasarkan area lokal yang telah diketahui seperti besar ukuran lapangan pertandingan sepak bola antar robot, Sehingga selalu melakukan perekaman gerak untuk mengetahui posisi sebenarnya.

Hasil dari pengujian yang didapatkan pada tugas akhir ini berupa perbandingan uji koordinat yang diketahui oleh sistem tanpa memperhatikan koordinat perpindahan robot yang sebenarnya dengan uji coba pergerakan robot yang diukur sesuai dengan keadaan perpindahan robot yang sebenarnya. Rata-rata *error* yang diketahui oleh sistem sebesar 0,399 cm sedangkan rata-rata *error* sebenarnya sebesar 2,108 cm.

Kata Kunci : *odometry*, kinematika, robot, robot sepak bola, lokalisasi, posisi.

KATA PENGANTAR

Segala puji syukur kehadirat Allah SWT, karena dengan rahmatnya dan hidayah-Nya penulis dapat menyelesaikan penyusunan Laporan Tugas Akhir yang berjudul “PENERAPAN KINEMATIKA UNTUK LOKALISASI PADA ROBOT SEPAK BOLA BERODA”. Laporan Tugas Akhir ini disusun dalam rangka penulisan laporan untuk memperoleh gelar Sarjana Komputer pada program studi S1 Sistem Informasi Stikom Surabaya.

Pada kesempatan ini, penulis ingin mengucapkan rasa terima kasih kepada pihak-pihak yang memberi dukungan dan masukan dalam menyelesaikan laporan Tugas Akhir ini. Oleh karena itu penulis ingin mengucapkan kepada:

1. Orang Tua dan Saudara-saudara saya tercinta yang telah memberikan dorongan dan bantuan baik moral maupun materi sehingga penulis dapat menempuh dan menyelesaikan Tugas Akhir ini.
2. Bapak Dr. Jusak selaku Dekan Fakultas Teknologi dan Informatika (FTI) Institut Bisnis dan Informatika Stikom Surabaya telah membantu proses penyelesaian Tugas Akhir yang dibuat oleh penulis dengan Baik.
3. Bapak Pauladie Susanto, S.Kom., M.T., selaku Ketua Program Studi S1 Sistem Komputer Stikom Surabaya, dan selaku Dosen Penguji atas ijin dan masukkan dalam menyusun Tugas Akhir ini.
4. Bapak Dr.Susijanto Tri Rasmana, S.Kom., M.T. dan Ibu Ira Puspasari, S.Si., M.T. selaku Dosen Pembimbing yang selalu memberi arahan dan bimbingan dalam menyelesaikan Tugas Akhir beserta laporan ini.
5. Semua staf dosen yang telah mengajar dan memberikan ilmunya.

6. Rekan-rekan Komunitas Stikom Robotik yang memberikan motivasi serta bantuan dalam penyelesaian Tugas Akhir ini.
7. Teman- teman seperjuangan SK angkatan 2015 dan semua pihak yang terlibat namun tidak dapat penulis sebutkan satu persatu atas bantuan dan dukungannya.
8. Serta semua pihak lain yang tidak dapat disebutkan secara satu per satu, yang telah membantu dalam menyelesaikan Tugas Akhir ini baik secara langsung maupun tidak langsung,

Penulis menyadari bahwa Laporan Tugas Akhir ini jauh dari kata sempurna, masih banyak kekurangan dalam menyusun laporan ini. Oleh karena itu dalam kesempatan ini, penulis meminta maaf apabila dalam Laporan Tugas Akhir ini masih banyak kesalahan baik dalam penulisan maupun Bahasa yang digunakan. Penulis juga memerlukan kritik dan saran dari para pembaca yang sifatnya membangun untuk kesempurnaan laporan yang telah penulis susun.



Penulis

DAFTAR ISI

	Halaman
HALAMAN JUDUL	i
HALAMAN SYARAT	ii
MOTTO	iii
HALAMAN PERSEMPAHAN.....	iv
HALAMAN PENGESAHAN.....	v
HALAMAN PERNYATAAN	vi
ABSTRAK.....	vii
KATA PENGANTAR	viii
DAFTAR ISI.....	x
DAFTAR GAMBAR.....	xiii
DAFTAR TABEL	xv
DAFTAR LAMPIRAN.....	xvi
BAB I PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Perumusan Masalah	2
1.3 Batasan Masalah	2
1.4 Tujuan.....	3
1.5 Sistematika Penulisan.....	3
BAB II LANDASAN TEORI	5
2.1 <i>Mobile Robot</i>	5
2.1.1 <i>Robot Holonomic</i>	6
2.1.2 <i>Kinematika</i>	7

2.2	Roda <i>Omnidirectional</i>	9
2.3	Motor DC.....	10
2.4	<i>Rotary Encoder</i>	11
2.5	<i>Odometry</i>	11
2.6	Arduino DUE.....	13
2.7	<i>Motor Driver EMS 30A</i>	15
2.8	UBEC (<i>Step Down</i>)	19
2.9	PWM	21
2.10	ARDUINO IDE	22
2.11	HC-05 modul <i>Bluetooth</i>	25
2.12	MIT App Inventor	27
BAB III ANALISIS DAN PERANCANGAN SISTEM		28
3.1	Perancangan Sistem Aktuator Robot	30
3.1.1	Perancangan Mekanik	30
3.1.2	Perancangan Sistem Kinematika	32
3.1.3	Perancangan Sistem <i>Odometry</i>	36
3.1.4	Konversi Jarak	38
3.1.5	Perhitungan RPM	41
3.2	Perancangan Elektronika Robot.....	42
3.3	Perancangan Aplikasi <i>Mobile</i>	44
3.4	Perancangan Komunikasi Robot.....	48
3.5	Perancangan Kontrol Robot.....	52
BAB IV PENGUJIAN		56
4.1	Uji Kecepatan Putar Motor	56
4.1.1	Tujuan Uji Kecepatan Putar Motor.....	56
4.1.2	Alat Yang Digunakan Pada Uji Kecepatan Putar Motor	56
4.1.3	Prosedur Pengujian Pada Uji Kecepatan Putar Motor	57
4.1.4	Hasil Pengujian Pada Uji Kecepatan Putar Motor.....	57

4.2	Uji Koordinat <i>Rotary Encoder</i>	58
4.2.1	Tujuan Uji Koordinat <i>Rotary Encoder</i>	58
4.2.2	Alat Yang Digunakan Pada Uji Koordinat <i>Rotary Encoder</i>	59
4.2.3	Prosedur Pengujian Pada Uji Koordinat <i>Rotary Encoder</i>	59
4.2.4	Hasil Pengujian Dari Uji Koordinat <i>Rotary Encoder</i>	60
4.3	Uji Pergerakan Robot.....	62
4.3.1	Tujuan Uji Pergerakan Robot.....	62
4.3.2	Alat Yang Digunakan Pada Uji Pergerakan Robot.....	62
4.3.3	Prosedur Pengujian Pada Uji Pergerakan Robot	62
4.3.4	Hasil Pengujian Pada Uji Pergerakan Robot	63
a.	Uji Pergerakan Persegi	64
b.	Uji Pergerakan Segitiga.....	65
4.4	Analisis Keseluruhan Pengujian Yang Telah Dilakukan	67
BAB V PENUTUP	69
5.1	Kesimpulan.....	69
5.2	Saran.....	70
DAFTAR PUSTAKA	71
LAMPIRAN	74
BIODATA PENULIS	80

DAFTAR GAMBAR

	Halaman
Gambar 2. 1 Robot KRSBI	5
Gambar 2. 2 Robot <i>Holonomic</i> (Thingbits, 2018).....	7
Gambar 2. 3 Kinematika	8
Gambar 2. 4 Roda <i>Omnidirectional</i>	9
Gambar 2. 5 Motor DC	10
Gambar 2. 6 <i>Pulse Rotary Encoder</i> (Nedelkovski, 2016).....	11
Gambar 2. 7 Proses <i>Odometry</i>	12
Gambar 2. 8 Arduino DUE (Eda, 2017)	13
Gambar 2. 9 <i>Motor Driver EMS 30A</i>	15
Gambar 2. 10 Tata Letak Komponen.....	17
Gambar 2. 11 UBEC (<i>Step Down</i>) (Tjahyadi, 2018).....	20
Gambar 2. 12 PWM (Firman, 2017).....	21
Gambar 2. 13 Arduino IDE <i>Startup</i> (Sinaryuda, 2017).....	23
Gambar 2. 14 Tampilan Lembar Kerja Arduino IDE (Sinaryuda, 2017)	24
Gambar 2. 15 Logo MIT App Inventor (Karen, 2017)	27
Gambar 3. 1 Blok Diagram Keseluruhan.....	29
Gambar 3. 2 Perancangan Mekanik.....	31
Gambar 3. 3 Persepsi Arah Hadap dan Pergerakan pada Lapangan.....	32
Gambar 3. 4 Konversi Arah Hadap	34
Gambar 3. 5 Keterangan Simbol Robot	37
Gambar 3. 6 Diameter Roda.....	39

Gambar 3. 7 Perhitungan Jarak	40
Gambar 3. 8 Perancangan Elektronik Robot.....	43
Gambar 3. 9 <i>Flowchart</i> Perancangan Aplikasi <i>Mobile</i>	45
Gambar 3. 10 Desain Aplikasi.....	46
Gambar 3. 11 Koneksi <i>Bluetooth</i>	47
Gambar 3. 12 Flowchart Penerimaan Data.....	49
Gambar 3. 13 Pisah Data.....	50
Gambar 3. 14 <i>Flowchart</i> Pisah Data.....	51
Gambar 3. 15 Blok Diagram Kontrol	52
Gambar 4. 1 Grafik PWM dan RPM Motor.....	58
Gambar 4. 2 Uji Pergerakan Persegi.....	64
Gambar 4. 3 Grafik Uji Gerak Persegi.....	65
Gambar 4. 4 Uji Pergerakan Segitiga	66
Gambar 4. 5 Garfik Uji Gerak Segitiga	67



DAFTAR TABEL

	Halaman
Tabel 2. 1 <i>Interface Header Motor Diver</i>	17
Tabel 2. 2 Power dan Motor <i>Connector</i>	18
Tabel 4. 1 Uji Kecepatan Putar Motor	57
Tabel 4. 2 Pengujian Koordinat Arah Gerak Maju	60
Tabel 4. 3 Pengujian Koordinat Arah Gerak Mundur	60
Tabel 4. 4 Pengujian Koordinat Arah Gerak Ke Samping Kiri.....	61
Tabel 4. 5 Pengujian Koordinat Arah Gerak Ke Samping Kanan.....	61
Tabel 4. 6 Pengujian Koordinat Rotasi Melawan Arah Jarum Jam.....	62
Tabel 4. 7 Pengujian Kecepatan Proporsi Motor.....	63
Tabel 4. 8 Perbandingan Target dan Error Terukur Persegi	64
Tabel 4. 9 Perbandingan Target dan Aktual Persegi	65
Tabel 4. 10 Perbandingan Target dan Error Terukur Segitiga	66
Tabel 4. 11 Perbandingan Target dan Aktual Segitiga.....	66

DAFTAR LAMPIRAN

Halaman

Lampiran 1. Program Robot 74



BAB I

PENDAHULUAN

1.1 Latar Belakang

Robot adalah perangkat yang dapat digunakan sebagai alat bantu manusia dalam menangani berbagai masalah yang telah ditemui sehingga manusia dapat menyelesaikan masalah tersebut secara efisien. Dalam melakukan tugasnya robot dapat bergerak secara manual yaitu dengan cara dikendalikan oleh manusia dari jarak tertentu guna mengantikan manusia dalam melaksanakan suatu tugas yang sulit dijangkau oleh manusia, maupun bergerak secara otomatis dengan cara menanamkan program yang dapat menyelesaikan suatu tugas pada robot. Salah satu faktor yang dapat mempengaruhi perkembangan robot di Indonesia ialah dengan adanya penyelenggaraan kompetisi robot dengan berbagai macam tema. Kontes Robot Sepak Bola Indonesia adalah salah satu tema atau divisi yang ada pada suatu kompetisi robot.

Pada bidang ini yaitu Robot Sepak Bola Beroda, robot ditanamkan kecerdasan untuk dapat bermain sepak bola secara otomatis sesuai dengan segala peraturan yang telah ditetapkan dengan cara meniru sistem kerja manusia saat bermain sepak bola yang sesungguhnya. Salah satu bagian terpenting dalam sistem kendali robot sepak bola beroda adalah pengaturan posisi robot, yang digunakan untuk mengatur formasi pertandingan dan menentukan lokalisasi posisi.

Penentuan posisi robot adalah aspek yang sangat penting untuk menentukan segala perencanaan pergerakan. Tanpa mengetahui posisi robot, robot hanya akan bergerak layaknya manusia buta yang tidak dapat menentukan lokasi dirinya sendiri

sehingga akan sulit dalam menentukan objek lainnya seperti letak gawang, batas lapangan dan posisi awal formasi pertandingan. Untuk mengatasi beberapa permasalahan di atas maka perlu dilakukan pembuatan sistem yang dirancang untuk dapat menentukan pergerakan dan menentukan posisi robot berdasarkan perekaman pergerakan robot.

Sistem kendali yang dilakukan untuk menentukan posisi robot adalah dengan cara merekam pergerakan robot yang bergerak dari titik awal posisi robot berada ke titik akhir posisi yang dituju. Dimana titik tersebut dapat diperoleh dari banyak sumber salah satunya adalah diperoleh dari perekaman *rotary encoder* dengan metode umpan balik *odometry*. Sistem kontrol yang digunakan sebagai penggerak robot ini menggunakan sistem kontrol kinematika agar *input* proses dapat lebih variatif dalam mengolah umpan balik dari *odometry*.

1.2 Perumusan Masalah

Berdasarkan latar belakang yang telah diuraikan di atas, diperoleh rumusan permasalahan sebagai berikut:

1. Bagaimana melakukan perancangan sistem kontrol kinematika untuk robot *holonomic*?
2. Bagaimana melakukan perancangan sistem umpan balik *odometry*?
3. Bagaimana melakukan lokalisasi robot yang dapat bergerak berdasarkan koordinat?

1.3 Batasan Masalah

Dalam sistem ini, agar tidak menyimpang dari tujuan yang nantinya akan

dicapai maka pembahasan masalah dibatasi pada hal – hal sebagai berikut:

1. Robot menggunakan rotary internal yang sudah terdapat dalam motor sehingga tidak dapat mengatasi masalah selip pada roda yang menimbulkan perbedaan perhitungan.
2. Robot berjalan tanpa memerlukan halangan yang ada.
3. Lapangan yang digunakan robot adalah lapangan yang datar atau tidak bergelombang.

1.4 Tujuan

Tujuan dari tugas akhir ini adalah sebagai berikut:

1. Robot dapat bergerak dengan sistem kontrol kinematika.
2. Robot dapat merekam pergerakan dengan menggunakan sistem umpan balik *odometry*.
3. Robot dapat melakukan lokalisasi yang dapat bergerak berdasarkan koordinat.

1.5 Sistematika Penulisan

Untuk memudahkan pembaca dalam memahami persoalan dan pembahasannya, maka penulisan laporan tugas akhir ini dibuat dengan sistematika sebagai berikut.

BAB I PENDAHULUAN

Pada bab ini membahas tentang latar belakang masalah dan penjelasan permasalahan secara umum, perumusan masalah serta batasan masalah yang dibuat, tujuan dari pembuatan tugas akhir dan sistematika penulisan buku.

BAB II LANDASAN TEORI

Pada bab ini membahas teori – teori yang berhubungan dan mendukung dalam pembuatan tugas akhir sperti *mobile robot*, arduino, kinematika, *odometry* dan literatur yang menunjang dalam pembuatan Tugas Akhir ini.

BAB III ANALISIS DAN PERANCANGAN SISTEM

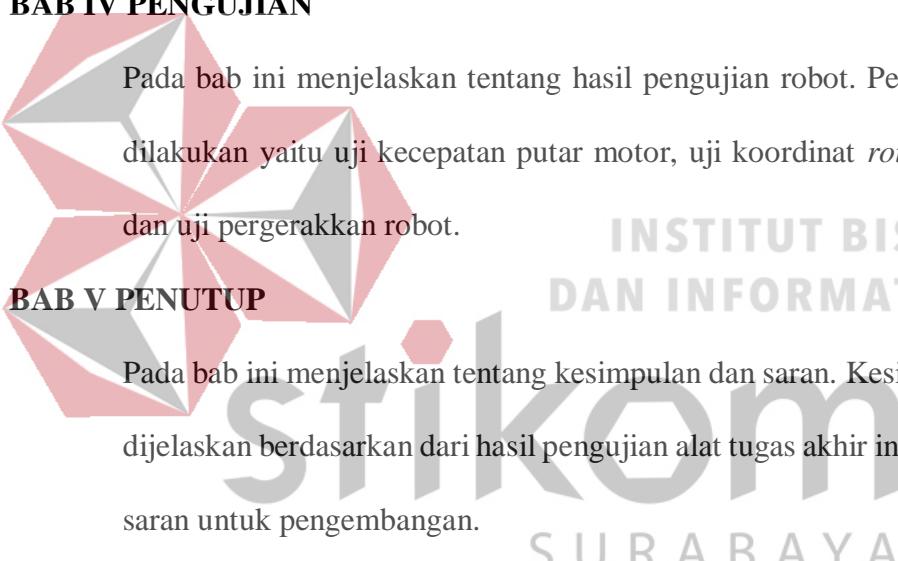
Pada bab ini membahas tentang perancangan sistem baik pada bagian perangakat keras, maupun perangkat lunak pada penerapan sistem kinematika ini.

BAB IV PENGUJIAN

Pada bab ini menjelaskan tentang hasil pengujian robot. Pengujian yang dilakukan yaitu uji kecepatan putar motor, uji koordinat *rotary encoder*, dan uji pergerakan robot.

BAB V PENUTUP

Pada bab ini menjelaskan tentang kesimpulan dan saran. Kesimpulan akan dijelaskan berdasarkan dari hasil pengujian alat tugas akhir ini, serta saran-saran untuk pengembangan.



BAB II

LANDASAN TEORI

2.1 *Mobile Robot*



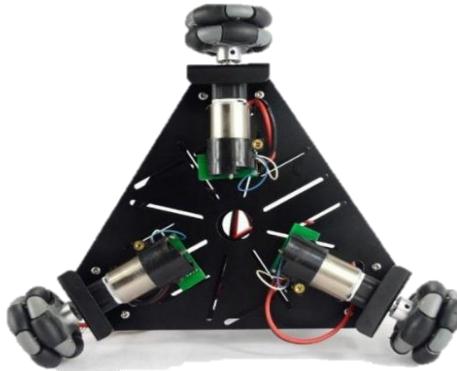
Gambar 2. 1 Robot KRSBI

Mobile robot adalah robot yang memiliki ciri khas mempunyai roda sebagai aktuatornya, yang memungkinkan robot untuk melakukan pergerakan. Pada tugas akhir ini robot yang digunakan adalah robot yang telah dibuat oleh tim robotik stikom surabaya yaitu robot sepak bola beroda pada gambar 2.1 dengan spesifikasi berikut:

1. Suplai tegangan sebesar 12-24V untuk motor penggerak, 12V untuk sistem elektronik, 5V untuk *mini PC*
2. Motor penggerak sebanyak 3 buah dengan tipe *Planetary Gear 36*
3. *Rotary encoder* internal pada motor dengan 7 pulse/rotation
4. Roda *omni* dengan diameter 100mm
5. Motor *driver* EMS 30 A
6. Kontroler menggunakan Arduino DUE
7. Transimisi data dengan *simulator* menggunakan ESP8266 WeMos D1 *Mini*



Robot holonomic adalah robot yang dapat bergerak ke segala arah tanpa melakukan rotasi sehingga robot memiliki ruang gerak bebas dalam melakukan pergerakan. Robot memungkinkan untuk melakukan pergerakan dengan banyak kombinasi seperti bergerak maju/mundur, geser kanan/kiri dan rotasi secara bersamaan, hal ini dikarenakan dengan adanya dukungan roda yang digunakan pada robot ini, roda yang digunakan adalah roda *omnidirectional*, roda khusus ini mempunyai roda kecil tambahan yang berporos tegak lurus pada roda inti, sehingga roda dapat bergerak segala arah. Sehingga robot dengan metode *holonomic* ini dapat melakukan pergerakan/mobilisasi robot lebih cepat, karena tanpa bermanuver saat belok. Desain robot *holonomic* pada umumnya dapat dilihat pada gambar 2.2.



Gambar 2. 2 Robot *Holonomic* (Thingbits, 2018)

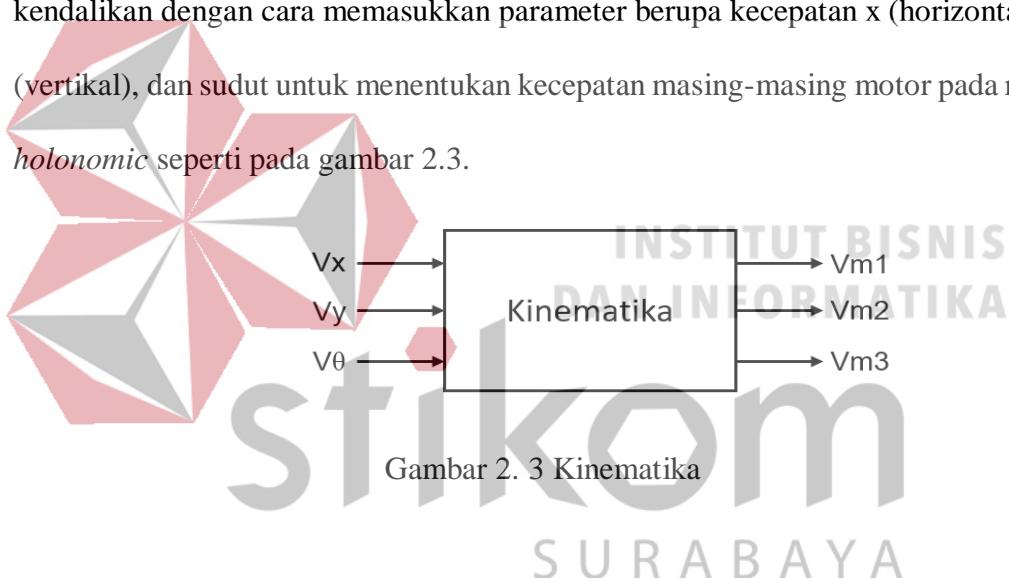
Pada umumnya robot di desain dengan pergerakan yang sudah direncanakan terlebih dahulu. Pada sistem pergerakkan konvensional, pergerakkan tidak mampu di kontrol pada setiap tingkat kebebasan dalam bergerak secara independen, sehingga hanya mampu bergerak ke beberapa arah yang sudah ditentukan sebelumnya. Ini disebut kendala *non-holomic* yaitu pencegahan roda kemudi dari selip, meskipun pada umumnya mampu menjangkau setiap lokasi dan orientasi dalam ruang 2 dimensi, namun memerlukan manuver dan perencanaan jalan yang rumit dan kompleks (Doroftei, 2007).

2.1.2 *Kinematika*

Kinematika robot adalah studi analisis pergerakan kaki atau roda robot terhadap sistem kerangka koordinat acuan yang diam atau bergerak tanpa memperhatikan gaya yang menyebabkan pergerakan tersebut. Model kinematika merepresentasikan hubungan *end effector* dalam ruang tiga dimensi dengan variabel sendi dalam ruang sendi. Model Kinematika Robot dalam kinematika dikenal

istilah *forward* kinematika dan *invers* kinematika. *Forward* kinematika adalah metode untuk menentukan orientasi dan posisi *end effector* dari besarnya sudut sendi dan panjang *link* kaki robot. Sedangkan *invers* kinematika merupakan kebalikan dari *forward* kinematika yaitu metode untuk mengetahui nilai sudut pada sendi-sendi yang diperlukan agar *end effector* dapat mencapai posisi yang dikehendaki (Setiawan, 2015).

Untuk mempermudah melakukan kontrol robot *holonomic* ini digunakanlah rumus kinematika, karena dengan metode ini robot dapat dikendalikan dengan cara memasukkan parameter berupa kecepatan x (horizontal), y (vertikal), dan sudut untuk menentukan kecepatan masing-masing motor pada robot *holonomic* seperti pada gambar 2.3.



Implementasi kinematika untuk menentukan pergerakan robot dapat dituliskan dengan persamaan *inverse* kinematika seperti dibawah ini:

$$\begin{pmatrix} Vm_1 \\ Vm_2 \\ Vm_3 \end{pmatrix} = \begin{pmatrix} -\sin\theta_1 & \cos\theta_1 & 1 \\ -\sin\theta_2 & \cos\theta_2 & 1 \\ -\sin\theta_3 & \cos\theta_3 & 1 \end{pmatrix} \begin{pmatrix} V_x \\ V_y \\ V_\theta \end{pmatrix} \quad (2-1)$$

Keterangan:

V_m = Kecepatan Motor (rpm)

θ =Sudut pemasangan roda pada robot ($^{\circ}$)

V_x =Kecepatan robot horizontal (m/s)

V_y =Kecepatan robot vertikal (m/s)

$R\omega$ =Kecepatan Sudut (rad/s)

Dengan matriks di atas dapat ditentukan persamaan kecepatan seluruh motor sesuai sudut masing masing (Rojas, 2006).

2.2 Roda *Omnidirectional*

Roda *Omnidirectional* telah menjadi populer untuk digunakan pada *mobile robot*, karena roda ini memungkinkan robot untuk berjalan dengan arah yang sama tanpa harus memutar terlebih dahulu jika robot bergerak maju maupun bergeser. Selain itu, gerakan translasi sepanjang jalur yang diinginkan bisa dikombinasikan dengan rotasi, sehingga robot itu tiba tujuan pada sudut yang benar.



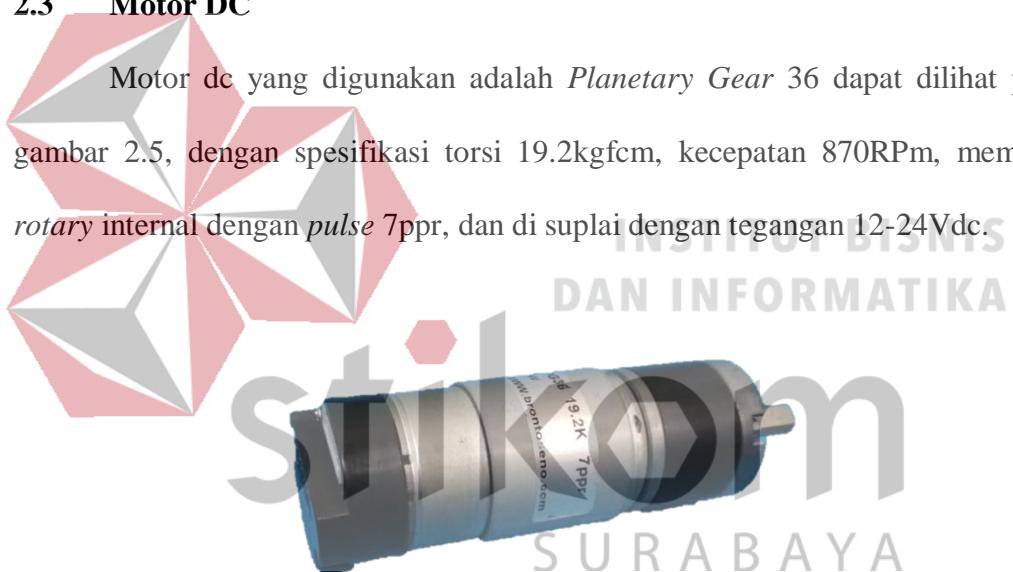
Gambar 2. 4 Roda *Omnidirectional*

Roda *Omnidirectional* sebagian besar didasarkan prinsip yang sama,

sementara roda menyediakan traksi dalam arah Seperti biasa pada sumbu motor, roda dapat meluncur tanpa geseran arah sumbu motor. Untuk mencapai ini, roda dibangun menggunakan roda yang lebih kecil yang terpasang di sepanjang pinggiran roda utama menggunakan rol dengan arah rotasi yang tidak paralel atau tegak lurus sumbu motor. Gambar 2.4 menunjukkan contoh jenis roda yang digunakan pada Tugas Akhir ini yaitu roda *omni* yang menggunakan diameter 100ml.

2.3 Motor DC

Motor dc yang digunakan adalah *Planetary Gear 36* dapat dilihat pada gambar 2.5, dengan spesifikasi torsi 19.2kgfcm, kecepatan 870RPM, memiliki *rotary internal* dengan *pulse* 7ppr, dan di suplai dengan tegangan 12-24Vdc.



Gambar 2. 5 Motor DC

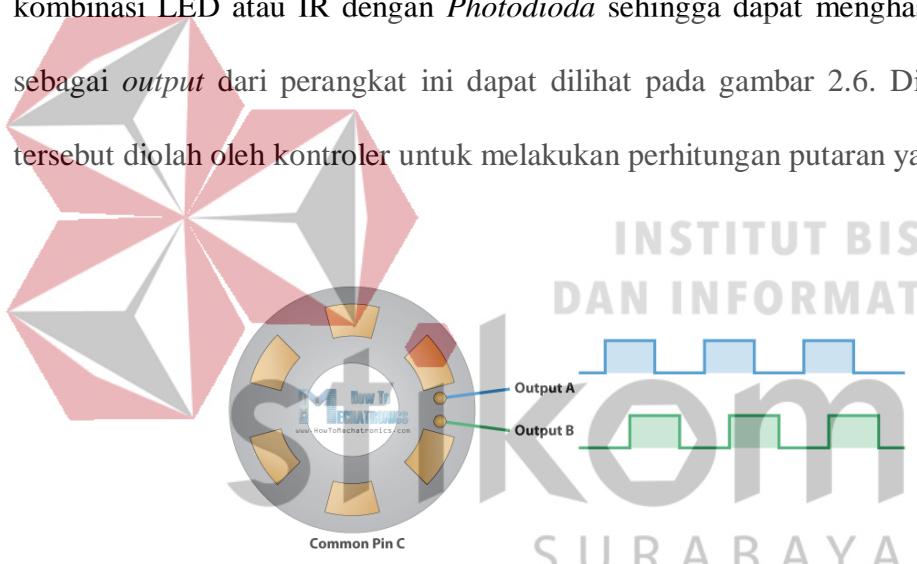
Rotary internal yang sudah termasuk pada motor sebenarnya tergolong *rotary* dengan *pulse* rendah namun *rotary* tersebut terpasang langsung pada putaran inti dinamo dan dengan adanya *gearbox* untuk *output* putaran sehingga *rotary* dapat lebih detil dalam melakukan pembacaan putaran.

Motor DC yang digunakan sebagai aktuator robot adalah motor *Planetary Gear-36* dengan perbandingan *gear down* sebesar 1:50 sehingga mempunyai torsi

yang cukup besar untuk penggerak robot pada Tugas Akhir ini, sebaliknya motor ini mempunyai kecepatan yang cukup pelan untuk melakukan pergerakan.

2.4 *Rotary Encoder*

Rotary Encoder merupakan suatu perangkat yang biasa digunakan untuk menghitung putaran poros, sehingga biasanya banyak digunakan untuk mendeteksi pergerakan/perpindahan posisi, cara kerja perangkat ini dengan membaca suatu lempengan yang mempunyai lubang ataupun bergerigi dengan menggunakan kombinasi LED atau IR dengan *Photodiode* sehingga dapat menghasilkan pulsa sebagai *output* dari perangkat ini dapat dilihat pada gambar 2.6. Dimana pulsa tersebut diolah oleh kontroler untuk melakukan perhitungan putaran yang terbaca.

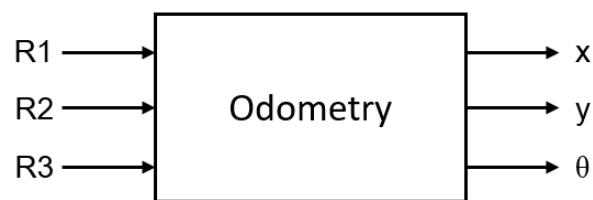


Gambar 2. 6 *Pulse Rotary Encoder* (Nedelkovski, 2016)

2.5 *Odometry*

Odometry adalah penggunaan data dari sensor pergerakan untuk memperkirakan perubahan posisi dari waktu ke waktu. *Odometry* digunakan untuk memperkirakan posisi relatif terhadap posisi awal, *odometry* membutuhkan sistem elektronik penunjang, bagian penting dari sistem elektronik penunjang *odometry* adalah sensor pergerakan. Sensor pergerakan yang bisa digunakan untuk sistem

odometry antara lain kamera, sensor inersia, maupun *rotary encoder*. *Rotary encoder* lebih disukai karena cara akses sensor yang relatif lebih mudah sehingga bisa langsung diproses dengan menggunakan mikrokontroler (Basori, 2014).



Gambar 2. 7 Proses *Odometry*

Seperti pada gambar 2.7 *odometry* bekerja dengan cara memasukkan parameter berupa *pulse* dari ketiga *rotary encoder* sehingga dapat menghasilkan nilai keluaran berupa x , y , θ untuk diolah sebagai koreksi umpan balik pergerakan.

Dalam penggunaan metode *odometry*, persamaan yang digunakan dalam melakukan pembacaan kombinasi beberapa *rotary encoder* dapat menggunakan rumus kinematika yaitu *forward* kinematika dengan mendapatkan nilai koordinat x , y , dan θ berdasarkan pembacaan kombinasi *rotary encoder*. Sesuai dengan *base* robot menggunakan 3 *rotary encoder* internal yang telah tersedia pada motor dc maka rumus forward kinematika dapat diperoleh sebagai berikut:

$$x = R_2 - R_1 \cos 60 - R_3 \cos 300 \quad (2-2)$$

$$y = R_1 \sin 60 - R_3 \cos 300 \quad (2-3)$$

$$\theta = \frac{R_1 + R_2 + R_3}{3} \quad (2-4)$$

Keterangan:

x =Koordinat horizontal (cm)

y=Koordinat vertikal (cm)

θ =Koordinat radian ($^{\circ}$)

R= Rotary Encoder

2.6 Arduino DUE



Gambar 2.8 Arduino DUE (Eda, 2017)

Arduino Due adalah board mikrokontroler yang berbasis pada CPU Atmel SAM3X8E ARM Cortex-M3. Ini adalah board Arduino pertama yang berbasis pada mikrokontroler ARM inti 32-bit. Ini memiliki 54 pin *input / output* digital (12 di antaranya dapat digunakan sebagai *output* PWM), 12 *input* analog, 4 UART (port serial perangkat keras), *clock* 84 MHz, koneksi USB OTG, 2 DAC (digital to analog) , 2 TWI, header SPI, header JTAG, tombol reset dan tombol hapus. Pada gambar 2.8 adalah bentuk fisik dari Arduino Due.

Peringatan: Tidak seperti kebanyakan board Arduino, board Arduino Due berjalan pada 3.3V. Tegangan maksimum yang dapat ditoleransi I / O pin adalah 3.3V. Menerapkan tegangan yang lebih tinggi dari 3.3V ke pin I / O manapun dapat

merusak board.

Board berisi segala sesuatu yang dibutuhkan untuk mendukung mikrokontroler; cukup hubungkan ke komputer dengan kabel *micro-USB* atau nyalakan dengan adaptor AC-ke-DC atau baterai untuk memulai. Arduino Due kompatibel dengan semua perisai Arduino yang bekerja di 3.3V dan sesuai dengan *pinout* Arduino 1.0.

Due mengikuti 1.0 *pinout*:

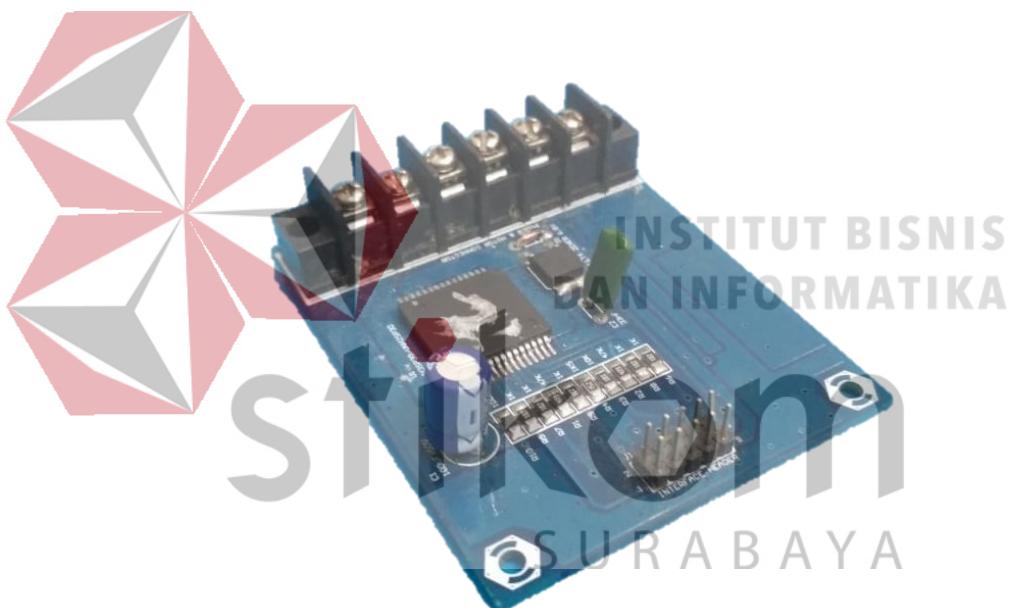
1. TWI: pin SDA dan SCL yang berada di dekat pin AREF.
2. IOREF: memungkinkan perisai terlampir dengan konfigurasi yang tepat untuk menyesuaikan tegangan yang disediakan oleh board. Hal ini memungkinkan kompatibilitas perisai dengan board 3.3V seperti *board* berbasis Due dan AVR yang beroperasi pada 5V.
3. Pin yang tidak terhubung, disediakan untuk penggunaan masa depan.

Spesifikasi:

- *Microcontroller AT91SAM3X8E*
- *Operating Voltage 3.3V*
- *Input Voltage (recommended) 7-12V*
- *Input Voltage (limits) 6-16V*
- *Digital I/O Pins 54 (of which 12 provide PWM output)*
- *Analog Input Pins 12*
- *Analog Output Pins 2 (DAC)*
- *Total DC Output Current on all I/O lines 130 mA*
- *DC Current for 3.3V Pin 800 mA*
- *DC Current for 5V Pin 800 mA*

- *Flash Memory 512 KB all available for the user applications*
- *SRAM 96 KB (two banks: 64KB and 32KB)*
- *Clock Speed 84 MHz*
- *Length 101.52 mm*
- *Width 53.3 mm*
- *Weight 36 g*

2.7 Motor Driver EMS 30A



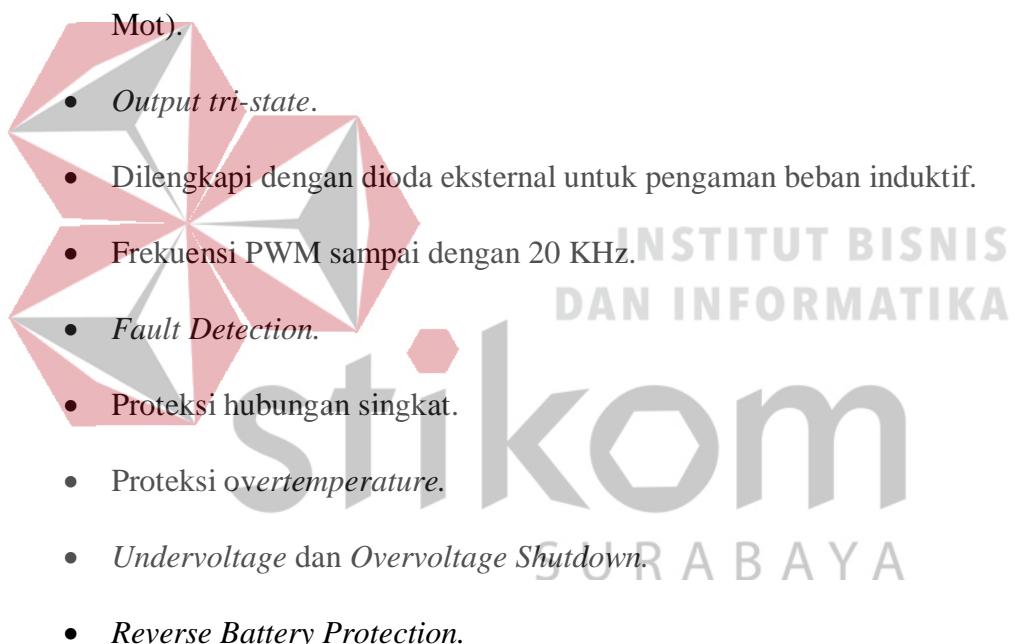
Gambar 2. 9 *Motor Driver EMS 30A*

Embedded Module Series (EMS) 30 A H-Bridge merupakan *driver H-Bridge* yang didisain untuk menghasilkan *drive 2 arah* dengan arus berkelanjutan sampai dengan 30A pada tegangan 5,5 Volt sampai 16 Volt. Modul ini dilengkapi dengan rangkaian sensor arus beban yang dapat digunakan sebagai umpan balik ke pengendali. Bentuk fisik dari motor driver ini terdapat pada gambar 2.9. Modul ini

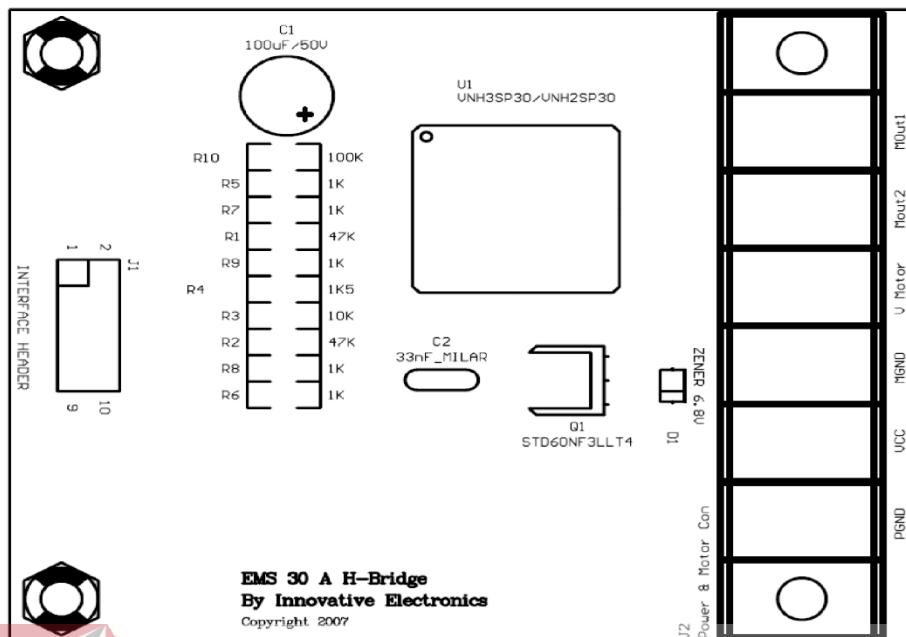
mampu men-drive beban-beban induktif seperti misalnya relay, solenoida, motor DC, motor stepper, dan berbagai macam beban lainnya.

Spesifikasi:

- Terdiri dari 1 *driver full H-Bridge* beserta rangkaian *current sense*.
- Mampu melewatkkan arus kontinyu 30 A.
- *Range* tegangan *output* untuk beban: 5,5 V sampai 16 V.
- *Input* kompatibel dengan level tegangan TTL dan CMOS.
- Jalur catu daya *input* (VCC) terpisah dari jalur catu daya untuk beban (V



Tata letak komponen dapat dilihat pada gambar 2.10 berikut ini:



Gambar 2. 10 Tata Letak Komponen

Modul H-Bridge memiliki 1 set *header* (J1) dan 1 set terminal konektor (J2).

Pada bagian ini akan dijelaskan deskripsi dan fungsi dari masing-masing *header* dan konektor tersebut.

Interface Header (J1) berfungsi sebagai *input* untuk antarmuka dengan *input-output* digital serta *output* analog dari modul H-Bridge. Pada tabel 2.1 berikut deskripsi dari masing-masing pin pada *Interface Header*:

Tabel 2. 1 *Interface Header Motor Diver*

NO. PIN	NAMA	I/O	FUNGSI
1	MIN1	I	Pin <i>input</i> untuk menentukan <i>output</i> MOUT1
2	MIN2	I	Pin <i>input</i> untuk menentukan <i>output</i> MOUT2

3	MEN1	I/O	Pin <i>enable</i> untuk <i>output MOUT1</i> Diberi logika <i>High</i> untuk mengaktifkan <i>half H-Bridge 1</i> , diberi logika <i>Low</i> secara eksternal untuk menonaktifkan <i>half H-Bridge 1</i> Jika terjadi kondisi <i>Fault</i> (<i>thermal shutdown, undervoltage, overvoltage</i> , dsb.), maka pin ini akan ditarik <i>Low</i> secara internal oleh modul <i>H-Bridge</i> untuk melaporkan adanya kondisi <i>Fault</i>
4	MEN2	I/O	Pin <i>enable</i> untuk <i>output MOUT2</i> Diberi logika <i>High</i> untuk mengaktifkan <i>half H-Bridge 2</i> , diberi logika <i>Low</i> secara eksternal untuk menonaktifkan <i>half H-Bridge 2</i> Jika terjadi kondisi <i>Fault</i> (<i>thermal shutdown, undervoltage, overvoltage</i> , dsb.), maka pin ini akan ditarik <i>Low</i> secara internal oleh modul <i>H-Bridge</i> untuk melaporkan adanya kondisi <i>Fault</i>
5	MCS	O	<i>Output</i> tegangan analog yang berbanding lurus dengan arus beban (<i>Range output 0 – 5 Volt</i>)
6	MPWM	I	Pin <i>input</i> untuk mengatur kerja modul <i>H-Bridge</i> secara PWM
7,9	VCC	-	Terhubung ke catu daya untuk <i>input</i> (5 Volt)
8,10	PGND	-	Titik referensi untuk catu daya <i>input</i>

Arus (dalam Ampere) yang dilewatkan oleh *H-Bridge* dapat dihitung dengan rumus:

$$I = \frac{\text{Tegangan}_\text{output}_\text{pada}_\text{pin}_\text{MCS} \times 11370}{1500} \quad (2-5)$$

Power & Motor Connector berfungsi sebagai konektor untuk catu daya dan beban. Berikut pada tabel 2.2 deskripsi dari masing-masing terminal pada *Power & Motor Connector*:

Tabel 2. 2 Power dan Motor *Connector*

NAMA	FUNGSI
PGND	Titik referensi untuk catu daya <i>input</i>
VCC	Terhubung ke catu daya untuk <i>input</i> (5 Volt)

MGND	Titik referensi untuk catu daya <i>output</i> ke beban
V MOTOR (V MOT)	Terhubung ke catu daya untuk <i>output</i> ke beban
MOUT2	<i>Output</i> ke beban dari <i>half H-Bridge</i> kedua
MOUT1	<i>Output</i> ke beban dari <i>half H-Bridge</i> pertama

Sebuah modul *H-Bridge* 30A dapat digunakan untuk mengatur kerja 1 buah motor DC secara 2 arah.

2.8 UBEC (*Step Down*)

Mengubah tegangan, tinggi ke rendah atau sebaliknya, memerlukan rangkaian yang tepat, agar daya dapat di-deliver dengan tingkat efisiensi setinggi mungkin. Menurunkan tegangan dengan menggunakan IC regulator seperti 7805, sangat umum digunakan. Regulator ini memiliki kemampuan menangani arus hingga 1A, dengan V_{in} minimal sama dengan 7V, untuk menghasilkan *output* 5V. Dengan perhitungan sederhana, bila $V_{in} = 9V$, maka disipasi daya ~ 4 Watt, satu nilai yang cukup besar (panas), atau menggunakan regulator linier tipe LDO, seperti 2940, yang juga memiliki kemampuan menangani arus hingga 1A, dengan V_{in} minimal sama dengan 5.5V, untuk menghasilkan *output* 5V.

Pilihan lain adalah regulator switching. Untuk kebutuhan mencatu motor servo atau rangkaian lain yang bekerja pada tingkat tegangan 5V – 6V, dapat menggunakan UBEC. UBEC – *Universal Battery Elimination Circuit* adalah rangkaian elektronik yang mengambil daya dari battery pack atau sumber DC lainnya, dan menurunkannya ke level tegangan 5V atau 6V. Tegangan *input* maksimum tergantung pada spesifikasi UBEC.

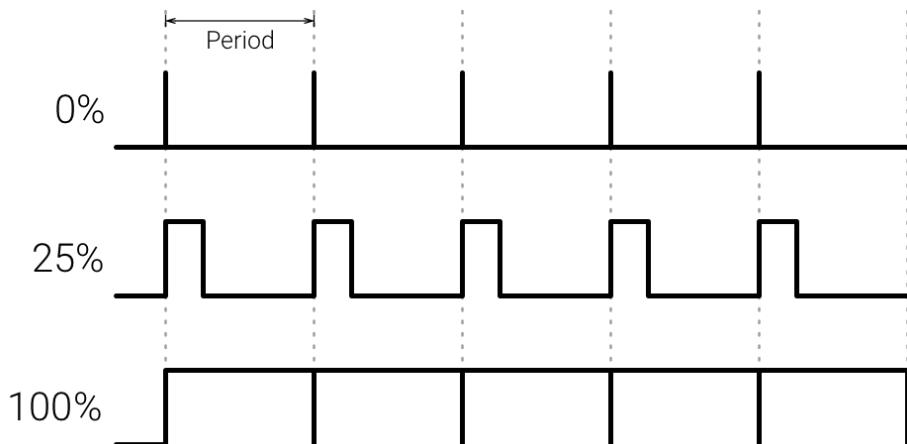


Gambar 2. 11 UBEC (*Step Down*) (Tjahyadi, 2018)

UBEC biasanya digunakan pada aplikasi yang memerlukan arus lebih tinggi, dan perangkat ini mampu mengantarkan daya dengan efisiensi hingga 92%. Ketika memilih UBEC, pastikan model UBEC yang dipilih memiliki rating arus yang sesuai dengan kebutuhan (beban). Bentuk fisik dari ubec ditampilkan pada gambar 2.11. Rangkaian lain yang juga sering dibutuhkan adalah DC-DC *Booster*. Sebagai contoh, satu produk DC-DC, mampu menghasilkan *output* 3.7V – 34V dengan *input* 3.7V – 34V. Artinya, dengan tegangan *input* minimum 3.7V dapat dihasilkan *output* maksimum 34V, dengan arus *input* maksimum 3A, serta mampu men-deliver daya dengan tingkat efisiensi hingga 90%.

Tegangan *input* tidak boleh lebih besar dari *output* yang dihasilkan. Dalam banyak aplikasi, khususnya aplikasi robotik, seringkali dibutuhkan kombinasi keduanya, agar dapat menggunakan satu catu battery pack (Tjahyadi, 2018).

2.9 PWM



Gambar 2. 12 PWM (Firman, 2017)

PWM (*Pulse Width Modulation*) adalah salah satu teknik modulasi dengan mengubah lebar pulsa (*duty cylce*) dengan nilai amplitudo dan frekuensi yang tetap. Pada gambar 2.12 ditampilkan perbandingan *duty cycle* pada PWM. Satu siklus pulsa merupakan kondisi high kemudian berada di zona transisi ke kondisi low. Lebar pulsa PWM berbanding lurus dengan amplitudo sinyal asli yang belum termodulasi. *Duty Cycle* merupakan representasi dari kondisi logika high dalam suatu periode sinyal dan di nyatakan dalam bentuk (%) dengan *range* 0% sampai 100%, sebagai contoh jika sinyal berada dalam kondisi high terus menerus artinya memiliki *duty cycle* sebesar 100%. Jika waktu sinyal keadaan high sama dengan keadaan low maka sinyal mempunyai *duty cycle* sebesar 50%.

Aplikasi penggunaan PWM biasanya ditemui untuk pengaturan kecepatan motor dc, pengaturan cerah/redup LED, dan pengendalian sudut pada motor servo. Contoh penggunaan PWM pada pengaturan kecepatan motor dc semakin besar nilai *duty cycle* yang diberikan maka akan berpengaruh terhadap cepatnya putaran motor. Apabila nilai *duty cylce*-nya kecil maka motor akan bergerak lambat (Firman,

2017).

Untuk membandingkannya terhadap tegangan DC, PWM memiliki 3 mode operasi yaitu:

1. *Inverted Mode.* Pada mode *inverted* ini jika nilai sinyal lebih besar dari pada titik pembanding (*compare level*) maka *output* akan di set high (5v) dan sebaliknya jika nilai sinyal lebih kecil maka *output* akan di set low (0v) seperti pada gelombang A pada gambar di atas.
2. *Non Inverted Mode.* Pada mode *non inverted* ini *output* akan bernilai high (5v) jika titik pembanding (*compare level*) lebih besar dari pada nilai sinyal dan sebaliknya jika bernilai low (0v) pada saat titik pembanding lebih kecil dari nilai sinyal seperti pada gelombang B pada gambar di atas.
3. *Toggle Mode.* Pada mode *toggle* *output* akan beralih dari nilai high (5v) ke nilai low (0v) jika titik pembanding sesuai dan sebaliknya beralih dari nilai low ke high.

2.10 ARDUINO IDE

Arduino IDE (*Integrated Development Environment*) adalah software yang di gunakan untuk memprogram di arduino, dengan kata lain Arduino IDE sebagai media untuk memprogram *board* arduino. Arduino IDE ini berguna sebagai *text editor* untuk membuat, mengedit, dan juga mevalidasi kode program. Bisa juga digunakan untuk meng-upload ke board Arduino. Kode program yang digunakan pada Arduino disebut dengan istilah Arduino “sketch” atau disebut juga *source code* arduino, dengan ekstensi file *source code* .ino. Gambar 2.13 menampilkan tampilan awal Arduino IDE saat pertama kali dibuka.



Gambar 2. 13 Arduino IDE *Startup* (Sinaryuda, 2017)

Editor *Programming* pada umumnya memiliki fitur untuk *cut / paste* dan untuk *find / replace* teks, demikian juga pada Arduino IDE. Gambar 2.14 menampilkan tampilan lembar kerja pada Arduino IDE. Pada bagian **keterangan aplikasi** memberikan pesan balik saat menyimpan dan mengekspor serta sebagai tempat menampilkan kesalahan. **Konsol log** menampilkan teks log dari aktifitas Arduino IDE, termasuk pesan kesalahan yang lengkap dan informasi lainnya. Pojok kanan bawah menampilkan port serial yang di gunakan. Tombol *toolbar* terdapat ikon tombol pintas untuk memverifikasi dan meng-upload program, membuat, membuka, dan menyimpan *sketch*, dan membuka monitor serial.



Gambar 2. 14 Tampilan Lembar Kerja Arduino IDE (Sinaryuda, 2017)

Verify pada versi sebelumnya dikenal dengan istilah *Compile*. Sebelum aplikasi di-upload ke board Arduino, biasakan untuk memverifikasi terlebih dahulu *sketch* yang dibuat. Jika ada kesalahan pada *sketch*, nanti akan muncul error. Proses *Verify* / *Compile* mengubah *sketch* ke binary code untuk di-upload ke mikrokontroller.

Upload tombol ini berfungsi untuk mengupload sketch ke *board* Arduino. Walaupun kita tidak mengklik tombol *verify*, maka sketch akan di-*compile*, kemudian langsung di-*upload* ke *board*. Berbeda dengan tombol *verify* yang hanya berfungsi untuk memverifikasi *source code* saja.

- **New Sketch** Membuka window dan membuat *sketch* baru.
- **Open Sketch** Membuka *sketch* yang sudah pernah dibuat. *Sketch* yang dibuat dengan IDE Arduino akan disimpan dengan ekstensi file .ino
- **Save Sketch** menyimpan *sketch*, tapi tidak disertai dengan mengkompile.
- **Serial Monitor** Membuka *interface* untuk komunikasi serial, nanti akan kita diskusikan lebih lanjut pada bagian selanjutnya.
- **Keterangan Aplikasi** pesan-pesan yang dilakukan aplikasi akan muncul di sini, misal **Compiling** dan **Done Uploading** ketika kita meng-*compile* dan meng-*upload sketch* ke *board* Arduino
- **Konsol log** Pesan-pesan yang dikerjakan aplikasi dan pesan-pesan tentang *sketch* akan muncul pada bagian ini. Misal, ketika aplikasi meng-*compile* atau ketika ada kesalahan pada sketch yang kita buat, maka informasi *error* dan baris akan diinformasikan di bagian ini.
- **Baris Sketch** bagian ini akan menunjukkan posisi baris kursor yang sedang aktif pada *sketch*.

2.11 HC-05 modul *Bluetooth*

HC-05 Adalah sebuah modul *Bluetooth SPP (Serial Port Protocol)* yang mudah digunakan untuk komunikasi serial *wireless* (nirkabel) yang mengkonversi port serial ke *Bluetooth*. HC-05 menggunakan modulasi *bluetooth* V2.0 + EDR (*Enhanced Data Rate*) 3 Mbps dengan memanfaatkan gelombang radio berfrekuensi 2,4 GHz.

Modul ini dapat digunakan sebagai slave maupun master. HC-05 memiliki 2 mode konfigurasi, yaitu AT mode dan *Communication* mode. AT mode berfungsi

untuk melakukan pengaturan konfigurasi dari HC-05. Sedangkan *Communication mode* berfungsi untuk melakukan komunikasi bluetooth dengan piranti lain.

Dalam penggunaannya, HC-05 dapat beroperasi tanpa menggunakan *driver* khusus. Untuk berkomunikasi antar *Bluetooth*, minimal harus memenuhi dua kondisi berikut:

1. Komunikasi harus antara *master* dan *slave*.
2. *Password* harus benar (saat melakukan pairing).

Jarak sinyal dari HC-05 adalah 30 meter, dengan kondisi tanpa halangan.

Adapun spesifikasi dari HC-05 adalah:

Hardware

- Sensitivitas -80dBm (*Typical*).
- Daya transmit RF sampai dengan +4dBm.
- Operasi daya rendah 1,8V – 3,6V I/O.
- Kontrol PIO.
- Antarmuka UART dengan baudrate yang dapat 26ntenna26m.
- Dengan 26ntenna terintegrasi.

Software

- *Default baudrate* 9600, Data bit : 8, Stop bit = 1, *Parity* : No Parity, Mendukung *baudrate* : 9600, 19200, 38400, 57600, 115200, 230400 dan 460800.
- Auto koneksi pada saat device dinyalakan (*default*).
- Auto *reconnect* pada menit ke 30 ketika hubungan putus karena *range* koneksi.

2.12 MIT App Inventor



Gambar 2. 15 Logo MIT App Inventor (Karen, 2017)

MIT App Inventor adalah aplikasi inovatif yang dikembangkan Google dan MIT untuk mengenalkan dan mengembangkan pemrograman android dengan mentrasformasikan bahasa pemrograman yang kompleks berbasis teks menjadi berbasis visual (*drag and drop*) berbentuk blok-blok. MIT App Inventor memungkinkan semua orang untuk membuat aplikasi yang berfungsi penuh untuk smartphone dan tablet. Aplikasi berbasis blok ini memfasilitasi pembuatan aplikasi yang kompleks dengan waktu proses pembuatan yang cukup singkat dibanding membuat aplikasi android pada umumnya. Proyek MIT App Inventor adalah untuk mendemokratiskan pengembangan perangkat lunak dengan memberdayakan semua orang, terutama generasi muda, untuk beralih dari konsumsi teknologi menjadi ke penciptaan teknologi. Gambar 2.15 adalah logo dari MIT App Inventor.

BAB III

ANALISIS DAN PERANCANGAN SISTEM

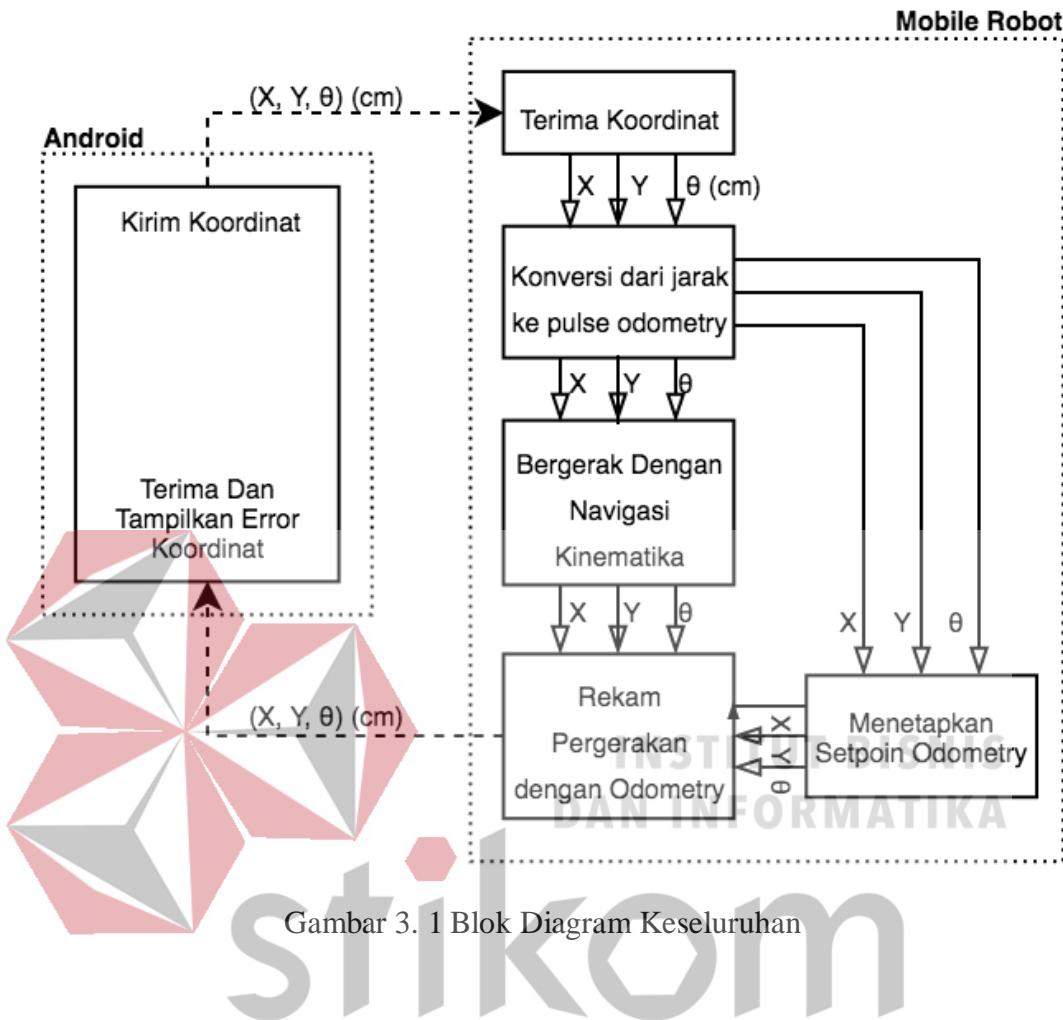
Pada bagian bab ini akan dijelaskan mengenai perancangan keseluruhan sistem. Perancangan perangkat lunak antara lain aplikasi smartphone untuk mengirim koordinat robot, proses perhitungan untuk konversi dari satuan jarak ke satuan pulsa yang digunakan sebagai setpoint *odometry*, perancangan sistem kendali robot menggunakan kinematika serta perancangan sistem perekaman gerak robot menggunakan *odometry*. Perancangan perangkat keras meliputi pengkabelan seluruh perangkat elektronika pada robot, perancangan tata letak aktuator. Dibagian bab ini juga disjelaskan mengenai blok diagram keseluruhan dari sistem lokalisasi robot ini.

Pada Tugas Akhir ini digunakan perhitungan rumus *inverse kinematika* sebagai penentuan proporsi kecepatan dimasing-masing motor saat diberikan kecepatan robot berupa nilai (x, y, θ) robot dapat berjalan dengan linear sesuai arah hingga sampai dengan tujuan koordinat.

Sebagai perhitungan umpan balik dari pergerakan robot yang dibaca oleh kombinasi 3 *rotary encoder* atau disebut juga dengan *odometry*, digunakanlah rumus *forward kinematika* untuk menentukan proporsi perhitungan dari pembacaan masing masing *rotary encoder* sehingga keluaran dari masing masing *rotary encoder* sudah berupa satuan koordinat yaitu (x, y, θ) .

Aplikasi android digunakan untuk memudahkan pengguna dalam melakukan kontrol maupun *monitoring* terhadap *error* yang telah diberikan oleh robot. Aplikasi ini berfungsi untuk memberikan koordinat tujuan kepada robot

sekaligus dapat menampilkan nilai error dari robot saat melakukan pergerakan.



Gambar 3.1 di atas menjelaskan tentang blok diagram keseluruhan sistem lokalisasi pada robot sepak bola beroda. Berikut penjelasan secara garis besar:

1. Dari aplikasi android mengirim koordinat target robot yang dibungkus dengan format “ $<X, Y, \theta>$ ” kemudian dikirim menuju robot melalui koneksi bluetooth.
2. Saat diterima robot data yang dibungkus tersebut dikupas kembali menjadi satuan data terpisah x, y , dan θ .
3. Konversi dilakukan saat data sudah menjadi data terpisah dengan tipe data integer. Konversi dalam merubah satuan jarak menjadi satuan pulsa untuk

setpoint odometry dilakukan dengan menghitung beberapa aspek yaitu diameter roda robot, panjang target lokasi, jumlah pulsa rotary dalam satu putaran.

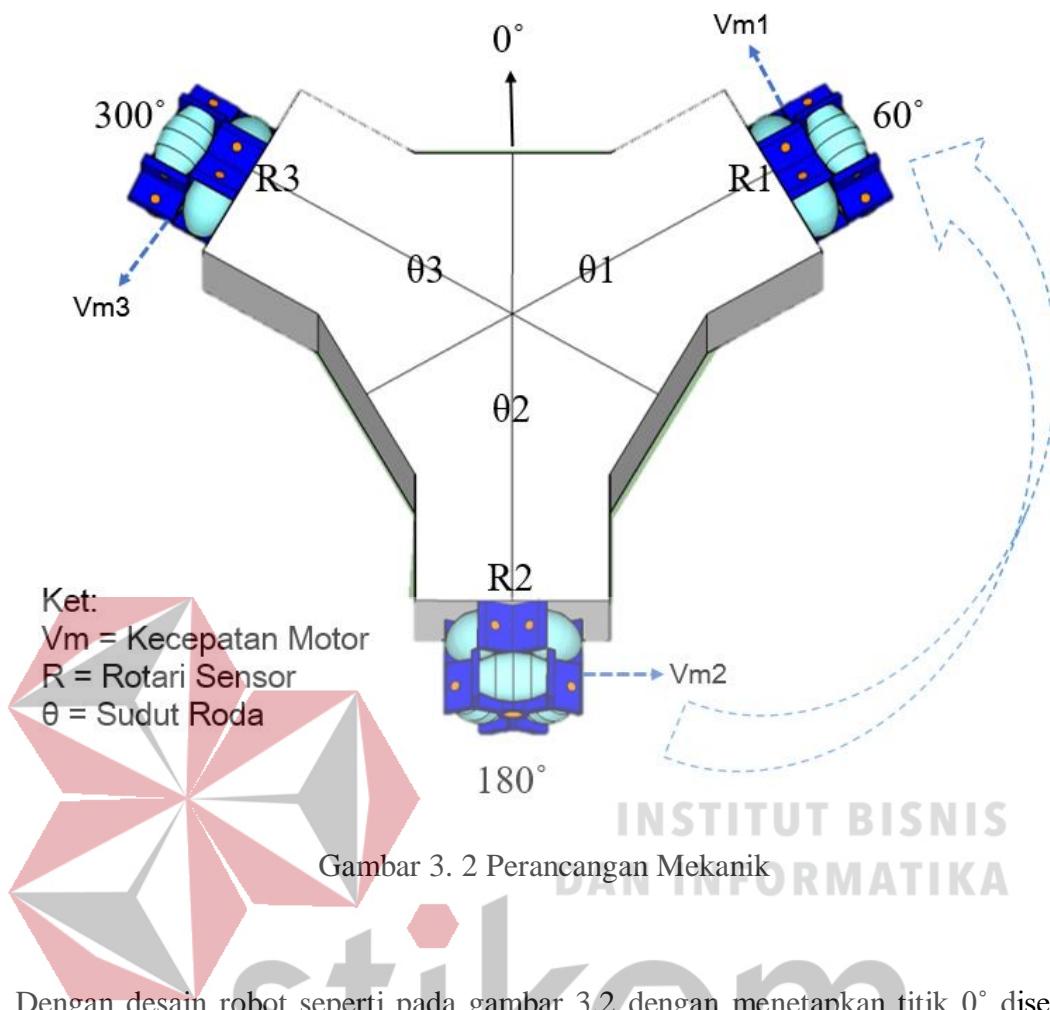
4. Setelah menjadi satuan pulsa hasil konversi tersebut ditetapkan sebagai *setpoint odometry*, dan ditetapkan sebagai *input* navigasi *inverse kinematika* untuk menentukan proporsi kecepatan masing-masing motor.
5. *Odometry* merekam pergerakan robot melalui kombinasi 3 *rotary encoder* secara *real time* dan selalu membandingkan dengan nilai *setpoint* hingga nilai *odometry* saat itu sama dengan *setpoint* yang dituju.
6. Pembacaan *odometry* tersebut dikirim ke *smartphone* untuk menampilkan *error* robot sebagai monitoring bagi pengguna.

3.1 Perancangan Sistem Aktuator Robot

Pada sub bab ini memaparkan mengenai seluruh perancangan aktuator atau penggerak robot sepak bola yang digunakan pada Tugas Akhir ini.

3.1.1 Perancangan Mekanik

Basis robot yang digunakan pada Tugas Akhir ini adalah tipe basis berbentuk (Y) dengan menggunakan 3 roda seperti pada gambar F.2 maka rumusan dapat dituliskan sesuai sudut roda yang telah didapat dari bentuk basis yang digunakan, yaitu 60° , 180° , 300° .



Dengan desain robot seperti pada gambar 3.2 dengan menetapkan titik 0° disela sudut 60° dan 300° , yang nantinya titik tengah ini digunakan sebagai acuan dari arah hadap robot sepak bola beroda. Dengan ditetapkannya titik arah hadap berdasarkan desain di atas maka memungkinkan mekanisme penendang dan penggiring bola ditetapkan pada titik tersebut. Sehingga tujuan dari desain mekanisme penggerak pada robot sepak bola ini selain menentukan konfigurasi tata letak roda juga sebagai penentuan tata letak mekanisme yang lain seperti mekanisme penendang dan penggiring bola.



Gambar 3. 3 Persepsi Arah Hadap dan Pergerakan pada Lapangan

Pada Gambar 3.3 adalah persepsi arah dan pergerakan robot berdasarkan koordinat pada lapangan sepak bola, dengan titik koordinat (0, 0) tepat di pojok bagian kiri bawah lapangan yang disesuaikan menghadap gawang lawan.

3.1.2 Perancangan Sistem Kinematika

Dalam melakukan kontrol pergerakan robot, penggunaan sistem kinematika menurut teori yang didapat adalah menggunakan *inverse kinematika* dengan mengacu pada persamaan 2-1, berikut adalah penjabaran persamaan berdasarkan teori yang didapat dengan implementasi rumus pada kontrol pergerakan robot *holonomic* beroda 3:

$$\begin{pmatrix} Vm_1 \\ Vm_2 \\ Vm_3 \end{pmatrix} = \begin{pmatrix} -\sin\theta_1 & \cos\theta_1 & 1 \\ -\sin\theta_2 & \cos\theta_2 & 1 \\ -\sin\theta_3 & \cos\theta_3 & 1 \end{pmatrix} \begin{pmatrix} V_x \\ V_y \\ R\omega \end{pmatrix} \quad (3-1)$$

Hasil perkalian matriks dari persamaan 3-1 adalah sebagai berikut:

$$\begin{pmatrix} Vm_1 \\ Vm_2 \\ Vm_3 \end{pmatrix} = \begin{pmatrix} -V_x\sin\theta_1 + V_y\cos\theta_1 + R\omega \\ -V_x\sin\theta_2 + V_y\cos\theta_2 + R\omega \\ -V_x\sin\theta_3 + V_y\cos\theta_3 + R\omega \end{pmatrix} \quad (3-2)$$

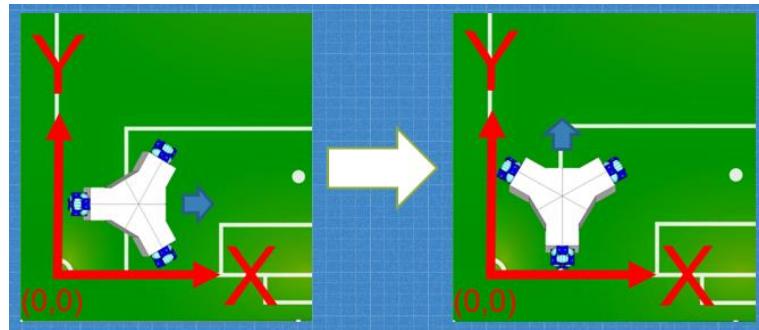
Maka hasil vektor kecepatan masing-masing motor berdasarkan persamaan 3-2 adalah:

$$Vm_1 = -V_x\sin\theta_1 + V_y\cos\theta_1 + R\omega \quad (3-3)$$

$$Vm_2 = -V_x\sin\theta_2 + V_y\cos\theta_2 + R\omega \quad (3-4)$$

$$Vm_3 = -V_x\sin\theta_3 + V_y\cos\theta_3 + R\omega \quad (3-5)$$

Berdasarkan desain mekanisme yang telah dibuat dan ketentuan persepsi arah hadap robot yang menghadap koordinat (Y) pada pembahasan bab sebelumnya, maka perlu dilakukan konversi arah hadap agar sesuai dengan ketepatannya dikarenakan pada rumus *inverse kinematika* mempunyai standard normal arah hadap robot yang menghadap koordinat (X).



Gambar 3. 4 Konversi Arah Hadap

Pada gambar 3.4 adalah analogi dari konversi arah hadap. Cara melakukan konversi arah hadap cukup dengan mengurangi sudut pemasangan seluruh roda dengan 90° .

Berikut adalah penulisan persamaan rumus untuk masing-masing kecepatan motor sesuai bentuk basis penempatan roda yang telah dirancang:

$$V_{m_1} = -V_x \sin(\theta_1 - 90^\circ) + V_y \cos(\theta_1 - 90^\circ) + R\omega \quad (3-6)$$

$$V_{m_2} = -V_x \sin(\theta_2 - 90^\circ) + V_y \cos(\theta_2 - 90^\circ) + R\omega \quad (3-7)$$

$$V_{m_3} = -V_x \sin(\theta_3 - 90^\circ) + V_y \cos(\theta_3 - 90^\circ) + R\omega \quad (3-8)$$

Keterangan:

V_m =Kecepatan motor (rpm)

V_y =Kecepatan robot vertikal (m/s)

V_x =Kecepatan robot horizontal (m/s)

$R\omega$ =Kecepatan sudut robot (rad/s)

θ =Sudut pemasangan roda ($\theta_1=60^\circ$, $\theta_2=180^\circ$, $\theta_3=300^\circ$)

Dengan adanya persamaan 3-6, 3-7, 3-8 maka telah diketahui proporsi masing masing kecepatan motor yang telah ditentukan oleh persamaan tersebut

sesuai dengan besar/kecil *input* yang diberikan terhadap kecepatan X (V_x), kecepatan Y (V_y), dan kecepatan sudut ($R\omega$). Rumus kinematika yang digunakan sangatlah fleksibel, rumus ini dapat disesuaikan dengan sudut pada tata letak pemasangan roda dan banyaknya roda yang digunakan, sehingga cukup mudah dalam memodifikasi tata letak roda sesuai kebutuhan pengguna. Untuk menentukan kecepatan robot pada Tugas Akhir ini maka diperlukan persamaan pembalik dari persamaan 3-6, 3-7, 3-8, sehingga dapat dituliskan menjadi persamaan 3-9, 3-10, 3-11 sebagai berikut:

$$V_x = Vm_2 - Vm_1 \cos 60 - Vm_3 \cos 300 \quad (3-9)$$

$$V_y = Vm_1 \sin 60 - Vm_3 \sin 300 \quad (3-10)$$

$$V_\theta = \frac{Vm_1 + Vm_2 + Vm_3}{L} \quad (3-11)$$

Keterangan:

V_x =Kecepatan robot horizontal (m/s)

V_y =Kecepatan robot vertikal (m/s)

V_θ =Kecepatan sudut robot (rad/s)

Vm =Kecepatan motor (rpm)

L =Jarak roda dengan titik pusat robot (cm)

Berikut adalah *listing* program dari penggunaan kontrol kinematika pada robot:

```
void kinematika(float xin, float yin, float sudutin)
{
    float kec_kanan = -xin*float(sin((60-90)*(PI/180.0))) +
        yin*float(cos((60-90)*(PI/180.0))) + sudutin;

    float kec_belakang = -xin*float(sin((180-90)*(PI/180.0))) +
        yin*float(cos((180-90)*(PI/180.0))) + sudutin;

    float kec_kiri = -xin*float(sin((300-90)*(PI/180.0))) +
        yin*float(cos((300-90)*(PI/180.0))) + sudutin;

    if(kec_kanan > 0) kanan_cw();
    else if(kec_kanan < 0) kanan_ccw();
    else kanan_stop();

    if(kec_kiri > 0) kiri_cw();
    else if(kec_kiri < 0) kiri_ccw();
    else kiri_stop();

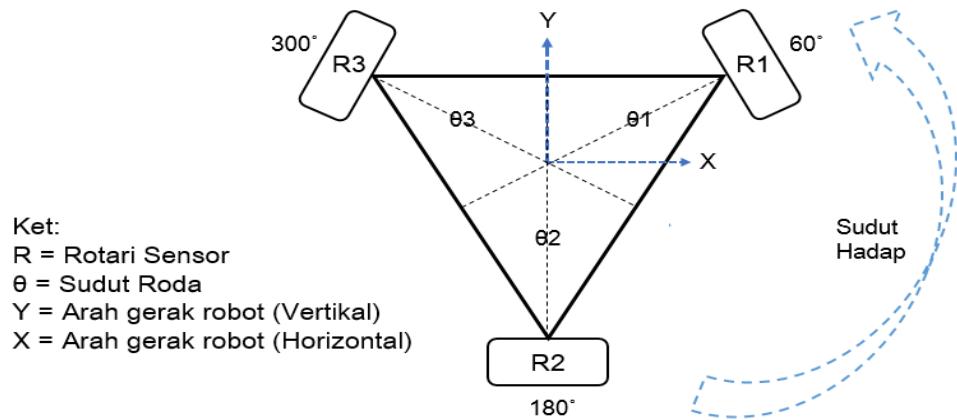
    if(kec_belakang > 0) belakang_cw();
    else if(kec_belakang < 0) belakang_ccw();
    else belakang_stop();

    kec_kanan = abs(kec_kanan);
    kec_kiri = abs(kec_kiri);
    kec_belakang = abs(kec_belakang);

    analogWrite(PWM1, kec_kiri);
    analogWrite(PWM2, kec_kanan);
    analogWrite(PWM3, kec_belakang);
}
```

3.1.3 Perancangan Sistem *Odometry*

Odometry adalah metode yang digunakan sebagai sistem umpan balik dari pembacaan kombinasi beberapa *rotary encoder* untuk merekam seluruh pergerakan robot sehingga dapat menghasilkan titik koordinat dan arah hadap robot.



Gambar 3. 5 Keterangan Simbol Robot

Sesuai dengan desain yang digunakan seperti pada gambar 3.5 maka persamaan rumus *odometry* yang digunakan adalah persamaan 3-12, 3-13, 3-14 sebagai berikut:

$$x = R_2 - R_1 \cos 60 - R_3 \cos 300 \quad (3-12)$$

$$y = R_1 \sin 60 - R_3 \cos 300 \quad (3-13)$$

$$\theta = \frac{R_1 + R_2 + R_3}{3} \quad (3-14)$$

Keterangan:

x = Koordinat horizontal robot (cm)

y = Koordinat vertikal robot (cm)

θ = Koordinat arah hadap robot (°)

R = Pulsa *rotary encoder* (Satuan)

Perlu dilakukan konversi pembacaan rotasi robot dikarenakan arah rotasi robot pada Tugas Akhir ini berotasi melawan arah jarum jam sehingga dapat

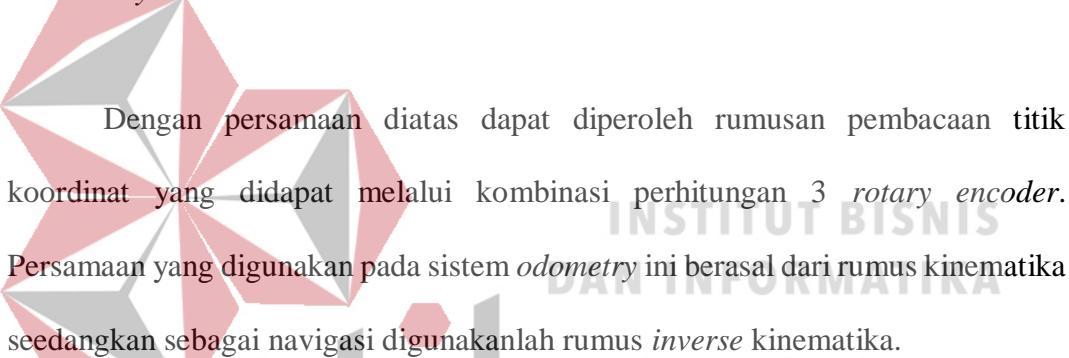
dituliskan bahwa nilai koordinat rotasi bernilai negatif (-θ), berikut konversi persamaan yang mengacu persamaan 3-14 menjadi persamaan 3-15 di bawah ini:

$$\begin{aligned}-\theta &= \left(\frac{R_1 + R_2 + R_3}{3} \right) \\ \theta &= -\left(\frac{R_1 + R_2 + R_3}{3} \right)\end{aligned}\tag{3-15}$$

Keterangan:

θ =koordinat radian ($^{\circ}$)

R = *Rotary encoder*



Berikut adalah *listing* program dari penggunaan *odometry* pada robot:

```
void odometry()
{
    x=r_belakang-(0.5*r_kanan)-(0.5*r_kiri);
    y=(0.87*r_Kanan)-(0.87*r_kiri);
    heading = -(r_kiri + r_kanan + r_belakang)/3;
}
```

3.1.4 Konversi Jarak

Konversi ini dilakukan untuk mendapatkan nilai *setpoint odometry* berupa total jumlah *pulse* yang nantinya nilai ini selalu dibandingkan dengan nilai error dari umpan balik *rotary encoder*. Dalam konversi ini perlu mengetahui beberapa aspek seperti diameter atau jari-jari dari roda *omni* yang digunakan untuk

pengukuran rotary encoder pada robot, jarak titik pusat dengan masing-masing roda pada robot, jumlah *pulse per rotation* pada *rotary encoder*, dan jarak koordinat tujuan itu sendiri yang dalam pengerjaan Tugas Akhir ini menggunakan satuan centi meter.



Gambar 3. 6 Diameter Roda

Perlu diketahui keliling roda yang terhubung dengan *rotary encoder*. Pada Tugas Akhir ini menggunakan *rotary* internal pada motor, oleh karena itu keliling roda yang diukur adalah keliling dari roda penggerak itu sendiri seperti pada gambar 3.6, yaitu sebesar 100mm dan total *pulse* dalam satu putaran adalah 356 *pulse*. Rumus yang digunakan adalah rumus keliling lingkaran seperti persamaan 3-16 dibawah ini :

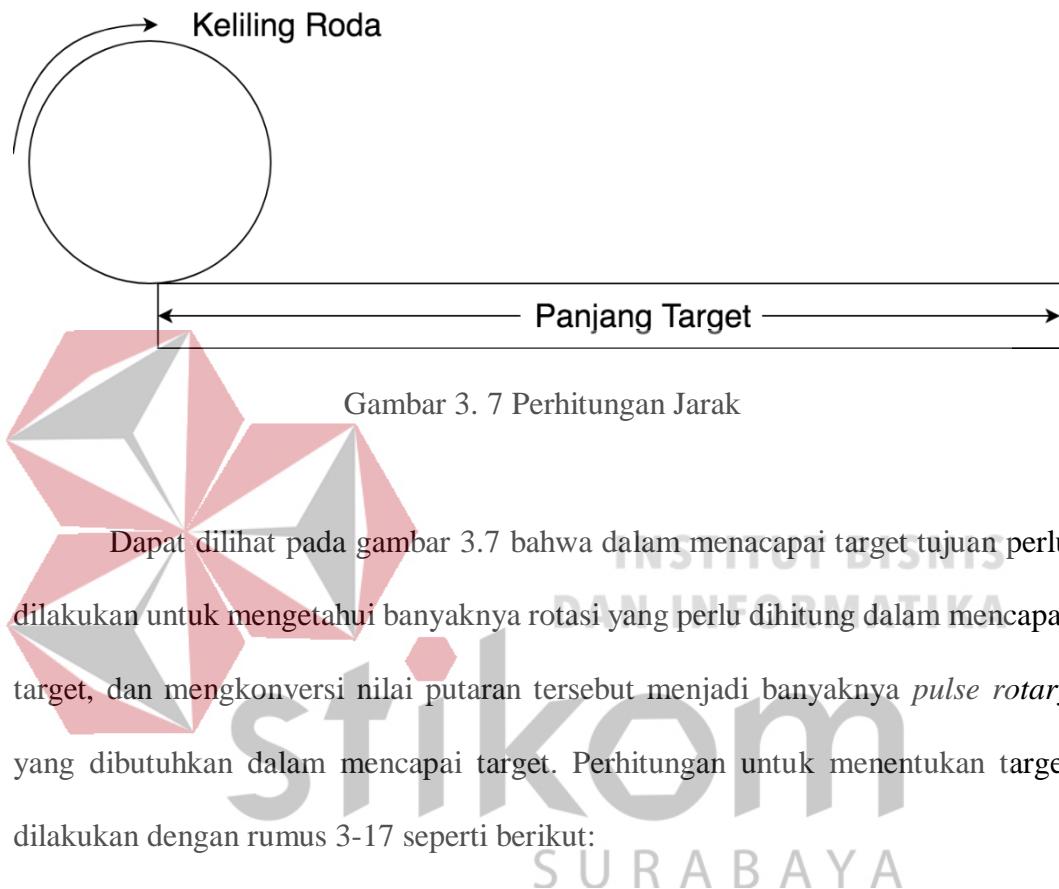
$$\text{Keliling Roda} = \pi 2r \quad (3-16)$$

Keterangan:

$$\Pi = 3.14159265$$

$$R = \text{Jari-jari roda (cm)}$$

Dari hasil perhitungan diatas dengan diketahui diameter roda *omni* sepanjang 100mm maka keliling roda *omni* yang didapat adalah sepanjang 31,42cm.



$$\text{Total pulse} \frac{\text{Pt}}{(\pi \cdot 2 \cdot r)} \cdot ppr \quad (3-17)$$

$$\text{Total pulse} \frac{\text{Pt}}{31,42} \cdot 356$$

Keterangan:

$$\pi = 3.14159265$$

$$r = \text{Jari-jari roda (cm)}$$

Pt = Panjang target tujuan (cm)

ppr = *Pulse per rotation* pada *rotary* (satuan) berdasarkan *rotary* yang digunakan pada tugas akhir ini adalah sebanyak 356 *pulse per rotation*

Dari perhitungan pada rumus 3-11 dapat diketahui hasil perhitungan target dari satuan cm ke total *pulse* yang diperlukan untuk mencapai target dari pembacaan *odometry* dengan.

3.1.5 Perhitungan RPM

Agar dapat mengetahui besar kecepatan masing-masing motor dalam satuan rotasi per menit maka gunakanlah perhitungan RPM persamaan 3-18 sebagai berikut:

$$RPM = \frac{R}{356} * 300$$

(3-18)

Keterangan:

RPM=Rotasi per menit

R=Nilai pulse *rotary encoder*

Konstanta 356= jumlah *pulse/rotation* dari *rotary encoder* yang digunakan

Konstanta 300= 60 detik * frekuensi *timer interrupt* (5Hz)

Berikut *listing* program perhitungan RPM beserta konfigurasi *timer interrupt* dengan interval 200ms:

```

int FREQ_1Hz = 5;

void startTimer(Tc *tc, uint32_t channel, IRQn_Type irq,
uint32_t frequency) {

    //Enable or disable write protect of PMC registers.
    pmc_set_writeprotect(false);
    //Enable the specified peripheral clock.
    pmc_enable_periph_clk((uint32_t)irq);

    TC_Configure(tc, channel,
    TC_CMR_WAVE|TC_CMR_WAVSEL_UP_RC|TC_CMR_TCCLKS_TIMER_CLOCK4);
    uint32_t rc = VARIANT_MCK/128/frequency;

    TC_SetRA(tc, channel, rc/2);
    TC_SetRC(tc, channel, rc);
    TC_Start(tc, channel);

    tc->TC_CHANNEL[channel].TC_IER = TC_IER_CPCS;
    tc->TC_CHANNEL[channel].TC_IDR = ~TC_IER_CPCS;
    NVIC_EnableIRQ(irq);
}

void TC3_Handler()
{
    rpma = (sa/356)*300;
    rpmb = (sb/356)*300;
    rpmc = (sc/356)*300;
    sa=0; sb=0; sc=0;
    Serial.print(int(rpma));
    Serial.print(" ");
    Serial.print(int(rpmb));
    Serial.print(" ");
    Serial.println(int(rpmc));
    TC_GetStatus(TC1, 0);
}

```

INSTITUT BISNIS
DAN INFORMATIKA

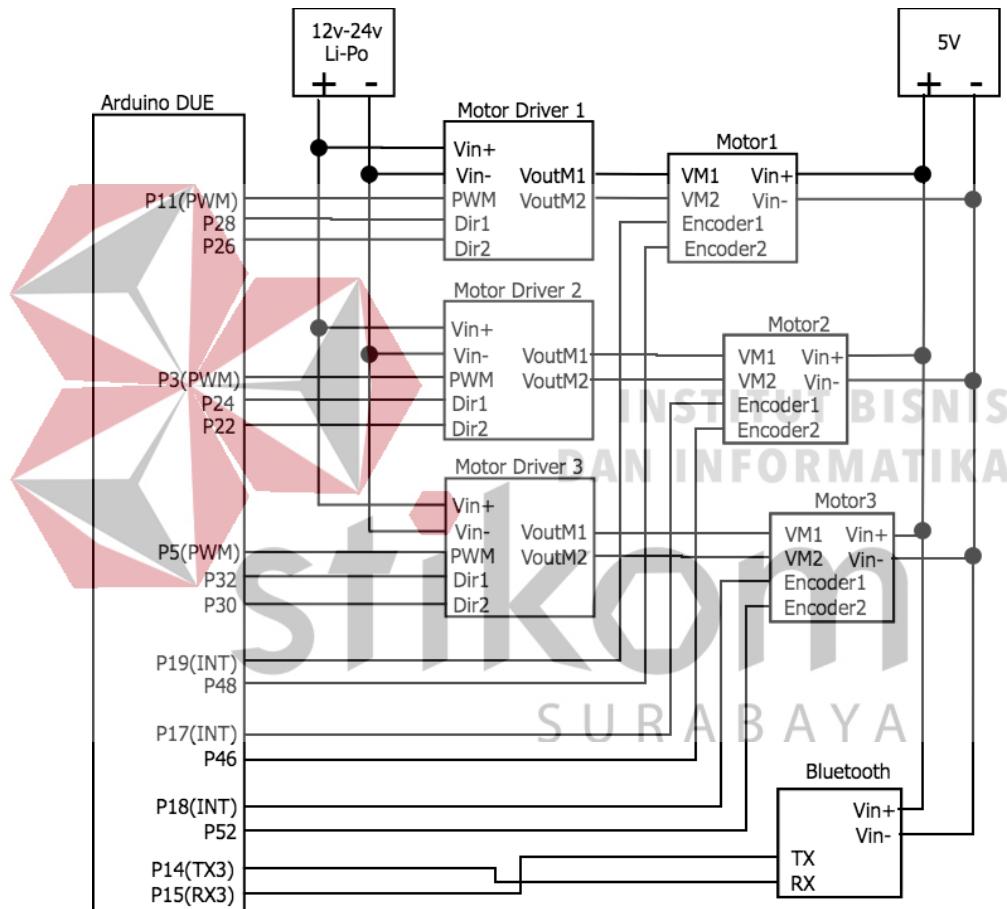


SURABAYA

3.2 Perancangan Elektronika Robot

Robot sepak bola pada Tugas Akhir ini menggunakan kontroler Arduino DUE yang sudah berbasis pada CPU Atmel SAM3X8E ARM Cortex-M3, sehingga sangat mumpuni dalam melakukan banyak multitasking sekaligus dengan kecepatan yang cukup tinggi. Arduino DUE digunakan pada Tugas Akhir ini dikarenakan adanya penggunaan *rotary encoder* yang mempunyai cara kerja dengan memanfaatkan pin *interrupt* pada kontroler untuk menghitung pulsa yang masuk sehingga dapat diketahui banyaknya putaran pada masing-masing *rotary*

encoder. Dengan menggunakan kontroler Arduino DUE ini pengguna tidak perlu khawatir untuk kekurangan pin dengan fungsi *interrupt*, dikarenakan setiap pin digital pada Arduino DUE dapat dilakukan *setting* untuk mode *interrupt*. Pada tugas akhir ini seluruh sistem pengendalian di kontrol melalui satu kontroler, atau tidak perlu dilakukan penambahan kontroler dengan mode *master/slave*.



Gambar 3. 8 Perancangan Elektronik Robot

Dapat dilihat pada gambar 3.8, sistem pada robot sepak bola ini mempunyai dua sumber tegangan yaitu sebesar 12/24V dan 5V, dimana sumber tegangan 12/24V digunakan sebagai sumber tegangan *input* motor driver untuk menggerakan aktuator berupa motor, dan untuk sumber tegangan 5V digunakan sebagai sumber

tegangan kontroler, motor *driver*, dan *rotary encoder*.

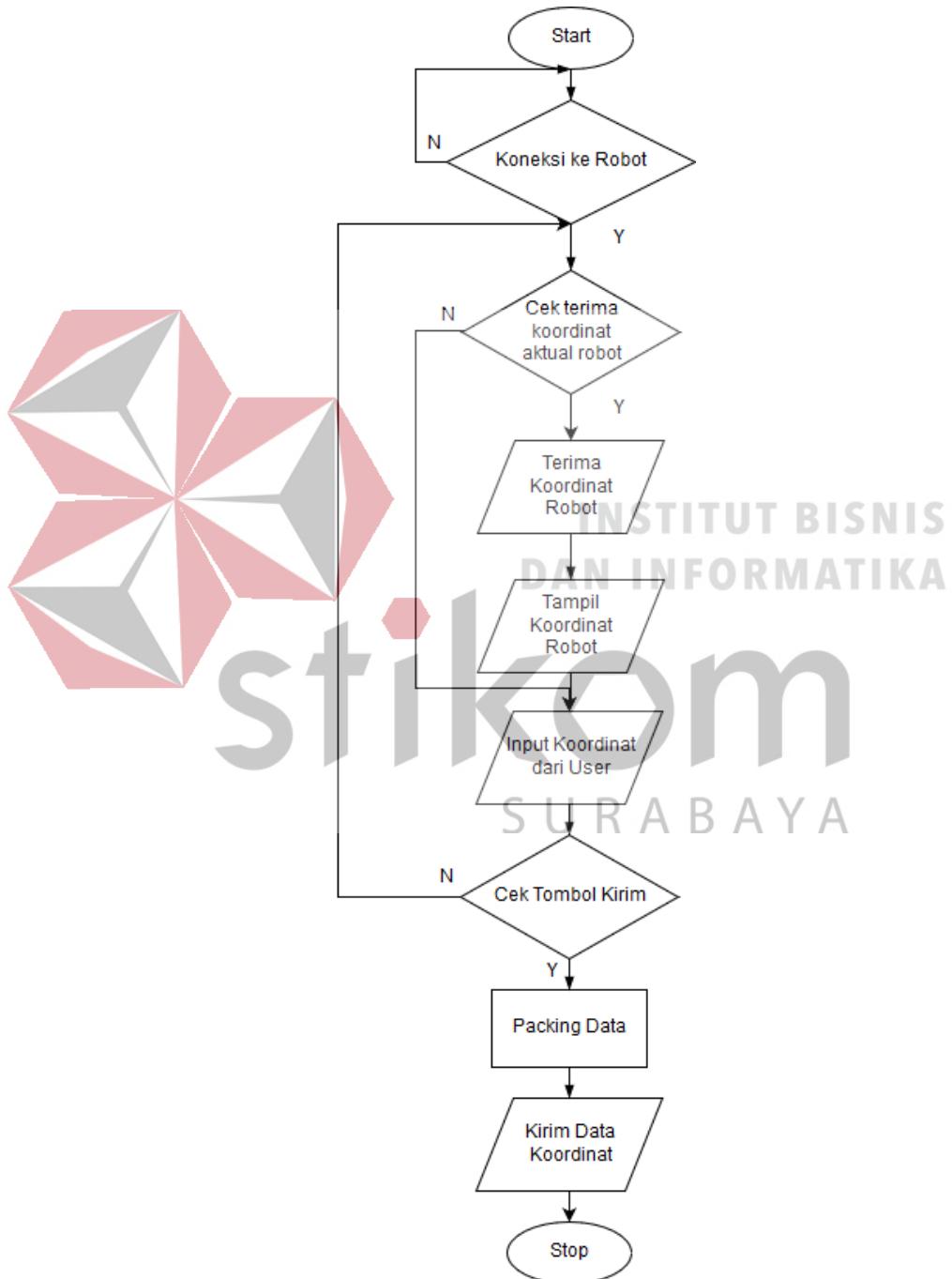
Motor *driver* yang digunakan adalah motor *driver* dengan kapasitas ampere yang cukup besar yaitu sebesar 30 ampere sesuai dengan tipenya EMS-30A, pada Tugas Akhir ini motor *driver* mengendalikan motor DC dengan sumber tegangan 12V yang berasal dari baterai *lithium polymer* 3 sel. Untuk mengendalikan motor driver ini diperlukan sebanyak 3 *masukkan* yaitu direksi 1, direksi 2, dan kontrol PWM. Direksi 1 dan direksi 2 digunakan sebagai menentukan arah putaran motor, saat diberi dengan kondisi (*high*) untuk direksi 1 dan (*low*) untuk direksi 2 maka motor akan berputar searah jarum jam dan berlaku juga sebaliknya, namun arah putaran motor juga dapat dipengaruhi dari *input* sumber tegangan motor yang dibalik untuk port VMotor pada motor driver. Untuk kontrol PWM dilakukan dengan memberikan *input* berupa pulsa dengan perbandingan *duty cycle* sesuai kebutuhan guna menentukan kecepatan putar motor DC.

Rotary encoder menggunakan rotary encoder internal yang ada pada motor DC. Dalam membaca arah putaran motor maka digunakanlah kedua *output* dari *rotary encoder* yaitu *output A* dan *output B*, dikarenakan kedua *output* tersebut mempunyai *output pulse* yang berdampingan. Salah satu *output* dari *rotary encoder* dihubungkan kepada pin kontroler yang telah diset menjadi pin *interrupt* untuk melakukan pembacaan putaran.

3.3 Perancangan Aplikasi *Mobile*

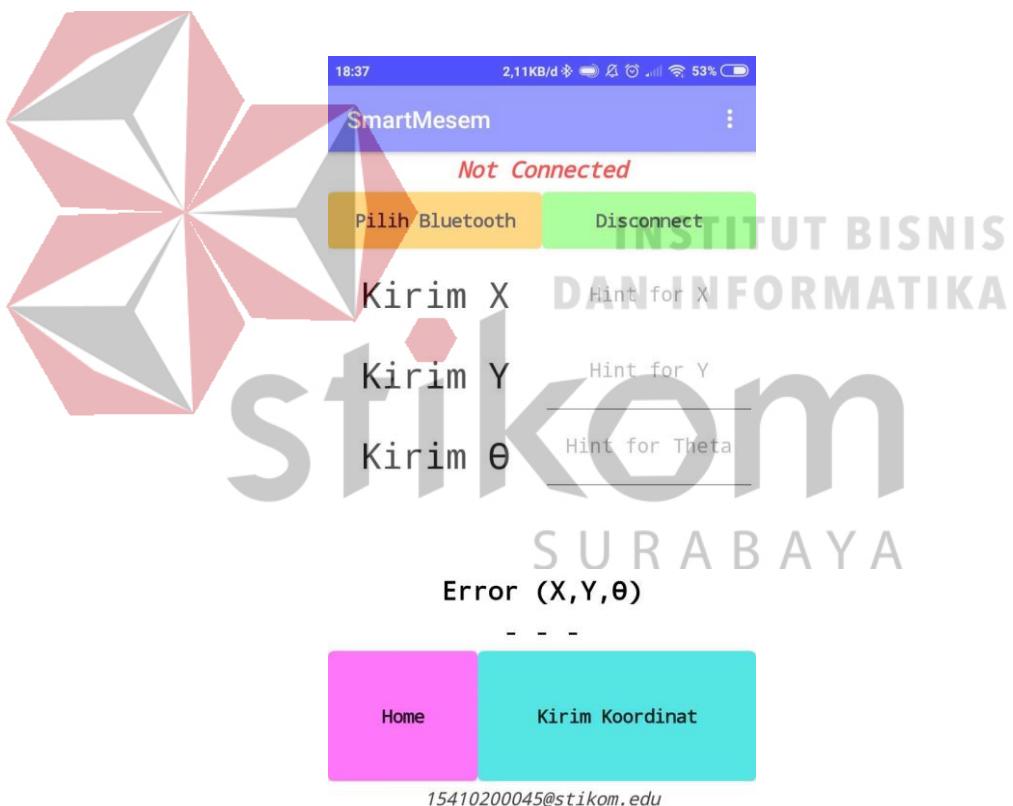
Untuk memudahkan pengguna dalam melakukan kontrol maupun *monitoring* terhadap *error* yang telah diberikan oleh robot, maka dibuatlah aplikasi *mobile* berbasis android untuk memberikan koordinat tujuan kepada robot sekaligus dapat

menampilkan nilai *error* dari robot saat melakukan pergerakan, transmisi data antara aplikasi dengan robot yaitu melalui koneksi bluetooth. Aplikasi ini dibuat menggunakan aplikasi berbasis web yaitu MIT App Inventor 2. Berikut adalah *flowchart* dari perancangan aplikasi ini:



Gambar 3. 9 *Flowchat* Perancangan Aplikasi *Mobile*

Saat pertama kali dijalankan sesuai *flowchart* pada gambar 3.9, pengguna harus melakukan koneksi terlebih dahulu untuk dapat terhubung dengan robot jika tidak maka kembali ke start. Saat sudah terkoneksi aplikasi dapat langsung menerima data koordinat dari robot serta menampilkannya dan melakukan cek *textbox* koordinat serta penekanan tombol “Kirim Koordinat”. Setelah koordinat terisi dan menekan tombol kirim maka data koordinat dibungkus dengan format yang telah ditentukan untuk dikirim menjadi satu paket data, yaitu dengan format ” $<X, Y, \theta>$ ”. Jika telah melakukan packing data maka data siap dikirimkan.



Gambar 3. 10 Desain Aplikasi

Gambar 3.10 adalah tampilan desain aplikasi untuk melakukan kontrol dan monitoring pada robot. Berikut adalah cara penggunaan aplikasi:

- Untuk melakukan kontrol, pengguna perlu menghubungkan koneksi terlebih dahulu pada robot dengan cara menekan tombol “Pilih Bluetooth” lalu pilih nama *bluetooth* sesuai dengan perangkat *bluetooth* yang ada pada robot yaitu “HC-05” berikut tampilan koneksi bluetooth pada gambar 3.11.



- Langkah selanjutnya yaitu memberikan *input* kepada masing-masing *texboxt* yang telah disediakan sesuai label yang ditentukan yaitu *input* koordinat x yang sejajar dengan label “Kirim X”, *input* koordinat y sejajar dengan label “Kirim Y”, dan *input* koordinat θ sejajar dengan label “Kirim θ”.
- Setelah itu tekan tombol “Kirim Koordinat” untuk mengirim koordinat yang telah dimasukkan pada robot.

4. Tombol “Home” berfungsi untuk mengisi seluruh *textbox* menjadi 0, atau mempersiapkan target koordinat menuju titik awal robot.

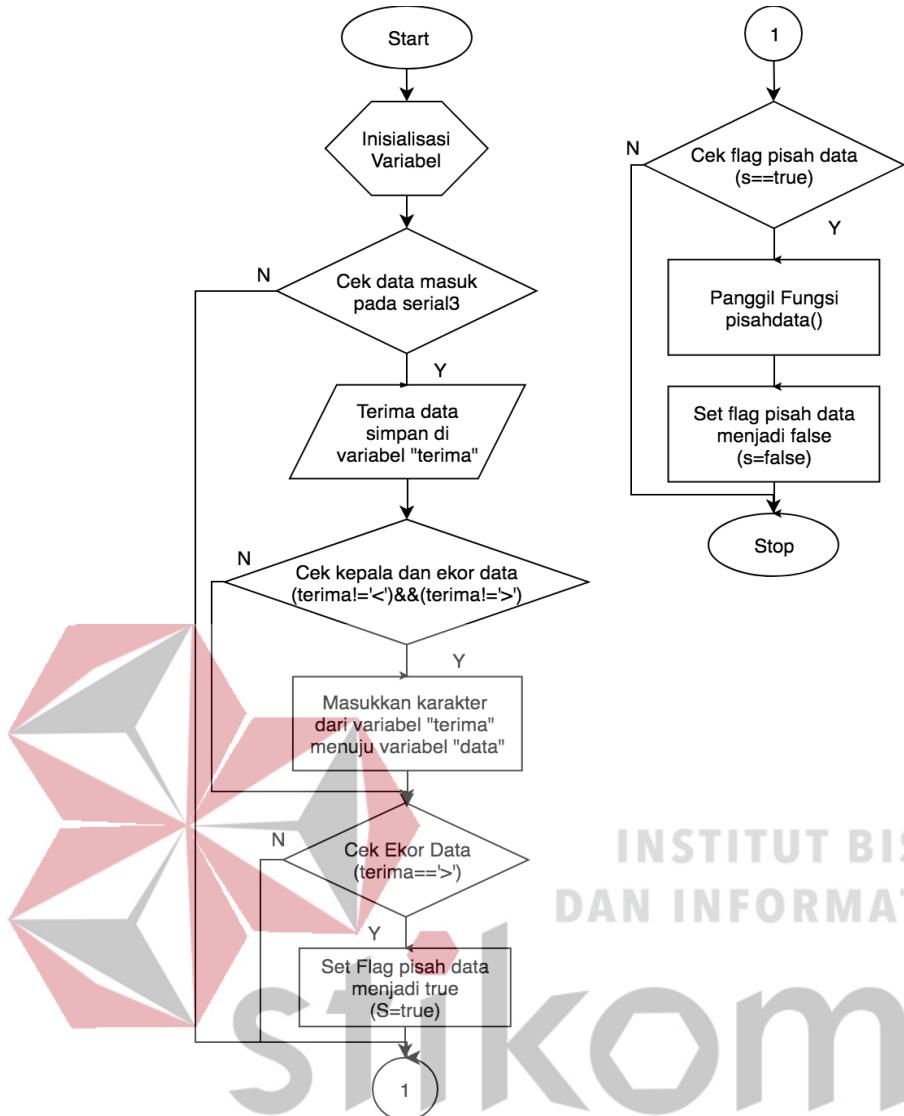
3.4 Perancangan Komunikasi Robot

Bagian ini membahas mengenai bagaimana robot menerima kemudian mempersiapkan data yang akan digunakan oleh sistem dan mengirim data untuk umpan balik bagi aplikasi *mobile*.

3.4.1. Terima Data Koordinat

Setalah aplikasi *mobile* mengirim data berupa koordinat tujuan untuk robot, maka perlu adanya proses penerimaan data dengan menggunakan komunikasi serial yang diterima dari bluetooth pada kontroler. Berikut adalah flowchart untuk melakukan penerimaan data koordinat pada gambar 3.12 di bawah ini:





Gambar 3. 12 Flowchart Penerimaan Data

Dan berikut adalah *listing* Program dari proses penerimaan data dapat dilihat di bawah ini:

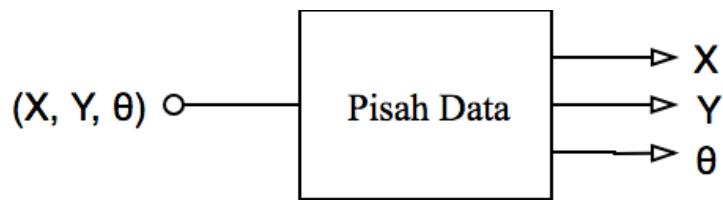
```

if(Serial3.available())
{
    terima = (char)Serial3.read();
    if((terima!='<')&&(terima!='>')) data+=terima;
    if(terima=='>')
        s=true;
}
if(s)
{
    pisahdata();
    s=false;
}

```

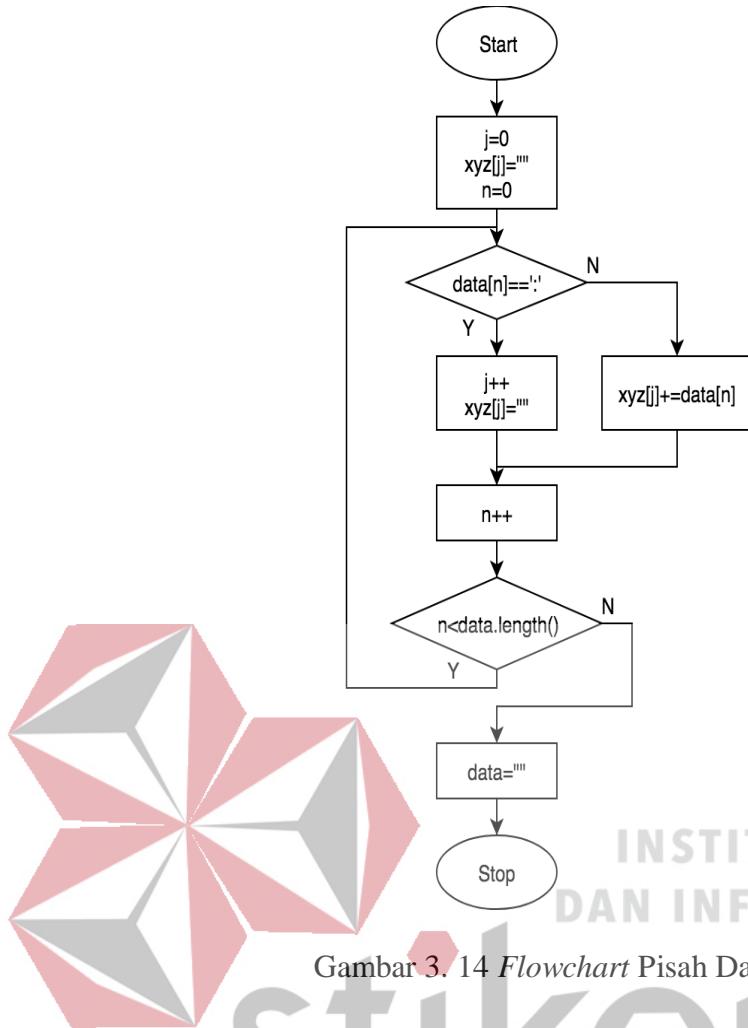
3.4.2. Pisah Data Koordinat

Setelah melakukan penerimaan data maka proses selanjutnya yang perlu dilakukan adalah pemisahan data.



Gambar 3. 13 Pisah Data

Dengan proses ini data yang awalnya masih berupa data string dan belum terkelompokkan menjadi satuan individu dapat diproses menjadi suatu data yang sudah siap untuk dilakukan proses perhitungan seperti pada gambar 3.13 sehingga mempermudah proses perhitungan pada sistem. Berikut adalah *flowchart* dari proses pemisahan data pada gambar 3.14 berikut ini:



Gambar 3. 14 Flowchart Pisah Data

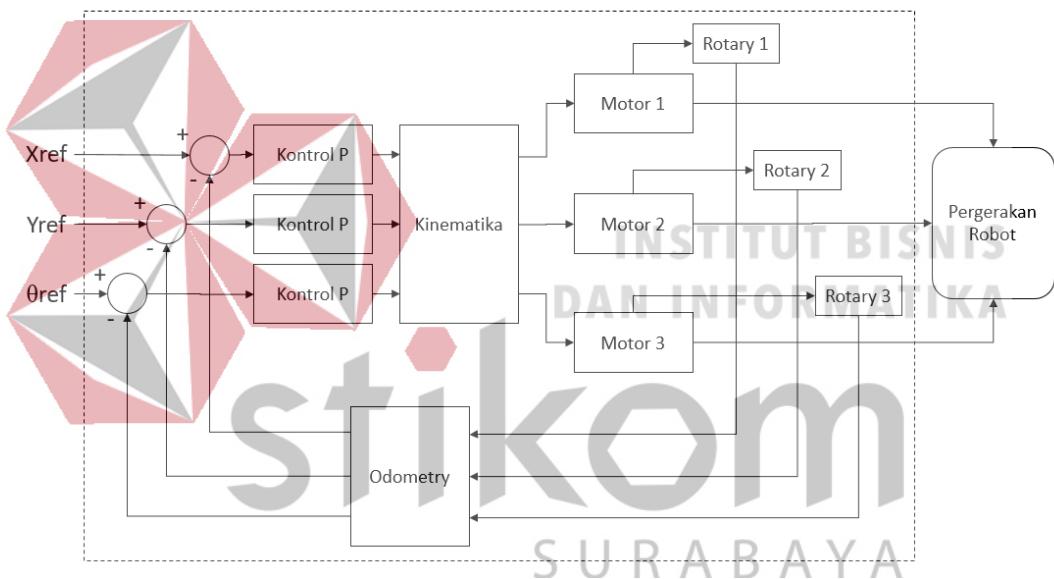
Berikut adalah *listing* Program dari proses pemisahan data:

```

void pisahdata()
{
    j=0;
    xyz[j]="";
    for(int n=0;n<data.length();n++)
    {
        if(data[n]==':')
        {
            j++;
            xyz[j]="";
        }
        else
        {
            xyz[j]+=data[n];
        }
    }
    data="";
}
  
```

3.5 Perancangan Kontrol Robot

Dalam perancangan kontrol untuk memungkinkan robot sepak bola beroda ini bisa melakukan lokalisasi dengan baik, maka ada beberapa metode yang digunakan dalam penggerjaan Tugas Akhir ini, beberapa metode tersebut adalah kontrol navigasi kinematika untuk robot *holonomic*, sistem umpan balik *odometry*, dan sistem pengaturan yang menggunakan kontrol proporsional. Berikut adalah blok diagram untuk perancangan seluruh sistem kontrol pada Tugas Akhir ini:



Gambar 3. 15 Blok Diagram Kontrol

Sesuai dengan blok diagram pada gambar 3.15 masukkan dari sistem kontrol di atas adalah berupa nilai koordinat referensi yaitu Xref, Yref, dan θRef koordinat inilah yang dituju oleh robot untuk melakukan pergerakan dalam lokalisasi. Titik koordinat ini nantinya ditetapkan oleh *odometry* untuk menjadi setpoint pergerakan, agar robot dapat melakukan pergerakan menuju titik tersebut.

Setelah *input* koordinat referensi diberikan, langkah selanjutnya *input*

tersebut diproses untuk ditetapkan sebagai masukkan kontrol kinematika, namun sebelum masuk kontrol kinematika terdapat kontrol proporsional pada sistem di atas yang berguna untuk mempercepat kontrol sistem menuju ke setpoint yang telah ditentukan oleh koordinat referensi. Berikut adalah perumusan kontrol proporsional pada sistem dengan rumus 3-19 di bawah ini:

$$p = K_p * (Setpoint - Aktual) \quad (3-19)$$

Keterangan:

p = kontrol proporsional.

K_p = konstanta untuk pengali *error*.

Setpoint = nilai yang ditentukan.

Aktual = nilai saat ini.

Dengan menggunakan rumus kontrol proporsional sistem akan menjadi lebih stabil untuk meminimalisir *error* yang disebabkan.

Listing program lokalisasi beserta kontrol proporsional adalah sebagai berikut:

```

int Target(int max_speed, int x_target, int y_target, int
heading_target)
{
    odometry();
    error_x = x_target - x;
    p_x = (int)(kp*error_x);
    if(p_x > max_speed) p_x = max_speed;
    else if(p_x < -max_speed) p_x = -max_speed;

    error_y = y_target - y;
    p_y = (int)(kp*error_y);
    if(p_y > max_speed) p_y = max_speed;
    else if(p_y < -max_speed) p_y = -max_speed;

    error_heading = heading_target - heading;
    p_heading = (int)(kp*error_heading);
    if(p_heading > max_speed) p_heading = max_speed;
    else if(p_heading < -max_speed) p_heading = -max_speed;

    kinematika(p_x, p_y, p_heading);
    delay(Ts);
    if(error_x<15 && error_x>-15 && error_y<15 && error_y>-15
    && error_heading<15 && error_heading>-15) done++;
    if(done == 0)
    {
        digitalWrite(buzz, ledon = !ledon);
        return 1;
    }
    else {
        digitalWrite(buzz, 0);
        kinematika(0,0,0);
        return 0;
    }
}

```

Untuk mengatasi *error steady state* pada koreksi error *odometry* dalam mencapai target koordinat digunakanlah konstanta toleransi koreksi *error* sebesar 15 (*pulse*) agar dapat mengatasi keadaan *error* stabil. Angka tersebut didapat setelah melakukan pengujian berdasarkan error koordinat terkecil (<1cm) dengan tanpa terjadi *error steady state*. Berikut *listing* program untuk membatasi toleransi koreksi *error odometry*:

```

if(error_x<15 && error_x>-15 && error_y<15 && error_y>-15 &&
error_heading<15 && error_heading>-15)

```

Perhitungan *odometry* mempunyai *input* berasal dari kombinasi ketiga rotary internal yang sudah terdapat pada motor DC, sistem umpan balik *odometry*

ini selalu melakukan koreksi error terhadap posisi aktual dengan posisi setpoint yang ditetapkan meskipun dalam kondisi tidak menerima perintah untuk melakukan pergerakan. Oleh karena itu dalam keadaan diam robot selalu melakukan koreksi posisi aktual terhadap *setpoint*, sehingga saat mendapat gangguan pergerakan dari luar yang secara paksa robot dapat kembali ke posisi terakhir perintah diberikan.

Kontrol navigasi kinematika pada sistem di atas digunakan untuk membagi proporsi kecapatan masing masing motor, sehingga dapat melakukan kontrol motor secara matematis saat menuju titik tertentu sesuai dengan tujuan koordinat. Kontrol kinematika berjalan berdasarkan *input* dari *error odometry* yang sudah melalui kontrol proporsional sehingga dapat disimpulkan bahwa *input* antar *odometry* dan kinematika hampir sama namun yang membuat berbeda pada *input* kinematika membunyai batas maksimal *input* atau dalam program yang dibuat bernama variabel “max_speed” variabel inilah yang mempengaruhi kecepatan robot. Berikut algoritma *input* kinematika beserta kontrol proporsional dari *listing* program:

```

error_x = x_target - x;
p_x = (int)(kp*error_x);
if(p_x > max_speed) p_x = max_speed;
else if(p_x < -max_speed) p_x = -max_speed;

error_y = y_target - y;
p_y = (int)(kp*error_y);
if(p_y > max_speed) p_y = max_speed;
else if(p_y < -max_speed) p_y = -max_speed;

error_heading = heading_target - heading;
p_heading = (int)(kp*error_heading);
if(p_heading > max_speed) p_heading = max_speed;
else if(p_heading < -max_speed) p_heading = -max_speed;

kinematika(p_x, p_y, p_heading);

```

Berdasarkan hasil pengujian dalam mengurangi *error steady state* saat mencapai koordinat target, nilai konstanta proporsional yang digunakan adalah sebesar (1,5). Diberikan nilai konstanta (>1) guna sedikit mempercepat respon menuju setpoint.

BAB IV

PENGUJIAN

Dalam bab ini akan dibahas tentang beberapa pengujian untuk sistem yang telah dirancang pada Tugas Akhir ini. Tujuan dari bab ini adalah untuk mengetahui tingkat keberhasilan terhadap perancangan sistem yang telah diajukan dan dikerjakan. Pada bab ini terdapat beberapa pengujian yaitu uji kecepatan putar motor, uji koordinat rotary encoder, dan uji pergerakan robot.

4.1 Uji Kecepatan Putar Motor

4.1.1 Tujuan Uji Kecepatan Putar Motor

Pengujian ini digunakan untuk mengetahui perbedaan kecepatan masing-masing motor yang digunakan dengan cara memberikan *input* kecepatan berupa PWM pada motor dan mengamati respon *output* kecepatan pada masing-masing motor (rpm). Dengan diketahuinya perbedaan respon kecepatan, nilai tersebut dapat ditentukan sebagai nilai untuk mensinkronkan kecepatan seluruh motor.

4.1.2 Alat Yang Digunakan Pada Uji Kecepatan Putar Motor

Peralatan yang dibutuhkan untuk pengujian ini adalah sebagai berikut:

1. Robot Sepak Bola
2. Laptop
3. Program Arduino untuk menampilkan nilai RPM
4. Penyangga Robot

4.1.3 Prosedur Pengujian Pada Uji Kecepatan Putar Motor

Langkah-langkah yang dilakukan dalam melakukan pengujian ini adalah sebagai berikut:

1. Mempersiapkan Robot Sepak Bola.
2. Melakukan transfer program dari laptop ke robot.
3. Menjalankan robot di atas penyangga robot yang telah disediakan.
4. Amati nilai PWM dan RPM masing-masing motor pada *Serial Monitor* yang disediakan oleh Arduino IDE.

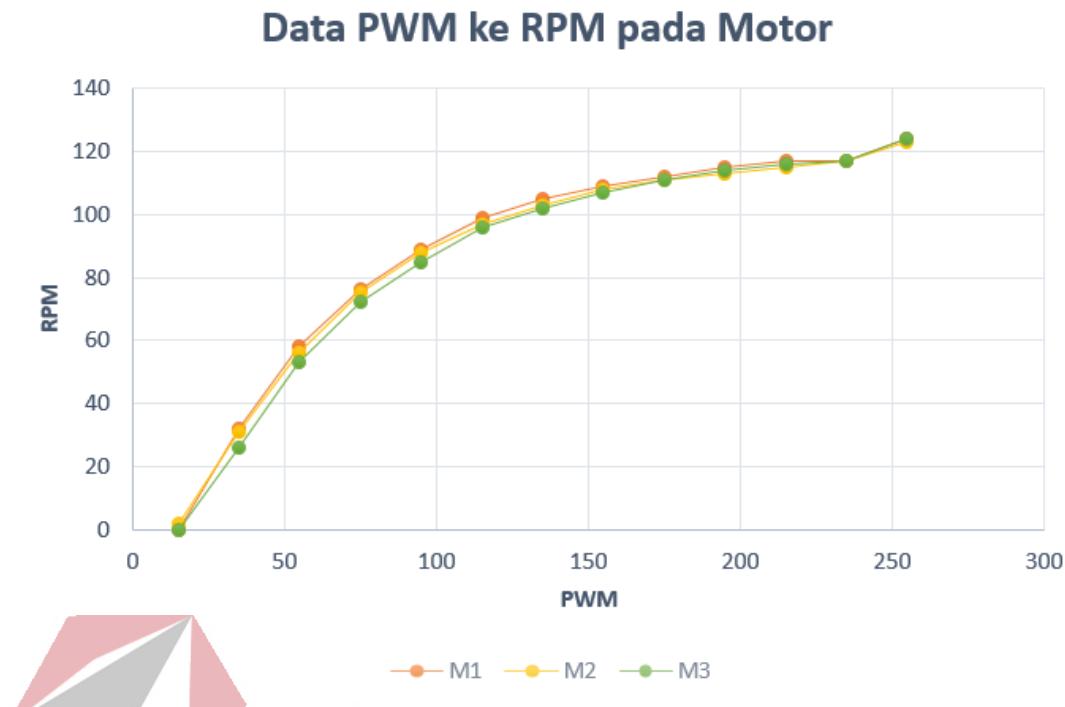
4.1.4 Hasil Pengujian Pada Uji Kecepatan Putar Motor

Berikut adalah tabel 4.1 hasil pengujian kecepatan masing-masing motor:

Tabel 4. 1 Uji Kecepatan Putar Motor

No	PWM	M1 (RPM)	M2 (RPM)	M3 (RPM)
1	255	124	123	124
2	235	117	117	117
3	215	117	115	116
4	195	115	113	114
5	175	112	111	111
6	155	109	108	107
7	135	105	103	102
8	115	99	97	96
9	95	89	88	85
10	75	76	75	72
11	55	58	56	53
12	35	32	31	26
13	15	0	2	0

Berikut adalah data pengujian dalam bentuk grafik pada gambar 4.1:



Gambar 4. 1 Grafik PWM dan RPM Motor

Hasil dari proses pengujian kecepatan masing-masing motor menunjukkan bahwa motor yang terpasang pada robot sepak bola beroda dalam Tugas Akhir ini mempunyai perbedaan kecepatan yang tidak signifikan. Seperti yang terlihat bahwa pada saat diberi *input* PWM yang tinggi kecepatan pada Motor 2 relatif lebih pelan dari motor lainnya, sedangkan pada saat diberi *input* PWM yang rendah Motor 3 yang berkecepatan lebih pelan dari motor lainnya.

4.2 Uji Koordinat *Rotary Encoder*

4.2.1 Tujuan Uji Koordinat *Rotary Encoder*

Pada tahap ini pengujian digunakan untuk mengetahui nilai pembacaan dari *rotary encoder* pada sistem tanpa memperhatikan pengukuran aktual.

4.2.2 Alat Yang Digunakan Pada Uji Koordinat *Rotary Encoder*

Peralatan yang dibutuhkan untuk pengujian ini adalah sebagai berikut:

1. Robot Sepak Bola
2. Laptop
3. *Smartphone* Android
4. Aplikasi Kontrol Robot

4.2.3 Prosedur Pengujian Pada Uji Koordinat *Rotary Encoder*

Langkah-langkah yang dilakukan dalam melakukan pengujian ini adalah sebagai berikut:

1. Mempersiapkan aplikasi Kontrol Robot pada *Smartphone*.
2. Mengkondisikan robot pada titik awal.
3. Menghubungkan koneksi antara robot dengan aplikasi pada *smartphone* menggunakan *bluetooth*.
4. Menjalankan aplikasi untuk mengkontrol robot dengan cara memberikan koordinat tujuan robot.
5. Pengujian dilakukan dengan cara memberikan target koordinat tunggal atau terarah yaitu robot bergerak maju, bergerak mundur, bergerak ke samping kiri, bergerak ke samping kanan, dan rotasi melawan arah jarum jam. Pengujian dilakukan dengan panjang target tujuan antara 30cm hingga 300cm.
6. Amati hasil *error* yang dapat dilihat pada aplikasi Kontrol Robot pada *smartphone*.

4.2.4 Hasil Pengujian Dari Uji Koordinat *Rotary Encoder*

Pada tabel 4.2 berikut adalah pengujian koordinat *rotary encoder* dengan arah gerak maju:

Tabel 4. 2 Pengujian Koordinat Arah Gerak Maju

No	TARGET		TERUKUR		ERROR	
	X	Y	X	Y	X	Y
1	0	30	-0,24	30,82	0,24	-0,82
2	0	60	-0,35	60,59	0,35	-0,59
3	0	90	-0,18	90,65	0,18	-0,65
4	0	120	-0,24	120,53	0,24	-0,53
5	0	150	-0,29	150,82	0,29	-0,82
6	0	180	-0,12	180,71	0,12	-0,71
7	0	210	-0,29	210,71	0,29	-0,71
8	0	240	-0,29	240,59	0,29	-0,59
9	0	270	-0,18	270,82	0,18	-0,82
10	0	300	-0,29	299,18	0,29	0,82
				Rata-rata	0,247	0,706

Pada tabel 4.3 berikut adalah pengujian koordinat *rotary encoder* dengan arah gerak mundur:

Tabel 4. 3 Pengujian Koordinat Arah Gerak Mundur

NO	TARGET		TERUKUR		ERROR	
	X	Y	X	Y	X	Y
1	0	-30	0	-30,53	0	0,53
2	0	-60	-0,06	-60,71	0,06	0,71
3	0	-90	0	-90,88	0	0,88
4	0	-120	0,06	-120,82	-0,06	0,82
5	0	-150	-0,06	-150,88	0,06	0,88
6	0	-180	0,12	-180,88	-0,12	0,88
7	0	-210	-0,06	-210,76	0,06	0,76
8	0	-240	-0,06	-240,88	0,06	0,88
9	0	-270	0	-270,59	0	0,59
10	0	-300	0,12	-300,76	-0,12	0,76
				Rata-rata	0,05	0,77

Pada tabel 4.4 berikut adalah pengujian koordinat *rotary encoder* dengan arah gerak

ke samping kiri:

Tabel 4. 4 Pengujian Koordinat Arah Gerak Ke Samping Kiri.

No	TARGET		TERUKUR		ERROR	
	X	Y	X	Y	X	Y
1	30	0	30,35	0	-0,35	0
2	60	0	60,65	0	-0,65	0
3	90	0	90,65	0	-0,65	0
4	120	0	120,88	-0,06	-0,88	0,06
5	150	0	150,76	-0,06	-0,76	0,06
6	180	0	180,76	0	-0,76	0
7	210	0	210,65	0,18	-0,65	-0,18
8	240	0	240,65	-0,18	-0,65	0,18
9	270	0	270,71	0,12	-0,71	-0,12
10	300	0	300,76	0	-0,76	0
					Rata-rata	0,682
						0,06

Pada tabel 4.5 berikut adalah pengujian koordinat *rotary encoder* dengan arah gerak ke samping kanan:

Tabel 4. 5 Pengujian Koordinat Arah Gerak Ke Samping Kanan.

No	TARGET		TERUKUR		ERROR	
	X	Y	X	Y	X	Y
1	-30	0	-30,41	-0,06	0,41	0,06
2	-60	0	-60,71	0,18	0,71	-0,18
3	-90	0	-90,65	0	0,65	0
4	-120	0	-120,82	0,12	0,82	-0,12
5	-150	0	-150,82	0,12	0,82	-0,12
6	-180	0	-180,76	-0,06	0,76	0,06
7	-210	0	-210,29	0	0,29	0
8	-240	0	-240,35	-0,06	0,35	0,06
9	-270	0	-270,47	0,18	0,47	-0,18
10	-300	0	-300,71	0	0,71	0
					Rata-rata	0,599
						0,078

Pada tabel 4.6 berikut adalah pengujian koordinat *rotary encoder* dengan rotasi melawan arah jarum jam:

Tabel 4. 6 Pengujian Koordinat Rotasi Melawan Arah Jarum Jam.

TARGET	TERUKUR	ERROR
0	0	0
45	45	0
90	90	0
135	135	0
180	180	0
225	225	0
270	270	0
315	315	0
360	360	0
Rata-rata		0

4.3 Uji Pergerakan Robot

4.3.1 Tujuan Uji Pergerakan Robot

Pada pengujian pergerakan robot ini digunakan untuk mengetahui ketepatan robot dalam melakukan perpindahan berdasarkan koordinat yang ditetapkan. pengujian pergerakan robot ini dilakukan dengan cara membandingkan pengukuran yang terukur oleh sistem dan pengukuran sebenarnya secara manual.

4.3.2 Alat Yang Digunakan Pada Uji Pergerakan Robot

Peralatan yang dibutuhkan untuk pengujian ini adalah sebagai berikut:

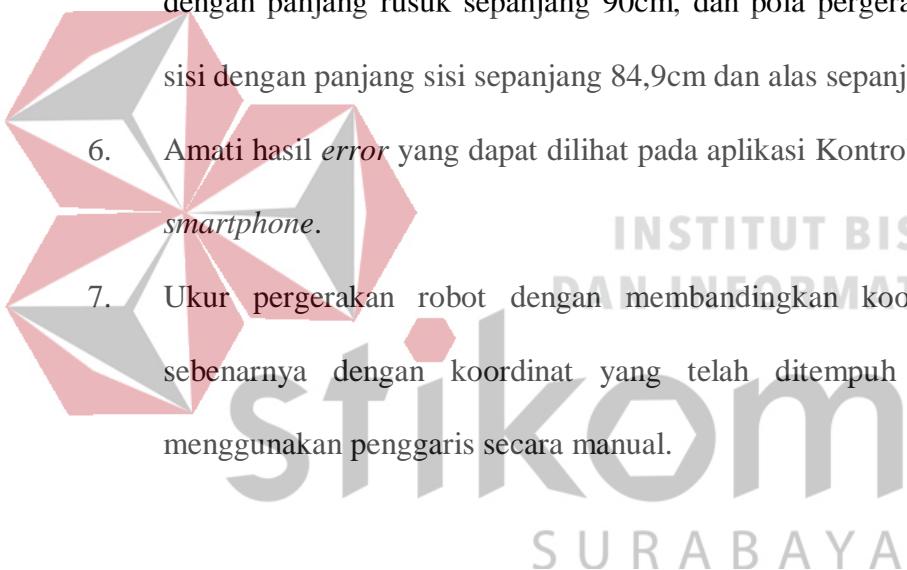
1. Robot Sepak Bola
2. Laptop
3. *Smartphone* Android
4. Aplikasi Kontrol Robot
5. Alat ukur panjang (penggaris)

4.3.3 Prosedur Pengujian Pada Uji Pergerakan Robot

Langkah-langkah yang dilakukan dalam melakukan pengujian ini adalah sebagai

berikut:

1. Mempersiapkan aplikasi Kontrol Robot pada *Smartphone*.
2. Mengkondisikan robot pada titik awal.
3. Menghubungkan koneksi antara robot dengan aplikasi pada *smartphone* menggunakan *bluetooth*.
4. Menjalankan aplikasi untuk mengkontrol robot dengan cara memberikan koordinat tujuan robot.
5. Terdapat dua pola pada pengujian ini yaitu pola pergerakan persegi dengan panjang rusuk sepanjang 90cm, dan pola pergerakan segitiga sisi dengan panjang sisi sepanjang 84,9cm dan alas sepanjang 120cm.
6. Amati hasil *error* yang dapat dilihat pada aplikasi Kontrol Robot pada *smartphone*.
7. Ukur pergerakan robot dengan membandingkan koordinat yang sebenarnya dengan koordinat yang telah ditempuh oleh robot menggunakan penggaris secara manual.



4.3.4 Hasil Pengujian Pada Uji Pergerakan Robot

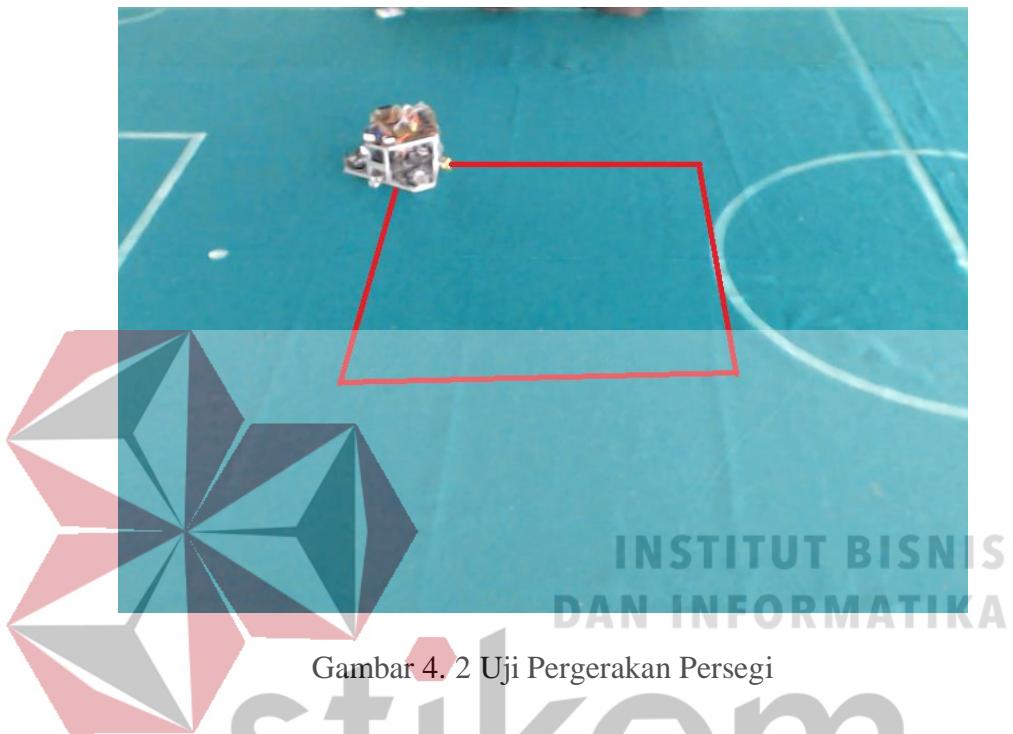
Sebelum melakukan uji pergerakan robot, pada tabel 4.7 berikut adalah pengujian kecepatan proporsi motor yang telah diberikan kinematika untuk mendistribusi kecapatan ke masing-masing motor saat diberi masukkan PWM dengan *duty cycle* sebesar 15,6% :

Tabel 4. 7 Pengujian Kecepatan Proporsi Motor

V-ROBOT	PWM1	PWM2	PWM3	V-M1	V-M2	V-M3
VX	20	40	-20	67	115	68
VY	34	0	34	114	0	115
Vθ	40	40	40	119	118	118

a. Uji Pergerakan Persegi

Berikut hasil pengujian pergerakan robot dengan pola gerak persegi dengan arah hadap tetap 0° dapat dilihat pada gambar 4.2 di bawah ini:



Pada tabel 4.8 di bawah ini menampilkan perbandingan angka koordinat target dengan pola gerak persegi yang diberikan pada robot dengan *error* yang terukur oleh sistem:

Tabel 4. 8 Perbandingan Target dan *Error* Terukur Persegi

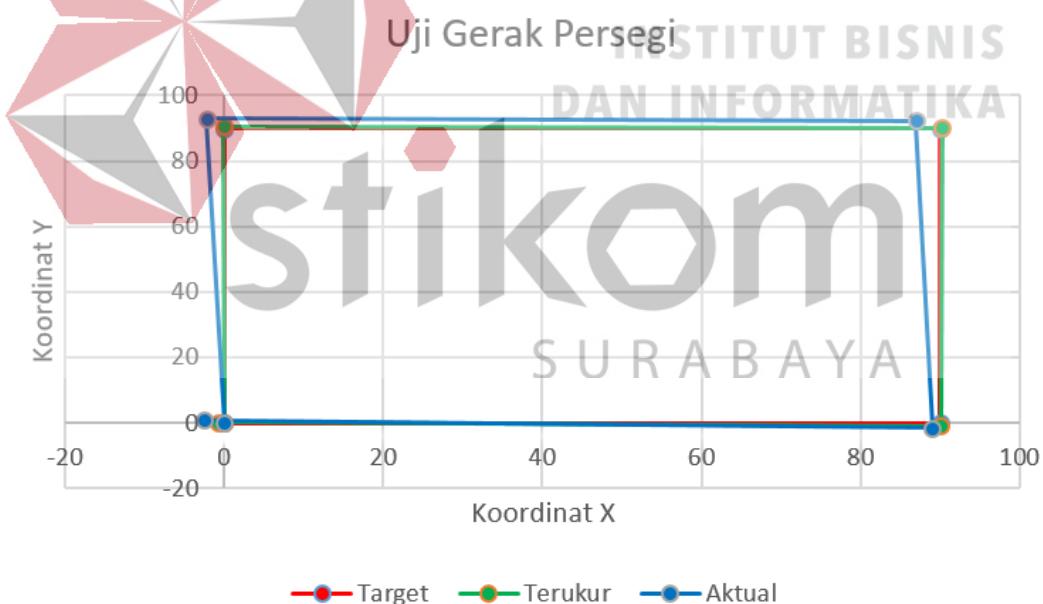
TARGET		TERUKUR		ERROR TERUKUR	
X	Y	X	Y	X	Y
0	90	-0,06	90,53	0,06	-0,53
90	90	90,24	90,12	-0,24	-0,12
90	0	90,12	-0,76	-0,12	0,76
0	0	-0,76	0,24	0,76	-0,24
		Rata-rata	0,295	0,413	

Tabel 4.9 berikut menampilkan perbandingan angka koordinat target dengan pola gerak persegi yang diberikan pada robot dengan pengukuran yang dilakukan secara manual untuk mengetahui *error* jarak yang sebenarnya atau *error* aktual:

Tabel 4. 9 Perbandingan Target dan Aktual Persegi

TARGET		AKTUAL		ERROR AKTUAL	
X	Y	X	Y	X	Y
0	90	-2,2	93	2,2	-3
90	90	87	92,2	3	-2,2
90	0	89	-1,5	1	1,5
0	0	-2,5	1	2,5	-1
		Rata-rata	2,175	1,925	

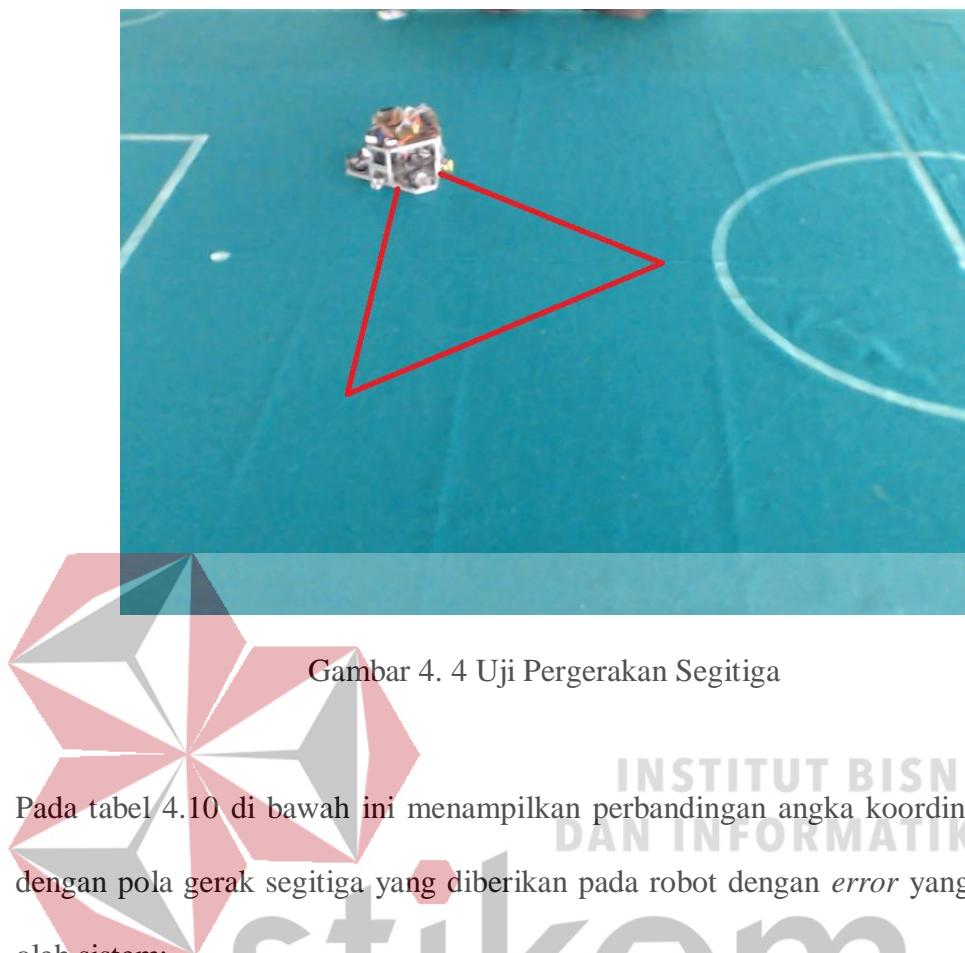
Pada gambar 4.3 menampilkan data pengujian pergerakan robot dengan pola gerak persegi dalam bentuk grafik:



Gambar 4. 3 Grafik Uji Gerak Persegi

b. Uji Pergerakan Segitiga

Berikut pengujian pergerakan robot dengan pola gerak segitiga dengan arah hadap tetap 0° dapat dilihat pada gambar 4.4:



Tabel 4. 10 Perbandingan Target dan *Error* Terukur Segitiga

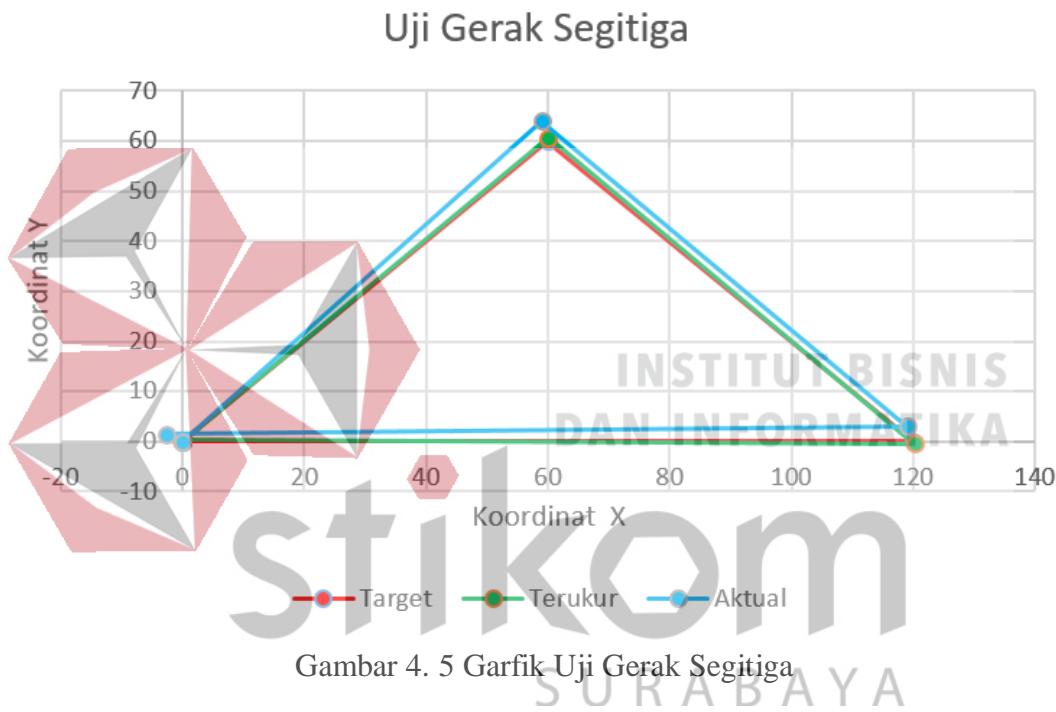
TARGET		TERUKUR		ERROR TERUKUR	
X	Y	X	Y	X	Y
60	60	60,06	60,65	-0,06	-0,65
120	0	120,18	-0,41	-0,18	0,41
0	0	-0,24	0,24	0,24	-0,24
		Rata-rata		0,16	0,433

Tabel 4.11 berikut menampilkan perbandingan angka koordinat target dengan pola gerak segitiga yang diberikan pada robot dengan pengukuran yang dilakukan secara manual untuk mengetahui *error* jarak yang sebenarnya atau *error* aktual :

Tabel 4. 11 Perbandingan Target dan Aktual Segitiga

TARGET		AKTUAL		ERROR AKTUAL	
X	Y	X	Y	X	Y
60	60	59	64	1	-4
120	0	119	3	1	-3
0	0	-2,5	1,5	2,5	-1,5
		Rata-rata		1,5	2,833

Pada gambar 4.5 menampilkan data pengujian pergerakan robot dengan pola gerak segitiga dalam bentuk grafik:



4.4 Analisis Keseluruhan Pengujian Yang Telah Dilakukan

Setelah dilakukannya beberapa pengujian diatas yaitu uji kecepatan motor, uji koordinat *rotary encoder*, dan uji pergerakkan robot. Analisa yang diperoleh dari hasil uji kecepatan motor menunjukkan bahwa masing-masing motor mempunyai perbedaan kecepatan yang tidak signifikan. Analisis yang diperoleh dari uji koordinat *rotary encoder* dengan uji pergerakkan robot menunjukkan bahwa *error* yang tidak diketahui sistem oleh pembacaan *rotary internal* cukuplah

besar setelah dilakukan perbandingan dengan pengukuran pergerakan robot yang sebenarnya dengan *error* yang diketahui oleh sistem.



BAB V

PENUTUP

Dalam bab ini berisi tentang kesimpulan dan saran yang berdasar pada hasil dari pengujian yang telah dilakukan pada Tugas Akhir ini, maka dapat diperoleh beberapa kesimpulan dan saran untuk pengembangan sistem berikutnya.

5.1 Kesimpulan

Berdasar hasil pengujian pada Tugas Akhir ini dapat disimpulkan beberapa kesimpulan sebagai berikut :

1. Hasil dari proses pengujian kecepatan masing-masing motor menunjukkan bahwa motor yang digunakan mempunyai perbedaan kecepatan yang tidak signifikan.
2. Hasil dari pengujian pembacaan koordinat oleh *rotary encoder* pada sistem tanpa memperhatikan pengukuran sebenarnya yaitu pada arah gerak maju memiliki rata-rata *error* sebesar 0,48 cm, pada arah gerak mundur memiliki rata-rata *error* sebesar 0,41 cm, pada arah gerak ke kanan memiliki rata-rata *error* sebesar 0,34 cm, pada arah gerak ke kiri memiliki rata-rata *error* sebesar 0,37 cm, dan pada gerak rotasi memiliki rata-rata *error* sebesar 0 cm
3. Hasil dari pengujian pergerakkan robot dengan perbandingan pengukuran yang terukur oleh sistem dan pengukuran sebenarnya secara manual yaitu pada uji pergerakkan dengan pola gerak persegi memiliki rata-rata *error* sebesar 2,05 cm, dan pada uji pergerakkan dengan pola gerak segitiga memiliki rata-rata *error* sebesar 2,2 cm.

5.2 Saran

Saran yang diberikan oleh penulis pada pengembangan Tugas Akhir ini selanjutnya adalah sebagai berikut:

1. Penggunaan kontrol PID atau *Fuzzy* untuk masing-masing motor agar dapat mengatasi perbedaan kecepatan pada motor yang tidak signifikan.
2. Penggunaan sensor *gyro* dalam sistem *odometry* sehingga robot selalu dapat mengukur arah hadapnya.
3. Penggunaan *rotary* eksternal agar dapat mengatasi *error* yang terjadi akibat selip pada roda.



DAFTAR PUSTAKA

Basori, S., Sulistiyanto, N., & Rif'an, M. (2014). IMPLEMENTASI ODOMETRY PADA ROBOT OTOMATIS KONTES ROBOT ABU INDONESIA.

Jurusan Teknik Elektro Fakultas Teknik Universitas Brawijaya.

Doroftei, I., Grosu, V., & Spinu, V. (2007). Omnidirectional Mobile Robot – Design and Implementation. *Gh. Asachi Technical University of Iasi Romania.*

Inthiam, J., & Deelertpiboon, C. (2014). Self-Localization and Navigation of Holonomic Mobile Robot using Omni-Directional Wheel. *King Mongkut's University of Technology*, 3.

Kariyanto, J. D., Alasiry, A. H., Ardila, F., & Hanafi, N. (2012). Navigasi Mobile Robot Berbasis Trajektori dan Odometry dengan Pemulihan Jalur Secara Otomatis. *Politeknik Elektronika Negeri Surabaya Electronics Engineering Polytechnic Institute of Surabaya Institut Teknologi Sepuluh Nopember Surabaya Kampus ITS Sukolilo*, 1-8.

Marta, B. S., Ardilla, F., & Besari, A. (2011). Path Tracking Pada Mobile Robot Dengan Umpan Balik Odometry. *Politeknik Elektronika Negeri Surabaya Kampus PENS-ITS Sukolilo, Surabaya*, 1-7.

Rochmanto, & Artha, R. (2014). IMPLEMENTASI ROBOT THREE OMNI-DIRECTIONAL MENGGUNAKAN KONTROLER PID PADA ROBOT KONTES ROBOT ABU INDONESIA (KRAI). *JURUSAN TEKNIK ELEKTRO FAKULTAS TEKNIK UNIVERSITAS BRAWIJAYA*.

- Rojas, R., & Förster, A. G. (2006). Holonomic Control of a robot with an omnidirectional drive. *o appear in KI - Künstliche Intelligenz, BöttcherIT Verlag*, 3.
- Setiawan, S., Firdaus, Rahmadya, B., & Derisma. (2015). PENERAPAN INVERS KINEMATIKA UNTUK PERGERAKAN KAKI ROBOT BIPED. *Fakultas Teknik Universitas Muhammadiyah Jakarta*.
- Eda, "Spesifikasi Arduino Due" 2017. [Online]. Available: <http://www.eda-channel.com/2017/11/spesifikasi-arduino-due.html>. [Accessed 10 October 2018].
- Christianto Tjahyadi, "DC-DC Converter" 2018. [Online]. Available: <http://christianto.tjahyadi.com/elektronika/ubec.html>. [Accessed 11 December 2018].
- Firman, "Tentang PWM (Pulse Width Modulation)" 2017. [Online]. Available: <http://kl301.ilearning.me/2015/05/19/tentang-pwm-pulse-width-modulation.html>. [Accessed 12 December 2018].
- Nedelkovski Dejan, "How Rotary Encoder Works and How To Use It with Arduino" 2016. [Online]. Available: <https://howtomechatronics.com/tutorials/arduino/rotary-encoder-works-use-arduino.html>. [Accessed 15 November 2018].
- Thingbits, "3WD 48mm Omni-Directional Triangle Mobile Robot Chassis" 2018. [Online]. Available: <https://www.thingbits.net/products/3wd-48mm-omni-directional-triangle-mobile-robot-chassis.html>. [Accessed 11 December 2018].

Sinaryuda, " Mengenal Aplikasi Arduino IDE dan Arduino Sketch" 2017. [Online].

Available: <https://www.sinaryuda.web.id/microcontroller/mengenal-aplikasi-arduino-ide-dan-arduino-sketch.html>. [Accessed 18 November 2018].

Karen, "About our new MIT App Inventor logo" 2017. [Online]. Available:

<http://appinventor.mit.edu/explore/blogs/karen/2017/08/about.html>.

[Accessed 12 December 2018].

