TUGAS AKHIR – TF 181801

RANCANG BANGUN SISTEM KONTROL PID *PATH FOLLOWING* PADA PROTOTIPE *UNMANNED AUTONOMOUS FORKLIFT*

REZA MALIKI AKBAR
NRP. 02311745000017

Dosen Pembimbing:

Dr. Katherin Indriawati, S.T., M.T.
Hendro Nurhadi, Dipl.-Ing, Ph.D.

DEPARTEMEN TEKNIK FISIKA

Fakultas Teknologi Industri dan Rekayasa Sistem

Institut Teknologi Sepuluh Nopember

Surabaya

2020

FINAL PROJECT – TF 181801

DESIGN OF PID PATH FOLLOWING CONTROL SYSTEM ON PROTOTYPE OF UNMANNED AUTONOMOUS FORKLIFT

REZA MALIKI AKBAR
NRP. 02311745000017

Advisor:

Dr. Katherin Indriawati, S.T., M.T.
Hendro Nurhadi, Dipl.-Ing, Ph.D.

ENGINEERING PHYSICS DEPARTMENT

Faculty of Industrial Technology and Systems Engineering

Institut Teknologi Sepuluh Nopember

Surabaya

2020

PERNYATAAN BEBAS PLAGIARISME

Saya yang bertanda tangan di bawah ini

Nama	:	Reza Maliki Akbar
NRP	:	02311745000017
Departemen/Prodi	:	Teknik Fisika/S1 Teknik Fisika
Fakultas	:	Fakultas Teknologi Industri dan Rekayasa Sistem
Perguruan Tinggi	:	Institut Teknologi Sepuluh Nopember

Dengan ini menyatakan bahwa Tugas Akhir dengan judul "Rancang Bangun Sistem Kontrol PID *Path Following* Pada Prototipe *Unmanned Autonomous Forklift*" adalah benar karya saya sendiri dan bukan plagiat dari karya orang lain. Apabila di kemudian hari terbukti terdapat plagiat pada Tugas Akhir ini maka saya bersedia menerima sanksi sesuai ketentuan yang berlaku.

Demikian surat pernyataan ini saya buat dengan sebenar-benarnya.

Surabaya, 23 Januari 2020

Yang membuat pernyataan,

Reza Maliki Akbar

NRP. 02311745000017

LEMBAR PENGESAHAN I

TUGAS AKHIR

RANCANG BANGUN SISTEM KONTROL PID *PATH FOLLOWING* PADA PROTOTIPE *UNMANNED AUTONOMOUS FORKLIFT*

Oleh:
Reza Maliki Akbar
NRP. 02311745000017

Surabaya, 23 Januari 2020

Menyetujui,

Pembimbing I,

Dr. Katherin Indriawati, S.T., M.T.
NIP. 19760523 200012 2 001

Pembimbing II,

Hendro Nurhadi, Dipl.-Ing., Ph.D.
NIP. 19751120 200212 1 002

Mengetahui,
Kepala Departemen
Teknik Fisika FTIRS-ITS

Dr. Suyanto, S.T., M.T.
NIP. 19711113 199512 1 002

RANCANG BANGUN SISTEM KONTROL PID *PATH FOLLOWING* PADA PROTOTIPE *UNMANNED AUTONOMOUS FORKLIFT*

TUGAS AKHIR

Diajukan Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Teknik
pada
Progam Studi S1 Departemen Teknik Fisika
Fakultas Teknologi Industri dan Rekayasa Sistem
Institut Teknologi Sepuluh Nopember

Oleh:
REZA MALIKI AKBAR
NRP. 02311745000017

Disetujui oleh Tim Penguji Tugas Akhir:

1. Dr. Katherin Indriawati, S.T., M.T. ..... (Pembimbing I)
2. Hendro Nurhadi, Dipl.-Ing., Ph.D. ..... (Pembimbing II)
3. Dr. Bambang L. Widjiantoro, S.T., M.T. ..... (Ketua Penguji)
4. Prof. Dr. Ir. Aulia Siti Aisjah, M.T. ..... (Penguji I)
5. Dyah Sawitri, S.T., M.T. ..... (Penguji II)
6. Moh. Kamalul Wafi, S.T., M.Sc. DIC ..... (Penguji III)

SURABAYA
Januari, 2020

RANCANG BANGUN SISTEM KONTROL PID *PATH FOLLOWING* PADA PROTOTIPE *UNMANNED AUTONOMOUS FORKLIFT*

Nama : Reza Maliki Akbar
NRP : 02311745000017
Departemen : Teknik Fisika FTIRS-ITS
Dosen Pembimbing I : Dr. Katherin Indriawati, S.T., M.T.
Dosen Pembimbing II : Hendro Nurhadi, Dipl.-Ing, Ph.D.

ABSTRAK

Salah satu teknologi yang sering digunakan dalam *material handling* dan *material transport* adalah *forklift*. Secara konvensional *forklift* biasanya dioperasikan oleh operator manusia. *Forklift* dapat diotomatisasi, agar efisien, meningkatkan keamanan, keselamatan, kesehatan kerja juga mengurangi risiko para operator dari kecelakaan. Gagasan inovasi *unmanned autonomous forklift* diciptakan dari permasalahan tersebut. Telah dirancang sistem *motion control* pada tugas akhir ini untuk prototipe *unmanned autonomous forklift* baik secara simulasi maupun perangkat keras dengan menggunakan metode kontrol PID, dan sistem odometri dilengkapi dengan *rotary encoder*. Variabel yang dikontrol adalah jarak dan variabel yang dimanipulasi adalah gaya pada kendaraan. Gaya pada kendaraan, pada prototipe diatur menggunakan pulsa motor. Parameter kontrol PID yang diterapkan pada simulasi, Kp=0,853253789; Ki=0,256576567; Kd=0,923253789. *Rise time* pada simulasi 7,46 detik; *settling time* 23,09 detik; eror lintasan 0,28%. Parameter kontrol PID yang diterapkan pada prototipe, Kp=0,97; Ki=0,3; Kd=1. Uji jarak dilakukan dengan variasi 50 cm hingga 200 cm (interval 25 cm). Satu variasi jarak dilakukan percobaan sebanyak 5 kali. Hasil rata-rata *error* mutlak 1,84 cm (3,67%).

Kata kunci : *material handling*, *material transport*, *unmanned autonomous forklift, rotary encoder*, odometri, kontrol PID.

DESIGN OF PID PATH FOLLOWING CONTROL SYSTEM ON PROTOTYPE OF UNMANNED AUTONOMOUS FORKLIFT

Name	***: Reza Maliki Akbar***
NRP	***: 02311745000017***
Department	***: Engineering Physics FITSE-ITS***
Advisor I	***: Dr. Katherin Indriawati, S.T., M.T.***
Advisor II	***: Hendro Nurhadi, Dipl-Ing, Ph.D.***

ABSTRACT

One technology that is often used in material handling and material transport is forklift. Conventionally, forklift is usually operated by human operators. Forklift can be automated, to make efficiency, in order to improve safety, and occupational health while also reducing the risk of operators from accident. The idea of innovating an unmanned autonomous forklift was created from this problem. A motion control system has been designed in this thesis for prototype unmanned autonomous forklift both in simulation and hardware using PID control method, also the odometry system is equipped with a rotary encoder. The controlled variable is distance and the manipulated variable is the driving force in vehicle. The driving force, on the prototype is regulated using motor pulses. PID control parameters applied to the simulation, Kp = 0.853253789; Ki = 0.256576567; Kd = 0,923253789. Rise time in simulation is 7,46 s; settling time is 23,09 s; trajectory error is 0.28%. PID control parameters applied to the prototype, Kp = 0.97; Ki = 0.3; Kd = 1. The distance test was conducted with a variation of 50 cm to 200 cm (with 25 cm interval). One of distance variation was done by doing experiment 5 times. The average absolute error is 1.84 cm (3.67%).

Keywords: material handling, material transport, unmanned autonomous forklift, rotary encoder, odometry, PID control.

KATA PENGANTAR

Puji syukur kehadirat Allah SWT yang senantiasa melimpahkan rahmat serta hidayah-Nya, serta shalawat serta salam kepada Nabi Muhammad SAW, hingga terselesaikannya tugas akhir beserta laporan yang berjudul **"Rancang Bangun Sistem Kontrol PID *Path Following* pada Prototipe *Unmanned Autonomous Forklift*"**. Penulis telah banyak memperoleh bantuan dari berbagai pihak dalam penyelesaian tugas akhir dan laporan tugas akhir ini. Penulis mengucapkan terimakasih kepada :

1. Bapak Dr. Suyanto, S.T., M.T. selaku Kepala Departemen Teknik Fisika yang telah memberikan petunjuk, ilmu, serta bimbingan selama menempuh pendidikan di Teknik Fisika.
2. Bapak Hendra Cordova, S.T., M. T., selaku Ketua Program Studi S1 Teknik Fisika periode 2014-2019 yang selama ini telah memberikan ilmu, nasihat, dan segala bentuk bantuan lainnya.
3. Ibu Dr. Katherin Indriawati, S.T., M.T., selaku dosen pembimbing yang telah memberikan ilmu, nasihat serta bimbingan yang sangat bermanfaat.
4. Bapak Hendro Nurhadi, Dipl-Ing, Ph.D, selaku dosen pembimbing yang juga telah dengan sabar memberikan petunjuk, ilmu serta bimbingan yang sangat bermanfaat.
5. Bapak Dr. Ir. Totok Soehartanto, D.E.A., selaku dosen wali yang telah membimbing penulis selama perkuliahan.
6. Kedua orang tua, terima kasih atas segala cinta, kasih sayang, doa, perhatian serta dukungan moril dan materil yang telah diberikan.
7. Bapak Dr. Aris Budiyarto, S.T., M.T., Bapak Ir. Bolo Dwiartomo, M.Sc, Bapak Suharyadi Pancono., Dipl. Ing. HTL,

M.T., para dosen Teknik Otomasi Manufaktur dan Mekatronika Politeknik Manufaktur Negeri Bandung, yang telah membantu, membimibing, memberikan saran dan solusi teknikal dan praktikal.

8. Bapak Eko Budi Utomo, S.ST, M.T., dosen Teknik Mekatronika Politeknik Elektronika Negeri Surabaya, yang telah membantu dan memberikan saran teknikal dan praktikal.
9. Bapak Benson Alex dan Bapak Sarmawi Djauhari dari pihak PT. Altraman (Alexindo Putra Mandiri) yang telah mendukung secara materil dan bantuan lainnya untuk tugas akhir ini.
10. Seluruh dosen karyawan dan civitas akademik Teknik Fisika, terimakasih atas segala bantuan dan kerjasamanya.
11. Seluruh teman-teman Lintas Jalur Teknik Fisika angkatan 2017 dan Laboratorium Fisika Rekayasa Departemen Teknik Fisika ITS, terima kasih untuk semuanya.
12. Teman-teman tim robotika bawah air dan amfibi, Banyubramanta ITS, terima kasih sudah mau berjuang bersama dan bertahan hingga sekarang.
13. Teman-teman UKM Maritime Challenge ITS dan PPNS, terima kasih atas segala dukungannya.
14. Semua pihak yang tidak dapat disebutkan satu persatu, terima kasih atas bantuannya.

Penulis sadar bahwa penulisan laporan tugas akhir ini tidak sempurna, namun semoga laporan ini dapat memberikan kontribusi berarti dan menambah wawasan yang bermanfaat bagi pembaca, keluarga besar Teknik Fisika khususnya, dan civitas akademik ITS pada umumnya. Semoga laporan tugas akhir ini dapat bermanfaat sebagai referensi pengerjaan laporan tugas akhir bagi mahasiswa yang lain.

Surabaya, 15 Januari 2020

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR NOTASI

θ	*heading angle*
ρ	*turning radius* pada *origin* robot (x,y)
φ	*steering angle*
v	kecepatan robot pada (x,y)
v_r	kecepatan referensi
θ_r	*heading angle* referensi
l	jarak antara as depan dan as belakang
l_c	jarak dari *origin* robot (x,y) ke titik pusat massa
w	setengah jarak antara dua roda dari setiap as
r	jari-jari roda
m	massa robot
J_b	momen massa inersia robot di sekitar sumbu vertikal melewati pusat massa robot
J_h	momen massa inersia roda di sekitar sumbu horizontal
J_v	momen massa inersia roda di sekitar sumbu vertikal
f_v	*driving force*
τ_φ	torsi *steering*
d_φ, d_v	koefisien *viscous damping*

BAB I
PENDAHULUAN

1.1 Latar Belakang

Proses pemuatan, pembongkaran dan pengangkutan bahan adalah salah satu masalah utama untuk setiap lokasi produksi pada industri dan memiliki dampak besar pada biaya produk (Chopra, 2013). Oleh karena itu, upaya untuk menemukan sistem yang menguntungkan dan fleksibel secara kontinyu menjadi sangat penting. Permintaan terhadap otomatisasi yang sangat tinggi akan mengubah operasi industri dan gudang secara signifikan ("Site Automation: Automated/Robotic On-Site Factories (Cambridge Handbooks on Construction Robotics), Thomas Bock, Thomas Linner, eBook - Amazon.com," 2015).

Forklift yang biasa dioperasikan oleh manusia perlu ditingkatkan untuk mencapai tugas industri otomatis yang efisien dalam proses penanganan material, sementara pada saat yang sama keselamatan operasionalnya harus terjamin (Swartz, 1999). Kemajuan pesat dalam teknologi sensor dan komputer merupakan peluang untuk mengembangkan *unmanned autonomous forklift*, sangat besar. Tugas mengendarai *forklift* membutuhkan pelatihan panjang bagi pengemudi untuk memastikan operasi yang aman dan terjamin (Jefferies, 2011). Pekerjaan mengendari *forklift* cukup melelahkan karena yang dilakukan adalah kegiatan repetisi yang monoton (Washington State Departement of LAbor&Industries, 2015). Hal tersebut merupakan kandidat ideal untuk diotomatisasikan.

Baru-baru ini harga sensor dan perangkat komputasi mengalami penurunan harga, beberapa peneliti menyadari bahwa teknologi *mobile robot* cukup siap untuk menjalankan *forklift* di gudang dan atau industri. Hal ini pun mendukung perkembangan revolusi industri 4.0 ("Smart Automation to Smart Manufacturing: Industrial Internet of Things, Uthayan Elangovan, eBook -

Amazon.com," 2016). *Unmanned autonomous forklift* menerima perintah dari manusia dapat dengan aman melaksanakan tugas yang memerlukan interaksi dengan orang dan benda di lingkungan kerja yang tidak diketahui. Ini termasuk kemampuan untuk mendeteksi dan memanipulasi palet dan muatannya, baik di tanah dan di tempat untuk menyimpan muatan, serta dapat melakukan pemetaan jalur kemudian meghindari rintangan. Meskipun beberapa produk ada di pasar, namun biaya yang dibutuhkan masih tinggi. Kemudian masih ada ruang untuk pengembangan sistem yang sederhana, ramah penggunaan, dan harganya cukup terjangkau.

Beberapa permasalahan tersebut dalam tugas akhir ini akan dirancang dan dikembangkan *forklift* yang dapat bergerak secara mandiri, mengangkat kargo dan memindahkannya ke lokasi target. *Forklift* harus mampu memetakan lingkungan kerja dan bergerak secara otomatis atau memungkinkan intervensi kendali dari manusia ketika dibutuhkan.

1.2 Rumusan Masalah

Berdasarkan latar belakang diatas maka permasalahan yang di angkat dalam tugas akhir ini adalah sebagai berikut :

a. Apakah kontroler PID dapat diimplementasikan pada sistem *motion control* untuk *unmanned autonomous forklift* secara simulasi?
b. Apakah kontroler PID dapat diimplementasikan pada sistem *motion control* untuk prototipe *unmanned autonomous forklift,* menggunakan perangkat keras secara *real-time*?

1.3 Tujuan

Tujuan dilakukannya tugas akhir ini adalah merancang dan mengimplementasikan secara simulasi sistem *motion control* PID serta membangun dan mengimplementasikan prototipe sistem *motion control* PID pada *unmanned autnomous forklift*.

1.4 Batasan Masalah

Adapun batasan masalah dalam tugas akhir ini adalah sebagai berikut:

a. Prototipe *forklift* dirupakan dalam bentuk *car-like* robot 4WD dengan motor penggerak di roda-roda depan.
b. Variabel pada simulasi yang dikontrol adalah jarak *forklift* dengan kesesuaian jalur yang sudah ditentukan dan variabel yang dimanipulasi adalah gaya kecepatan pada kendaraan.
c. Variabel pada prototipe yang dikontrol adalah jarak dan variabel yang dimanipulasi adalah pulsa pada motor DC.
d. Sensor jarak yang digunakan pada prototipe *forklift* adalah *rotary encoder*.
e. Lintasan yang dilalui adalah lintasan lurus, pada percobaan prototipe, lintasan memiliki variasi jarak 50 cm, 75 cm, 100 cm, 125 cm, 150 cm, 175 cm, 200 cm.

1.5 Sistematika Penulisan

Penyusunan laporan kerja praktik disusun dengan struktur yang terarah. Adapun sistematika penulisan dibuat dengan urutan sebagai berikut,

a. BAB I PENDAHULUAN
   Berisikan tentang latar belakang, rumusan masalah, tujuan, dan batasan masalah tugas akhir yang dikerjakan. Bab ini ditutup dengan pembahasan sistematika penulisan yang digunakan.
b. BAB II TEORI PENUNJANG
   Berisikan mengenai teori-teori penunjang daripada tugas akhir ini. Mulai dari teknologi pemindahan barang, *forklift*, mikrokontroler, kontrol PID, kinematika dan dinamika robot, pemetaan jalur dan navigasi robot, sensor-sensor, dan aktuator-aktuator pada sistem.
c. BAB III METODOLOGI PENELITIAN
   Bab ini menjelaskan bagan alur secara keseluruhan berupa metodologi penelitian yang dilakukan pada tugas akhir ini.

d. BAB IV ANALISA DATA DAN PEMBAHASAN
   Bab ini berisikan hasil dari data yang diperoleh setelah melakukan simulasi, pengujian sistem.
e. BAB V KESIMPULAN DAN SARAN
   Bab ini berisi kesimpulan dan saran daripada tugas akhir ini.

BAB II
TINJAUAN PUSTAKA

2.1 Metode Pemindahan Barang

Banyak orang mengartikan *material transport* sama dengan *material handling.* Dua istilah ini memiliki definisi yang berbeda. *Material handling* adalah ilmu tentang pemindahan (*transport*), penyimpanan (*storage*), pengamanan, dan pengontrolan material (Frazelle, 2002). *Material transport* merupakan bagian dari *material handling*. Istilah lainnya yaitu *material moving*. Aktivitas ini menjadi sangat penting pada praktiktnya dalam operasi perusahaan. Mulai dari industri pertambangan, manufaktur, kimia dan proses, sehingga *material transport* memiliki alokasi biaya yang cukup besar. Terutama dari sisi *supplier* atau pemasok, maka dari itu perkembangan teknologinya sangat pesat untuk memenuhi kebutuhan sistem logistik yang modern. Dalam pengelompokannya berdasarkan tenaga penggeraknya, pemindah barang dibagi menjadi dua yaitu digerakkan dengan tenaga manusia dan digerakkan dengan tenaga mesin (Ray, 2008).

2.1.1 Alat Pemindah Barang dengan Tenaga Manusia

Alat pemindah barang ini termasuk cara yang paling konvensional. Berikut beberapa contoh alat pemindah barang yang digerakkan dengan tenaga manusia :

a. *Manual Hand Truck*

Alat pemindah barang secara manual atau dengan tenaga manusia yang tidak memiliki mekanisme tambahan, cukup konvensional. Variasi *hand truck* ini cukup banyak, karena rata-rata desain alat menyesuaikan kondisi operasi.

b. *Hydraulic Handlift*

Jenis alat yang digunakan untuk memindahkan barang dilengkapi dengan mekanisme pengangkat secara hidrolik.

c. *Electric Handlift*
Jenis alat yang digunakan untuk memindahkan barang dilengkapi dengan mekanisme pengangkat motor elektrik yang mana catu dayanya dari baterai.

2.1.2 Alat Pemindah Barang dengan Tenaga Mesin

Alat pemindah barang ini mekanisme penggeraknya dan tenaganya berasal dari mesin. Berikut beberapa contoh alat pemindah barang yang digerakkan dengan tenaga mesin :

a. *Forklift*
Alat pemindah barang secara manual atau dengan tenaga manusia yang tidak memiliki mekanisme tambahan, cukup konvensional. Variasi *hand truck* ini cukup banyak, karena rata-rata desain alat menyesuaikan kondisi operasi.
b. *Car on rails heavy duty load*
Jenis alat yang digunakan untuk memindahkan barang dilengkapi dengan mekanisme pengangkat secara hidrolik.
c. *Crane*
Jenis alat yang digunakan untuk memindahkan barang dilengkapi dengan mekanisme pengangkat motor elektrik yang mana catu dayanya dari baterai.
d. *Automated Guided Vehicle*
Kendaraan tanpa dikenderai namun dikendalikan oleh komputer dengan menggunakan gelombang radio atau jalur sensor.

2.2 *Forklift*

Forklift adalah peralatan penanganan material dan logistik. Istilah penanganan material dan logistik menggambarkan kegiatan untuk setiap proyek konstruksi atau kegiatan yang membutuhkan untuk mengambil dan memindahkan barang mentah, barang telah diproses, barang yang telah selesai dikerjakan, alat-alat, suplai, atau peralatan untuk *maintenance* (Navy Construction Force, 2008). Setiap operasi membutuhkan proses pengangkatan, penurunan atau pemindahan benda-benda yang telah

diklasifikasikan. Forklift dirancang untuk melakukan berbagai tugas tersebut dalam beberapa kondisi tertentu.

Sebuah *forklift* perlu dibutuhkan perawatan tersendiri untuk menjaga keawetannya, pergantian *sparepart forklift* pada bagian-bagian yang rusak menjadi keharusan guna memperpanjang masa guna *forklift* dan agar tidak berimbas pada bagian-bagian yang lain. Perawatan *forklift* tidak hanya sebatas pada pergantian *sparepart forklift* yang telah rusak namun termasuk cara pakainya sendiri, untuk lebih mengenal pemakaian sebuah *forklift* berikut ini beberapa bagian dari *forklift* yang harus diketahui (Jefferies, 2011):

a. *Fork* adalah bagian utama dari kendaraan *forklift* yang berfungsi sebagai penopang untuk mebawa dan mengangkat barang. *Fork* berbentuk dua buah logam lurus dengan panjang rata-rata 2,5m. Posisi peletakan barang di atas *pallet* masuk ke dalam *fork* juga menentukan beban maksimal yang dapat diangkat sebuah *forklift*.
b. *Carriage* adalah bagian dari *sparepart forklift* yan berfungsi sebagai penghubung antara *mast* dan *fork*. Di tempat inilah *fork* melekat. *Carriage* juga berfungsi sebagai sandaran dan pengaman bagi barang-barang dalam *pallet* untuk transportasi atau pengangkatan.
c. *Mast* adalah bagian utama terkati dengan fungsi kerja sebuah *fork* dalam *forklift*. *Mast* adalah satu bagian yang berupa dua buah logam yang dikonstruksikan pada tiang diintegrasikan dengan sistem hidrolik. Berfungsi untuk *lifting* juga *tilting*.
d. *Overhead guard* adalah pelindung bagi seorang pengemudi *forklift.* Fungsi pelindungan ini terkait dengan *safety user* dari kemungkinan terjadinya barang yang jatuh saaat diangkat atau diturunkan.
e. *Counterweight* adalah bagian peneyeimbang beban dari sebuah *forklift.* Letakanya berlawanan dengan posisi *fork*.

Gambar 2.1 *Electric forklift* (Toyota Industrial Equipment, 2018)

Menurut daya yang digunakannya, *forklift* dibagi menjadi beberapa kategori yaitu (Lencrow Fokrlifts, 2018) :

a. *Diesel forklift*
b. *LPG/petrol forklift*
c. *Electric forklift*

Pada tugas akhir ini digunakan *forklift* dengan skala 1:10 yaitu RC *electric forklift* Huina 1577 2.4 GHz dengan 8-*channel*.

Gambar 2.2 RC *forklift* Huina 1577 (Amazon.com, 2018)

2.3 Kinematika dan Dinamika *Mobile Robot*

Robot dapat dianalisa dalam dua dominan kajian, yaitu analisa kinematik dan dinamik. Analisa kinematik berkaitan dengan gerakan robot tanpa memandang efek inersia atau kelembaman yang terjadi ketika robot melakukan gerakan, sedang analisa dinamik berhubungan dengan efek inersia dari struktur dari struktur robot secara fisik hasil dari gerakan yang ditimbulkan oleh torsi aktuator ketika robot sedang melakukan pergerakan (Supriyanto dkk., 2010). Konversi sudut ke koordinat Cartesian pada robot 1 DOF adalah dasar dari transformasi dalam analisis kinematik. Untuk sistem robot lebih dari 1 DOF (dan kenyataannya, robot aplikasi selalu memiliki lebih dari 1 DOF) analisa kinematik mutlak diperlukan. Sistem mobile robot lebih rumit kinematiknya, karena berkaitan dengan prinsip gerak *holonomic* dan *non-holonomic*.

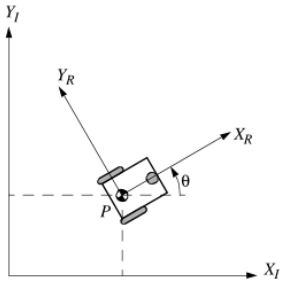
Kinematika adalah studi paling dasar tentang bagaimana sistem mekanik berperilaku. Dalam *mobile robotics*, perlu dipahami perilaku mekanik robot baik untuk merancang robot mobile yang sesuai untuk tugas dan untuk memahami cara membuat perangkat lunak kontrol untuk sebuah contoh perangkat keras robot *mobile* (Siegwart & Nourbakhsh, 2004).

Mobile robot bukanlah sistem mekanik kompleks pertama yang memerlukan analisis rumit. Robot manipulator telah menjadi subyek studi intensif selama lebih dari tiga puluh tahun. Dalam beberapa hal, robot manipulator telah jauh lebih kompleks daripada robot-robot bergerak awal: robot las standar mungkin memiliki lima atau lebih sambungan, sedangkan robot-robot bergerak awal adalah mesin penggerak diferensial yang sederhana. Dalam beberapa tahun terakhir, komunitas robotik telah mencapai pemahaman yang cukup lengkap tentang kinematika dan bahkan dinamika (yaitu berkaitan dengan kekuatan dan massa) dari manipulator robot. Komunitas robotik mobile memiliki banyak pertanyaan kinematik yang sama dengan komunitas manipulator robot. Ruang kerja robot manipulator sangat penting karena

menentukan kisaran posisi yang mungkin yang dapat dicapai oleh *end-effector*-nya terhadap perlengkapannya terhadap lingkungan. Ruang kerja robot seluler juga sama pentingnya karena ia mendefinisikan berbagai kemungkinan pose yang dapat dicapai robot mobile di lingkungannya. Pengontrolan lengan robot mendefinisikan cara di mana keterlibatan aktif motor dapat digunakan untuk berpindah dari pose ke pose di ruang kerja. Demikian pula, kemampuan pengendalian robot bergerak menentukan jalur dan lintasan yang mungkin di ruang kerjanya.

Dinamika robot menempatkan kendala tambahan pada ruang kerja dan lintasan karena pertimbangan massa dan gaya. Robot *mobile* juga dibatasi oleh dinamika; misalnya, pusat gravitasi yang tinggi membatasi radius putar yang praktis dari robot yang cepat, seperti mobil karena bahaya berguling.

Perbedaan utama antara *mobile robot* dan lengan manipulator juga memperkenalkan tantangan signifikan untuk estimasi posisi. Manipulator memiliki satu ujung yang tetap pada lingkungan. Mengukur posisi efektor ujung lengan hanyalah masalah memahami kinematika robot dan mengukur posisi semua sendi menengah. Posisi manipulator dengan demikian selalu dapat dihitung dengan melihat data sensor saat ini tetapi robot bergerak adalah otomat mandiri yang dapat sepenuhnya bergerak dengan memperhatikan lingkungannya.



Gambar 2.3 Kinematik *mobile robot*, referensi global dan lokal (Petersen dkk., 2000)

Tidak ada cara langsung untuk mengukur posisi robot bergerak secara instan. Gerak robot harus diintegrasikan dari waktu ke waktu. Ketidakakuratan estimasi gerakan mnejadi tambahan karena selip dan jelas bahwa mengukur posisi robot bergerak justru merupakan tugas yang sangat menantang. Proses memahami gerakan robot dimulai dengan proses menggambarkan kontribusi yang diberikan setiap roda untuk gerakan. Setiap roda memiliki peran dalam memungkinkan seluruh robot bergerak. Dengan cara yang sama, masing-masing roda juga memberlakukan batasan pada gerakan robot, misalnya menolak untuk selip secara lateral. Pada bagian berikutnya, diperkenalkan notasi yang memungkinkan ekspresi gerak robot dalam kerangka referensi global serta kerangka referensi lokal robot sebagaimana diilustrasikan pada Gambar 2.3. Kemudian, dengan menggunakan notasi ini, didemonstrasikan konstruksi model gerak kinematik sederhana ke depan, menggambarkan bagaimana robot secara keseluruhan bergerak sebagai fungsi geometri dan perilaku roda individu. Selanjutnya, secara formal mendeskripsikan batasan kinematik roda individu, dan kemudian menggabungkan batasan kinematik ini untuk mengekspresikan seluruh kendala kinematis robot. Dengan alat ini, seseorang dapat mengevaluasi jalurnya dan lintasan yang menentukan kemampuan manuver robot.

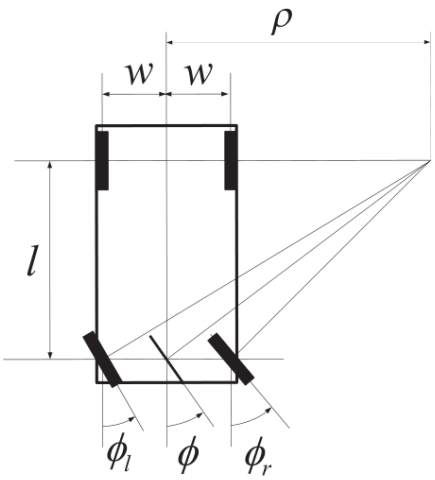
2.4 Model Kinematika dan Dinamika *Forklift-Like* Robot

Pemodelan kinematika dan dinamika *forklift* diambil dari referensi model *Forklift-Like Wheeled Robot* (Lee, 2014). Persamaan yang digunakan pada pemodelan yaitu persamaan *state-space*.

2.4.1 Model Kinematik

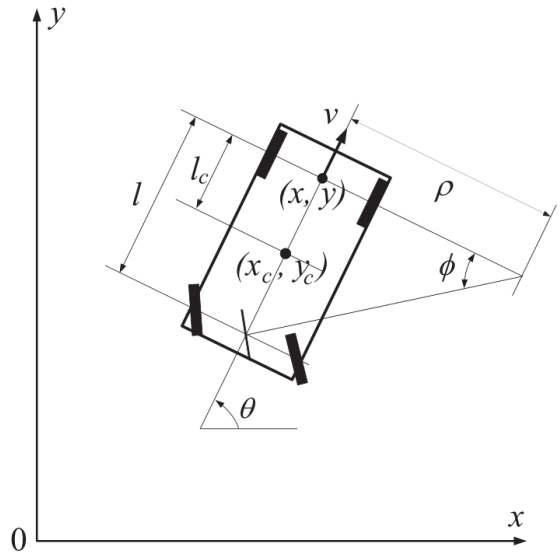
Pada Gambar 2.4 ditunjukkan model *forklift-like mobile robot*, di mana l adalah jarak antara poros depan dan poros belakang; w menunjukkan jarak setengah antar kedua roda tiap porosnya, ρ adalah radius belok dari *origin* robot; $\varphi, \varphi_r, \varphi_l$ menunjukkan *steering angle*, sudut roda kanan, sudut roda kiri.

S*teering system* didesain dan diasumsikan tanpa slip pada roda atau slipnya diabaikan. Dengan demikian $\tan\varphi = \frac{l}{\rho}, \tan\varphi_r = \frac{l}{\rho - w}, \tan\varphi_l = \frac{l}{\rho + w}$ mengikuti pada Gambar 2.4. Variabel ρ dieliminasi dari fungsi tangen, yang menghasilkan $\tan\varphi_r = \frac{l\tan\varphi}{(l - w\tan\varphi)}$, $\tan\varphi_l = \frac{l\tan\varphi}{(l + w\tan\varphi)}$ sudut roda kiri dan sudut roda kanan dapat dengan mudah dihitung untuk diberikan *l* dan *w* sebagai fungsi *steering angle* φ. Hasilnya kontrol dari *steering angle* φ sama dengan φ_r dan φ_l.



Gambar 2.4 Bentuk *steering* dari *forklift-like wheeled robot* (Lee, 2014)

Gambar 2.5 merupakan ilustrasi parameter-parameter kinematik *forklift*, di mana (x, y) adalah posisi *origin* robot dalam *frame*; (x_c, y_c) adalah posisi pusat massa robot; l_c adalah jarak dari *origin* robot (x, y) ke titik pusat massa; v adalah kecepatan robot di (x, y); ρ adalah radius belokan *origin* robot (x, y); φ dan θ adalah *steering angle* dan *heading angle* dari *forklift*.



Gambar 2.5 Parameter kinematik dari *forklift* (Lee, 2014)

Berdasarkan kondisi mengabaikan slip pada roda, batasan-batasan (*constraints*) kinematik *nonholonomic* dapat diturunkan menjadi :

$$\dot{x} = v\cos\theta \tag{2.1}$$

$$\dot{y} = v\sin\theta \tag{2.2}$$

$$\dot{\theta} = -\frac{v}{l}\tan\varphi \tag{2.3}$$

Di mana batasan terakhir diturunkan dari $\dot{\theta} = -\frac{v}{\rho}$ dengan $\rho = \frac{l}{\tan\theta}$, sebagaimana ρ didefinisikan pada Gambar 2.5.

2.4.2 Model Dinamik

Persamaan gerak *forklift-like mobile* robot diturunkan dengan menerapkan persamaan Lagrange. Dalam penurunan ini, energi potensial pada robot diasumsikan nol.

Energi kinetik pada robot yang dipertimbangkan adalah jumlah energi translasi dari massa robot, energi rotasi empat roda di sekitar sumbu horizontal, energi rotasi robot di sekitar sumbu vertikal yang melewati pusat massa robot, dan energi rotasi dari empat roda di sekitar sumbu vertikal robot melewati pusat masssa robot. Gambar 2.5 dijadikan acuan, kecepatan rata-rata roda belakang sama dengan detik φ kali dari roda depan dikarenakan kemudinya.

Karena energi kinetik sama dengan Lagrangean *L* untuk *forklift-like* robot, persamaan gerak robot dapat diturunkan dengan menggunakan persamaan Lagrange (Essen & Nijmeijer, 2001) untuk v dan φ.

$$M\dot{q} + c + Dq = f \tag{2.4}$$

Vektor $\dot{q}$ adalah vektor keadaan 2 x 1 didefiniskan sebagai $\dot{q} \equiv (\mathrm{v}, \dot{\varphi})^{\mathrm{T}}$; f adalah vektor input 2 × 1 yang didefinisikan sebagai $\mathrm{f} \equiv (\mathrm{f_v}, \tau_\varphi)^{\mathrm{T}}$; matriks M 2 x 2 *positive-definite symmetric*, matriks 2 × 1 Coriolis dan vektor gaya sentrifugal *c*, dan matriks 2 × 2 *viscous damping*, D didefinisikan sebagai berikut :

$$M = \begin{bmatrix} \left(m + \frac{2}{r^2}Jh\right) + \frac{2}{r^2}J_h \sec^2\varphi \\ + \frac{1}{l^2}(l_c^2 m + J_b + 4J_v)\tan\varphi - \frac{2}{l}J_v \tan\varphi \end{bmatrix} \quad \begin{matrix} -\frac{2}{l}J_v \tan\varphi \\ 2J_v \end{matrix} \\ -\frac{2}{l}J_v \tan\varphi \end{bmatrix} \tag{2.5}$$

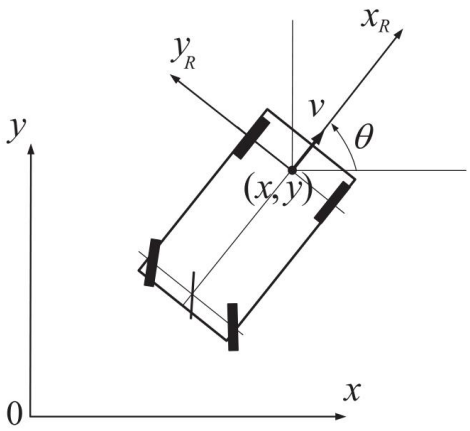
$$c = \begin{Bmatrix} (2\left[\frac{1}{l^2}\left(l_c^2 m + J_b + 4J_v\right) + \frac{2}{r^2}J_h\right]\tan\varphi \sec^2\varphi v\dot{\varphi} - \frac{2}{l}J_v \sec^2\varphi\dot{\varphi}^2) \\ -\left[\frac{1}{l^2}\left(l_c^2 m + J_b + 4J_v\right) + \frac{2}{r^2}J_h\right]\tan\varphi \sec^2\varphi v^2 \end{Bmatrix} \tag{2.6}$$

$$D = \begin{bmatrix} d_v & 0 \\ 0 & d_\varphi \end{bmatrix} \tag{2.7}$$

Radius dari roda adalah *r*; *m* adalah massa dari robot; J_b adalah momen massa inersia robot di sekitar sumbu vertikal melewati pusat massa robot; J_h dan J_v adalah momen massa inersia roda di sekitar sumbu horizontal dan vertikal yang masing-masing melewati pusat massa roda; d_v dan d_φ menunjukkan koefisien *viscous damping*; f_v dan τ_φ menunjukkan *driving force* dan torsi *steering*.

2.4.3 Pembangkit Trayektori (*Trajectory Generator)*

Pemodelan pembangkit trayektori (*trajectroy generator* mengaplikasikan kontrol kinematik untuk menciptakan *real-time trajectory*. *Tracking error* e_B ditentunkan berdasarkan *frame* dasar *xy* yang ditunjukkan pada Gambar 2.6.



Gambar 2.6 Dua *frame* robot yang berbeda dan bergerak (Lee, 2014)

Secara definisi persamaan *tracking error* dtunjukkan sebagai berikut :

$$e_B \equiv \begin{Bmatrix} e_{Bx} \\ e_{By} \\ e_{B\theta} \end{Bmatrix} = \begin{Bmatrix} x_d - y \\ y_d - y \\ \theta_d - \theta \end{Bmatrix} \tag{2.8}$$

Dengan x_d, y_d, θ_d adalah nilai posisi $x, y,$ dan θ yang diinginkan.

Tracking error yang dinotasikan e_B pada *frame* dasar, ditransformasikan menjadi e_R, *tracking error frame* robot lokal x_R, y_R dihubungkan dengan *origin* robot yang ditunjukkan pada Gambar 2.6.

$$e_R \equiv \begin{Bmatrix} e_{Bx} \\ e_{By} \\ e_{B\theta} \end{Bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{Bmatrix} e_{Bx} \\ e_{By} \\ e_{B\theta} \end{Bmatrix} \tag{2.9}$$

Persamaan eror kinematik diturunkan dengan mendiferensiasikan persamaan (2.9), mengacu pada batasan-batasan kinematik (2.1) – (2.3)

$$\dot{e_x} = \dot{\theta} e_y - v + v_d \cos e_\theta \tag{2.10}$$

$$\dot{e_y} = -\dot{\theta} e_x + v_d \sin e_\theta \tag{2.11}$$

$$\dot{e_\theta} = -\dot{\theta}_d - \dot{\theta} \tag{2.12}$$

Tracking error pada *frame* robot lokal dipertimbangkan, pergerakan robot dikontrol dari sudut pandang robot, layaknya mengemudi sebuah mobil.

Skema kontrol kinematik dirancang berdasarkan persamaan kesalahan kinematik (2.10)-(2.12). Skema kontrol kinematik menghitung kecepatan referensi *v* dan laju waktu referensi *heading angle* $\dot{\theta}$ daripada robot. Nilai $k_z = 1$ dikarenakan posisi *forklift* pada sumbu z tidak mengalami perubahan.

$$v_r = v_d \cos e_\theta + k_x e_x \tag{2.13}$$

$$\dot{\theta}_r = \dot{\theta}_d + \frac{k_y}{k_z} v_d e_y + k_\theta \sin e_\theta \tag{2.14}$$

2.4.4 Penyederhanaan Model Dinamik

Model dinamik dibuat menjadi lebih sederhana untuk tercapainya kesederhanaan desain kontrol. *Steering angle* biasanya tidak besar dalam praktiknya untuk kendaraan roda 4 saat sedang bergerak. Semua *tan φ* dapat diabaikan dalam dinamik nonlinier (2.4); namun, efek dari notasi v^2 dalam *steering* dinamik (φ) mungkin terlalu signifikan untuk mengemudi kecepatan tinggi. Dengan demikian, model dinamik (2.4) dapat disederhanakan menjadi

$$\dot{v} = \frac{1}{m_v} (-d_v v + f_v) \tag{2.15}$$

$$\ddot{\varphi} = \frac{1}{J_\varphi} \left(-d_\varphi \dot{\varphi} + c_\varphi \tan \varphi \sec^2 \varphi \, v^2 + \tau_\varphi\right) \tag{2.16}$$

Di mana

$$m_v = m + \frac{4}{r^2} J_h \tag{2.17}$$

$$J_\varphi = 2J_v \tag{2.18}$$

$$c_\varphi = \frac{1}{l^2} (l_c^2 m + J_b + 4J_v) + \frac{2}{r^2} J_h \tag{2.19}$$

2.5 Perencanaan Jalur dan Navigasi Pada Robot

Perencanaan jalur adalah hal penting untuk *autonomous mobile robot* yang memungkinkan untuk menemukan jalur terpendek atau optimal antara dua titik. Jika tidak, jalur yang optimal bisa menjadi jalur yang meminimalkan jumlah belokan, jumlah pengereman atau apa pun yang diperlukan oleh aplikasi tertentu (Petersen dkk., 2000).

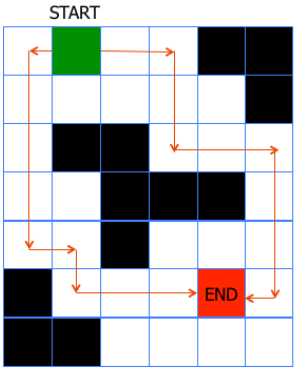
Perencanaan jalur memerlukan peta lingkungan dan robot untuk mengetahui lokasinya sehubungan dengan peta. Robot diasumsikan dapat melokalkan dirinya untuk saat ini, dilengkapi dengan peta, dan mampu menghindari rintangan sementara di jalannya.

Navigasi *mobile robot* yang aman dan efektif membutuhkan algoritma perencanaan jalur yang efisien karena kualitas jalur yang dihasilkan sangat memengaruhi aplikasi robot. Minimalisasi jarak yang ditempuh biasanya adalah tujuan utama dari proses navigasi karena memengaruhi metrik lain seperti waktu pemrosesan dan konsumsi energi.

Terdapat tiga fungsi utama yang mendasar dalam navigasi robot yaitu sebagai berikut (Koubaa dkk., 2018):

1. Lokalisasi : Dapat menentukan lokasi robot pada lingkungannya. Beberapa metode dapat diterapkan untuk lokalisasi, yaitu penggunaan kamera, GPS, sensor *ultrasound*, *laser rangefinder*, LiDAR (*Light Detection and Ranging*). Lokasi dapat ditetapkan sebagai referensi simbolis relatif terhadap lingkungan lokal (misal: titik tengah pada suatu kamar), dinyatakan sebagai koordinat topologi (misal: kamar No. 24) atau dapat diekspresikan dengan menggunakan koordinat absolut (misal: lintang, bujur, ketinggian).
2. Pemetaan : Robot membutuhkan peta lingkungannya untuk mengidentifikasi di mana robot telah bergerak sejauh angka yang telah ditentukan. Peta membantu robot untuk mengetahui arah dan lokasi. Peta dapat ditempatkan secara manual ke dalam memori robot (misal: representasi grafik, representasi matriks) atau dapat secara bertahap dibangun sementara robot menemukan lingkungan baru.
3. *Motion planning* atau *path planning* (perencanaan jalur) : Untuk menemukan jalur untuk *mobile robot*, posisi tujuan harus diketahui lebih dahulu oleh robot, yang mana robot membutuhkan skema pengalamatan yang tepat dan dapat diikuti oleh robot. Skema pengalamatan berfungsi untuk menunjukkan

ke robot di mana ia akan mulai dari posisi awal. Sebagai contoh, robot dapat diminta untuk pergi ke ruangan tertentu di lingkungan kantor dengan hanya memberikan nomor ruangan sebagai alamatnya. Dalam skenario lain, alamat dapat diberikan dalam notasi korrdinat absolut ataupun relatif.



Gambar 2.7 Algoritma Dijkstra untuk perencanaan jalur robot (Reality Bytes, 2018)

Kecerdasan harus ditanamkan ke dalam robot untuk memastikan pelaksanaan tugas yang sedang dipertimbangkan dan memenuhi misi secara efisien. Penanaman kecerdasan ke dalam sistem robot memaksakan penyelesaian sejumlah besar masalah penelitian seperti navigasi yang merupakan salah satu masalah mendasar dari sistem *mobile robot*. Robot harus mengetahui posisinya relatif terhadap posisi tujuannya untuk menyelesaikan tugas navigasi dengan sukses,. Robot juga harus mempertimbangkan bahaya lingkungan di sekitarnya dan menyesuaikan tindakannya untuk memaksimalkan kesempatan untuk mencapai tujuan.

Robot dikurangi menjadi titik-massa dan semua hambatan di lingkungan berkembang setengah dari perpanjangan terpanjang

robot dari pusatnya untuk menghadapi perwujudan fisik robot, yang mempersulit proses perencanaan jalan. Representasi ini dikenal sebagai ruang konfigurasi karena mengurangi representasi robot ke koordinat x dan y di dalam bidang datar. Sebagian besar algoritma perencanaan juga tidak mempertimbangkan orientasi robot. Jika ini diinginkan, dimensi tambahan perlu diperkenalkan ke ruang konfigurasi (Nehmzow, 2003).

2.6 *Rotary Encoder*

Rotary encoder adalah perangkat elektromekanik yang dapat memonitor gerakan dan posisi. *Rotary encoder* umumnya menggunakan sensor optik untuk menghasilkan serial pulsa yang dapat diartikan menjadi gerakan, posisi, dan arah ("Basics of Rotary Encoders: Overview and New Technologies | Machine Design," 2019). Sehingga posisi sudut suatu poros benda berputar dapat diolah menjadi informasi berupa kode digital oleh *rotary encoder* untuk diteruskan oleh rangkaian kendali. *Rotary encoder* umumnya digunakan pada pengendalian robot, *motor drive*, dsb.

Rotary encoder tersusun dari suatu piringan tipis yang memiliki lubang-lubang padabagian 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 perangkat 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. Gambar 2.8 menunjukkan bagan skematik sederhana dari *rotary encoder*. Semakin banyak deretan pulsa yang dihasilkan pada satu putaran menentukan akurasi *rotary encoder* tersebut, akibatnya semakin

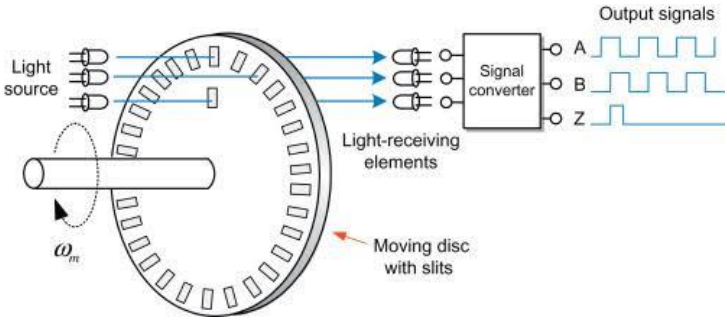
banyak jumlah lubang yang dapat dibuat pada piringan menentukan akurasi *rotary encoder* tersebut (S., F., & Ara, 2013).

Odometri adalah penggunaan data dari pergerakan aktuator untuk memperkirakan perubahan posisi dari waktu ke waktu. Odometri digunakan untuk memperkirakan posisi relatif terhadap posisi awal (Susanto, 2018). Perhitungan jumlah pulsa yang dihasilkan oleh sensor *rotary encoder* setiap satuan ukuran yang kemudian dikonversi menjadi satuan millimeter digunakan untuk memperkirakan posisi relatif robot.

Jumlah pulsa setiap satu kali putaran roda didapatkan digunakan rumus sebagai berikut:

$$Keliling\ roda = 2\ \times\ \pi\ \times r \tag{2.20}$$

$$Pulsa\ per\ mm = \frac{resolusi\ encoder}{keliling\ roda} \tag{2.21}$$

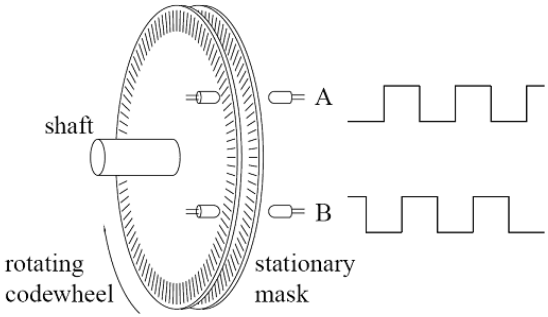


Gambar 2.8 *Rotary encoder* ("Rotary encoder system. | Download Scientific Diagram," 2019)

Terdapat dua jenis sensor *rotary encoder* yaitu *incremental encoder* dan *absolute encoder*.

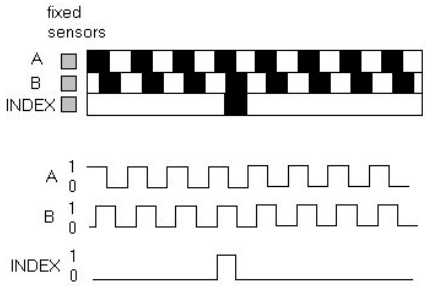
2.6.1 *Incremental Encoder*

Incremental encoder adalah sensor enkoder yang menghasilkan gelombang pulsa digital hasil dari pembacaan setiap resolusi yang dilewati oleh sensor tersebut. Pada umumnya terdiri dua *channel* yaitu channel A dan B. Ketika poros berputar maka pulsa akan dihasilkan pada setiap *channel* yang berhubungan dengan kecepatan motor sedangkan antara A dan B menghasilkan arah putaran. Cara mengukur arah putaran dapat diketahui dengan mengetahui *channel* mana yang mendahului terhadap *channel* satunya, dapat ditentukan dengan melihat perbedaan fasa antara kedua channel tersebut yang selalu berbeda fasa seperempat sinyal. *Incremental encoder* menggunakan pembacaan kode biner yang dapat membaca dua kondisi sekaligus sehingga dapat membaca dua arah yang berbeda.



Gambar 2.9 Susunan *incremental encoder* (“Rotary Encoder - Northwestern Mechatronics Wiki,” 2019)

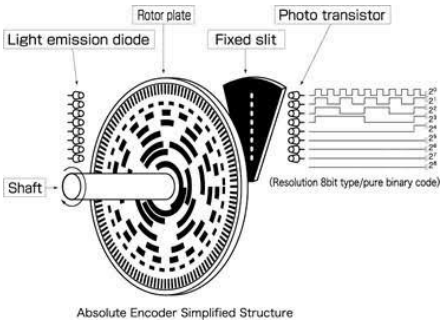
Arah putaran dapat ditentukan melalui level salah satu sinyal terhadap sinyal transisi pada saat transisi. Salah satu contohnya adalah ketika channel A = 0 dan B = 1 menunjukkan arah putaran searah jarum jam, sebaliknya B = 0 dan A = 1 menunjukkan arah berlawanan jarum jam.



Gambar 2.10 Sinyal keluaran *incremental encoder* (“Rotary Encoder - Northwestern Mechatronics Wiki,” 2019)

2.6.2 *Absolute Encoder*

Absolute encoder merupakan sensor enkoder yang menghasilkan keluaran kombinasi biner yang mutlak di setiap sudutnya. Hal tersebut disebabkan oleh struktur piringan *absolute encoder* yang mempunyai jalur lingkaran-lingkaran konsentris dari dekat poros ke bagian luar poros, semakin dekat dengan poros, maka bagian transparan dan gelap akan semakin sedikit. Pada saat tidak diberi tegangan, sensor masih menyimpan kode biner di sudut terakhir yang didapatkan dan kode binernya berbeda beda disetiap sudut sehingga *output*-nya cukup banyak. *Port input* yang dibutuhkan banyak untuk membaca sensor ini.



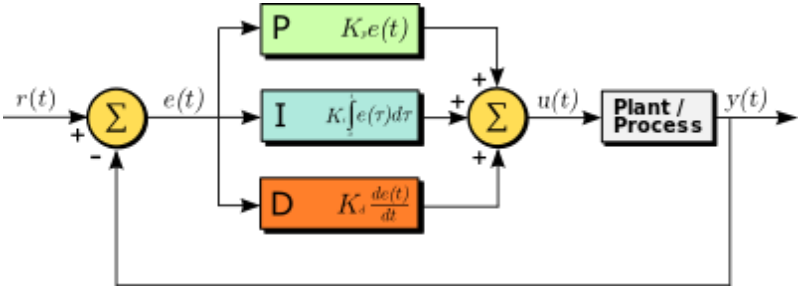
Gambar 2.11 *Absolute encoder* (“Position Sensors | Capacitive Inductive LVDT Rotary Encoder Working,” 2019)

2.7 Kontrol PID

Kontroler proporsional-integral-derivatif (*PID controller*) adalah mekanisme umpan balik loop kontrol generik yang banyak digunakan dalam sistem kontrol industri. Kontroler PID memperbaiki kesalahan antara *process variable* yang diukur dan *set point* yang diinginkan dengan menghitung dan kemudian mengeluarkan tindakan korektif yang dapat menyesuaikan proses tersebut (Ogata, 1997).

Perhitungan kontroler PID (algoritma) melibatkan tiga parameter terpisah; yaitu nilai proporsional, integral, dan derivatif. Nilai proporsional menentukan reaksi terhadap kesalahan saat ini, nilai integral menentukan reaksi berdasarkan jumlah kesalahan terbaru, dan nilai derivatif menentukan reaksi terhadap laju perubahan kesalahan. Penjumlahan dari ketiga tindakan ini digunakan untuk menyesuaikan proses melalui elemen kontrol seperti posisi *control valve*, atau kecepatan dan posisi motor.

Tuning dilakukan pada tiga konstanta dalam algoritma pengontrol PID, PID dapat memberikan aksi kontrol yang dirancang untuk persyaratan proses tertentu. Respon pengontrol dapat diuraikan dalam kaitannya dengan responsifnya pengontrol terhadap suatu kesalahan, sejauh mana pengontrol melampaui titik setel dan tingkat osilasi sistem.



Gambar 2.12 Kontroler PID (“How Does a PID Controller Work - Structure & Tuning Methods,” 2019)

Adapun persamaan kontroler PID yaitu :

$$u(t) = K_p \left(e(t) + \frac{1}{T_i} \int_0^t e(t)dt + T_d \frac{de(t)}{dt} \right) \qquad (2.22)$$

Keterangan :

$u(t)$: *output* dari kontroler PID atau *manipulated variable*

K_p : konstanta proporsional

T_i: konstanta integral

T_d: konstanta derivatif

$e(t)$: *error* (selisih antara *set point* dengan *level* aktual)

Persamaan kontroler PID (2.20) dapat juga dituliskan sebagai berikut :

$$u(t) = K_p e(t) + K_i \int_0^t e(t)dt + K_d \frac{de(t)}{dt} \qquad (2.23)$$

Dengan :

$K_i = K_p \times \frac{1}{T_i}$ dan $K_d = K_p \times T_d$

Beberapa aplikasi mungkin memerlukan hanya menggunakan satu atau dua mode untuk memberikan kontrol sistem yang sesuai. Ini dicapai dengan mengatur gain dari output kontrol yang tidak diinginkan ke nol. Pengontrol PID akan disebut pengontrol PI, PD, P atau I jika tidak ada tindakan kontrol masing-masing. Pengontrol PI sangat umum, karena tindakan turunan sangat sensitif terhadap kebisingan pengukuran, dan tidak adanya nilai integral mencegah sistem mencapai nilai targetnya karena tindakan kontrol.

2.8 Motor DC

Motor DC adalah motor listrik yang memerlukan suplai tegangan arus searah pada kumparan medan untuk diubah menjadi energi gerak mekanik. Kumparan medan pada motor dc disebut stator (bagian yang tidak berputar) dan kumparan jangkar disebut

rotor (bagian yang berputar). Motor arus searah, sebagaimana namanya, menggunakan arus langsung yang tidak langsung/*direct-unidirectional* ("Teori Motor DC Dan Jenis-Jenis Motor DC," 2019).

Motor DC adalah piranti elektronik yang mengubah energi listrik menjadi energi mekanik berupa gerak rotasi. Pada motor DC terdapat jangkar dengan satu atau lebih kumparan terpisah. Tiap kumparan berujung pada cincin belah (komutator). Dengan adanya insulator antara komutator, cincin belah dapat berperan sebagai saklar kutub ganda (*double pole, double throw switch*). Motor DC bekerja berdasarkan prinsip gaya Lorentz, yang menyatakan ketika sebuah konduktor beraliran arus diletakkan dalam medan magnet, maka sebuah gaya (yang dikenal dengan gaya Lorentz) akan tercipta secara ortogonal diantara arah medan magnet dan arah aliran arus.

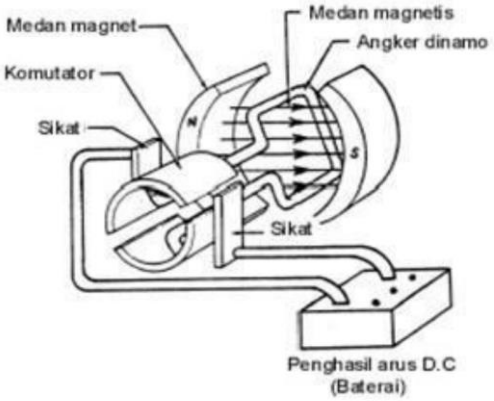
2.8.1 Komponen Utama Motor DC

Motor DC tersusun dari dua bagian yaitu bagian diam (stator) dan bagian bergerak (rotor) ("Motor DC, Pengertian, Karakteristik, Bagian & Jenis Motor DC," 2019). Stator motor arus searah adalah badan motor atau kutub magnet (sikat-sikat), sedangkan yang termasuk rotor adalah jangkar lilitanya. Pada motor, kawat penghantar listrik yang bergerak tersebut pada dasarnya merupakan lilitan yang berbentuk persegi panjang yang disebut kumparan.

Tiga bagian utama pada motor DC yaitu :

a. Kutub medan. Motor DC sederhana memiliki dua kutub medan, kutub utara dan kutub selatan. Garis magnetik energi membesar melintasi ruang terbuka di antara kutub-kutub dari utara ke selatan. Untuk motor yang lebih besar atau lebih kompleks terdapat satu atau lebih elektromagnet.

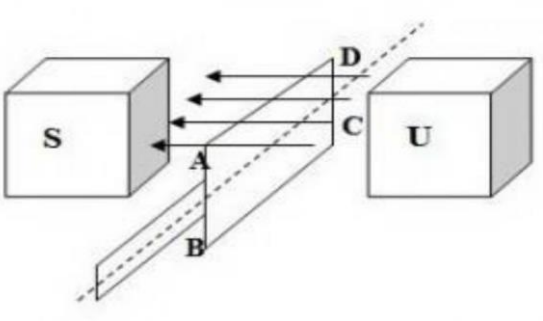
b. Kumparan motor DC. Bila arus masuk menuju kumparan motor DC, maka arus ini akan menjadi elektromagnet. kumparan motor DC yang berbentuk silinder, dihubungkan ke as penggerak untuk menggerakan beban. Untuk kasus motor DC yang kecil, kumparan motor DC berputar dalam medan magnet yang dibentuk oleh kutub-kutub, sampai kutub utara dan selatan magnet berganti lokasi. Jika hal ini terjadi, arusnya berbalik untuk merubah kutub-kutub utara dan selatan kumparan motor DC.
c. Komutator motor DC. Komponen ini terutama ditemukan dalam motor DC. Kegunaannya adalah untuk membalikan arah arus listrik dalam kumparan motor DC dan juga membantu dalam transmisi arus antara kumparan motor DC dan sumber daya.



Gambar 2.13 Bagian-bagian pada motor DC (“Motor DC, Pengertian, Karakteristik, Bagian & Jenis Motor DC,” 2019)

2.8.2 Prinsip Kerja Motor DC

Kumparan ABCD terletak dalam medan magnet serba sama dengan kedudukan sisi aktif AD dan CB yang terletak tepat lurus arah fluks magnet (“Prinsip Kerja Motor DC,” 2019). Sedangkan sisi AB dan DC ditahan pada bagian tengahnya, sehingga apabila sisi AD dan CB berputar karena adanya gaya Lorentz, maka kumparan ABCD akan berputar. Hasil perkalian gaya dengan jarak pada suatu titik tertentu disebut momen, sisi aktif AD dan CB akan berputar pada porosnya karena pengaruh momen putar.



Gambar 2.14 Prinsip kerja motor DC (“Prinsip Kerja Motor DC,” 2019)

Pada daerah dibawah kutub-kutub magnet besarnya momen putar tetap karena besarnya gaya lorentz. Hal ini berarti bahwa kedudukan garis netral sisi-sisi kumparan akan berhenti berputar. Supaya motor dapat berputar terus dengan baik, maka perlu ditambah jumlah kumparan yang digunakan. Kumparan-kumparan harus diletakkan sedemikian rupa sehingga momen putar yang dialami setiap sisi kumparan akan saling membantu dan menghasilkan putaran yang baik. Dengan pertimbangan teknis, maka kumparan-kumparan yang berputar tersebut dililitkan pada suatu alat yang disebut jangkar, sehingga lilitan kumparan itupun disebut lilitan jangkar.

2.9 Modul FC-03

Modul FC-03 adalah sensor *optocoupler* yang terintegrasi dengan komparator LM393 dalam satu modul. Modul ini dapat digunakan untuk menghitung pulsa dan mengukur kecepatan motor ("Motor Encoder RPM Speed Counter Interrupter Sensor Module FC-03 | QQ Online Trading," 2019). Penggunaannya dilakukan dengan menempatkan poros *encoder* di antara *optocoupler* untuk dihitung lubang-lubang pada poros *encoder*.

Gambar 2.15 Modul sensor FC-03 ("Motor Encoder RPM Speed Counter Interrupter Sensor Module FC-03 | QQ Online Trading," 2019)

Modul ini memiliki fitur sebagai berikut:

a. Arus : 15mA
b. Tegangan Kerja : DC 3.3V – 5V
c. Sinyal Output : *Output Digital Switching* (0 dan 1)
d. Dimensi : 3.2 cm x 1.4cm

Pin Out :

a. VCC = *Pin supply* tegangan 3.3V – 5 V
b. GND = *Ground*
c. D0 = TTL *switch* sinyal output
d. A0 = N/A

2.10 Mikrokontroler Arduino Mega 2560

Mikrokontroler adalah suatu *chip* berupa IC (*Integrated Circuit*) yang dapat menerima sinyal *input*, mengolahnya dan memberikan sinyal *output* sesuai dengan program yang diisikan ke dalamnya ("What Is a Microcontroller? The Defining Characteristics and Architecture of a Common Component - Technical Articles," 2019). Sinyal *input* mikrokontroler berasal dari sensor yang merupakan informasi dari lingkungan sedangkan sinyal *output* ditujukan kepada actuator yang dapat memberikan efek ke lingkungan. Jadi secara sederhana mikrokontroler dapat diibaratkan sebagai otak dari suatu perangkat yang mampu berinteraksi dengan lingkungan sekitarnya.

Mikrokontroler pada dasarnya adalah komputer dalam satu *chip*, yang di dalamnya terdapat mikroprosesor, memori, jalur *input* dan *output* (I/O), dan perangkat pelengkap lainnya("What is a Microcontroller? - Definition from Techopedia," 2019). Kecepatan pengolahan data pada mikrokontoler lebih rendah jika dibandingkan dengan PC (*Personal Computer*). Pada PC kecepata mikroprosesor yang digunakan saat ini telah mencapai orde GHz, sedangkan kecepatan operasi mikrokontroler pada umumnya berkisar antara 1-16 MHz. Begitu juga kapasitas RAM dan ROM pada PC yang bisa mencapai orde *gigabyte*, dibandingkan dengan mikrokontroler yang hanya berkisar pada orde *byte* hingga *kilobyte*.

Arduino Mega 2560 adalah papan mikrokontroler ATmega2560, berdasarkan *datasheet* memiliki 54 digital pin *input*/*output*, yang mana 15 pin dapat digunakan sebagai *output* PWM, 16 *analog input*, 4 pin UART (*hardware port serial*), osilator kristal 16 MHz, koneksi USB, jack listrik, *header* ICSP, dan tombol reset ("Arduino - ArduinoMega2560," 2019). Ini berisi semua yang diperlukan untuk mendukung mikrokonroler, hanya menghubungkannya ke komputer dengan kabel USB atau *power*

supply dengan adaptor AC-DC atau baterai. Arduino Mega kompatibel dengan sebagian besar *shield*, dirancang untuk Arduino Dueminalove atau Diecimilia. Arduino Mega 2560 R3 adalah versi terbaru dari Arduino Mega 2560. Kelebihan lain pada Arduino Mega 2560 yaitu tidak menggunakan *chip driver* FTDI USB-to-serial ("Arduino Mega 2560 Board: Specifications, and Pin Configuration," 2019).

Gambar 2.16 Arduino Mega 2560 ("Arduino - ArduinoMega2560," 2019)

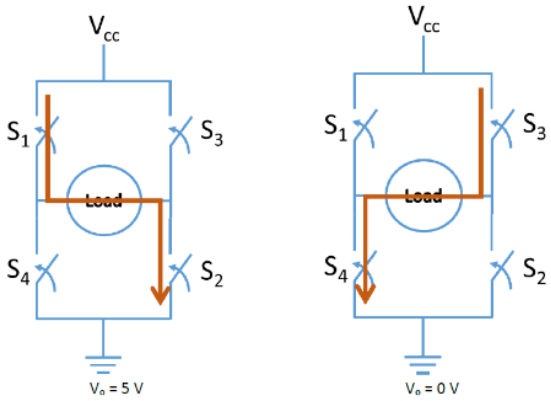
Tabel 2.1 Spesifikasi Arduino Mega 2560

No.	Deskripsi	Spesifikasi
1.	Mikrokontroler	ATmega 2560
2.	Tegangan kerja	5V
3.	*Input voltage (recommended)*	7-12V
4.	*Input voltage (maxium)*	6-20V
5.	*Digital input/output* (I/O)	54 pin (14 PWM)
6.	*Analog input*	16 pin
7.	DC *current per* I/O	40mA
8.	DC *current* 3,3V	50mA
9.	*Flash memory* 32KB	8 KB *bootloader*
10.	SRAM	8 KB
11.	EEPROM	4 KB
12.	*Clock speed*	16 MHz

2.11 *Driver* L298N

Driver motor DC L298N merupakan driver motor DC yang dapat digunakan untuk mengatur kecepatan motor dan mengendalikan arah putaran motor ("In-Depth: Interface L298N DC Motor Driver Module with Arduino," 2019). Komponen utamanya adalah IC L298, pada IC tersebut terdapat rangkaian *H-bridge* yang merupakan rangkaian elektronik yang memungkinkan tegangan masuk ke beban dengan arah yang berbeda.

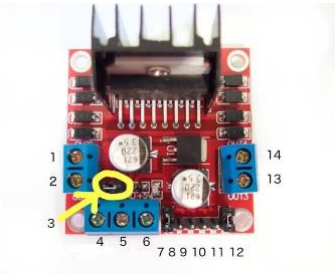
Prinsip kerja *H–bridge* yang sangat sederhana ("H-Bridges – the Basics | Modular Circuits," 2019) dapat dilihat pada Gambar 2.17 Apabila saklar S1 dan S2 aktif maka arus akan mengalir dari +V menuju S1 dan menggerakan motor searah jarum jam. Sedangkan apabila S3 dan S4 yang aktif arus akan +V menuju S3 maka motor akan bergerak berlawanan dengan jarum jam. Untuk membuat motor bergerak *counter clockwise* dan *clockwise* rangkaian *H-bridge* hanya mengatur polaritas yang diberikan kepada motor untuk mengatur arah putarannya.



Gambar 2.17 Cara kerja H-Bridge *clockwise* dan *counter-clockwise* ("H-Bridges – the Basics | Modular Circuits," 2019)

Driver L298N saat ini sudah tersedia modul *driver* motor, memiliki empat pin IN1, IN2, IN3 dan IN4 untuk menerima masukan logika berupa logika 1 dan 0 dari mikrokontroler yang digunakan (“Arduino DC Motor Control Tutorial - L298N | PWM | H-Bridge - HowToMechatronics,” 2019). Terdapat pin ENA dan ENB untuk mengaktifkan *driver* motor A dan B dengan memberikan nilai PWM. Pada modul *driver* motor juga terdapat regulator 5V untuk menghasilkan tegangan +5V yang berasal dari input +12V. Pada Gambar 2.18 menunjukan modul *driver* L298N sebagai berikut.

(a.)



(b.)

Gambar 2.18 (a.) *Driver* L298N
(b.) Konfigurasi pin modul *driver* L298N
(“Arduino DC Motor Control Tutorial - L298N | PWM | H-Bridge - HowToMechatronics,” 2019)

Konfigurasi pin modul *driver* L298N ditampilkan pada tabel sebagai berikut :

Tabel 2.2 Konfigurasi pin modul *driver* L298N

No. Pin	**Keterangan**
1.	OUT1 (Motor 1)
2.	OUT2 (Motor 1)
3.	Jumper 12V (untuk input > 12V)
4.	Input +12V
5.	Ground
6.	Input +5V
7.	ENA
8.	IN1
9.	IN2
10.	IN3
11.	IN4
12.	ENB
13.	OUT3 (Motor 2)
14.	OUT4 (Motor 2)

Spesifikasi modul *driver* L298N ditampilkan pada tabel sebagai berikut :

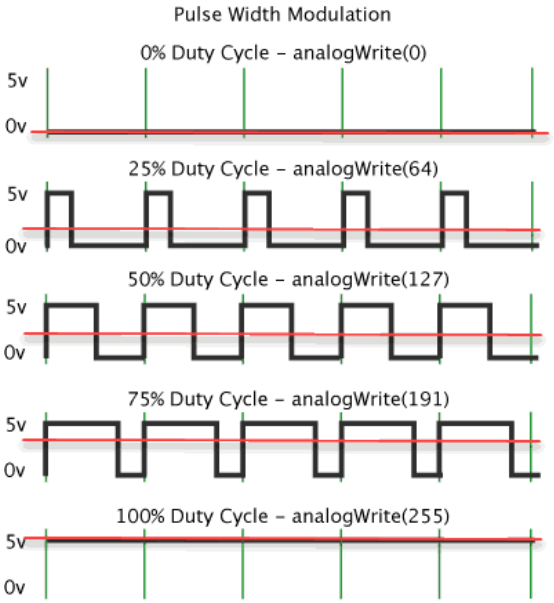
Tabel 2.3 Spesifikasi modul *driver* L298N

Bagian	**Spesifikasi**
Double H-Bridge Drive Chip	L298N
Logical voltage	5V
Drive voltage	5V-35V
Logical current	0-36mA
Drive current	2A (maks.)
Daya maksimal	25W
Dimensi	43x43x26mm
Massa	26 gram

2.12 *Pulse Width Modulation* (PWM)

Pulse Width Modulation (PWM) adalah metode yang sering digunakan untuk mengatur nilai tegangan rata-rata dengan modulasi lebar pulsa ("What is a Pulse Width Modulation (PWM) Signal and What is it Used For? - National Instruments," 2019). PWM merupakan salah satu cara untuk mengatur kecepatan motor DC dengan mengatur tegangan rata-rata yang masuk ke dalam motor tersebut, tegangan yang diberikan berupa gelombang segi empat dengan *duty cycle* tertentu.

Duty cycle merupakan perbandingan antara waktu pulsa tegangan aktif terhadap periodenya. Beberapa contoh bentuk PWM untuk beberapa nilai *duty cycle* dapat dilihat pada Gambar 2.19.



Gambar 2.19 PWM dengan *duty cycle* yang berbeda-beda ("Pulse Width Modulation - learn.sparkfun.com," 2019)

Pengaturan kecepatan motor dengan metoda PWM ini dilakukan dengan mengatur *duty cycle* yang di berikan. *Duty cycle* didapat dengan Persamaan (2.24) berikut ini:

$$Duty\ cycle = \frac{t_{on}}{T} \times 100\% \quad (2.24)$$

Keterangan :

t_{on} = panjang gelombang aktif

T = periode gelombang

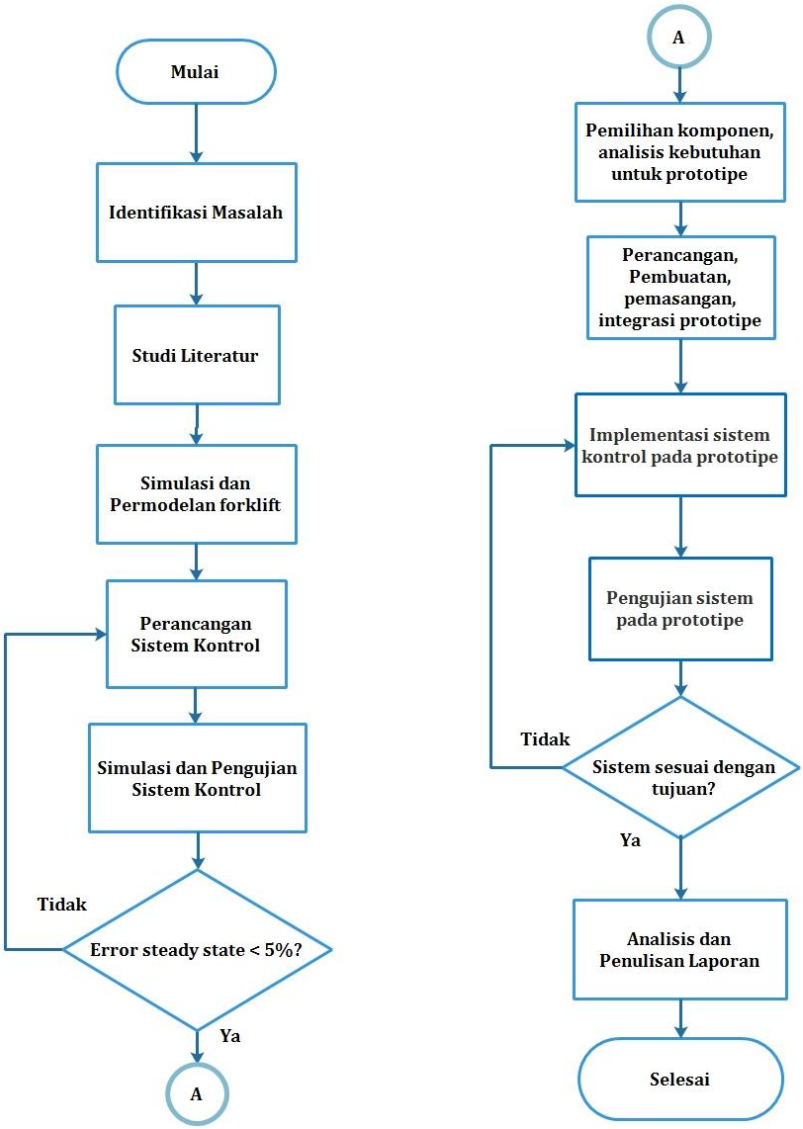
Pengaturan kecepatan motor dengan PWM harus memperhatikan frekuensi dari PWM yang digunakan. Frekuensi ini berhubungan dengan respon frekuensi dari motor DC yang digunakan ("Frekuensi, Duty Cycle, PWM," 2019). Frekuensi PWM yang terlalu tinggi biasanya mengakibatkan motor tidak merespon terhadap tengangan PWM yang masuk karena tegangan yang rapat.

BAB III
METODOLOGI PENELITIAN

Metodologi yang digunakan untuk mencapai tujuan dari tugas akhir ini dijelaskan melalui diagram alir yang ditunjukkan pada Gambar 3.1 yang mendeskripsikan tahap-tahap penelitian yang dimulai dari mengindentifikasi masalah lalu studi literatur. Kemudian membuat pemodelan berserta simulasi dari *unmanned autonomous forklift* yang hendak dibuat dan diteliti. Pemodelan menentukan dalam perancangan sistem kontrol, lalu disimulasikan dan diujikan sistem kontrol hingga tercapai kriteria *error steady-state* kurang dari 5%. Setelah perancangan dan pengujian sistem kontrol untuk *forklift*, dilakukan analisis kebutuhan, pemilihan komponen dilanjut dengan perancangan prototipe. Perancangan prototipe direalisasikan dengan pembuatan, pemasangan dan integrasi sistem secara keseluruhan baik dari segi mekanik dan elektronik. Ketika semua telah terpasang dan terintegrasi, sistem kontrol pada prototipe *unmanned autonomous forklift* diimplementasikan lalu diuji dengan jarak tertentu. Selama diuji tentu didapatkan data-data pengujian untuk prototipe *unmanned autonomous forklift*, setelah didapat tadi, data-data tersebut diolah untuk diinterpretasikan sebagai informasi dari performa dan keberhasilan prototipe *unammned autonomous forklift*.

3.1 Studi Literatur

Kajian studi dilakukan terhadap permasalahan yang ada menggunakan metode studi literatur pada penelitian-penelitian yang telah dilakukan sebelumnya. Beberapa referensi menjelaskan tentang rancang bangun dan pengembangan *unmanned autunomous forklift* yang bervariasi (Abdellatif dkk., 2018), (Tamba, Hong, & Hong, 2009).



Gambar 3.1 Bagan alir metodologi penelitian

3.2 Pemodelan *Forklift*

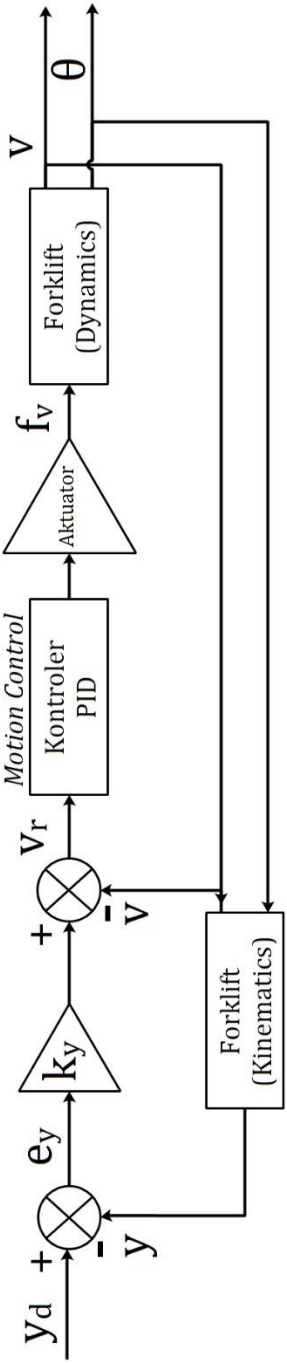
Pemodelan forklift diambil dari referensi model *Forklift-Like Wheeled Robot* (Lee, 2014), sebagaimana pemodelan kinematik dan dinamiknya diturunkan untuk *forklift* dengan empat roda. Pada robot ini merupakan tipe *car-like* di mana sistemnya *non-linear* dan *coupled*. Secara kinematis, laju waktu *heading angle* proporsional terhadap kecepatan mengemudi dan sudut tangen dari *steering angle*. Sehingga, *heading angle car-like* robot tidak dapat dikendalikan ketika kecepatan mengemudinya nol. Jadi *steering angle* juga kecepatan mengemudi harus dikendalikan bersamaan untuk memenuhi kendali arah dari *car-like* robot.

Pemodelan *forklift-like mobile* robot hampir mirip dengan *car-like mobile* robot. Perbedaan utamanya adalah sistem *rear-steering*, sedangkan *car-like* robot menggunakan sistem *front-steering*. Dinamika dari *car-like* robot dan *forklift-like* robot yaitu *non-linear* dan *coupled*. *Forklift-like* robot juga termasuk merupakan dalam *non-minimum-phase dynamic systems*, yang mana membuat kontrolnya semakin kompleks dan rumit.

3.3 Perancangan, Simulasi, dan Pengujian Sistem Kontrol

3.3.1 Desain *motion control*

Gambar 3.2 merupakan diagram blok *motion control* untuk prototipe *unmanned autonomous forklift*. Masing-masing blok memiliki persamaan yang diambil dari referensi (Lee, 2014). Pada blok kinematika *forklift* (*forklift kinematics*) terdapat persamaan (2.1), (2.2), dan (2.3) yang merupakan persamaan kinematika dari *forklift-like* robot itu sendiri. *Forklift* bergerak pada lintasan yang lurus, dengan parameter l = 0.12 m, maka nilai $\theta = 90^o$, dengan demikian



Gambar 3.2 Diagram blok *motion control* prototipe *forklift*

$$\dot{x} = 0 \tag{3.1}$$

$$\dot{y} = v \tag{3.2}$$

$$\dot{\theta} = 0 \tag{3.3}$$

Dinamika *forklift* (*forklift dynamics*) digunakan sebagai *plant* dengan persamaan (2.15) dan (2.16) yang mana kedua persamaan ini merupakan penyederhanaan dari persamaan (2.4). Variabel v_r didapat dari *trajectory generator*, yang berfungsi sebagai pembangkit trayektori secara *real-time* mengacu pada persamaan (2.13) yaitu

$$v_r = k_y \times e_y \tag{3.4}$$

Blok *motion control* menggunakan persamaan PID pada umumnya, yaitu persamaan (2.23) atau dituliskan kembali sebagai berikut

$$f_v = K_p e(t) + K_i \int_0^t e(t)dt + K_d \frac{de(t)}{dt} \tag{3.5}$$

Dengan persamaan *e* yaitu

$$e = v_r - v \tag{3.6}$$

Sinyal kontrol padap prototipe tidak langsung berupa gaya atau *driving force* (f_v) akan tetapi berupa pulsa motor sehingga perlu ditambahkan fungsi transfer berupa *gain* aktuator, karena perubahan yang cepat dan *time constant*-nya kecil sekali diasumsikan fungsi transfer sebagai orde 0 menjadi *gain* saja. Parameter pada *gain* aktuator diberi nilai sebesar 200 yang didapatkan dari *trial* dan *error*.

3.3.2 Simulasi dan Pengujian Sistem Kontrol

Setelah dari penurunan persamaan secara matematis, dilakukan simulasi pemodelan pada aplikasi Matlab Simulink. Simulasi mengikuti sesuai dengan diagram blok *motion control*

pada Gambar 3.2. Sehingga persamaan-persamaan pada Gambar 3.2 ditransformasikan dalam bentuk blok-blok, subsistem di Simulink. Diagram blok Simulink secara keseluruhan ada pada lampiran.

Ketika melakukan simulasi dilakukan pengujian terhadap sistem kontrol untuk memenuhi kriteria sistem dengan *error steady-state* kurang dari 5%. Maka dari itu dilakukan *fine tuning* selama pengujian terhadap sistem kontrol hingga mendapatkan parameter Kp, Ki, Kd yang sesuai dengan kriteria sebagaimana yang telah disebutkan. Performa sistem kontrol juga diuji dilihat dari jarak yang diinginkan (*setpoint*) dengan hasil jarak yang telah ditempuh.

3.4 Pemilihan Komponen, Analisis Kebutuhan Untuk Prototipe

Setelah simulasi dilakukan, didapatkan data-data dari simulasi yang berguna untuk menentukan komponen, alat-alat yang dibutuhkan pada sistem.

Adapun komponen-komponen utama yang dibutuhkan dalam pembuatan prototipe *unammned autonomous forklift* ini adalah :

a. Prototipe robot *car-like* 4WD beserta roda bannya
b. Motor DC disertai *gearbox*, 4 buah (untuk kontrol posisi cukup 2 buah)
c. Mikrokontroler Arduino Mega 2560, 1 buah
d. Modul FC-03, 2 buah
e. Driver L298N, 1 buah

Pada praktiknya, digunakan robot *car-like* 4WD sebagai prototipe sistemnya, dikarenakan pencarian *rotary encoder* yang sesuai dengan kondisi fisik prototipe *forklift* dan juga apabila dilakukan pemasangan *rotary encoder* pun dikhawatirkan akan merusak prototipe *forklift.* Selain itu dibutuhkan pula tambahan sensor lainnya beserta rangkaian penguatnya. Dikarenakan

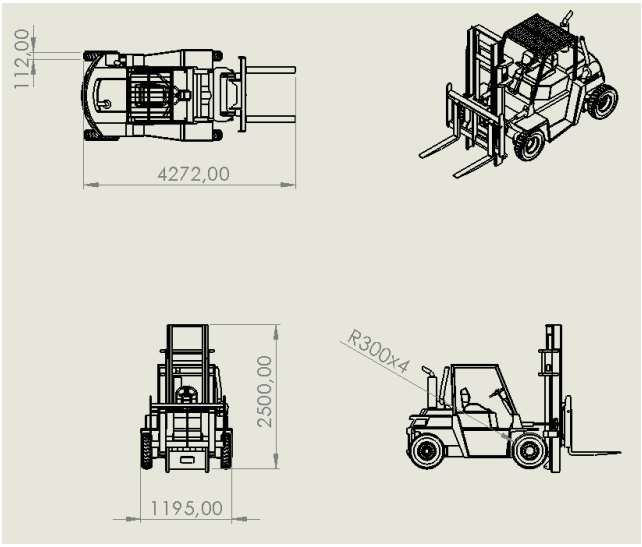
modelnya tidak persis sama, kesamaannya hanya pada bagian depan yaitu peletakan motor akselerasi, sehingga fungsi *rear-steering* tidak bisa diaplikasikan. Oleh karena itu sistem hanya bisa bergerak maju (ataupun mundur jika programnya disesuaikan) sesuai jarak yang telah ditentukan.

3.5 Perancangan, Pembuatan, Pemasangan, dan Integrasi Prototipe

3.5.1 Perancangan mekanik

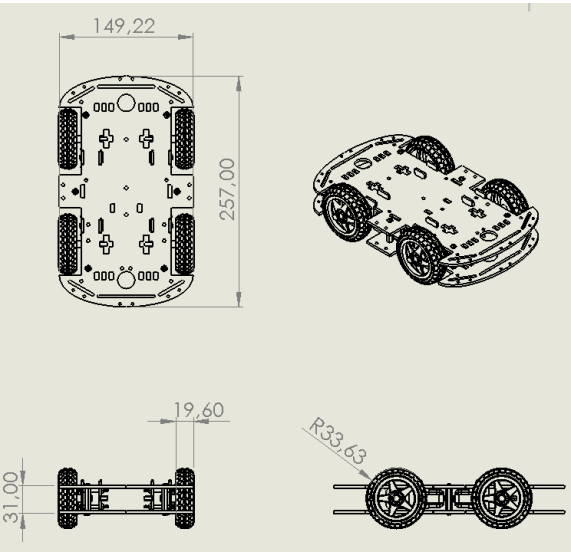
Berikut merupakan gambar tiga dimensi *computer-aided design* dari prototipe. Sebagaimana yang telah dijelaskan pada subbab 3.4, dengan segala keterbatasan yang ada, pada tugas akhir ini prototipe *forklift* pada Gambar 3.3 dirupakan dengan *car-like* robot 4WD pada Gambar 4.4. Persamaan yang digunakan dalam *plant forklift* yaitu kinematika pada persamaan (2.1), (2.2), (2.3) dan dinamika pada persamaan (2.15) dan (2.16)

a. Prototipe *forklift*
   Panjang : 4,27 m
   Lebar : 1,19 m
   Tinggi : 2,5 m
   Massa : 1000 kg
   Diameter roda: 600 cm
   Jumlah motor : 3 buah
   Jumlah roda ban : 4 buah



Gambar 3.3 CAD *forklift*

b. *Car-like* robot 4WD
Panjang : 25,7 cm
Lebar : 14,92 cm
Tinggi : 3,1 cm
Massa : 670 gram
Diameter roda : 66,10 mm
Jumlah motor : 4 buah
Jumlah roda ban : 4 buah



Gambar 3.4 CAD *car-like* robot 4WD

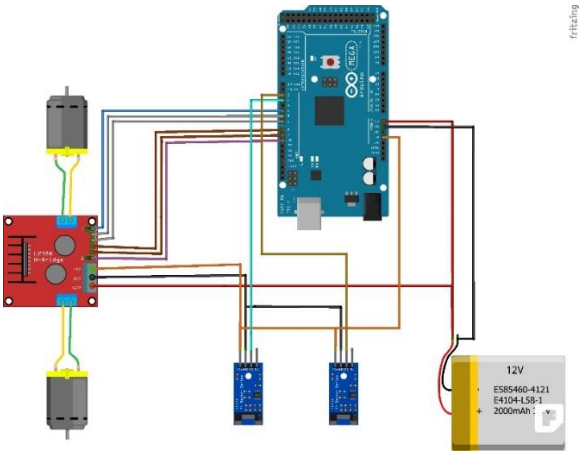
3.5.2 Perancangan elektronik

Sebelum dilakukan perancangan elektronik dilakukan analisa perencanaan komponen yang tersedia di dalam *forklift*.

Pada prototipe *unmanned autonomous forklift* didapatkan tiga buah motor penggerak, yaitu :

a. Motor depan sebagai akselerasi
b. Motor *fork*, untuk menangkat dan menurunkan barang
c. Motor belakang sebagai *steering* atau kemudi

Namun dikarenakan *rotary encoder* tidak dapat dipasang pada prototipe *forklift* melihat kondisi fisiknya, sehingga digunakan prototipe *car-like* robot 4WD. Dengan demikian cukup dibutuhkan dua buah motor di depan saja untuk pergerakan akselerasinya. Sehingga perancangan skematik elektroniknya seperti berikut.



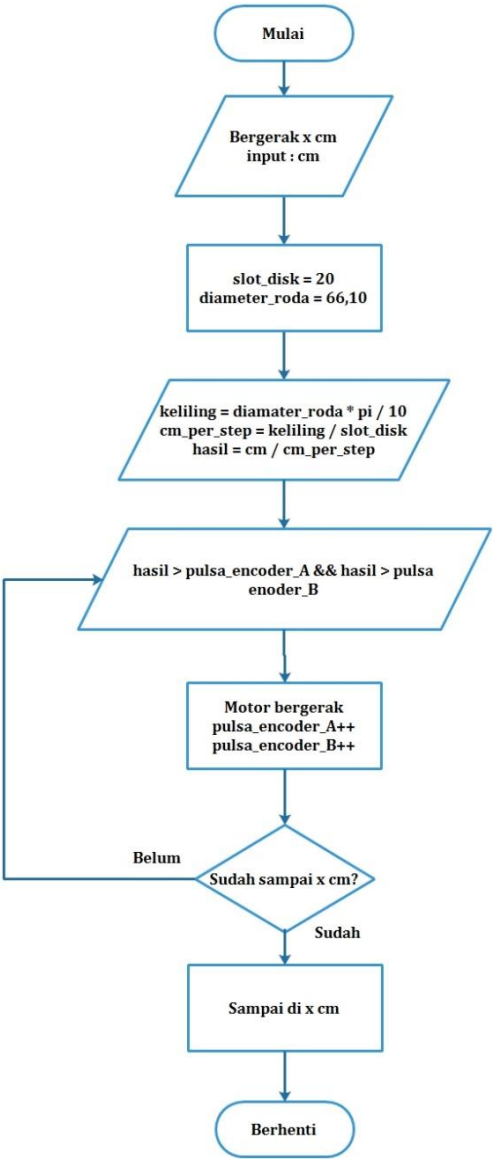
Gambar 3.5 Skematik elektronik

Tabel 3.1 Daftar masukan dan keluaran sistem

Input	**Output**
D10 Arduino	ENA
D9 Arduino	IN1
D8 Arduino	IN2
D5 Arduino	ENB
D7 Arduino	IN3
D6 Arduino	IN4
D3 Arduino	D0 FC-03 (kanan)
D2 Arduino	D0 FC-03 (kiri)
OUT1, OUT2	Motor +, Motor – (kanan)
OUT3, OUT4	Motor +, Motor – (kiri)
+12V	Vin Arduino, +12V driver
-12V	GND
5V Arduino	VCC FC-03, 5V driver
GND Arduino	GND FC-03, GND driver

3.5.3 Perancangan program/*software*

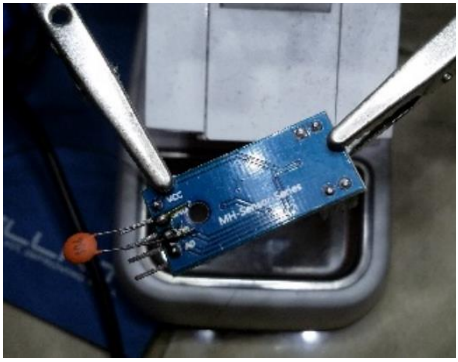
Program odometri dirancang untuk menuju pada posisi tertentu dengan jarak satuannya cm. Pada masukan program yaitu nilai jarak yang ingin ditempuh. Setelah itu program akan memproses di mana terdapat konstanta slot_disk enkoder yaitu bernilai 20, artinya 20 lubang pada disk *rotary encoder*. Diameter roda pun dimasukkan pada program yaitu sebesar 66,10 cm. Kemudian, memasuki pada program odometrinya. Untuk mendapatkan hasil tempuh harus diketahui keliling roda, kemudian dimasukkan ke centimeter per langkah yang rumusnya keliling roda dibagi jumlah lubang disk *rotary encoder*. Setelah itu program akan menghitung putaran *encoder* sesuai kalkulasi yang telah dibuat dan dimasukkan sebelumnya. Jika nilai hasil masih besar dari pulsa encoder, maka motor akan terus bergerak. Mikrokontroler akan melakukan iterasi terus menerus hingga nilai hasil kurang dari pulsa *encoder*. Jika nilai hasil sudah memenuhi, maka robot akan berhenti pada jarak yang telah ditentukan. Bagan alur terdapat pada Gambar 3.6.



Gambar 3.6 Bagan alir program odometri

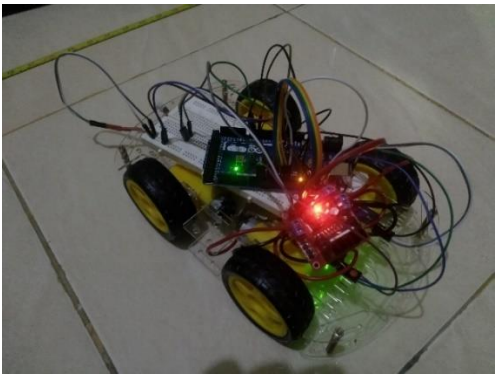
3.5.4 Pembuatan, pemasangan dan integrasi prototipe

Pembuatan, pemasangan, dan integrasi sistem dilakukan setelah melakukan segala tahap perancangan. Kalibrasi sensor dilakukan dengan menambahkan kapasitor sebagai *filter* untuk mengurangi sinyal *rebound* diilustrasikan pada Gambar 3.7.



Gambar 3.7 Penambahan kapasitor untuk mengurangi *rebound*

Proses mekanik, elektronik, dan program atau *software* diintegrasikan dan dikombinasikan ke prototipe *forklift* diilustrasikan pada Gambar 3.8.



Gambar 3.8 Prototipe yang telah diintegrasikan

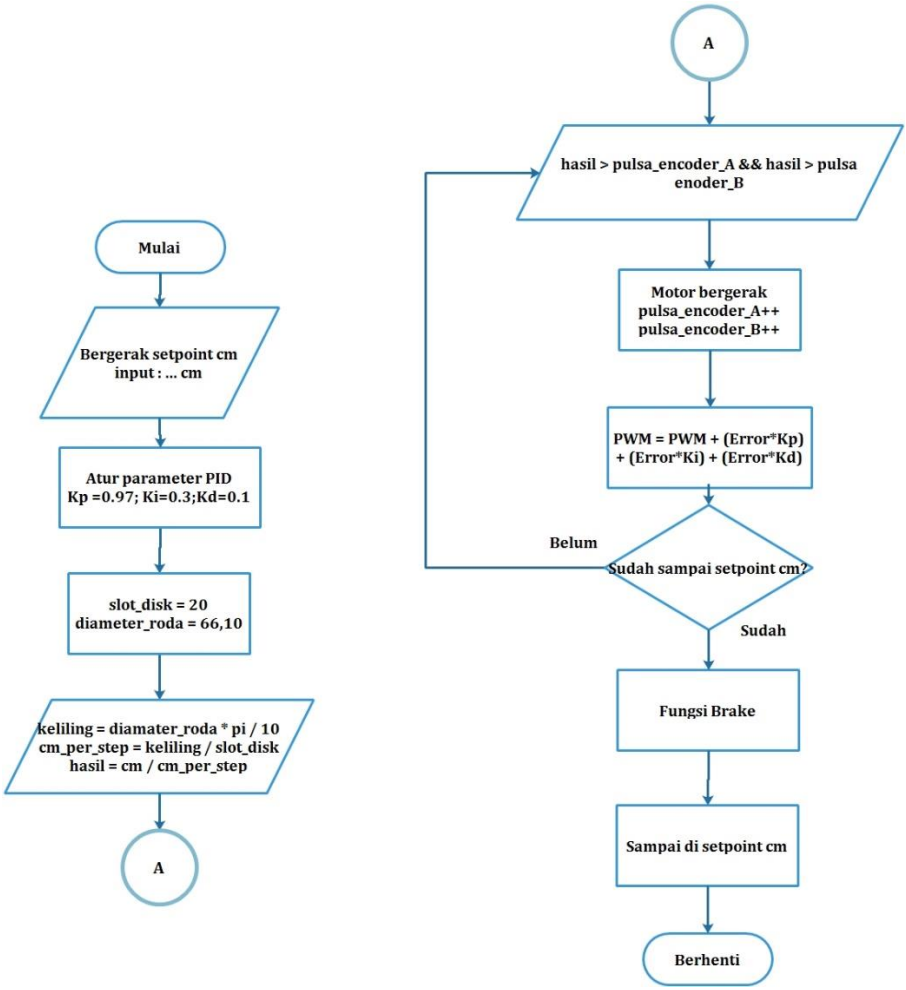
3.6 Implementasi Sistem Kontrol Pada Prototipe

Program odometri yang telah dibuat belum menggunakan tambahan berupa program kontroler PID, sehingga hasil keluaran dari program nilainya tidak ada kontrolernya, hanya mengacu pada nilai yang ada di pengulangan *while* program. Dengan ditambahkannya program kontroler PID pada program odometri yang sebelumnya, nilai *setpoint* dapat terjaga dengan kontroler yang telah diprogram. Implementasinya nilai-nilai parameter *gain* Kp, Ki, dan Kd dimasukkan. Kemudian nilai *setpoint*-nya juga dimasukkan. Program kontroler PID akan meghitung *error* dari pulsa-pulsa *encoder*, kontroler akan mengeksekusi pada PWM motor. Jika sudah mencapai *setpoint* yang diinginkan fungsi *brake* pada *driver* motor diaktifkan, sehingga *forklift* sampai pada jarak yang telah ditentukan. Bagan alur program sistem kontrol ditunjukkan pada Gambar 3.9.

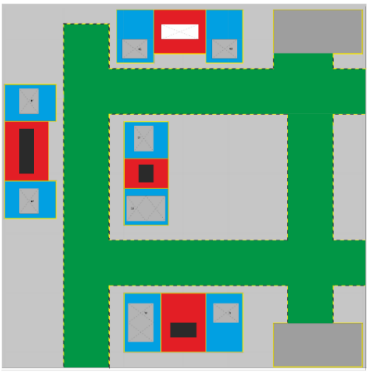
3.7 Pengujian Sistem

Pengujian sistem dilakukan dengan eksperimen pada prototipe *unmanned autonomous forklift* dengan metode-metode yang telah ditentukan, mengacu pada teori yang ada dan tinjauan pustaka.

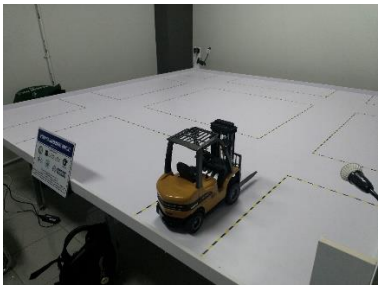
Pada penelitian tugas akhir ini yang dibuat dan diteliti adalah prototipe *forklift* sehingga pengujian pun dilakukan pada media yang lebih kecil. Prototipe *forklift* yang dirupakan *car-like* robot 4WD diuji pada meja miniatur pabrik dengan luas $2{,}4\ \text{m} \times 2{,}4\ \text{m}$ dengan desainnya ditunjukkan pada Gambar 3.10, implementasinya pada Gambar 3.11. *Workstation* yang ada pada meja pabrik disesuaikan dengan luasan yang ada. Sehingga di dapat 6 titik yaitu empat untuk tempat permesinan, satu *raw material*, satu tempat penyimpanan barang yang sudah selesai.



Gambar 3.9 Implementasi sistem kontrol pada prototipe



Gambar 3.10 Desain meja miniatur pabrik



Gambar 3.11 Hasil meja miniatur pabrik yang sudah jadi

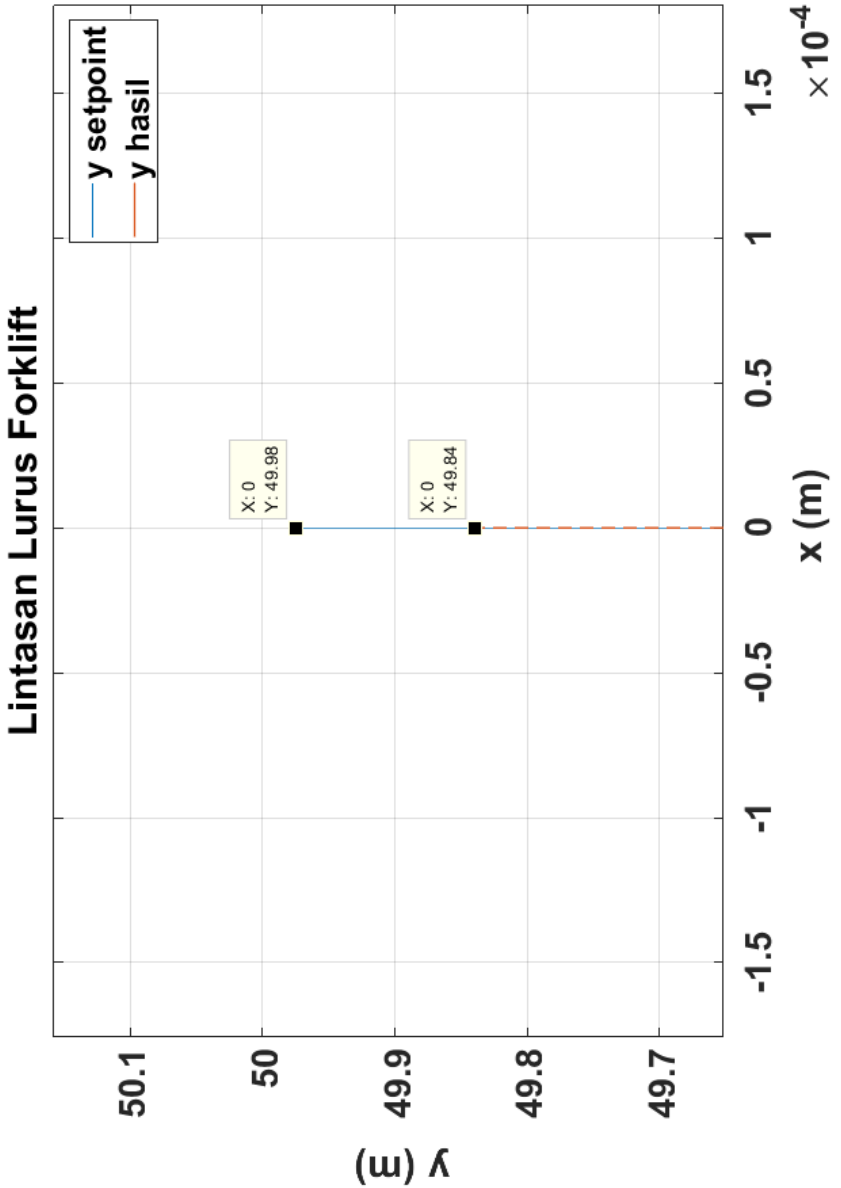
BAB IV
ANALISIS DATA DAN PEMBAHASAN

4.1 Hasil Simulasi Sistem Kontrol

Pada simulasi dimasukkan parameter-parameter kinematik dan dinamik untuk diproses di *workspace* menggunakan *.m file, agar dari Simulink dapat membaca parameter-parameternya. Parameter-parameter kinematik dan dinamik yang dimasukkan yaitu r=0,661; m=0,67; l=0,12; lc=0,06; Jb=0,3; Jh=0,2; Jv =0,1; dv=0,2; dphi=0,1. Selain parameter kinematik dan dinamik, dimasukkan juga nilai-nilai parameter *gain* kx=0,5; ky=0,5; kz=1; ktheta=1. Trayektori diatur dari *trajectory generator* dengan ketentuan-ketentuan $v_d = 5t\ m/s$ untuk $0 \leq t \leq 6\ s$ dan $v_d = 30\ m/s$ untuk $t \geq 6\ s$ dan $\theta_d = 0{,}2 * t$ untuk seluruh $t \geq 0\ s$. Trayekotri merupakan lintasan lurus dengan *setpoint* 50 meter. *Simulation stop time* pada Simulink diatur pada waktu 100 detik.

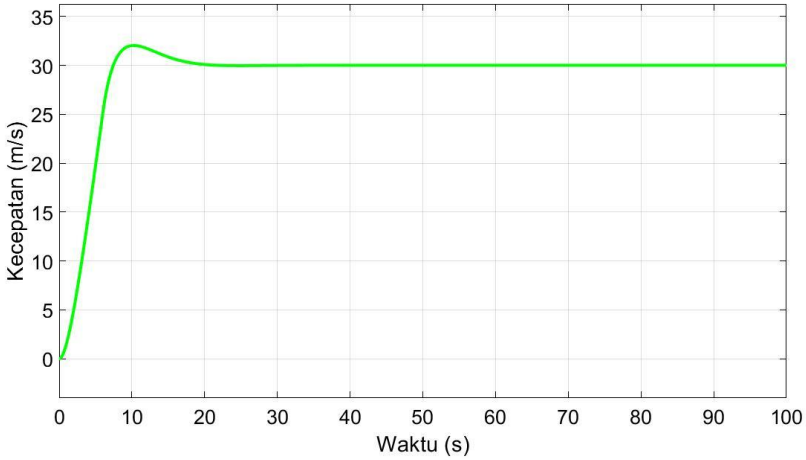
Fine tuning dilakukan pada *motion controller*, dengan konfigurasi parameter-parameter kontrol PID yaitu Kp=0,853253789; Ki=0,256576567; Kd=0,923253789. Hasil simulasi berupa gerakan lurus pada sumbu y.

Gambar 4.1 menunjukkan lintasan lurus *forklift* pada simulasi yang telah dijalankan. Hasil simulasi tersebut *error*-nya bernilai 0,28%. *Error* tersebut diperoleh dari selisih antara nilai *y setpoint*=49,98 m dengan *y* hasil=49,84 m.



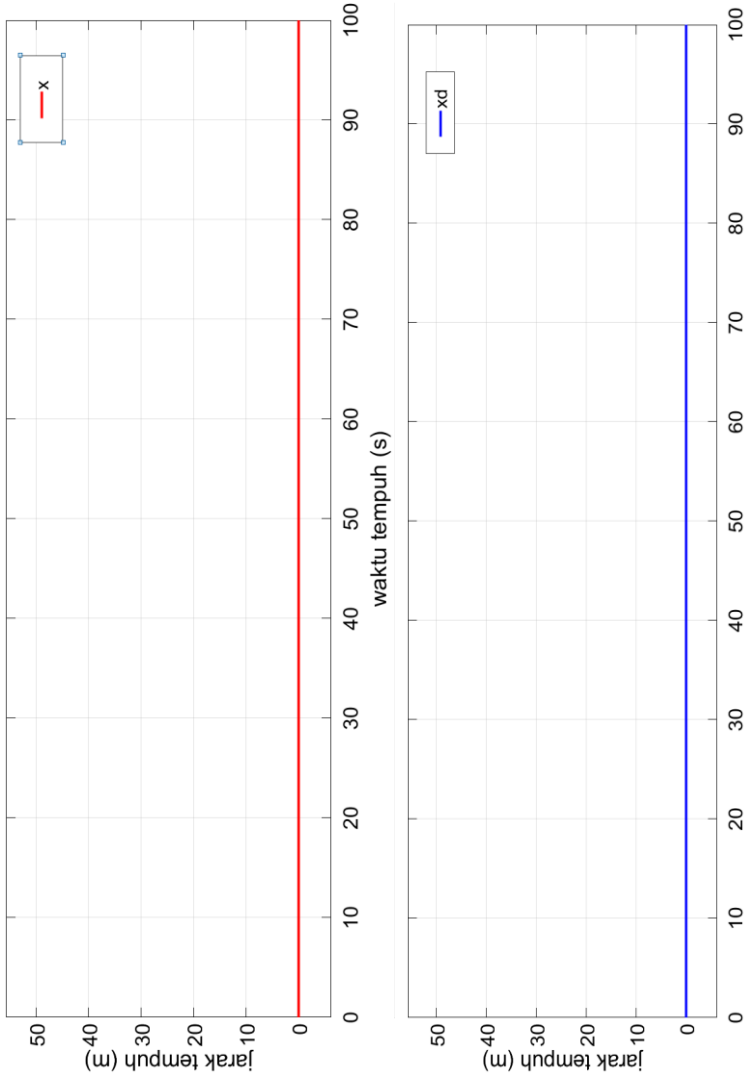
Gambar 4.1 Grafik lintasan lurus *forklift*

Respon kecepatan (*v*) *forklift* ditunjukkan pada Gambar 4.2. Karakterisik respon kecepatan *forklift*-nya yaitu: *delay time* 4,12 detik, *rise time* 7,46 detik, *peak time* 10,03 detik, *overshoot* 6,73%, *settling time* 23,09 detik

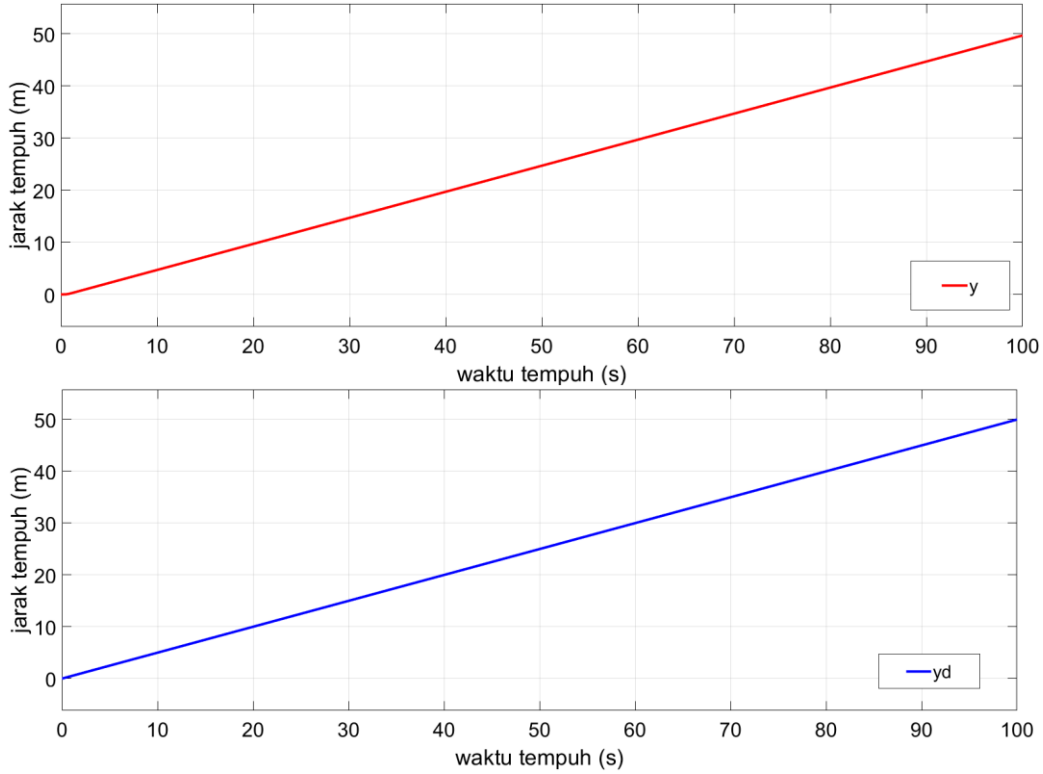


Gambar 4.2 Respon kecepatan *forklift*

Mengacu pada lintasan lurus yang berjarak 50 m, dengan waktu tempuh 100 detik, dapat diperoleh grafik jarak tempuh terhadap waktu tempuh pada Gambar 4.3 dan Gambar 4.4. Gambar 4.4 menunjukkan pada sumbu *y* didapatkan v = 0,5 m/s, sedangkan pada Gambar 4.3 menunjukkan sumbu *x* = 0 m/s (konstan) dikarenakan *forklift* tidak mengalami perubahan sudut, sesuai dengan lintasan lurus. Diperoleh perbandingan hasil yang sama, dari *setpoint* (nilai yang diinginkan) dan hasil perhitungan simulasi (hasil tempuh). Grafik dibandingkan terpisah karena hasilnya sangat berimpit.



Gambar 4.3 Grafik jarak tempuh terhadap waktu tempuh pada sumbu x

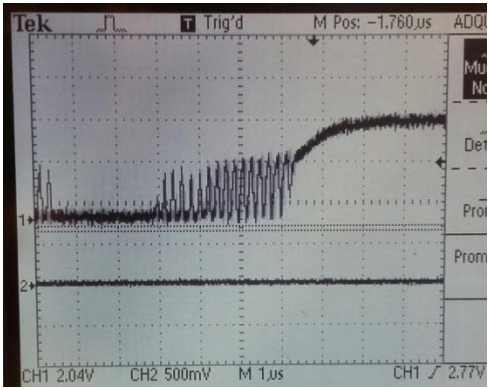


Gambar 4.4 Grafik jarak tempuh terhadap waktu tempuh pada sumbu *y*

4.2 Kalibrasi Sensor Modul FC-03

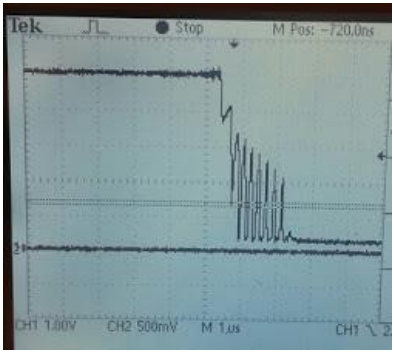
Permasalahan yang sering ditemui ketika menggunakan modul sensor FC-03 adalah pengondisian sinyalnya yang kurang baik. Hal lain juga ketika sensor ini dipasangkan dengan mikrokontroler Arduino Mega, sensornya membaca pulsa 4 kali lebih banyak daripada yang dihasilkan oleh enkodernya. Sensor ini sangat sensitif terhadap gangguan pada pin VCC dan GND. Oleh karena itu dibutuhkan *filter* untuk mengeliminasi gangguan tersebut.

Gambar 4.5 merupakan ilustrasi awal sinyal pulsa *rotary encoder* terlihat bahwa terdapat *rebound* pada saat kondisi transien, sehingga sebetulnya sinyal digital tidak sepenuhnya kotak sempurna. Data yang diterima oleh Arduino Mega2560 menjadi tidak sempurna juga.



Gambar 4.5 Pulsa di bagian awal terdapat *rebound*

Ini terjadi juga pada bagian akhir pulsa, ditunjukkan pada Gambar 4.6, terdapat *rebound* juga. Arduino sangat sensitif dan membaca *rebound* ini sebagai pulsa yang benar, yaitu diterima dan diproses oleh Arduino, padahal sebetulnya bukan merupakan pulsa yang benar.

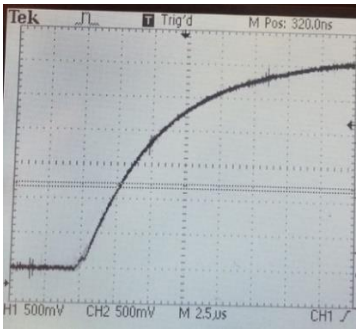


Gambar 4.6 Pulsa di bagian akhir terdapat *rebound*

Hal yang dilakukan untuk mengantisipasi *rebound* bisa dengan ditambahkannya kapastior 100nF (104) dan membuat program dengan *anti debouncing*. Dengan adanya kapasitor, sinyal-sinyal *rebound* dapat di-*filter*.

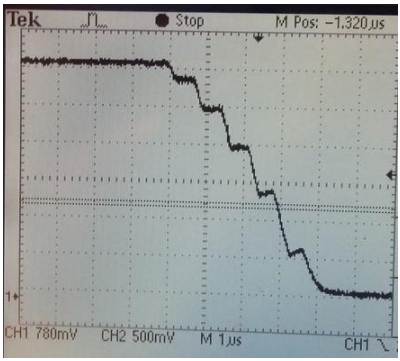
Efek setelah dilakukan penambahan kapasitor dan modifikasi program maka didapatkan hasil sebagai berikut :

Pulsa pada awal sinyal menjadi halus tidak ada banyak *noise* dan *spike* kecil-kecil yang disebabkan *rebound* ditunjukkan pada Gambar 4.7.



Gambar 4.7 Pulsa di bagian awal, *rebound*-nya hilang

Gambar 4.8 merupakan hasil dari *filter* kapasitor pada sensor, pulsa pada akhir sinyal *spike*-nya berkurang, jumlah *noise* berkurang juga dibandingkan dengan pulsa sinyal yang ada pada Gambar 4.6.



Gambar 4.8 Pulsa di bagian akhir, *rebound*-nya berkurang

4.3 Uji Jarak Pada Prototipe

Uji jarak dilakukan menggunakan komunikasi serial program. Penalaan parameter PID dengan cara *trial and error* didapatkan masing-masing parameter Kp=0,97; Ki=0,3; Kd=1.

Uji coba pada program dilakukan dengan melakukan komunikasi serial. Tujuan daripada diuji program menggunakan komunikasi serial agar dapat dilihat hasil *output* pada *serial monitor* Arduino dan divalidasi hasilnya.

Dilakukan uji jarak dengan variasi jarak 50 cm, 75 cm, 100 cm, 125 cm, 150 cm, 175 cm, dan 200 cm (memiliki interval 25 cm). Satu variasi jarak dilakukan percobaan sebanyak 5 (lima) kali. Perhitungan yang dilakukan yaitu untuk mendapatkan eror mutlak dan standar deviasinya. Diperoleh hasil sebagai berikut :

Tabel 4.1 Hasil uji jarak

No. [1]	Percobaan [2]	Jarak yang diinginkan (cm) [3]	Jarak sebenarnya (cm) [4]	Eror Mutlak (cm) [5]	Eror Mutlak (%) [6]	Rata-rata eror mutlak (cm) [7]	Rata-rata eror mutlak (%) [8]	Standar Deviasi [9]
1	1	50	55,1	5,1	10,2	1,82	3,64	1,97
	2	50	52,1	2,1	4,2			
	3	50	49,8	0,2	0,4			
	4	50	51,2	1,2	2,4			
	5	50	50,5	0,5	1			
2	1	75	73,2	1,8	2,4	1,84	3,68	0,91
	2	75	76,8	1,8	2,4			
	3	75	74,6	0,4	0,53			
	4	75	77,8	2,8	3,73			
	5	75	77,4	2,4	3,2			
3	1	100	103,2	3,2	3,2	1,8	3,6	1,042
	2	100	101,5	1,5	1,5			
	3	100	99,4	0,6	0,6			
	4	100	102,5	2,5	2,5			
	5	100	98,8	1,2	1,2			
4	1	125	126,7	1,7	1,36	1,84	3,68	1,258
	2	125	128,6	3,6	2,88			
	3	125	123,4	1,6	1,28			
	4	125	127,2	2,2	1,76			
	5	125	124,9	0,1	0,08			
5	1	150	151,1	1,1	0,73	1,86	3,72	0,680
	2	150	152,1	2,1	1,4			
	3	150	148,4	1,6	1,07			
	4	150	148,4	1,6	1,07			
	5	150	147,1	2,9	1,93			

Tabel 4.1 Lanjutan

[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]
6	1	175	175,4	0,4	0,23	1,88	3,76	1,130
	2	175	178,4	3,4	1,94			
	3	175	177,5	2,5	1,43			
	4	175	173,4	1,6	0,91			
	5	175	176,5	1,5	0,86			
7	1	200	201,5	1,5	0,75	1,82	3,64	1,047
	2	200	203,3	3,3	1,65			
	3	200	199,4	0,6	0,3			
	4	200	198,7	1,3	0,65			
	5	200	202,4	2,4	1,2			

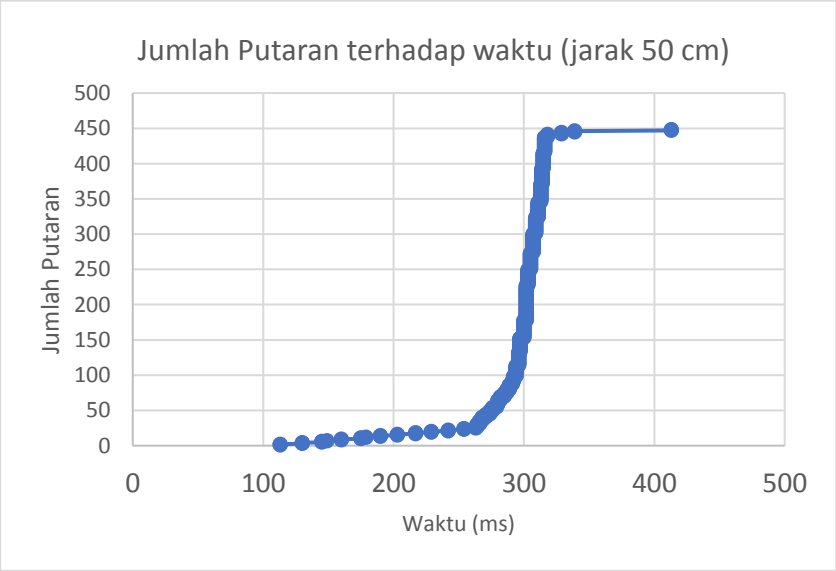
Diperoleh hasil rata-rata error uji jarak pada prototipe dengan variasi jarak 50 cm, 75 cm, 100 cm, 125 cm, 150 cm, 175 cm, 200 cm sebagai berikut :

a. Pada jarak 50 cm memiliki rata-rata eror mutlak 1,82 cm (3,64%) dengan standar devasi 1,97.
b. Pada jarak 75 cm memiliki rata-rata eror mutlak 1,84 cm (3,68%) dengan standar deviasi 0,91.
c. Pada jarak 100 cm memiliki rata-rata eror mutlak 1,8 cm (3,6%) dengan standar deviasi 1,042.
d. Pada jarak 125 cm memiliki rata-rata eror 1,84 cm (3,68%) dengan standar deviasi 1,258.
e. Pada jarak 150 cm memiliki rata-rata eror mutlak 1,86 cm (3,72%) dengan standar deviasi 0,680.
f. Pada jarak 175 cm memiliki rata-rata eror 1,88 cm (3,76%) dengan standar deviasi 1,130.
g. Pada jarak 200 cm memiliki rata-rata eror mutlak 182 cm (3,64%) dengan standar deviasi 1,047.

Hasil rata-rata eror mutlak keseluruhan percobaan 1,84 cm (3,67%). Dari semua variasi jarak dapat diambil rentang rata-rata eror mutlak berkisar pada 1,8-1,88.

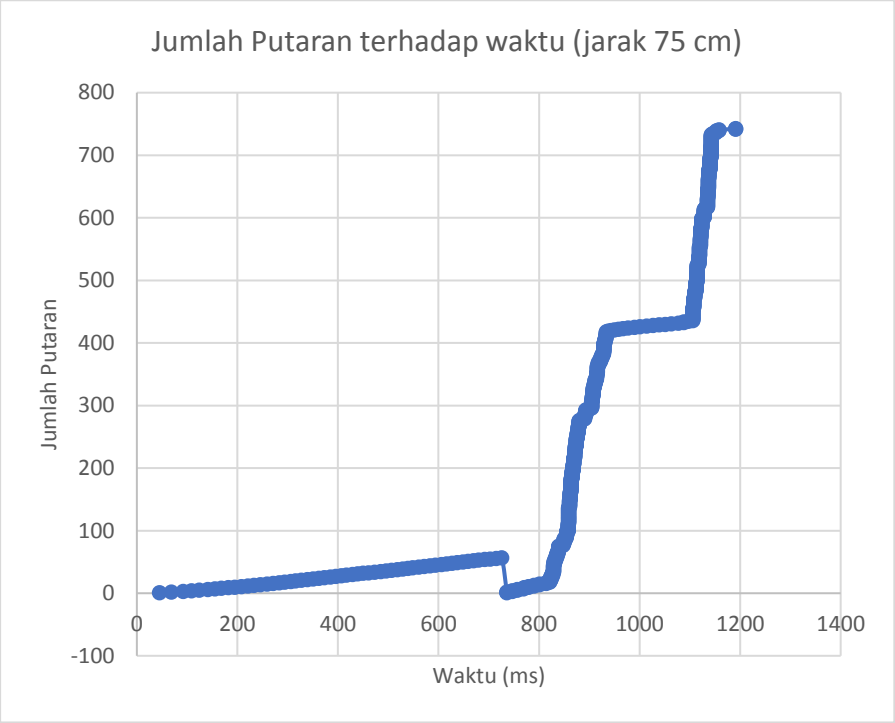
Dilakukan pengambilan data jumlah putaran *rotary encoder* terhadap waktu. Tujuannya adalah sebagai pembanding daripada hasil simulasi dan hasil eksperimental, melihat kemiripan *trendline* pada daerah *transient.* Grafik dihasilkan dengan melakukan *plotting* data keluaran dari putaran *rotary encoder*, dan untuk waktu diambil dari *timer millis internal* mikrokontroler Arduino Mega2560.

Gambar 4.9 mennujukkan respon kecepatan (dalam nilai jumlah putaran *rotary encoder*) pada percobaan $y_d = 50\ cm$. Hasil ini menunjukkan bahwa *rotary encoder* memerlukan waktu 413 ms untuk menghasilkan nilai kecepatan konstan (yaitu 448 putaran).



Gambar 4.9 Grafik jumlah putaran terhadap waktu (jarak 50 cm)

Gambar 4.10 mennujukkan respon kecepatan (dalam nilai jumlah putaran *rotary encoder*) pada percobaan $y_d = 75\ cm$. Hasil ini menunjukkan bahwa *rotary encoder* memerlukan waktu 1191 ms untuk menghasilkan nilai kecepatan konstan (yaitu 743 putaran).

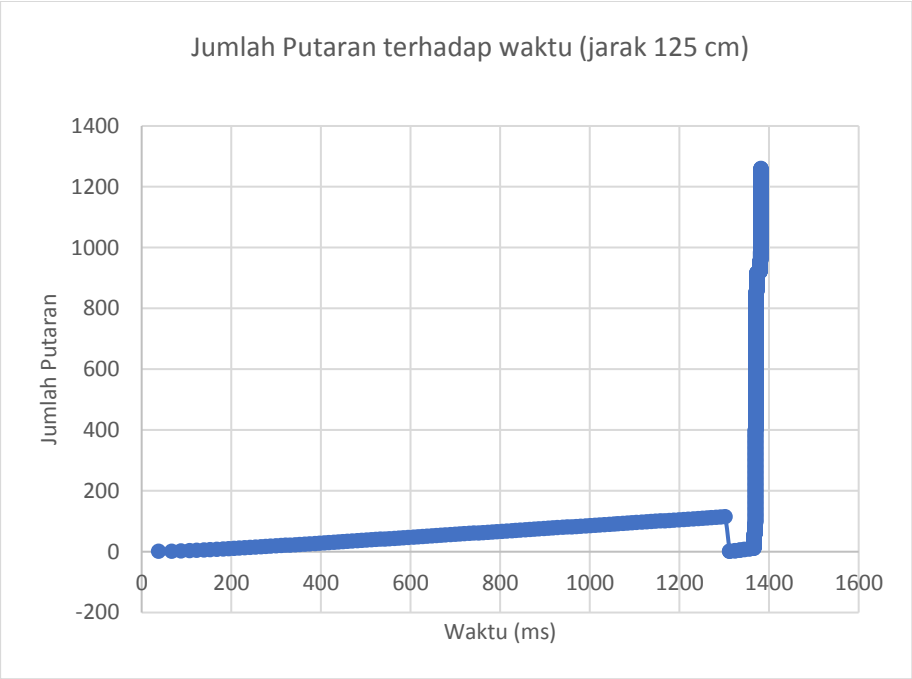


Gambar 4.10 Grafik jumlah putaran terhadap waktu (jarak 75 cm)

Gambar 4.11 mennujukkan respon kecepatan (dalam nilai jumlah putaran *rotary encoder*) pada percobaan $y_d = 100\ cm$. Hasil ini menunjukkan bahwa *rotary encoder* memerlukan waktu 743 ms untuk menghasilkan nilai kecepatan konstan (yaitu 916 putaran).

Gambar 4.11 Grafik jumlah putaran terhadap waktu (jarak 100 cm)

Gambar 4.12 mennujukkan respon kecepatan (dalam nilai jumlah putaran *rotary encoder*) pada percobaan $y_d = 125\,cm$. Hasil ini menunjukkan bahwa *rotary encoder* memerlukan waktu 1382 ms untuk menghasilkan nilai kecepatan konstan (yaitu 1260 putaran).



Gambar 4.12 Grafik jumlah putaran terhadap waktu (jarak 125 cm)

Gambar 4.9 hingga Gambar 4.11 yaitu percobaan pada jarak 50 cm, 75 cm, 100 cm memiliki kecendrungan *trendline* yang menyerupai dengan respon hasil simulasi. Walaupun pada percobaan jarak 75 cm, grafiknya terjadi fluktuasi pada kisaran waktu 600-800 ms.

Gambar 4.12 yaitu percobaan pada jarak 125 cm, terdapat anomali pada waktu yang ditampilkan. Hal ini dikarenakan kemampuan mikrokontroler Arduino khususnya dalam kecepatan *clock* 16 MHz yang tidak sebanding dengan kecepatan *rotary encoder* ketika berputar dengan nilainya besar juga waktu yang cukup lama, sehingga terjadi seperti *delay*. Penyebab lain adalah akibat Arduino Mega2560 memiliki *memory* yang kecil, yaitu SRAM (*Static Random Access Memory*) berukuran 4 KB ("Arduino Mega 2560 Board: Specifications, and Pin Configuration," 2019), ketika proses *data logging* respon yang diberikan Arduino cukup lama, waktu sebenarnya berjalan namun pada prosesnya *read and write* data tidak terjadi secara normal.

Data pada percobaan pada jarak 150 cm hingga 200 cm memiliki anomali yang sama, sehingga grafiknya tidak ditampilkan. Percobaan *data logging* di atas 100 cm dapat disimpulkan Arduino Mega2560 tidak mampu untuk merekam datanya.

(Halaman ini sengaja dikosongkan)

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil analisis data yang telah dilakukan, didapatkan beberapa kesimpulan sebagai berikut :

a. Telah berhasil dibuat sistem *motion control* PID prototipe *unmanned autonomous* dengan lintasan lurus secara simulasi dengan menggunakan nilai-nilai parameter Kp=0,853253789; Ki=0,256576567; Kd=0,923253789, dengan *rise time* pada simulasi 7,46 detik; *settling time* 23,09 detik; eror lintasan 0,28%.
b. Telah berhasil rancang bangun sistem *motion control* PID prototipe *unmanned autonomous forklift* dengan lintasan lurus yang mana jaraknya bervariasi dari 50 cm hingga 200 cm dengan interval 25 cm, menggunakan nilai-nilai parameter Kp=0,97; Ki=0,3; Kd=1, dengan rata-rata *error* mutlak 1,84 cm (3,67%).

5.2 Saran

Penambahan sensor-sensor untuk meningkatkan lokalisasi dan pemetaan jalur robot sangat diperlukan. Tidak akan cukup apabila hanya mengandalakan satu buah sensor. Pada penelitian selanjutnya kontrol terhadap kemudi (*steering*) harus disertai dengan sensor dan aktuator yang memadai.

(Halaman ini sengaja dikosongkan)

DAFTAR PUSTAKA

Abdellatif, M., Shoei, M., Talaat, O., Gabalah, M., M. Elbably, M., & Saleh, S. (2018). Design of an Autonomous Forklift Using Kinect. *MATEC Web of Conferences*, *153*(04005).

Amazon.com. (2018). RC Forklift Toys Huina 1577 2.4 GHz 8-ch. Diambil 29 Januari 2019, dari https://www.amazon.com/Game-Forklift-Rotation-Demonstration-Children/dp/B07B8N7XVZ

Arduino - ArduinoMega2560. (2019). Diambil 25 Desember 2019, dari https://www.arduino.cc/en/Guide/ArduinoMega2560

Arduino DC Motor Control Tutorial - L298N | PWM | H-Bridge - HowToMechatronics. (2019). Diambil 26 Desember 2019, dari https://howtomechatronics.com/tutorials/arduino/arduino-dc-motor-control-tutorial-l298n-pwm-h-bridge/

Arduino Mega 2560 Board: Specifications, and Pin Configuration. (2019). Diambil 25 Desember 2019, dari https://www.elprocus.com/arduino-mega-2560-board/

Basics of Rotary Encoders: Overview and New Technologies | Machine Design. (2019). Diambil 25 Desember 2019, dari https://www.machinedesign.com/automation-iiot/sensors/article/21831757/basics-of-rotary-encoders-overview-and-new-technologies

Chopra, S. 1960-. (2013). *Supply chain management : strategy, planning, and operation*. Pearson.

Essen, H. Van, & Nijmeijer, H. (2001). Nonlinear model predictive control of constrained mobile robot. *Proceeding Europe Control Conference*, 1157–1162.

Frazelle, E. (2002). *World-class warehousing and material handling*. McGraw-Hill.

Frekuensi, Duty Cycle, PWM. (2019). Diambil 26 Desember 2019, dari https://sunupradana.info/pe/2019/05/11/frekuensi-duty-cycle-pwm/

H-Bridges – the Basics | Modular Circuits. (2019). Diambil 26

Desember 2019, dari https://www.modularcircuits.com/blog/articles/h-bridge-secrets/h-bridges-the-basics/

How Does a PID Controller Work? - Structure & Tuning Methods. (2019). Diambil 25 Desember 2019, dari https://www.elprocus.com/the-working-of-a-pid-controller/

In-Depth: Interface L298N DC Motor Driver Module with Arduino. (2019). Diambil 26 Desember 2019, dari https://lastminuteengineers.com/l298n-dc-stepper-driver-arduino-tutorial/

Jefferies, R. (2011). *Forklift operator training*. Roger Jefferies.

Koubaa, A., Bennaceur, H., Chaari, I., Trigui, S., Ammar, A., Sriti, M. F., … Javed, Y. (2018). Introduction to mobile robot path planning. In *Studies in Computational Intelligence* (Vol. 772, hal. 3–12). Springer Verlag. https://doi.org/10.1007/978-3-319-77042-0_1

Lee, H.-H. (2014). Modeling and trajectory control of a forklift-like wheeled robot. *ASME International Mechanical Engineering Congress and Exposition, Proceedings (IMECE), 4A*, 1–8. https://doi.org/10.1115/IMECE2014-37081

Lencrow Fokrlifts. (2018). Different Types of Forklift and Their Common Use. Diambil 5 November 2018, dari https://www.lencrowforklifts.com.au/news/different-types-of-forklift-and-their-common-use/

Motor DC, Pengertian, Karakteristik, Bagian & Jenis Motor DC. (2019). Diambil 25 Desember 2019, dari http://zonaelektro.net/motor-dc/

Motor Encoder RPM Speed Counter Interrupter Sensor Module FC-03 | QQ Online Trading. (2019). Diambil 25 Desember 2019, dari http://qqtrading.com.my/motor-rpm-speed-counter-sensor-module-fc-03

Navy Construction Force. (2008). Chapter 12 Forklift - Navy Construction Force Manual. In *Navy Construction Force Manual* (hal. 12–14). Nevada: Navy Construction Force.

Nehmzow, U. (2003). *Mobile Robotics: A Practical Introduction*.

Mobile Robotics: A Practical Introduction. Springer London. https://doi.org/10.1007/978-1-4471-0025-6

Ogata, K. (1997). *Modern Control Engineering*. Englewood Cliffs: Prentice Hall.

Petersen, I. R., Ugrinovskii, V. A., Savkin, A. V., Chandrasekharan, P. C., Tamba, T. A., Hong, B., … Ogata, K. (2000). *Mobile Robot Kinematics*. *Autonomous Mobile Robots* (Vol. 2). Cambridge: Navy Construction Force. https://doi.org/10.1007/978-1-4471-0447-6

Position Sensors | Capacitive Inductive LVDT Rotary Encoder Working. (2019). Diambil 25 Desember 2019, dari https://www.electronicshub.org/position-sensors/

Prinsip Kerja Motor DC. (2019). Diambil 25 Desember 2019, dari http://elektronika-dasar.web.id/prinsip-kerja-motor-dc/

Pulse Width Modulation - learn.sparkfun.com. (2019). Diambil 26 Desember 2019, dari https://learn.sparkfun.com/tutorials/pulse-width-modulation/all

Ray, S. (2008). *Introduction to materials handling*. New Age International (P) Ltd., Publishers.

Reality Bytes. (2018). Graph-Based Path Planning: Dijkstra's Algorithm. Diambil 29 Januari 2019, dari https://realitybytes.blog/2017/07/11/graph-based-path-planning-dijkstras-algorithm/

Rotary Encoder - Northwestern Mechatronics Wiki. (2019). Diambil 25 Desember 2019, dari http://hades.mech.northwestern.edu/index.php/Rotary_Encoder

Rotary encoder system. | Download Scientific Diagram. (2019). Diambil 25 Desember 2019, dari https://www.researchgate.net/figure/Rotary-encoder-system_fig5_3432265

S., M. B., F., A., & Ara, B. (2013). Path Tracking Pada Mobile Robot Dengan Umpan Balik Odometry.

Siegwart, R., & Nourbakhsh, I. (2004). Mobile Robot Kinematics. In *Autonomous Mobile Robots* (hal. 39–41). Cambridge: MIT

Press.

Site Automation: Automated/Robotic On-Site Factories (Cambridge Handbooks on Construction Robotics), Thomas Bock, Thomas Linner, eBook - Amazon.com. (2015). Diambil 25 Desember 2019, dari https://www.amazon.com/Site-Automation-Automated-Site-Construction-ebook/dp/B01DPNK0EY/

Smart Automation to Smart Manufacturing: Industrial Internet of Things, Uthayan Elangovan, eBook - Amazon.com. (2016). Diambil 25 Desember 2019, dari https://www.amazon.com/Smart-Automation-Manufacturing-Industrial-Internet-ebook/dp/B07RLWH6C3

Supriyanto, R., Hustinawati, K., A. B., N., R. W., Permadi, Y., & Sa'ad, A. (2010). *Buku Ajar Robotika*.

Susanto, V. E. P. (2018). *Kendaraan Otonom Menggunakan Kendali Berbasis Rute Dengan Metode Odometry*. Politeknik Negeri Bandung. Diambil dari http://digilib.polban.ac.id/gdl.php?mod=browse&op=read&id=jbptppolban-gdl-vitoekapra-8026

Swartz, G. (1999). *Forklift safety : a practical guide to preventing powered industrial truck incidents and injuries*. Government Institutes.

Tamba, T. A., Hong, B., & Hong, K.-S. (2009). A Path Following Control of an Unmanned Autonomous Forklift. *International Journal of Control, Automation, and Systems Vol. 7 (1)*, 113–122.

Teori Motor DC Dan Jenis-Jenis Motor DC. (2019). Diambil 25 Desember 2019, dari http://elektronika-dasar.web.id/teori-motor-dc-dan-jenis-jenis-motor-dc/

Toyota Industrial Equipment. (2018). Toyota Forklift. Diambil 29 Januari 2019, dari https://www.toyotaforklift.com/

Washington State Departement of LAbor&Industries. (2015). Forklift Safety Guide. *Director*. Diambil dari www.Lni.wa.gov/Safety

What is a Microcontroller? - Definition from Techopedia. (2019). Diambil 25 Desember 2019, dari

https://www.techopedia.com/definition/3641/microcontroller

What Is a Microcontroller? The Defining Characteristics and Architecture of a Common Component - Technical Articles. (2019). Diambil 25 Desember 2019, dari https://www.allaboutcircuits.com/technical-articles/what-is-a-microcontroller-introduction-component-characteristics-component/

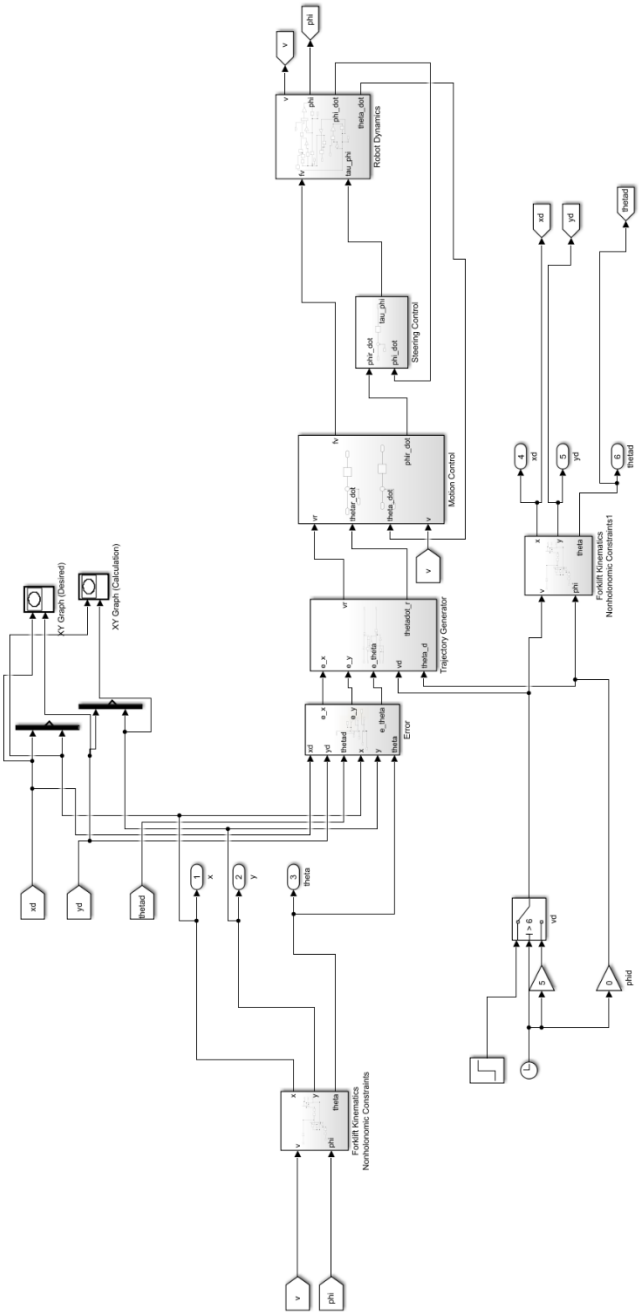
What is a Pulse Width Modulation (PWM) Signal and What is it Used For? - National Instruments. (2019). Diambil 26 Desember 2019, dari https://knowledge.ni.com/KnowledgeArticleDetails?id=kA00Z0000019OkFSAU&l=en-ID

(Halaman ini sengaja dikosongkan)

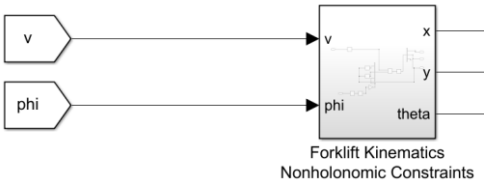
LAMPIRAN

(Halaman ini sengaja dikosongkan)

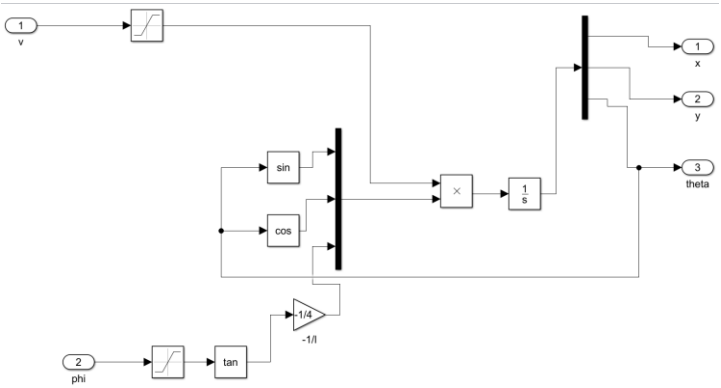
SIMULINK FORKLIFT ALTRAMAN



1. Kinematika forklift (*non-holonomic constraints*)

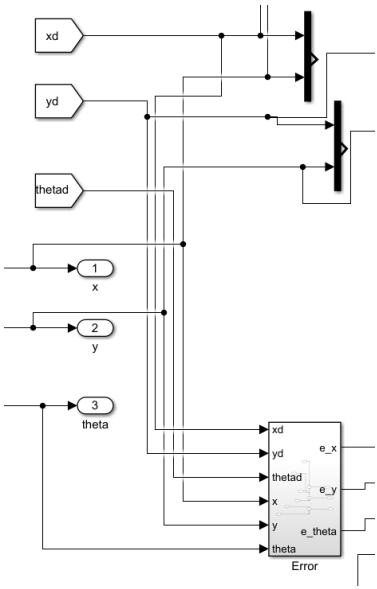


Gambar A.1 Subsistem kinematika *forklift*

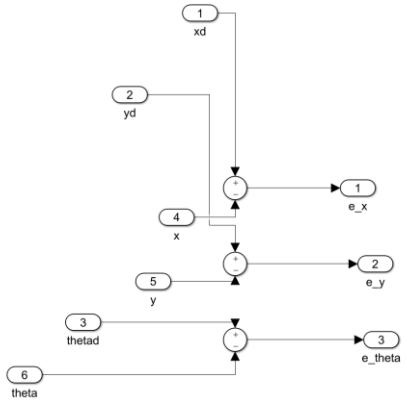


Gambar A.2 Pemodelan kinematika *forklift*

2. Error

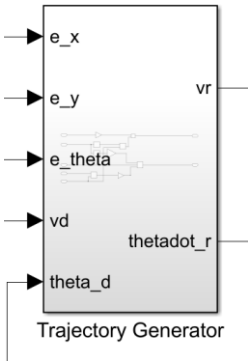


Gambar A.3 Subsistem error

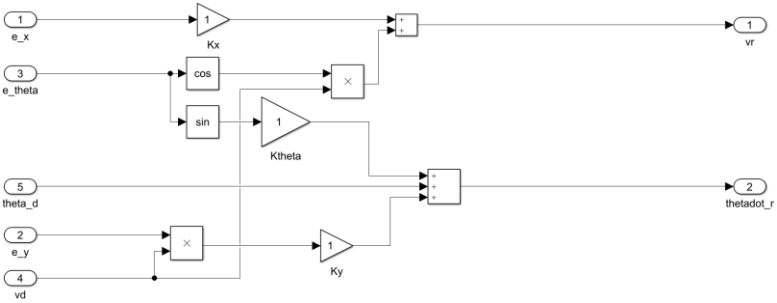


Gambar A.4 Pemodelan error untuk *trajectory generator*

3. *Trajectory generator*

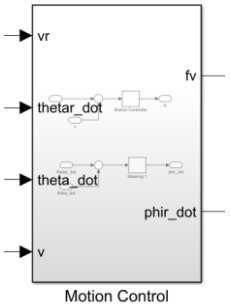


Gambar A.5 Subsistem *trajectory generator*

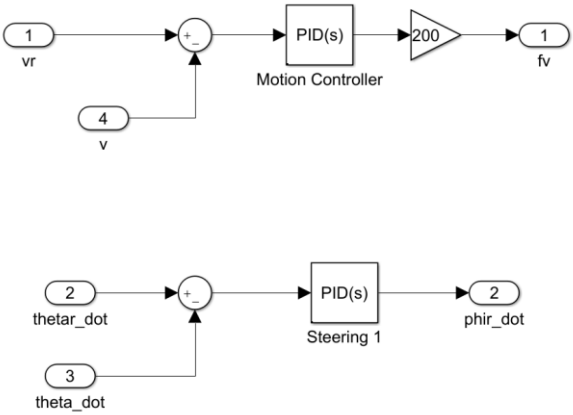


Gambar A.6 Pemodelan *trajectory generator*

4. *Motion control*

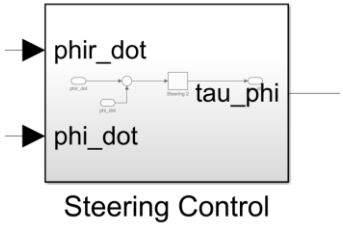


Gambar A.7 Subsistem *motion control*

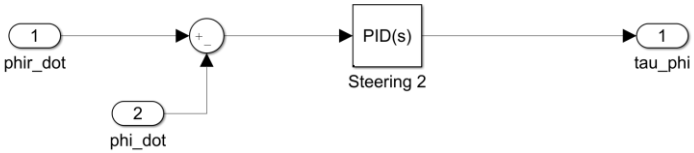


Gambar A.8 Pemodelan *motion control*

5. *Steering control*

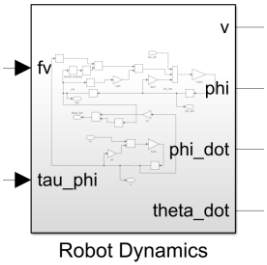


Gambar A.9 Subsistem *steering control*

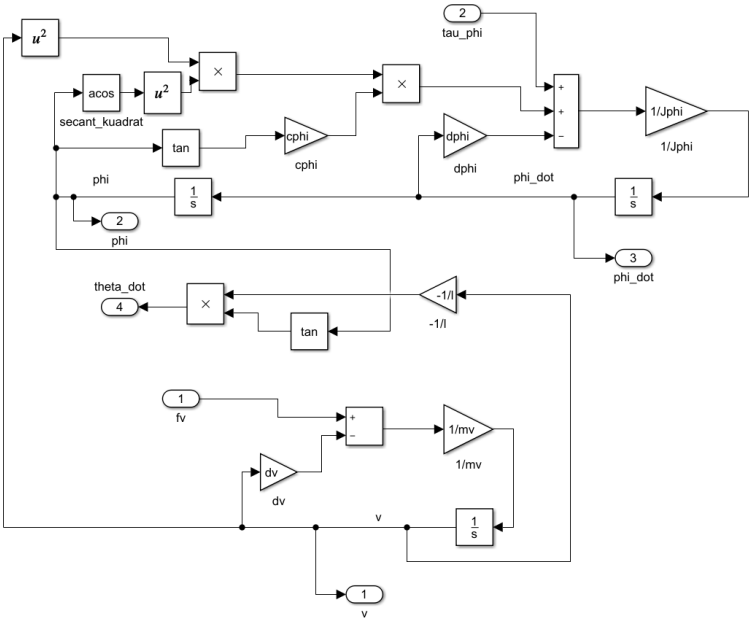


Gambar A.10 Pemodelan *steering control*

6. Dinamika robot

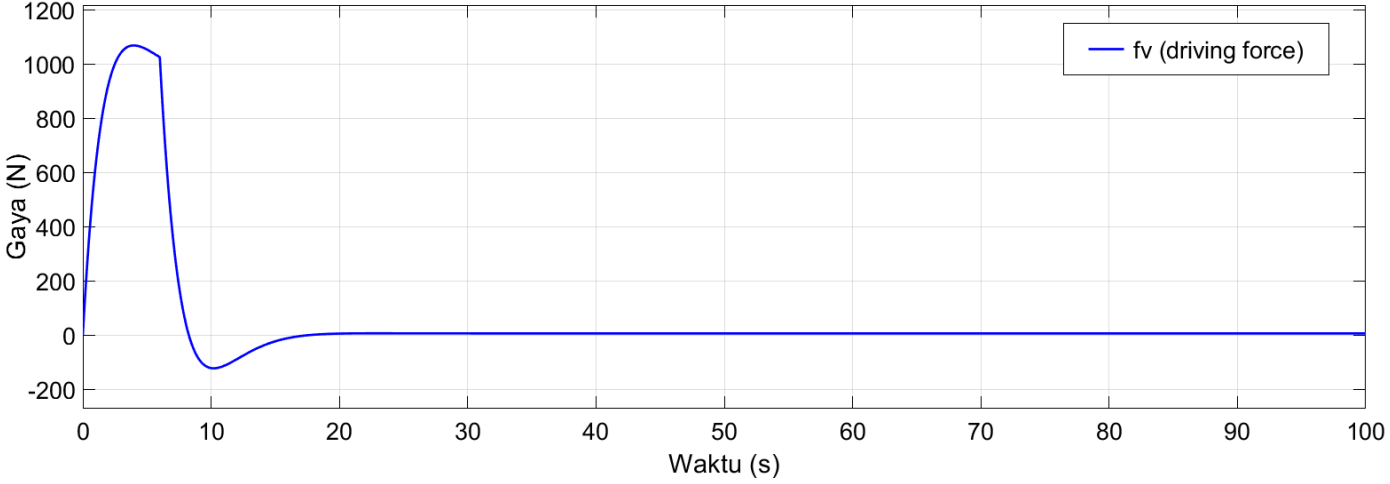


Gambar A.11 Subsistem dinamika robot



GambarA.12 Pemodelan dinamika robot

(Halaman ini sengaja dikosongkan)



Scope fv_motion (*Driving Force*)

(Halaman ini sengaja dikosongkan)

param_kindyn.m

```
%kinematics and dynamics gain
%--------------------------------------
r=0.661; %jari-jari dari roda dalam meter
l=0.12; %jarak dari titik poros (axle) depan ke
belakang RC forklift dalam meter
lc=l/2; %jarak dari titik poros depan ke titik
tengah (mass center) dalam meter
%mv=1000; %massa RC forklift dalam kg
m=0.67; %massa RC forklift dalam kg
Jb=0.3; %momen inersia titik tengah massa kg m2
Jh=0.2; %momen inersia horizontal kg m2
Jv=0.1; %momen inersia vertikal kg m2
dv=0.2; %visocus damping coeff. kg/s
dphi=0.1; %visocus damping coeff. kg m2/s
cphi = 1/l^2*(lc^2*m+Jb+4*Jv)+2/r^2*Jh;
Jphi = 2* Jv;
mv = m+4/r^2*Jh;

%fv: gaya kemudi
%tau_phi : torsi steering

% control gain
%--------------------------------------
% x,y,z, theta merupakan parameter-parameter
gain kontrol
% untuk trajectory generation
Kx =0.5;
Ky=0.5;
Kz=1;
Ktheta=1;
```

(Halaman ini sengaja dikosongkan)

Odometry_ForkliftAltraman.ino

```
/*

  Unmanned Autonomous Forklift Program

  Supported by PT. Altraman

  Odometry


  “Around here we don't look backwards for very
long.

  We keep moving forward, opening up new doors

  and doing new things,

  because we're curious...

  and curiosity keeps leading us down new
paths.”

  -Walt Disney


  @maliki_borneo

  Manufacturing Automation & Mechatronics
Engineering Polman Bandung

  Engineering Physics (Lintas Jalur) ITS
Surabaya

*/

#include <PID_v1.h>

double Kp=0.97, Ki=0.3, Kd=1;

double Setpoint, Input, Output;

PID myPID(&Input, &Output, &Setpoint, Kp, Ki,
Kd, DIRECT);
```

```
// Constants for Interrupt Pins

// Change values if not using Arduino Uno


#define pi 3.14159265358979323846264338327795

const byte MOTOR_A = 3;  // Motor 2 Interrupt
Pin - INT 1 - Right Motor

const byte MOTOR_B = 2;  // Motor 1 Interrupt
Pin - INT 0 - Left Motor


// Constant for steps in disk

const float stepcount = 20.00;  // 20 Slots in
disk, change if different


// Constant for wheel diameter

const float wheeldiameter = 66.10; //67; //
Wheel diameter in millimeters, change if
different


// Integers for pulse counters

volatile int counter_A = 0;

volatile int counter_B = 0;



// Motor A right


int enA = 10; //pwm
```

```
int in1 = 9;
int in2 = 8;

// Motor B left

int enB = 5; //pwm
int in3 = 7;
int in4 = 6;

// Interrupt Service Routines

// Motor A pulse count ISR
void ISR_countA()
{
  counter_A++;  // increment Motor A counter
value
}

// Motor B pulse count ISR
void ISR_countB()
{
  counter_B++;  // increment Motor B counter
value
}
```

```
// Function to convert from centimeters to steps
int CMtoSteps(float cm) {


  int result;  // Final calculation result
  float circumference = (wheeldiameter * pi) /
10; // Calculate wheel circumference in cm
  float cm_step = circumference / stepcount;  //
CM per Step


  float f_result = cm / cm_step;  // Calculate
result as a float
  result = (int) f_result; // Convert to an
integer (note this is NOT rounded)
  Input = f_result;
Serial.print("cm : "); Serial.println(cm);
  Serial.print("Keliling Roda : ");
Serial.println(circumference);
  Serial.print("Langkah per cm : ");
Serial.println(cm_step);
  Serial.print("Hasil Float : ");
Serial.println(f_result);
  Serial.print("Hasil Integer : ");
Serial.println(result);
  return result;  // End and return result


}
```

```
// Function to Move Forward
void MoveForward(int steps, int mspeed)
{
   counter_A = 0;  //  reset counter A to zero
   counter_B = 0;  //  reset counter B to zero

   // Set Motor A forward
   digitalWrite(in1, HIGH);
   digitalWrite(in2, LOW);

   // Set Motor B forward
   digitalWrite(in3, HIGH);
   digitalWrite(in4, LOW);

   // Go forward until step value is reached
   while (steps > counter_A && steps >
counter_B) {

    if (steps > counter_A) {
    analogWrite(enA, mspeed);
    } else {
    analogWrite(enA, 0);
    }
    if (steps > counter_B) {
```

```
    analogWrite(enB, mspeed);
    } else {
    analogWrite(enB, 0);
    }
   }

  // Stop when done
  analogWrite(enA, 0);
  analogWrite(enB, 0);
  counter_A = 0;  //  reset counter A to zero
  counter_B = 0;  //  reset counter B to zero

}

// Function to Move in Reverse
void MoveReverse(int steps, int mspeed)
{
   counter_A = 0;  //  reset counter A to zero
   counter_B = 0;  //  reset counter B to zero

   // Set Motor A reverse
  digitalWrite(in1, LOW);
  digitalWrite(in2, HIGH);
```

```
  // Set Motor B reverse
  digitalWrite(in3, LOW);
  digitalWrite(in4, HIGH);

   // Go in reverse until step value is reached
   while (steps > counter_A && steps >
counter_B) {

    if (steps > counter_A) {
    analogWrite(enA, mspeed);
    } else {
    analogWrite(enA, 0);
    }
    if (steps > counter_B) {
    analogWrite(enB, mspeed);
    } else {
    analogWrite(enB, 0);
    }
    }

  // Stop when done
  analogWrite(enA, 0);
  analogWrite(enB, 0);
  counter_A = 0;  //  reset counter A to zero
```

```
  counter_B = 0;  //  reset counter B to zero

}

// Function to Spin Right
void SpinRight(int steps, int mspeed)
{
   counter_A = 0;  //  reset counter A to zero
   counter_B = 0;  //  reset counter B to zero

   // Set Motor A reverse
  digitalWrite(in1, LOW);
  digitalWrite(in2, HIGH);

  // Set Motor B forward
  digitalWrite(in3, HIGH);
  digitalWrite(in4, LOW);

   // Go until step value is reached
   while (steps > counter_A && steps >
counter_B) {

    if (steps > counter_A) {
    analogWrite(enA, mspeed);
```

```
    } else {
    analogWrite(enA, 0);
    }
    if (steps > counter_B) {
    analogWrite(enB, mspeed);
    } else {
    analogWrite(enB, 0);
    }
   }

  // Stop when done
  analogWrite(enA, 0);
  analogWrite(enB, 0);
  counter_A = 0;  //  reset counter A to zero
  counter_B = 0;  //  reset counter B to zero

}

// Function to Spin Left
void SpinLeft(int steps, int mspeed)
{
   counter_A = 0;  //  reset counter A to zero
   counter_B = 0;  //  reset counter B to zero
```

```
   // Set Motor A forward
  digitalWrite(in1, HIGH);
  digitalWrite(in2, LOW);

  // Set Motor B reverse
  digitalWrite(in3, LOW);
  digitalWrite(in4, HIGH);

   // Go until step value is reached
   while (steps > counter_A && steps >
counter_B) {

    if (steps > counter_A) {
    analogWrite(enA, mspeed);
    } else {
    analogWrite(enA, 0);
    }
    if (steps > counter_B) {
    analogWrite(enB, mspeed);
    } else {
    analogWrite(enB, 0);
    }
  }
```

```
  // Stop when done
  analogWrite(enA, 0);
  analogWrite(enB, 0);
  counter_A = 0;  //  reset counter A to zero
  counter_B = 0;  //  reset counter B to zero

}

void StopBrake()
{

  digitalWrite(in1, LOW);
  digitalWrite(in2, LOW);

  digitalWrite(in3, LOW);
  digitalWrite(in4, LOW);
}

void setup()
{
  Serial.begin(9600);
  // Attach the Interrupts to their ISR's
  attachInterrupt(digitalPinToInterrupt
(MOTOR_A), ISR_countA, RISING);  // Increase
counter A when speed sensor pin goes High
```

```
  attachInterrupt(digitalPinToInterrupt
(MOTOR_B), ISR_countB, RISING);  // Increase
counter B when speed sensor pin goes High


  // Test Motor Movement  - Experiment with your
own sequences here
  Setpoint = 5;
  myPID.SetMode(AUTOMATIC);
  myPID.Compute();
  //MoveForward(Setpoint, 0);
  MoveForward(CMtoSteps(Setpoint), 255);  //
Forward half a metre at 255 speed
  delay(1000);  // Wait one second
  StopBrake();
 Serial.print("Count A : ");
  Serial.println(counter_A);
   Serial.print("Count B : ");
  Serial.println(counter_B);


  Serial.print("Setpoint : ");
  Serial.println(Setpoint);
  Serial.print("Input : ");
  Serial.println(Input);
  Serial.print("Output : ");
  Serial.println(Output);
//  SpinRight(3,100);
```

```
//  delay(1000);  // Wait one second
//  StopBrake();
//  MoveForward(CMtoSteps(3), 150);  // Forward
half a metre at 255 speed
//  delay(1000);  // Wait one second
//  StopBrake();




}
//10 steps itu 1 m
// 5 cm 100cm
void loop()
{



}
//Time taken = current time – previous time
//timetaken = millis()-pevtime; //timetaken in
millisec
//rpm=(1000/timetaken)*60;
//Velocity = 2π × RPS × radius of wheel.
//v = radius_of_wheel * rpm * 0.104
```

(Halaman ini sengaja dikosongkan)

BIODATA PENULIS

Reza Maliki Akbar, A.Md; dilahirkan di Nanga Pinoh, Kalimantan Barat, tanggal 7 Mei 1995. Penulis telah menyelesaikan pendidikan di TK Aisiyah Bustanul Athfal 10 Bandung, SDN Dr. Cipto Bandung, SMPN 1 Bandung, SMKN 1 Cimahi (STMN Pembangunan Bandung) kompentensi keahlian Rekayasa Perangkat Lunak, D3 di Politeknik Manufaktur Negeri Bandung (Politeknik Mekanik Swiss-ITB) jurusan Teknik Otomasi Manufaktur dan Mekatronika (masuk tahun 2013) dan sedang menempuh pendidikan S1 Lintas Jalur Teknik Fisika di ITS (masuk tahun 2017) hingga sekarang. Penulis pernah bekerja di perusahaan kontraktor otomasi dan elektrikal. Banyak proyek yang telah dibuat penulis, yaitu proyek yang beruhubungan dengan pemrograman, elektronika, robotika, sistem otomasi dan kontrol, serta *Internet of Things* (IoT). Selama di ITS, penulis aktif di tim robotika bawah air dan amfibi, sebagai *founder* dan *general manager*, Banyubramanta ITS serta aktif juga di UKM Robotika, UKM *Maritime Challenge*, dan bergabung di IEEE ITS *Student Branch*. Bagi pembaca yang memiliki kritik, saran atau ingin berdiskusi lebih lanjut mengenai tugas akhir ini maka dapat menghubungi penulis melalui email : maliki.17023@mhs.its.ac.id atau rezamaliki.akbar@gmail.com.