ITS
Institut
Teknologi
Sepuluh Nopember

PROYEK AKHIR

PROTOTIPE ROBOT PENGIKUT PADA IMPLEMENTASI ROBOT SWARM UNTUK MEMBENTUK FORMASI MENGIKUTI PEMIMPIN

FOLLOWER ROBOT PROTOTYPE ON IMPLEMENTATION SWARM ROBOTS WITH THE FOLLOW THE LEADER FORMATION

Oleh:

Hakim Sa'adi
NRP. 7104 040 050

Dosen Pembimbing :

Endah Suryawati Ningrum, S.T., M.T.
NIP. 19750112.200012.2.001

Dr, Ir. Endra Pitowarno, M.Eng.
NIP. 19620630.198701.1.001

Ali Husein Alasiry, S.T., M.Eng.
NIP. 19731027.200003.1.001

JURUSAN TEKNIK ELEKTRONIKA
POLITEKNIK ELEKTRONIKA NEGERI SURABAYA
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA

2010

PROYEK AKHIR

PROTOTIPE ROBOT PENGIKUT PADA IMPLEMENTASI ROBOT SWARM UNTUK MEMBENTUK FORMASI MENGIKUTI PEMIMPIN

FOLLOWER ROBOT PROTOTYPE ON IMPLEMENTATION SWARM ROBOTS FOR FOLLOW THE LEADER FORMATION

Oleh:

Hakim Sa'adi
NRP. 7104.040.050

Dosen Pembimbing :

Endah Suryawati Ningrum, S.T, M.T.
NIP. 19750112.200012.2.001

Dr. Ir. Endra Pitowarno, M.Eng
NIP. 19620630.198701.1.001

Ali Husein Alasiry, S.T, M.Eng.
NIP. 19731027.200003.1.001

HALAMAN JUDUL

JURUSAN TEKNIK ELEKTRONIKA
POLITEKNIK ELEKTRONIKA NEGERI SURABAYA
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA

2010

PROTOTIPE ROBOT PENGIKUT PADA IMPLEMENTASI ROBOT SWARM UNTUK MEMBENTUK FORMASI MENGIKUTI PEMIMPIN

Oleh :

HAKIM SA'ADI
NRP. 7104.040.050

Proyek Akhir ini Diajukan Sebagai Salah Satu Syarat Untuk Memperoleh Gelar Sarjana Sains Terapan (S.ST.)
Periode Wisuda Maret 2011
di
Politeknik Elektronika Negeri Surabaya
Institut Teknologi Sepuluh Nopember
Surabaya

Di setujui dan disahkan pada tanggal Februari 2011

Dosen Penguji Proyek Akhir	Dosen Pembimbing
1.	1.
Eru Puspita, S.T, M.Kom	Endah S. Ningrum, S.T, M.T
NIP. 19691231.199501.1.001	NIP. 19750112.200012.2.001
2.	2.
Legowo Sulistyo, S.ST, M.Eng	Dr. Ir. Endra Pitowarno, M.Eng
NIP. 19651122.199103.1.005	NIP. 19620630.198701.1.001
3.	3.
Reesa Akbar, S.T	Ali Husein Alasiry, S.T, M.Eng
NIP. 19750729.200112.1.001	NIP. 19731027.200003.1.001

Mengetahui :
Ketua Jurusan Teknik Elektronika

Ir. Rika Rokhana, M.T.
NIP. 19690905.199802.2.001

ABSTRAK

Robot *Swarm* adalah sekumpulan robot dengan struktur fisik relatif sederhana dan kesamaan perilaku yang mampu bekerja sama dari hasil interaksi antar robot dan interaksi antara robot dengan lingkungannya. Dari berbagai macam jenis robot *swarm*, formasi dengan Perilaku Mengikuti Pemimpin (*Follow The Leader*) menjadi tema yang sangat menarik karena sering kita jumpai dalam kehidupan sehari – hari. Kehandalan dari konsep *Follow The Leader* terletak pada kemampuan robot Pengikut (*Follower*) dalam mengikuti robot Pemimpin (*Leader*) dengan jarak yang relatif konstan.

Sesuai dengan tujuannya untuk *Follow The Leader* maka kemampuan pemetaan atau lokalisasi adalah syarat mutlak yang harus ada pada robot pengikut. Penentuan posisi relatif robot pemimpin dapat dilakukan dengan mengkombinasikan antara jarak antar robot dengan perubahan gerakan roda pada robot pemimpin. Karena biasanya roda robot sering mengalami slip yang mengakibatkan kesalahan perhitungan. Maka pengukuran jarak antar robot dilakukan dengan menggunakan sensor ultasonik yang dimodifikasi dengan menambahkan reflektor dengan tujuan efisien daya dan biaya.

Kata kunci : *Follow The Leader, Lokalisasi, Pemetaan, reflector, ultrasonik.*

ABSTRACT

The development of robotics technology today has entered the area Swarm robotics. It is the science of learning about the set of robots with relatively simple physical structure and the similarity of behavior that can work together from the results of interaction between robots and its environment. Of the various types of robot swarm, Behavior Following formation with leader (Follow The Leader) became very interesting issues because we often encounter in daily life. The Robustness of Follow The Leader Concepts is carry on the ability of the Follower robot to follow The Leader Robot with keeping the specified relative distance.

In accordance with the purpose to Follow The Leader then mapping or localization capability is a necessary condition that must exist on the follower robot. Determination of the relative position of the leader robot can be done by combining the distance between the robot with the wheels on the robot motion changes the leader robot. Because usually wheeled robots often slip resulting in calculation errors. So the inter-robot distance measurements performed using a modified ultasonic sensors by adding a reflector for the purpose of efficient and cheap power.

Keywords : *Follow The Leader, Localization,mapping, reflector, ultrasonic.*

KATA PENGANTAR

Segala puji dan syukur bagi Allah, Tuhan yang Maha Esa, Maha Agung, yang Maha Memiliki dan Mengetahui, Sumber ilmu dan Sumber segala kebenaran. Sholawat salam tercurah kepada baginda Nabi Muhammad SAW, juga kepada para sahabat, pengikut dan orang-orang yang berada dijalannya hingga akhir jaman.

Hanya karena petunjuk dan pertolongan Allah SWT serta iringan doa dari kedua orang tua dan saudara serta segala bantuan dari dosen pembimbing dan pengerahan segenap usaha, akhirnya kami dapat menyelesaikan proyek akhir yang berjudul :

"PROTOTIPE ROBOT PENGIKUT UNTUK IMPLEMENTASI ROBOT SWARM DENGAN FORMASI MENGIKUTI PEMIMPIN"

Seperti pepatah yang berbunyi *"tiada gading yang tak retak"*, maka proyek akhir ini jauh dari sempurna, masih terdapat banyak kekurangan. Oleh karena itu penulis mengharapkan masukan dan saran dari pembaca sekalian guna tercapainya hasil yang lebih baik, segala saran dan kritik dapat disampaikan di : *hqm_saad@yahoo.co.id*.

Semoga apa yang telah penulis tuangkan dalam proyek akhir ini dapat memberikan manfaat bagi rekan-rekan semua. Amin.

Surabaya, januari 2011

Penulis

UCAPAN TERIMA KASIH

Segala puji dan syukur bagi Allah, Tuhan yang Maha Esa, Maha Agung, yang Maha Memiliki dan Mengetahui, Sumber ilmu dan Sumber segala kebenaran. Sholawat salam tercurah kepada baginda Nabi Muhammad SAW, juga kepada para sahabat, pengikut dan orang-orang yang berada dijalannya hingga akhir jaman.

Dibalik terselesaikannya buku ini, ada orang-orang yang dengan sabar, bijak dan memiliki wawasan yang luas yang mendorong dan memberi arahan pada penulis untuk menyelesaikannya.

- Khusus kepada para pembimbing Proyek Akhir : Endah Suryawati Ningrum, M.T, Dr. Endra Pitowarno, dan Ali Husein Alasiry, M.Eng atas segala kesabaran dan kebijakannya selama ini. Beliau bertiga adalah guru penulis dalam beberapa tahun terakhir untuk bidang studi robotika. Dari merekalah wawasan penulis tentang dunia robotika terbuka. Dunia yang selama ini terasa jauh namun sebenarnya dekat.

- Khusus kepada sahabat, teman diskusi sekaligus rekan dalam mengerjakan Proyek Akhir ini, saudara Mifta Roni Prasetya yang dengan sabar dan istiqomah berusaha untuk mencapai hasil yang diharapkan dan ditargetkan.

- Khusus kepada kedua orang tua penulis. Dari beliau berdua penulis mengenal tentang arti dan tujuan hidup. Merekalah yang telah membesarkan serta mendidik penulis untuk mampu bersikap jujur, terbuka, toleran, kreatif, berani, sabar, tahan banting dan bijaksana. Mereka memiliki peran sangat penting dan tak terhingga, hingga ucapan terimakasih saja tidak akan pernah cukup untuk menggambarkan wujud penghargaan penulis.

- Khusus Kepada para penguji seminar Proyek Akhir : Eru Puspita, M.Kom, Reesa Akbar, S.T, dan Legowo Sulistyo, M.Eng, Atas segala saran, kritik dan revisi dan kesempatan yang diberikan.

- Kepada Kajur dan Sekjur jurusan Teknik Elektronika : Rika Rohana, M.T dan Bambang Sumantri, M.Sc, atas kesempatan, bantuan dan toleransinya untuk menyelesaikan Proyek Akhir ini.
- Kepada Hari Oktavianto, M.Sc selaku panitia Proyek Akhir.
- Kepada Agus Indra Gunawan, M.Sc atas kesempatan yang diamanahkan kepada penulis untuk belajar di lab automation TC102 dan untuk segala saran yang sangat berharga selama ini.
- Kepada Wiwit Priantono, teknisi Lab Embedded untuk pinjaman alat, perangkat dan tempat, dan sebagai teman begadang.
- Kepada Paulus susetyo, S.T untuk sarannya.
- Kapada Prof. William Spears dari universitas wyoming, atas segala sarannya walaupun hanya lewat dunia maya.
- Kepada Dr. Dadet Pramadihanto selaku Direktur Politeknik Elektronika Negeri Surabaya. Atas kesempatan yang diberikan penulis untuk menimba bahkan kalau mampu menguras semua ilmu yang diajarkan di PENS-ITS.
- Kepada Aris pratiarso, M.T selaku Asisten Direktur II untuk keringanan biaya kuliahnya.
- Kepada Pakde Jamil, Aqil Azizi, Ulfa, Lutfi Khusniati, Taufik Al-Asyari, Khasin Anwari atas segala saran, kritik, canda dan dorongannya selama ini.
- Kepada Alumni Elka ’08 khususnya Kelas EBD4 : Mohammad Khozain, Muamar Qhadafi, Tommi Adi, Osa Utomo, Tutut Kurniadi, Putut Hadi, Hanny Isnar, Danan Januar, Efendi Rona, Yudha Sangputro, Husnul Fuad, Hendra Susanto, Hariski Priyo, Ahmad Fauzi, Nurdin Ardiansyah, Dimas Sultan, Andrian Herman, Rio Sektiaji, Syukron Ali, Taqwan Nurcholis, Anis Fahruddin, Rum Susetyo, Yunta Dwi, Novita Astin, Siwi Dian, Rizki Dian, Noviani Rahmawati, Eva Kusuma. Atas segala epik pertemanan, diskusi dan

bantuan selama ini baik secara moral, spirit maupun finansial. Semoga tali silaturrohim yang selama ini terjalin tetap terikat kuat.

- Kepada Heres, alim Prasojo, Shoim, dan Ainul atas bantuannnya yang bersifat produktif dan bukannya konsumtif.
- Kepada Agus suhariyanto, Septian, dan Umar, eko, putut krian, atas segala bantuannya baik itu pinjaman motor, komponen bahkan kadang makan.
- Kepada alumni dan anggota lab Automation : Rosyid, Arif pakdeno, Otos, Ainul rokhim, Aqib maymoon, Agus setyabudi, Yusuf alfian, Herman gusnadi, Amat roni, Harun arrosyid, Hendi hermawan, andri.
- Kepada teman asrama : Pak Juwari, Pak ji, Pak Naryo, Yanto, Edi, Karso, Toyo, Warobai, Sobir, Rudi, Iwan Setiawan, Roman, Sumari, Sumanto, Endro Subko, Haris, Yudha, Rohmat, Aditya, Musbikhin, Hanif, Susilo, Zainul, Bowo, Iwan, Ipin, Mualim, Aang, Zainal Asikin, Zainal, Darmaji, Riski, Asrul, Basuki, Wahyu, Ali, Joko.
- Untuk ratusan orang lainnya yang telah banyak memberikan inspirasi dan bantuan baik secara langsung maupun tak langsung namun belum penulis cantumkan namanya. Penulis mengucapkan terimakasih dan penghargaan yang tiada terhingga.

Semoga Allah mencatat segala bantuan mereka sebagai ibadah dan melipatkan pahalanya serta mengganti dan membalas segala kebaikan mereka dengan yang lebih baik. Dan hanya dengan ridho dan anugerah Allah S.W.T saja buku ini dapat selesai. Alhamdulillah

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I
PENDAHULUAN

1.1 Latar Belakang

Sesuai dengan tujuan penciptaannya, kemajuan dibidang teknologi robotika semakin memberikan kemudahan bagi manusia. Didalam buku "ROBOTIKA : Desain,Kotrol dan Kecerdasan Buatan". Dr. Endra Pitowarno menyebutkan bahwa Awal mula Robotika eksis di dunia diawali dengan pertunjukan lakon komedi berjudul *Rossum's Universal Robot* yang ditulis oleh Karl Capek pada tahun 1921 yang mengisahkan tentang sebuah mesin menyerupai manusia, mampu bekerja secara terus menerus tanpa lelah [1]. Perkembangan teknologi robotika pun jauh berkembang dengan dukungan kemajuan teknologi dan disiplin ilmu lainnya antara lain: teknologi sensor, komunikasi, metode kontrol dan lain-lain.

Salah satu pengembangan teknologi robotika adalah bidang *Swarm Robotics* yaitu salah satu cabang ilmu robotika yang mempelajari tentang sekumpulan robot dengan struktur fisik relatif sederhana dan kesamaan perilaku yang mampu bekerja sama dari hasil interaksi antar robot dan interaksi antara robot dengan lingkungannya [2]. Umumnya konsep – konsep tersebut masih terinspirasi oleh kehidupan di alam (*Biological Inspiration*) sehingga menjadi sangat aplikatif dalam kehidupan kita sehari-hari. Dalam penelitiannya professor mclurkin membagi skema robot swarm kedalam beberapa skema : *Beacon Navigation, follow the leader, Match Orientation,* dan *Orbit robot* adalah beberapa contoh perilaku (*behaviors*) dari robot swarm [3]. Untuk melakukan tugas yang sangat sulit dan kompleks diperlukan robot yang memiliki kemampuan yang lebih namun ini membuat robot semakin rumit dan mahal. Dapat dibayangkan jika robot tersebut mengalami masalah disaat tugas yang diberikan belum selesai dieksekusi sepenuhnya, misi akan gagal sepenuhnya. Disinilah letak keunggulan konsep robot swarm bahwa menggunakan beberapa robot yang sederhana dan murah lebih efisien daripada menggunakan satu robot yang rumit dan mahal.

Dari berbagai macam jenis robot *swarm*, formasi dengan Perilaku Mengikuti menjadi tema yang sangat menarik karena sering kita jumpai

dalam kehidupan sehari – hari tanpa kita sadari. Secara garis besar inspirasi tema perilaku mengikuti pemimpin didapat dari inspirsi alam dan kebiasaan hidup manusia. Contoh yang sering kita jumpai jalannya bebek dan semut dan formasi angsa ketika terbang. Sedang dari kebiasaan manuasi semisal konvoi kendaraan di jalan raya. Perilaku berkonvoi di jalan raya sangat rentan terjadi kecelakaan karena sedikit kelalaian salah satu pengendara dari anggota konvoi akan menjadi pemicu kecelakaan dalam skala yang besar sehingga diharapkan konsep Perilaku Mengikuti Pemimpin dapat diaplikasikan untuk menjadi solusi dari permasalahan tersebut.

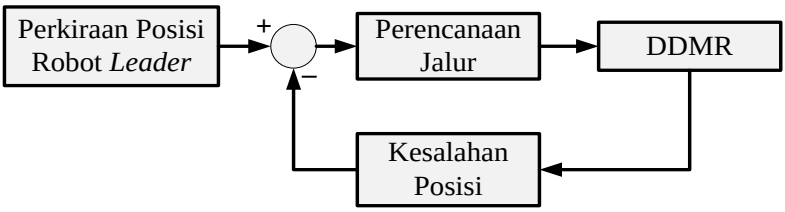
Pada tugas akhir ini akan dibangun *Swarm Robots* dengan formasi *follow the leader* (selanjutnya akan disebut mengikuti Pemimpin) yang terdiri dari sebuah mobile robot *leader* (selanjutnya akan disebut dengan robot pemimpin) dan satu atau lebih mobile robot *follower* (selanjutnya akan disebut dengan robot pengikut). Semua mobile robot tersebut harus mampu bergerak secara mandiri tanpa dikendalikan lagi oleh manusia. Agar mampu bergerak secara mandiri maka wahana tersebut memerlukan sensor-sensor untuk mengolah data informasi tentang kondisi lingkungan sekitar sebagai *input* kendali pergerakan. Kehandalan dari konsep Perilaku Mengikuti Pemimpin terletak pada kemampuan manuver robot Pengikut dalam mengikuti robot Pemimpin oleh karena itu robot pengikut harus memiliki kemampuan untuk mengetahui posisi dirinya terhadap wahana/robot lain dan juga sebaliknya.

Dewasa ini semakin banyak para engineer yang mengembangkan Robot swarm karena penggunaan beberapa robot yang mempunyai tugas tertentu akan lebih menghemat sumberdaya baik itu biaya maupun tenaga serta kemudahan dalam perancangan dibanding satu robot yang multifungsi. Aplikasi robot swarm di bidang militer yaitu pemetaan dan survei area sedangkan aplikasi lain di bidang eksplorasi luar angkasa yaitu pengumpulan sample batu dan pencarian jejak air. Perilaku Mengikuti Pemimpin adalah salah satu jenis dari robot swarm.

Robot pemimpin adalah robot yang lintasannya dijadikan acuan oleh robot pengikut di belakangnya, robot pemimpin dilengkapi dengan sensor halangan sedang robot pengikut adalah robot buta yang mengandalkan informasi yang diberikan oleh robot pemimpin. Karena itu untuk memenuhi konsep mengikuti pemimpin maka robot pengikut harus mampu mengetahui posisi dari robot pemimpin. Banyak metode yang dapat diterapkan untuk pemetaan mulai dengan informasi global,

dalam hal ini menggunakan GPS, atau menggunakan informasi local seperti vision based system. System dengan GPS sangat tidak baik digunakan pada robot berukuran kecil, sedangkan vision based akan lambat dalam prosesnya dan memerlukan dukungan perangkat keras yang tidak murah dan berkemampuan rendah. Penentuan posisi relatif robot pemimpin dapat dilakukan dengan mengkombinasian antara arah hadap kedua robot terhadap kutub bumi. Banyak jenis sensor yang dapat diaplikasikan untuk keperluan penghitungan jarak. Selama ini sensor yang sering dipakai untuk keperluan pengukuran jarak adalah sensor yang bekerja pada medium cahaya dan suara(ultrasonik). Medium ultrasonik adalah medium yang paling sesuai untuk kasus ini. Karena suara/ultrasonik mempunyai kecepatan yang jauh lebih rendah dibanding dengan cahaya ataupun gelombang radio, namun justru nilai kurang disini merupakan celah yang dapat kita manfaatkan. Untuk sinkronisasi antara robot pemimpin dengan robot induk digunakan medium gelombang radio.

Salah satu jenis mobile robot yang paling populer adalah mobile robot berpenggerak diferensial atau *Differential Drive Mobile Robot* (DDMR) yakni mobile robot dengan dua buah penggerak berupa motor yang masing-masing bergerak secara independen kemudian ditambahkan roda bebas atau *castor* untuk menjaga keseimbangan. Metode kendali ini lebih populer dikarenakan perhitungan kinematika dan dinamikanya lebih sederhana. Sedangkan untuk kendali kecepatan robot dengan sistem umpan balik.



Gambar 1.1 *: Blok diagram kerja robot pengikut*

1.2 Tujuan

Merancang dan membuat robot yang mampu bergerak mengikuti trajektori robot didepannya yang didefinisikan sebagai robot pemimpin

dengan jarak tertentu dan dapat berkoordinasi antar robot menggunakan komunikasi nirkabel radio frekuensi**.** Metode pengukuran jarak menggunakan sensor ultrasonik dengan prinsip Time Of Flight.

1.3 Batasan Masalah

Pemberian batasan-batasan atau ruang lingkup bertujuan untuk memfokuskan terhadap masalah-masalah yang dikemukakan. Berikut adalah ruang lingkup dalam pembuatan Proyek akhir ini:

1. Robot pengikut bergerak mengikuti robot pemimpin dengan jarak tertentu secara konstan menggunakan panduan data jarak antar robot dan data posisi robot pemimpin yang diwakili dengan peruabahan gerakan roda.
2. Pada posisi awal (start) robot pengikut diasumsikan berada dibelakang robot dengan koordinat yang telah ditentukan.
3. Robot menggunakan kemudi differensial dan pengujian dilakukan pada permukaan rata dan tidak licin.

1.4 Metodologi

Dalam pengerjaan proyek akhir ini, penulis melakukan perencanaan awal berupa rancangan system yang akan melandasi semua pekerjaan yang akan dikerjakan pada tugas akhir ini. Rancangan itu berupa :

- Studi literatur meliputi pembahasan materi yang berkaitan dengan Robot *Swarm* dengan *Follow The Leader Behaviors, PID control,* Kinematika *Differential Drive Mobile Robot,* Metode *Time Of Flight* untuk pengukuran jarak
- Mempelajari secara singkat materi yang berkaitan dengan kinematika DDMR.
- Mempelajari tentang lembar data dari sensor ultrasonik, modul komunikasi RF Xbee Pro, modul kompas elektronik CMPS03.
- Mempelajari lembar data mikrokontroler yang memungkinkan untuk melakukan algoritma kontrol secara keseluruhan secara efisien.
- Mempelajari materi tentang pengaturan konfigurasi fitur-fitur pada mikrokontroler yang akan digunakan.
- Merakit mekanik DDMR beserta *Printed Circuit Board* (PCB).

- ❖ Melakukan kalibrasi, pengujian dan pengambilan data dari peranti sensor CMPS03, rotary encoder, transducer ultrasonik dan peranti komunikasi Xbee Pro. Pengambilan data dimaksudkan untuk mengetahui performansi peranti yang diuji terhadap beberapa faktor yang dianggap berpotensi mengintervensi kinerja.
- ❖ Melakukan perencanaan pembagian tugas dari mikrokontroler-mikrokontroler yang digunakan agar task-task dapat dikerjakan dengan efisien. Hal ini dilakukan disebabkan task-task yang cukup banyak dan terbatasnya kapasitas memori mikrokontroler.
- ❖ Membuat program untuk masing-masing mikrokontroler sesuai dengan pembagian kerja.
- ❖ Menguji coba dan mengevaluasi kinerja program secara terpisah.
- ❖ Menguji coba program dan system secara keseluruhan.

1.5 Sistematika Pembahasan

Sistematika proyek akhir dibagi menjadi lima bab, yang susunannya sebagai berikut:

BAB I Pendahuluan

Pada bab pertama ini berisi tentang latar belakang umum, latar belakang, tujuan, metodologi, batasan masalah, dan sistematika pembahasan.

BAB II Studi Literatur

Pada Bab kedua berisi literatur tentang Perilaku Mengikuti Pemimpin *Behaviors*, Kinematika dan Dinamika DDMR, kontrol Motor DC, dan *Rotary Encoder*.

BAB III Desain dan Eksperimen Perangkat Keras

Pada Bab ketiga membahas tentang proses perencanaan sampai dengan pembuatan perangkat keras meliputi rangkaian transducer ultrasonik, kompas elektronik, rotary encoder, modul komunikasi *Radio Frequency* Xbee Pro beserta teknik antarmuka pada rangkaian mikrokontroler yang disajikan per

sub-bab. Masing-masing sub bab akan ditampilkan data hasil eksperimen.

BAB IV Perencanaan Dan Eksperimen Pergerakan Robot Berdasarkan Algoritma Perilaku Mengikuti Pemimpin

Pada Bab keempat menyajikan desain perangkat lunak dari algoritma gerakan robot pengikut mengikuti robot pemimpin yang didalamnya terdapat algoritma pengukuran jarak berdasarkan *Time Of Flight* dan perencanaan rute (*path planning*) dalam bentuk flowchart, dan listing program. selain desain juga akan diberikan data hasil eksperimen yang akan menggambarkan performansi perangkat lunak tersebut.

BAB V Kesimpulan

Pada bab kelima ini berisi kesimpulan dan saran yang berkaitan dengan keseluruhan proyek akhir ini.

BAB II
STUDI LITERATUR

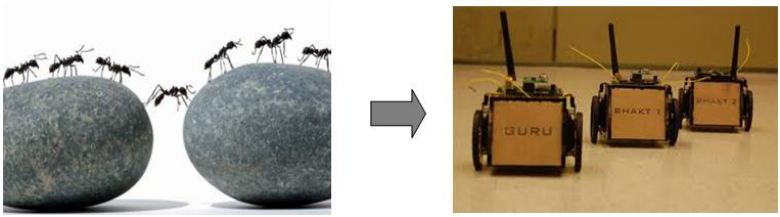
Studi literatur merupakan langkah awal, dengan mempelajari literatur yang kemungkinan nanti akan diperlukan. Dalam pengerjaan Proyek Akhir ini studi literatur akan terus mengiringi sampai pengerjaan proyek akhir selesai, karena tentunya akan sangat diperlukan dalam mendesain maupun menganalisa

2.1 Tujuan dan Metodologi

Studi literatur ini mempunyai tujuan untuk mendapatkan teori teori yang diperlukan dalam pengerjaan Proyek Akhir. Sedangkan metodologi studi literatur mengambil dari beberapa sumber literatur seperti buku, paper serta sumber-sumber lain yang diperlukan.

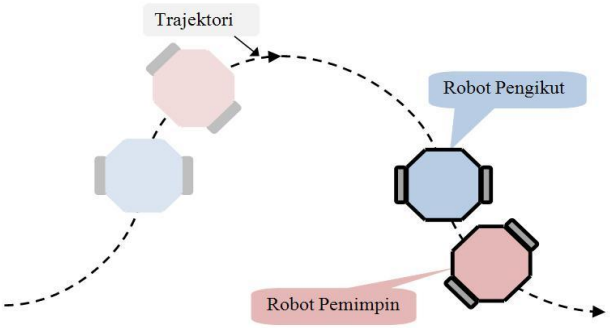
2.2 Robot *Swarm* dengan formasi Perilaku Mengikuti Pemimpin

Perilaku mengikuti pemimpin atau lebih dikenal dengan *Follow The Leader* merupakan salah satu skema dari robot swarm. Robot-robot *Swarm* sendiri memiliki pengertian sejumlah robot dengan struktur fisik relatif sederhana dan kesamaan perilaku yang mampu bekerja sama dari hasil interaksi antar robot dan antara robot dengan lingkungannya. Para peneliti terus berupaya mengembangkan algoritma robot swarm dengan tetap mengambil inspirasi dari kehidupan di alam (*Biological Inspiration*). Pada gambar 2.1 dijelaskan contoh biologikal inspiration yang mendasari algoritma mengikuti pemimpin.



Gambar 2.1 *: Perilaku koloni semut yang mendasari penelitian algoritma perilaku mengikuti pemimpin behavior pada robot GuruBhakts yang dikembangkan oleh Jitender Bishnoi dan Rahul Khosla[4]*

Dalam formasi mengikuti pemimpin ini terdapat satu robot yang berperan sebagai robot pemimpin dan sebagian yang lain sebagai robot pengikut. P Chandhak dalam tesis S2 nya menyatakan bahwa robot pengikut harus mampu mengikuti trajektori robot didepannya yang diidentifikasikan sebagai robot pemimpin dengan jarak yang telah ditentukan dengan relatif konstan dalam kecepatan dan akselerasi yang bervariasi, bahkan keadaan berhenti mendadak [5] seperti yang diilustrasikan pada gambar 2.2. Robot pengikut bergerak mengikuti lintasan yang dibuat oleh robot pemimpin sehinggga informasi tentang posisi relatif antar robot dengan cepat dan akurat sangat diperlukan.



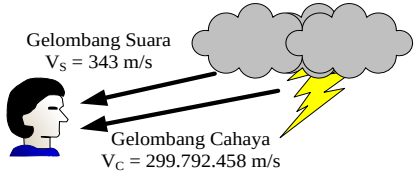
Gambar 2.2 *: Ilustrasi perilaku mengikuti pemimpin*

Metode pemetaan letak antar robot dapat dengan menggunakan informasi global maupun informasi lokal. Informasi global dapat diperoleh dengan menggunakan GPS sedang informasi lokal dapat diperoleh dengan menggunakan kamera (vision based system), menggunakan sensor ultrasonik, line of sight dengan inframerah dan lainnya. Namun menggunakan GPS untuk robot dengan dimensi dan skala kecil sangat tidak praktis dan akurat karena GPS mempunyai kesalahan posisi yang cukup besar. Selain itu untuk penggunaan pada ruang 2 dimensi maka pengetahuan posisi hanya pada batasan bidang yang sama, pada bidang horizontal saja atau bidang vertical saja. Dan jika menggunakan vision based system selain dibutuhkan perangkat keras yang rumit hasilnya dikhawatirkan tidak akurat apabila mobilitas robot sangat tinggi.

2.3 Prinsip *Time Of Flight* Untuk Pengukuran Jarak

Time of Arrival (ToA) atau juga disebut *Time of flight* (ToF) adalah waktu tempuh yang diperlukan sinyal radio dari sebuah pemancar (*transmitter*) sampai diterima oleh penerima (*receiver*) [6]. Dalam hubungannya antara kecepatan cahaya diruang hampa dengan frekuensi sinyal pembawa maka dapat diperoleh jarak antara transmitter dengan receiver. Pengukuran jarak tersebut hanya berlaku untuk pengukuran diruang hampa udara sehingga hasilnya akan berbeda jika dilakukan dengan media lain namun fakta ini sering diabaikan.

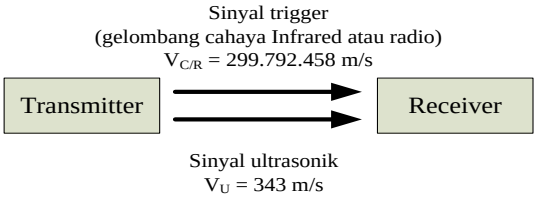
Berbeda dengan metode *Time of Arrival* yang hanya menggunakan waktu kedatangan absolut pada sebuah transduser, metode *Difference in Time of Arival* lebih memperhatikan selisih waktu antara waktu keberangkatan pada sebuah *station* dengan waktu kedatangan di *station* lain [6]. Metode tersebut dapat dianalogikan dengan bagaimana kita mengetahui seberapa jauh jarak petir dengan mengukur jeda waktu saat kilat petir menyambar dengan suara petir terdengar.



Gambar 2.3 *: Prinsip Time Of Arrival dalam fenomena alam petir*

Terinspirasi dari fenomena kilat dan suara petir di atas metode DTOA yang kami gunakan dapat dijelaskan sebagai berikut:

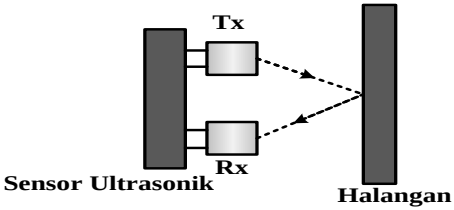
1. Robot pemimpin akan memancarkan gelombang radio sebagai sinyal trigger dan gelombang ultrasonik dalam waktu yang hamper bersamaan.



Gambar 2.4 *: Prinsip TDOA untuk pengukuran jarak pada sensor ultrasonik*

2. Gelombang ultrasonik dan gelombang radio dipancarkan dalam waktu hamper bersamaan namun merambat dengan kecepatan yang jauh berbeda.
3. Robot pengikut akan menerima sinyal trigger terlebih dahulu dan memulai menghitung dengan waktu gelombang ultrasonik sampai pada dirinya. Sehingga didapatkan jarak antar 2 robot.

Sensor pengukur jarak yang populer dalam aplikasi robotika adalah sensor ultrasonik dan sensor *Position Sensitive Device* (PSD). Kedua sensor bekerja berdasarkan sinyal pantul (echo) yang ditangkap oleh penerima. Pada sensor ultrasonik data jarak terukur sebanding dengan waktu antara sinyal dikirim dengan sinyal echo diterima. Desain sensor ultrasonik pada umumnya menempatan transmitter (Tx) dan receiver (Rx) dalam satu board sehingga sinyal ultrasonik akan mengalami pemantulan sebelum diterima oleh perangkat receiver seperti yang diilustrasikan pada gambar 2.5.



Gambar 2.5 *: Sensor Ultrasonik dengan Tx dan Rx dalam satu board*

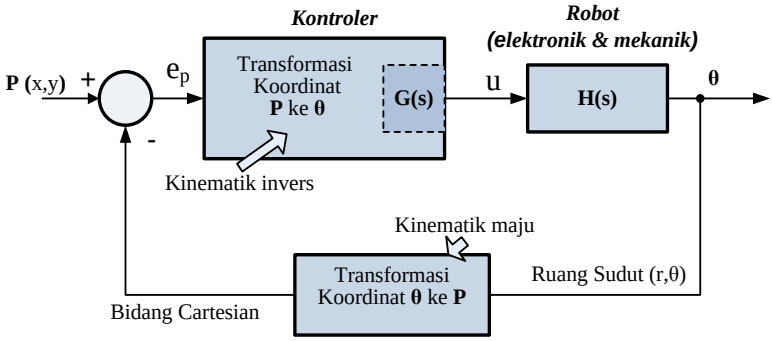
Desain sensor pada gambar 2.5 sangat menguntungkan dalam hal sinkronisasi proses pengukuran. Jika metode time of flight digunakan

untuk mengukur jarak antara pemancar dengan penerima yang terpisah maka prosedur sinkronisasi menjadi lebih rumit seperti pada gambar 2.4. Media sinyal trigger dari gelombang elektromagnetik yang termodulasi menjadi multak diperlukan.

2.4 Kinematika Mobile Robot Dengan Kemudi Diferensial

Mobile Robot dapat dianalisa dalam dua domain kajian yaitu analisa kinematika yang berkaitan dengan gerakan robot tanpa memandang efek inersia/kelembaman ketika robot bergerak dan analisa dinamika yang berhubungan dengan efek inersia dari struktur fisik robot yang ditimbulkan oleh torsi aktuator. Sistem robotik secara garis besar terdiri dari sistem kontroler, elektronik dan mekanik robot seperti blok diagram pada gambar 2.6

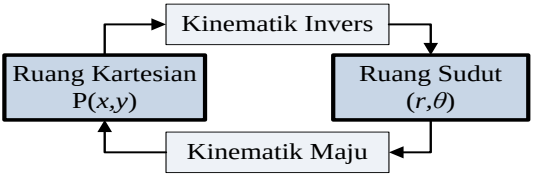
G(s) adalah persamaan matematik kontroler sedangkan H(s) adalah persamaan sistem robotik secara fisik. Komponen r_i adalah masukan referensi dalam fungsi waktu berupa referensi posisi, kecepatan dan percepatan yang bervariasi dan kontinyu sehingga membentuk suatu konfigurasi trajektori. Komponen e_p adalah error, u adalah sinyal keluaran dari kontroler dan y adalah keluaran dalam domain sudut perputaran roda mobile robot.



Gambar 2.6 *: Diagram system control robotic*

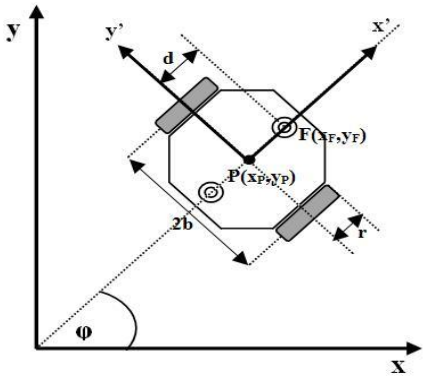
Jika input merupakan fungsi dari suatu koordinat vektor posisi dan orientasi P(x,y). Keluaran yang diukur dari gerakan robot adalah

sudut perputaran roda. Dalam pemrograman, dibutuhkan pemetaan ruang kerja robot berupa posisi titik tertentu pada bagian robot yang dinyatakan sebagai koordinat kartesian 2D sehingga perlu dilakukan transformasi koordinat antara ruang kartesian dengan ruang sudut ini. Di dalam gambar 2.7 dinyatakan sebagai kinematik invers dan kinematik maju. Kombinasi antara transformasi koordinat P ke θ dengan kontroler G(s) disebut sebagai kontroler kinematik. Masukan berupa sinyal error P ($\mathbf{e_p}$) dan keluaran berupa sinyal kemudi **u** untuk aktuator.



Gambar 2.7 *: Transformasi kinematik maju dan kinematik invers*

Mobile Robot dengan kendali *Differential Drive* berpenggerak dua buah roda kiri-kanan yang dikendalikan secara independen seperti yang diilustrasikan pada gambar 2.8 Mobile robot bergerak dalam bidang 2-Dimensi sehingga dapat diasumsikan bergerak dalam sumbu XY saja, sementara kontur dan ketinggian medan sebagai sumbu Z dapat diabaikan.



Gambar 2.8 *: DDMR pada bidang kartesian 2 dimensi*

Berikut ini adalah parameter umum dalam kalkulasi kinematika pada mobile robot berpenggerak *Differential Drive*:

φ	: sudut hadap robot terhadap sumbu x bidang cartesian
2b	: lebar robot diukur dari garis tengah antar roda
r	: jari-jari roda
(X_0,Y_0)	: koordinat acuan di tubuh robot terhadap sumbu XY
X,Y	: sumbu global robot
X',Y'	: sumbu lokal robot

Jika robot diasumsikan berjalan relatif pelan tanpa slip antara kedua roda dengan permukaan medan dan titik F (castor) sebagai acuan analisa maka komponen x dan y dapat dinotasikan dalam persamaan non-holonomic sebagai berikut,

$$\dot{x}_F \sin\varphi - \dot{y}_F \cos\varphi + \dot{\varphi} d = 0 \qquad (2.1)$$

Masalah klasik dalam kontrol kinematik DDMR adalah terdapat dua aktuator namun parameter kontrolnya lebih dari dua, yaitu x untuk gerakan ke arah X (1DOF) dan y untuk arah Y (1DOF) yang dukur relatif terhadap perpindahan titik P, dan gerakan sudut hadap φ yang diukur dari garis hubung titik P dan F terhadap sumbu X.

Dari persamaan 2.1 Nampak bahwa derajat kebebasan dalam kontrol kinematiknya berjumlah tiga yaitu (*x, y, φ*) karena ketiga parameter ini perlu dikontrol secara simultan untuk mendapatkan gerakan non-holonomic

Persamaan kinematik untuk DDMR dapat dinyatakan dalam persamaan kecepatan berikut

$$\begin{bmatrix} \dot{x}_F \\ \dot{y}_F \\ \dot{\varphi}_F \end{bmatrix} = \mathbf{T_{NH}} \begin{bmatrix} \dot{\boldsymbol{\theta}}_L \\ \dot{\boldsymbol{\theta}}_R \end{bmatrix} \quad \text{atau} \quad \dot{\boldsymbol{q}}(t) = \mathbf{T_{NH}}(\mathbf{q})\, \dot{\boldsymbol{\theta}}(t) \qquad (2.2)$$

TNH adalah matriks transformasi non-holonomik $\dot{\boldsymbol{\theta}}_L$ dan $\dot{\boldsymbol{\theta}}_R$ adalah kecepatan radial roda kiri dan kanan, *q* adalah sistem koordinat umum robot.

$$Q = [x_F, y_F, \varphi]^T \text{ atau } q = \begin{bmatrix} x_F \\ y_F \\ \varphi \end{bmatrix} \qquad (2.3)$$

Jika $\mathbf{T_{NH}}$ diuraikan dari persamaan 2.1 Dengan memperhatikan gambar 2.8 maka didapat ditentukan

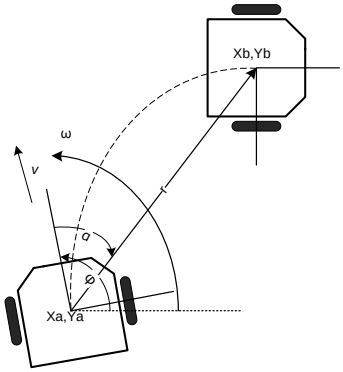
$$\mathbf{T_{NH}\,(q)} = \begin{bmatrix} \frac{r}{2}cos\varphi + \frac{d\,.\,r}{2b}sin\varphi & \frac{r}{2}cos\varphi - \frac{d\,.\,r}{2b}sin\varphi \\ \frac{r}{2}sin\varphi - \frac{d\,.\,r}{2b}cos\varphi & \frac{r}{2}sin\varphi + \frac{d\,.\,r}{2b}cos\varphi \\ -\frac{r}{2b} & \frac{r}{2b} \end{bmatrix} \qquad (2.4)$$

Kinematik inversnya dapat ditulis,

$$\dot{\boldsymbol{\theta}}(t) = \mathbf{T_{NH}^{-1}}\,(\boldsymbol{q})\dot{\boldsymbol{q}}(t) \qquad (2.5)$$

2.5 Pencarian target

Pencarian target dilakukan dengan cara menggabungkan data jarak dengan data perubahan gerak roda robot. Pada gambar 2.9 diilustrasikan metode pencarian posisi robot pemimpin dan pergerakan robot pengikut mengikuti robot pemimpin.



Gambar 2.9 : *Robot pemimpin dan robot pengikut.*

dalam gambar 2.9 *x,y* adalah koordinat cartesian robot, *φ* adalah sudut yang dibentuk dari posisi robot terhadap sumbu *x*, v adalah kecepatan linear robot , dan ω adalah kecepatan sudut(angular) robot. Pada pencarian target dimana posisi awal robot pemimpin adalah di

(xb,yb,φb) dan posisi robot pengikut adalah *(xa,ya,φa), α* didefinisikan sebagai perubahan sudut antara posisi awal dan akhir, dengan posisi sudut akhir adalah φb maka secara mate-matis dapat dinyatakan:

$$r = \sqrt{(xa - xb)^2} + (ya - yb)^2 \quad (2.6)$$

$$\varphi b = \text{atan}\,(y, x) \quad (2.7)$$

$$\alpha = \varphi a - \varphi b \quad (2.8)$$

atan adalah invers tangen pada 4 kuadran. Dengan cara mensbtitusikan koordinat baru r, φb, dan α model kinematik menjadi seperti pada persamaan (2.9).

$$\dot{r} = -v\cos(\varphi b - \varphi a) \quad (2.9)$$

$$\dot{\varphi b} = v\,\frac{\sin(\alpha)}{r} \quad (2.10)$$

$$\dot{\varphi a} = \omega \quad (2.11)$$

Dengan mengganti α pada φb-φa didapatkan,

$$\dot{r} = -v\cos(\alpha) \quad (2.12)$$

$$\dot{\alpha} = -\omega + v\,\frac{\sin(\alpha)}{r} \quad (2.13)$$

$$\dot{\varphi b} = v\frac{\sin(\alpha)}{r} \quad (2.14)$$

Yang perlu diperhatikan adalah, bahwa persamaan diatas hanya dikatakan benar jika error *e* sama dengan nol. Oleh karena itu trajectory diantara 2 posisi tidak dihitung sebagai suatu kepastian, tetapi dengan menerapkan error pergerakan yang sangat kecil pada keadaan yang sebenarnya, misalnya odometri error yang diakibatkan oleh pembacaan rotary encoder yang kurang presisi.

Dengan memperhatikan persamaan *Lyapunov*, yaitu:

$$V = V_1 + V_2 = \tfrac{1}{2}\lambda r^2 + \tfrac{1}{2}(\alpha^2 + h\varphi b^2);\ \lambda, h > 0 \quad (2.15)$$

V1 dan V2 dianggap sebagai akar rata-rata error vektor jarak r dan sejajar dengan vector[$\alpha, \sqrt{h\varphi b}$]. Dengan menurunkan v, persamaan *Lyapunov* menjadi,

$$\dot{V} = \dot{V}_1 + \dot{V}_2 = \lambda r\dot{r} + \frac{1}{2}\left(\alpha\dot{\alpha} + h\varphi b\dot{\varphi b}\right)$$
$$= \lambda r cos\alpha + \alpha\left[-\omega + v\frac{\sin\alpha(\alpha + h\varphi b)}{\alpha r}\right] \quad (2.16)$$

Sehingga dapat diuraikan dari persamaan 2.16, mengakibatkan suatu kondisi dimana $\dot{V}_1$ dapat bernilai negatif, jika kecepatan linear *v* mengikuti aturan berikut,

$$v = (\gamma cos\alpha)r;\ \gamma > 0 \quad (2.17)$$

Dengan mengganti *v* pada rumus persamaan 2.17, $\dot{V}_1$menjadi,

$$\dot{V}_1 = -(\lambda sin^2\alpha)r^2 \leq 0 \quad (2.18)$$

$\dot{V}_1$ konvergen didefinisikan sebagai batas positif, analogi untuk $\dot{V}_2$

$$\dot{V}_2 = \alpha\left[-\omega + \gamma\frac{\cos\alpha\, sin\alpha}{\alpha}(\alpha + h\varphi b)\right] \quad (2.19)$$

Sehingga dari persamaan diatas, mengakibatkan suatu kondisi dimana $\dot{V}_2$ dapat bernilai negatif, jika kecepatan angular ω, mengikuti aturan sebagai berikut:

$$\omega = k\alpha\left[\lambda\frac{\cos\alpha\, sin\alpha}{\alpha}(\alpha + h\varphi b)\right];(k > 0) \quad (2.20)$$

Dengan mengganti ω pada persamaan 2.20, $\dot{V}_2$ menjadi,

$$\dot{V}_2 = k\alpha^2 \leq 0 \quad (2.21)$$

Dengan mengganti $\dot{V}_1$ dan $\dot{V}_2$, didapatkan solusi dari *V*,

$$\dot{V} = \dot{V}_1 + \dot{V}_2 = -(\gamma\, cos^2\alpha)r^2 - k\alpha \leq 0 \quad (2.22)$$

Fungsi *lyapunov* kovergen ke batas bilangan positif. Dengan menggunakan persamaan diatas dan menerapkannya kedalam model kinematik pada model persamaan 2.9 sampai persamaan 2.11, sehingga didapatkan,

$$\dot{r} = -(\gamma cos^2\alpha)r \quad (2.23)$$

$$\dot{\varphi b} = -(k\alpha - \gamma h \frac{\cos\alpha \sin\alpha}{\alpha} r(0) > 0 \quad (2.24)$$

$$\dot{\varphi b} = \gamma \cos\alpha \sin\alpha \quad (2.25)$$

Karena r dan φb kovergen ke nol, sehingga berakibat $\dot{r}$ dan $\dot{\varphi b}$ kovergen ke nol pula.

2.6 Penelusuran Trajektory (*Trajectory Tracking)*

Tugas yang sulit dalam pengaturan gerak robot adalah pada bagian pengaturan kecepatan input roda, untuk menelusuri trajektory refferensi yang telah diperoleh dari pengiriman robot pemimpin. Pertama-tama yang harus dilakukan yaitu mengatur kesalahan posisi, dengan mengasumsikan posisi refferensi $q_{ref} = (x_{ref}, y_{ref}, \theta_{ref})^T$, dan posisi aktual robot adalah $q_{act} = (x_{act}, y_{act}, \theta_{act})^T$, kesalahan posisi $q_e = (x_e, y_e, \theta_e)^T$ adalah transformasi dari posisi refferensi q_{ref} pada sebuah sistem koordinat lokal dengan posisi asal (x_{act}, y_{act}) dan sumbu x dengan arah hadap robot adalah θ_{act}. Sehingga dapat diwuudkan dalam persamaan matematis, sebagai berikut,

$$\begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix} = \begin{bmatrix} \cos\theta & sin\theta \\ -sin\theta & cos\theta \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_{ref} - x_{act} \\ y_{ref} - y_{act} \\ \theta_{ref} - \theta_{act} \end{bmatrix} \quad (2.26)$$

Model refferensi kecepatan linear dan angular (v,ω) yang bisa menyebabkan q_e kovergen ke nol, diberikan oleh (I. Kolmanovsky dkk, 1995; fierro dan lewis, 1998) adalah sebagai berikut,

$$V_{des} = V_{ref} \cos e_3 + K_1 e_1 \quad (2.27)$$

$$\omega_{des} = \omega_{ref} + v_{ref}\ k_2 e_2 + K_3 \sin e_3 \quad (2.28)$$

Dengan K1,K2,K3 adalah konstanta positif

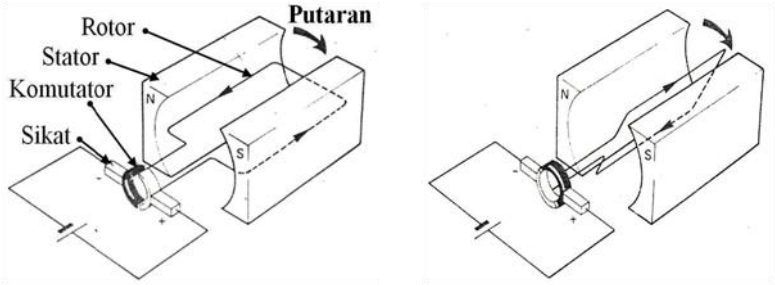
2.7 Motor DC

Motor merupakan elemen dasar sistem pergerakan sebuah wahana bergerak karena mampu merubah arus listrik menjadi tenaga mekanik yang berupa putaran rotor. Penyusun utama motor DC adalah *Stator* (kumparan pada motor dc yang tidak berputar), *Rotor* (kumparan jangkar yang berputar) dan *Komutator*.

Gambar 2.10 *: Fisik Motor DC dengan Gearbox*

Secara umum mekanisme kerja untuk motor DC adalah:

- Arus listrik dalam medan magnet akan memberikan gaya.
- Jika kawat yang membawa arus dibengkokkan menjadi sebuah loop, maka kedua sisi loop (sudut kanan medan magnet) mendapatkan gaya pada arah yang berlawanan.
- Pasangan gaya menghasilkan tenaga putar / torsi untuk memutar kumparan.
- Motor memiliki beberapa loop pada dinamonya untuk memberikan tenaga putaran yang lebih seragam dan medan magnetnya dihasilkan oleh kumparan medan.
- Kecepatan motor DC dapat diatur dengan menaik-turunkan tegangan pada kumparan medan yang dapat dilakukan dengan mengatur duty cycle / *Pulse Width Modulation* (PWM).



(a) (b)

Gambar 2.10 *: (a) Komponen penyusun motor DC (b) Putaran rotor*

2.8 Piranti Sensor

Sistem navigasi mobile robot ini menggunakan kombinasi antara sensor jarak dengan perubahan kecepatan roda robot. Sensor-sensor tersebut digunakan untuk keperluan lokalisasi, dimana ada dua kategori sensor dari sudut pandang robot yaitu sensor lokal (*On-Board*) yang dipasang pada bodi robot dan sensor global, yaitu sensor yang diinstal di luar bodi robot namun masih dalam lingkungan kerjanya dan saling berkomunikasi, sebagai contoh: GPS. Penggunaan GPS digunakan untuk wilayah kerja yang sangat luas sehingga tidak cocok jika digunakan untuk aplikasi lokalisasi di dalam ruangan (*Indoor*).

2.8.1 Sensor Ultrasonik

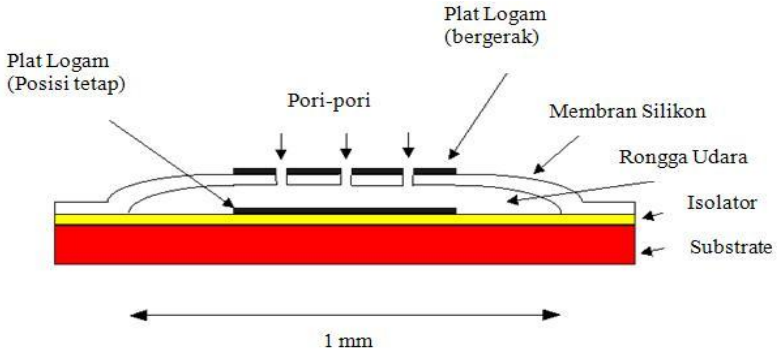
Ultrasonik adalah gelombang bunyi dengan frekuensi di atas daerah audio manusia (di atas 20 KHz). Gelombang ultrasonik diproduksi dengan dua cara kerja yaitu dengan efek piezo-elektrik dan magneto-striksi namun piezo-elektrik adalah prinsip yang paling sering digunakan. Frekuensi kerja untuk transduser ultrasonik pada umumnya adalah 40 KHz yang dihasilkan rangkaian osilator dan keluaran dari osilator dilanjutkan menuju penguat sinyal. Besarnya frekuensi ditentukan oleh komponen kalang RLC / kristal. Penguat sinyal akan memberikan sebuah sinyal listrik yang diumpankan ke piezoelektrik dan terjadi reaksi mekanik sehingga bergetar dan memancarkan gelombang yang sesuai dengan besar frekuensi pada osilator.

(a) (b)

Gambar 2.11 *: sensor ultrasonik*

Efek piezo-elektrik dapat dibuat dari kuarsa dan keramik (lead zirconate-titanate). Setelah muatan dua-kutub mengalami polarisasi, bahan tersebut mampu berubah panjangnya jika ada medan listrik yang dikenakan sepanjang arah polarisasi. Sebaliknya jika bahan itu diregangkan secara mekanis ke arah polarisasinya akan membangkitkan tegangan listrik yang membentanginya. Perubahan mekanis maksimum terjadi bila sinyal yang dikenakan berada pada frekuensi yang sama dengan frekuensi resonansi dari transduser sehingga dapat digunakan sebagai transmitter dan receiver.

Receiver berfungsi sebagai penerima gelombang pantulan dari transmitter yang dikenakan pada permukaan benda atau gelombang langsung LOS (Line of Sight) dan memiliki reaksi yang membangkitkan tegangan listrik saat gelombang dengan frekuensi yang resonan.



Gambar 2.12 *: Piranti Piezo-elektrik*

Sebuah gelombang ultrasonik berjalan melaui sebuah medium, secara matematis besarnya jarak dapat dihitung sebagai berikut:

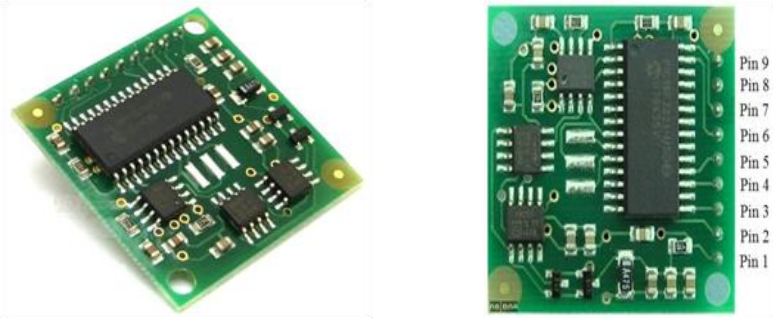
$$s = (v.t) / 2 \quad (2.29)$$

dimana,
s = jarak (meter),
v= kecepatan suara (344 m/detik)
t = waktu tempuh (detik)

Saat gelombang ultrasonik menumbuk suatu penghalang, sebagian gelombang tersebut dipantulkan dan sebagian yang lain diserap. Kemudian sistem mengukur waktu yang diperlukan untuk pemancaran gelombang sampai kembali ke sensor dan menghitung jarak target dengan menggunakan kecepatan suara dalam medium.

2.8.2 Kompas Elektronik

Dalam navigasi mobile robot, menetukan arah hadap adalah mutlak diperlukan. Modul kompas elektronik yang akan digunakan adalah cmps03 yang menggunakan sensor medan magnet KMZ51 buatan phillips yang cukup sensitif terhadap kutub magnet bumi. Terdapat dua teknik untuk mengakses data dari modul cmps03 yaitu dengan menggunakan modulasi sinyal pulsa (PWM) atau dengan komunikasi I2C.



Gambar 2.13 *: Kompas Elektronik CMPS03*

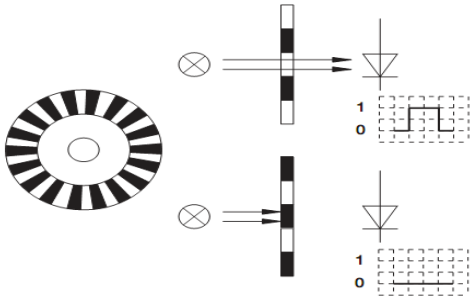
Tabel 2.1 : Konfigurasi Pin kompas elektronik CMPS03

Pin	Keterangan
1	+ 5 Volt
2	Serial Clock (SCL) untuk koneksi I2C
3	Serial Data (SDA) untuk koneksi I2C
4	Pulse Width Modulation (PWM)
5	Tidak dihubungkan
6	Kalibrasi
7	50/60 Hertz
8	Tidak dihubungkan
9	Ground

2.8.3 Rotary Encoder

Perangkat ini digunakan sebagai sensor untuk mengukur sudut, posisi, kecepatan dan percepatan dengan merubah gerakan rotasi mekanik menjadi sinyal pulsa digital sehingga dapat diproses oleh rangkaian counter, tachometer, dan mikrokontroler.

Perangkat *rotary encoder* biasanya berupa piringan kaca atau plastik yang memiliki daerah tembus cahaya dan daerah buram yang dilengkapi dengan sumber cahaya infra merah dan photodioda pada sisi yang berseberangan. Piringan tersebut biasanya dipasang dalam poros motor.



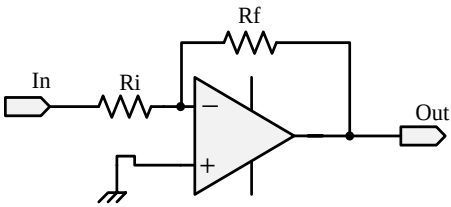
Gambar 2.14 *: Prinsip kerja rotary encoder*

2.9 Pengkondisi Sinyal

Rangkaian pengkondisi sinyal digunakan untuk mengkondisikan suatu sinyal masukan yang dapat berupa penguatan, pelemahan, penyaringan dan lain-lain agar sinyal dapat dibaca dan diolah dengan baik oleh mikrokontroler ataupun piranti lainnya.

2.9.1 Penguat Membalik (Inverting Amplifier)

Pada rangkaian inverting amplifier, input non-inverting di-ground-kan sedangkan input inverting sebagai masukan. Dengan mengasumsikan, opamp mempunyai *open loop gain* yang tidak berhingga, maka perbedaan tegangan antara input inverting dan input non-inverting sama dengan nol (Ed=0). Pada kondisi ini, input *inverting* disebut *virtual ground*. Arus yang mengalir pada Ri adalah V_{IN}/R_1 dan arus pada RF adalah V_{OUT}/R_F. Rangkaian inverting amplifier ditunjukkan pada gambar 2.15.



Gambar 2.15 *: Rangkaian penguat membalik (Inverting Amplifier)*

Penguatan tegangan pada inverting amplifier sama dengan harga resistor feedback dibagi dengan harga resistor input. Tanda minus menunjukkan adanya perbedaan fasa antara input dan output

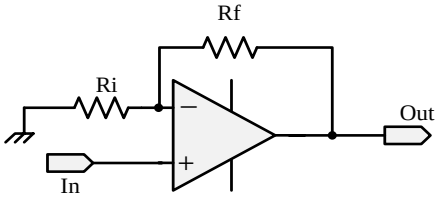
$$-\frac{Vi}{Ri} = \frac{Vo}{Rf} \quad (2.30)$$

$$Vo = \left[-\frac{Rf}{Ri}\right] * Vi \quad (2.31)$$

$$A = \frac{Vo}{Vi} = -\frac{Rf}{Ri} \quad (2.32)$$

2.9.2 Penguat Tak Membalik (Non-Inverting Amplifier)

Sering kali dibutuhkan penguat yang memberikan keluaran yang sefasa dengan masukannya serta memenuhi hubungan Rf dengan Ri. Rangkaian non inverting amplifier menjawab tantangan tersebut. Gambar 2.16 menunjukkan rangkaian non-inverting amplifier



Gambar 2.16*: Rangkaian penguat tak membalik (non inverting amplifier)*

Dengan asumsi tegangan antara tegangan terminal inverting (-) dan non-inverting (+) adalah 0 volt, berarti tegangan di titik A sama dengan Vi. Arus yang mengalir pada Ri sama dengan arus yang mengalir pada Rf, yaitu :

$$i = \frac{Vi}{Ri} \quad (2.33)$$

$$Vo = \left[1 + \frac{Rf}{Ri}\right] * Vi \quad \text{atau} \quad (2.34)$$

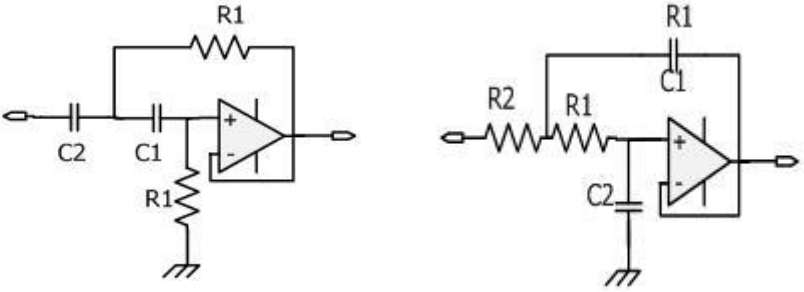
$$A = \frac{Vo}{Vi} = \left[1 + \frac{Rf}{Ri}\right] \quad (2.35)$$

2.9.3 Penapis Lolos (Pass Filter)

Rangkaian Pass filter berguna untuk meneruskan sinyal berfrekuensi tertentu dan meredam sinyal berfrekuensi selainnya. Sinyal dapat berupa sinyal listrik seperti perubahan tegangan maupun data-data digital seperti citra dan suara.

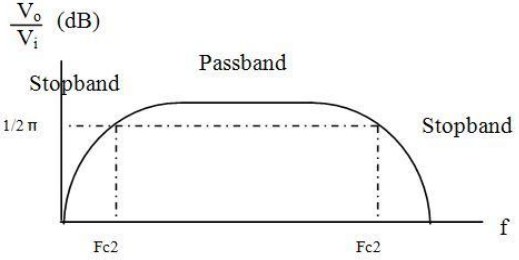
Untuk kebutuhan pass filter, terdapat 2 filter sekaligus yaitu High pass filter dan Low pass filter. Filter low pass didapat dengan meletakkan kumparan secara seri dengan sumber sinyal dan meletakkan kapasitor secara paralel dengan sumber sinyal. Sedangkan high pass didapat dengan meletakkan kapasitor secara seri dengan sumber sinyal

dan kumparan secara pararel dengan sumber sinyal. Penggabungan keduanya menghasilkan filter T dan filter phi. Contoh penggunaan pass filter pada aplikasi audio, yaitu pada tone control dan equalizer. Kumparan yang dirangkai seri dengan sumber tegangan akan meredam frekuensi tinggi dan meneruskan frekuensi rendah, sedangkan sebaliknya kapasitor yang dirangkai seri akan meredam frekuensi rendah dan meneruskan frekuensi tinggi.



Gambar 2.17 *:* Rangkaian High Pass dan Low Pass Filter

Untuk sinyal berupa data digital dapat difilter dengan melakukan operasi matematika seperti konvolusi. Finite impulse response (FIR) dan Infinite impulse response (IIR) . Contoh aplikasi low-pass filter pada data digital adalah memperhalus gambar dengan Gaussian blur. Gambar 2.18 menunjukkan kurva umum karakteristik pass filter.



Gambar 2.18 *: Kurva umum karakteristik pass filter*

Keterangan :

$\frac{Vo}{Vi}$ $(\boldsymbol{db})$: amplitudo respon
fc : frekuensi cut-off

Batas frekuensi antara sinyal yang dapat diteruskan dan yang diredam (frekuensi cut-off) yang dapat ditentukan dengan persamaan:

$$fc = \frac{1}{2\pi RC} \quad (2.36)$$

Dimana,
R = Nilai hambatan; C = Nilai kapasitor

Untuk penyederhanaan, frekuensi cutoff didefinisikan sebagai titik pemisah antara passband dan stopband. Frekuensi cut-off sebuah filter sebenarnya adalah frekuensi pada saat penguatan tegangannya turun menjadi 0,707 atau $1/\sqrt{2}$ kali dari penguatan pass band-nya.

Berdasarkan persamaan 2.21 Jika $V_o > V_i$ maka terjadi penguatan dan dB menjadi positif. Sebaliknya, jika $V_o < V_i$ maka terjadi pelemahan (atenuasi) sehingga dB menjadi negatif.

$$db = -20\, log_{10} \frac{Vo}{Vi} \quad (2.37)$$

Ada beberapa tipe filter, yaitu :

1. Butterworth : Menghasilkan kerataan pass-band yang maksimal sehingga sering digunakan sebagai filter anti-aliasing pada aplikasi data konverter dimana dibutuhkan level sinyal yang tepat pada seluruh passband.

2. Chebyshev : Menghasilkan penguatan roll-off yang lebih tinggi diatas f_c. penguatan pada passband tidak monoton, tapi mengandung ripple dari magnitud konstan. Semakin tinggi ripple pada frekuensi passband, semakin tinggi pula roll-off filter. Meningkatnya orde filter berpengaruh dari magnitud ripple pada rolloff filter berkurang. Filter orde genap menghasilkan ripple diatas 0 dB, sementara filter orde ganjil menghasilkan ripple di bawah 0 dB. Filter ini sering digunakan pada bank filter, dimana sinyal frekuensi lebih penting daripada penguatan konstan.

3. Bessel : Filter ini memiliki respon fase yang linear melalui rentang frekuensi yang lebar, menghasilkan grup delay konstan di dalam rentang frekuensi tersebut. Bassel juga bersifat mentransmisikan gelombang kotak. Gain passband pada bassel tidak serata filter Butterworth, transisi dari passband ke stop band tidak setajam pada filter Chebyshev.
4. Eliptik : Filter elektronik dengan ripple yang diratakan pada passband dan stopband. Ripple di setiap band dapat diatur dan t mempunyai transisi penguatan lebih cepat antara pass band dan stop band dengan ripple yang telah diberikan. Jika ripple di stop band mendekati nol, filter ini menjadi filter Chebyshev tipe I. Saat ripple di pass band mendekati nol, filter ini menjadi filter Chebyshev tipe II, jika keduanya mempunyai nilai ripple mendekati nol, menjadi filter Butterworth. faktor ripple menentukan ripple pass band, sedangkan gabungan antara faktor ripple dan faktor selektifitas menentukan ripple stop band.

Halaman ini sengaja dikosongkan

BAB III
DESAIN DAN EKSPERIMEN PERANGKAT KERAS

Pada bab ini akan dibahas mengenai desain fisik Mobile Robot dengan kemudi diferensial (*Differential Drive*) yang didalamnya juga terdapat desain dan data eksperimen rangkaian kemudi motor DC, rangkaian pengendali (*Controller*), rangkaian sensor dan komunikasi nirkabel. Perangkat keras robot akan dibahas per bagian secara terpisah agar dapat diketahui performa masing-masing bagian tersebut.

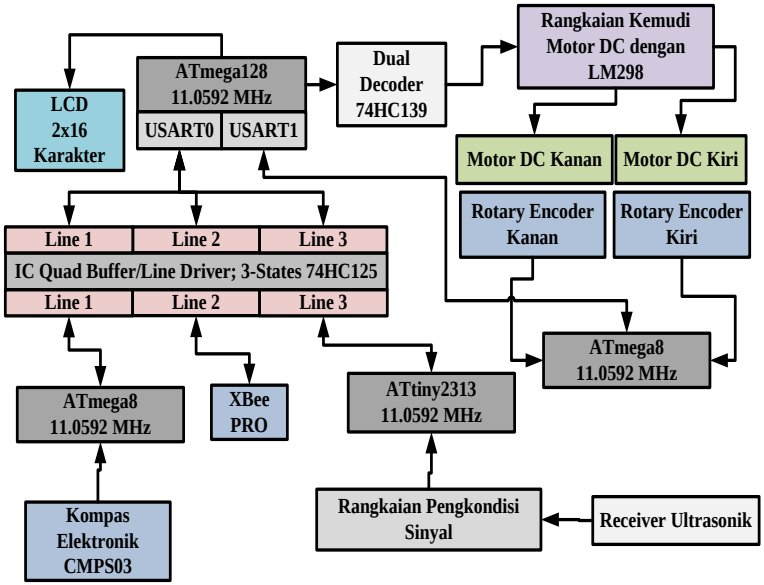
3.1 Tujuan dan Metodologi

Eksperimen dilakukan agar diperoleh data yang akurat mengenai skema kontrol, apakah dapat bekerja dengan baik.

Metode yang dilakukan meliputi rancangan fisik mekanik robot mobile, desain berupa skematik dari rangkaian mikrokontroler, sistem sensor dan komunikasi nirkabel. Data eksperimen diambil per bagian/blok dilengkapi dengan kode program. Selain kode program juga akan dibahas mengenai fitur-fitur yang tersedia dari mikrokontroler.

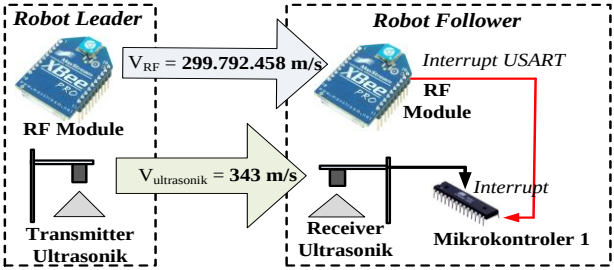
3.2 Desain Perangkat Keras

Pembuatan perangkat keras dilakukan berdasarkan blok diagram pada gambar 3.1. perlu diketahui bahwa pada tugas akhir ini dibuat dua buah mobile robot yang masing masing berfungsi sebagai robot pemimpin dan robot pengikut. Di blok diagram gambar 3.1 perangkat keras terdiri dari perangkat sensor, modul komunikasi nirkabel, sistem kontroler, driver motor, dan aktuator berupa dua buah motor DC yang sudah dilengkapi dengan gearbox. Pembeda robot pengikut dan pemimpin yang paling utama terletak pada adanya sensor halangan. Pada robot pemimpin sensor halangan menggunakan sensor jarak infrared GPD12 sedang robot pengikut sama sekali tidak mempunyai sensor halangan atau dengan kata lain, robot pengikut adalah robot buta yang hanya mengandalkan informasi yang diberikan oleh robot pemimpin.



***Gambar 3.1** : Blok Diagram Perangkat Keras Robot*

Pada kedua robot, perangkat sensor (kompas elektronik dan sensor ultrasonik) bekerja secara terintegrasi dengan komponen RF module untuk mendukung proses penentuan posisi relatif (*localization*) berdasarkan metode Time Of Flight (TOF). Arah hadap robot pemimpin dibaca menggunakan modul CMPS03 kemudian dikirimkan secara nirkabel menggunakan RF Module X-Bee Pro. Selain berisi informasi pembacaan arah hadap robot pemimpin dan peruabahan gerak roda robot, paket data yang dikirimkan juga berfungsi sebagai sinyal trigger kepada robot pengikut untuk segera melakukan proses perhitungan dimana pada saat yang bersamaan robot pemimpin juga menyebarkan sinyal ultrasonik yang natinya akan diterima pada rangkaian penerima pada robot pengikut.

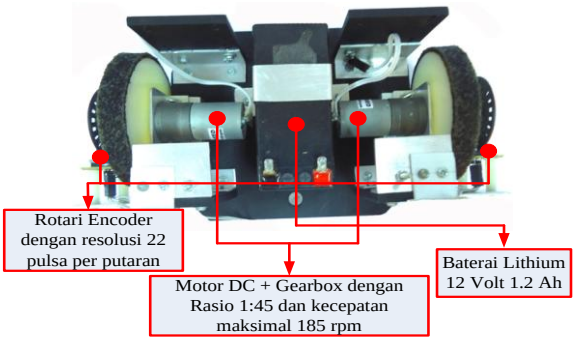


Gambar 3.2 *: Implementasi Konsep Time Of Flight*

Robot pengikut dilengkapi dengan rangkaian detektor atau penerima ultrasonik berupa rangkaian pengkondisi sinyal yang terdiri dari rangkaian penguat lengkap dengan rangkaian regulator catu daya -10 Volt untuk IC TL082, filter,dan komparator yang akan dibahas lebih detail pada sub-bab 3.2.3 mengenai sistem sensor.

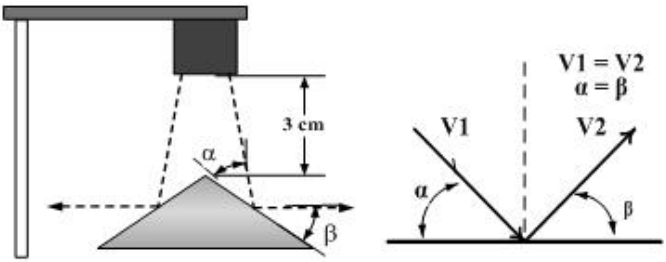
3.2.1 Mekanik Robot

Mekanik robot dibuat menggunakan bahan-bahan yang ringan namun cukup kuat seperti papan acrylic dan aluminium. Komponen gerak robot terdiri dari dua buah motor DC 12V yang sudah dilengkapi gearbox internal dengan rasio 1:45 dan kecepatan maksimal 185 rpm (*rotation per minute*). Mekanik robot dilengkapi dengan baterai *Lithium* yang dapat diisi ulang (*rechargeable*).



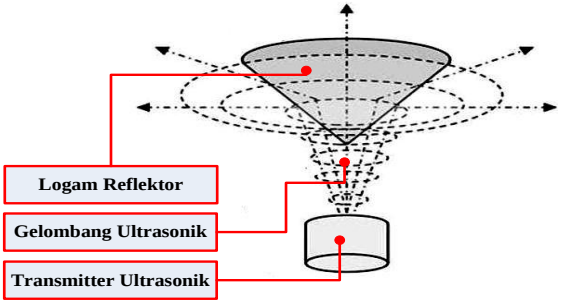
Gambar 3.3 *: Disain mekanik Mobile Robot dengan kemudi differensial*

Pada robot pemimpin dan robot pengikut keduanya dilengkapi dengan transduser ultrasonik. Untuk menyebarkan sinyal ultrasonik ke segala arah diperlukan 12 buah sensor pemancar yang mempunyai sudut elevasi pemancar sebesar 30° demikian juga untuk sisi receiver terdapat 12 sensor penerima sehingga hal ini dinilai kurang praktis dan boros biaya.



Gambar 3.4 *: Hukum pantulan gelombang*

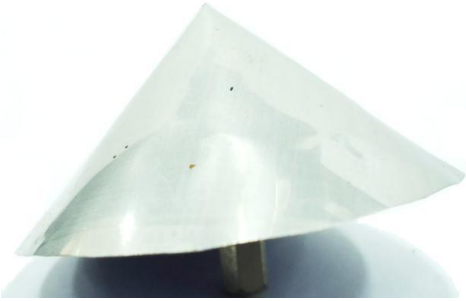
Berdasarkan teori propagasi gelombang bahwa gelombang bunyi dapat dipantulkan oleh permukaan yang keras dan disebut sebagai Gema (echo). Pantulan gelombang bunyi mematuhi Hukum Pantulan yaitu sudut datang sama dengan sudut pantulan seperti yang dijelaskan pada gambar 3.4. Dari fenomena tersebut, kami akan menggunakan logam tipis berbentuk kerucut yang berguna untuk memantulkan gelombang ultrasonik ke segala arah.



Gambar 3.5 *: Ilustrasi penyebaran sinyal ultrasonic ke segala arah menggunakan reflector berdimensi kerucut*

Reflektor terbuat dari bahan logam seperti aluminium atau besi , solid maupun lempengan tipis. Pembuatan reflektor dapat dilakukan dengan memanfaatkan kaleng softdrink bekas yang terbuat dari alumunium. Selain mudah dan murah dalam pembuatan, setelah dilakukan pengujian ternyata kerucut penyebar dari kaleng softdrink bekas tersebut mampu memantulkan gelombang ultrasonik dengan baik.

Gelombang bunyi dapat merambat dengan lebih cepat pada medium benda padat atau cair daripada udara yaitu lima belas kali lebih cepat dalam medium logam dan empat kali lebih cepat pada medium air. Kondisi yang akan memberi pengaruh kuat dalam sistem ini adalah suhu medium udara lingkungan sekitar. Dalam keadaan udara yang kering pada suhu 0^o C cepat rambat gelombang bunyi adalah lebih kurang 331 ms^{-1} sedangkan pada suhu kamar cepat rambat gelombang bunyi adalah lebih kurang 343 ms^{-1}. Semakin dingin suhu udara semakin lambat laju rambat gelombang. Sistem ini didesain bekerja pada suhu ruangan sehingga asumsi kecepatan rambat yang digunakan adalah 343 ms^{-1}.



Gambar 3.6 *: reflector berbentuk kerucut dari logam*

Rodney heill[7] menjelaskan bahwa dalam membuat *cone reflector* terdapat beberapa aturan, aturan tersebut seperti pada persamaan 3.1

$$r = \frac{z^2}{2000} - 500 \qquad (3.1)$$

$$1000 \leq z \leq 1871 \; dan \;\; 0 \leq r \leq 1250$$

Dimana satuan diukur dalam inchi/1000

r adalah jari-jari reflektor sedangkan z adalah jarak antara transduser ultrasonik dengan pucuk *cone reflector*.

Gambar 3.7 *: Reflektor Ultrasonik*

Gambar 3.8 *: Receiver ultrasonic dengan reflektor*

Posisi pemasangan transmitter dengan reflektor yang ideal dapat dilakukan berdasarkan pengujian apakah sinyal yang dipancarkan dapat diterima dengan baik oleh receiver. Pengujian dilakukan dengan

mengatur ketinggian antara transmitter dengan reflektor. Dari pengujian yang telah dilakukan didapatkan ketinggian yang ideal adalah kurang lebih 3 cm dengan posisi titik tengah membran ultrasonik harus lurus tepat dengan titik tengah kerucut penyebar. Menurut lembar data sensor ultrasonik, membran sensor ini menghasilkan sinyal ultrasonik dengan frekwensi 40KHz dan besar sudut elevasi 40°.

Reflektor yang dibuat terbukti ampuh untuk memantulkan sinyal akustik tersebut namun untuk jarak yang jauh sinyal tersebut mengalami pelemahan. Untuk mengantisipasi hal tersebut maka rangkaian penguatan dua tahap diharapkan dapat menguatkan kembali sinyal tersebut. Sensor ini juga dilengkapi dengan rangkaian komparator yang berguna untuk mengatur batasan sinyal input dianggap sebagai sinyal asli atau dengan kata lain sebagai rangkaian penghalang cacat/noise. Terdapat dua resistor multitone yang masing-masing digunakan untuk mengatur batas ambang positif dan negatifnya.

3.2.2 Rangkaian Kemudi (*Driver*) Motor DC

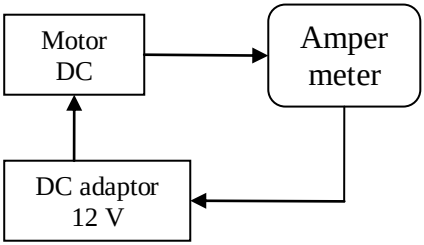
Dari pengukuran yang telah dilakukan terhadapap motor DC yang digunakan dalam system robot. Didapatkan data kurang lebih yaitu beban awal sebelum stadystate sebesar 0.3 A sedang pada stadystate sebesar 0.05A.

Table 3.1 data motorDC

parameter	nilai
Beban awal	0.3A
Beban normal	0.05A
Tegangan kerja	12 V
rpm	185

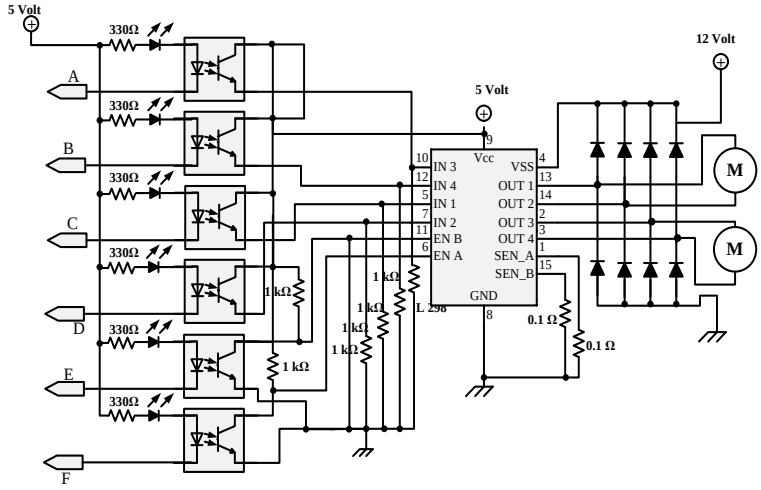
Dengan berbekal data pada tabel 3.1 maka rangkaian driver yang digunakan harus mampu memberikan arus sebesar 0.3Ampere. namun jika batas arus rangkaian driver dipaksakan mendekati batas arus maksimal motor akan menghawatirkan untuk kelangsungan rangkaian itu sendiri, oleh karena itu dalam desain ini, batas tersebut ditingkatkan sampai pada level 2A. dan berdasarkan referensi dari beberapa buku

Proyek Akhir (Ahmad Nasrul Aziz, 2009) maka dipilihlah driver dengan menggunakan komponen IC L298 buatan STMicroelectronics.



***Gambar 3.9** : konfigurasi pengukuran Arus motorDC*

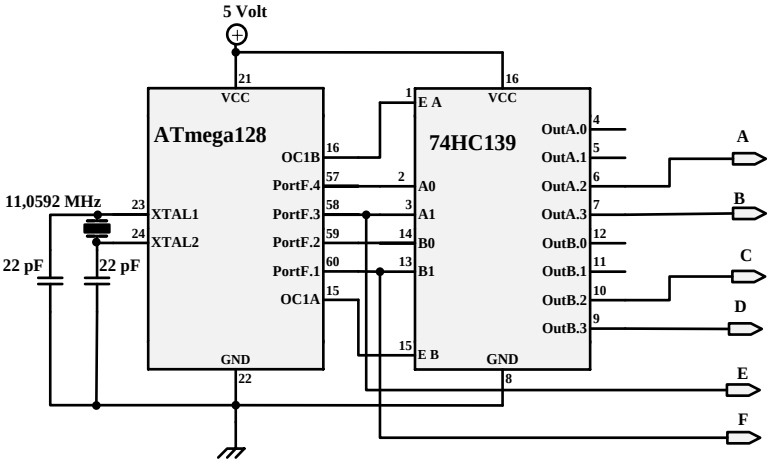
L298 sangat mampu menjawab tantangan terhadap kebutuhan arus yang cukup besar, bahkan dengan mempararel antar aoutputnya akan didapatkan arus sekitar 4A.



Gambar 3.10 *: Rangkaian Kemudi Motor DC*

Rangkaian kemudi motor DC berguna untuk memberikan penguatan arus ke motor melalui switching, yaitu dengan menggunakan *Pulse Width Modulation* (PWM). Hal ini dilakukan mengingat prinsip kerja motor DC adalah berdasarkan jumlah arus yang diberikan, semakin besar arus yang melalui kumparan motor, semakin cepat putaran motor dan demikian juga sebaliknya. Tegangan masukan diatur berdasarkan *Dutty Cycle* PWM yaitu perbandingan jumlah *Timing On* dibandingkan jumlah *Timing Off* dalam satu periode. Tegangan keluaran yang dihasilkan PWM adalah tegangan rata-ratanya.

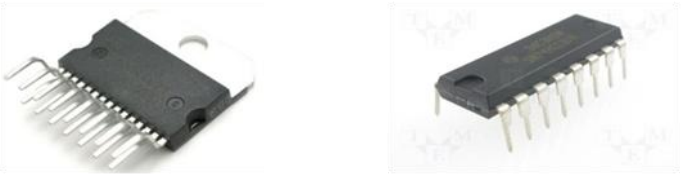
Rangkaian kemudi motor DC dilengkapi dengan decoder 74LS139 yang berguna untuk merubah model konfigurasi pengaktifan rangkaian kemudi dan arah putar motor sehingga masukan rangkaian kemudi adalah 3 jalur yaitu: PWM, Arah Putar dan Rem.



Gambar 3.11 *: Skema Rangkaian Dekoder untuk rangkaian Kemudi Motor*

L298 adalah jenis IC driver motor yang dapat mengendalikan arah putaran dan kecepatan motor DC ataupun Motor stepper. Mampu mengeluarkan output tegangan untuk Motor dc dan motor stepper sebesar 50 volt. IC l298 terdiri dari transistor-transistor logik (TTL) dengan gerbang nand yang memudahkan dalam menentukkan arah putaran suatu motor dc dan motor stepper. Dapat mengendalikan 2 untuk motor dc namun pada hanya dapat mengendalikan 1 motor stepper.

Penggunaannya paling sering untuk robot *line follower*. Bentuknya yang kecil memungkinkan dapat meminimalkan pembuatan robot line follower.



(a) (b)

Gambar 3.12 *: (a) IC Driver Motor Dual Full Bridge L298 (b) IC Dual Dekoder Dua Input Empat Output 74HC139*

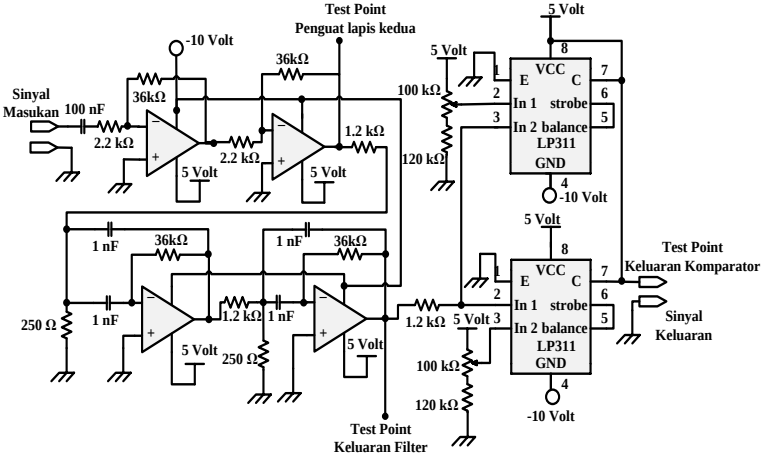
Gambar 3.13 *: Rangkaian Kemudi Motor DC*

3.2.3 Sistem Sensor

Sinyal ultrasonik yang dipancarkan dari pemancar akan mengalami pemantulan sebanyak dua kali sebelum akhirnya ditangkap oleh penerima. Sehingga sinyal tersebut rentan terjadi pelemahan. Sinyal ultrasonik yang tertangkap penerima harus difilter dan dikuatkan terlebih dahulu agar dapat diolah oleh kontroller. Proses filter dan penguatan akan dibahas pada sub-bab selanjutnya. Dalam pengujian

yang kami lakukan sinyal ultrasonik yang dipancarkan masih dapat tertangkap dengan baik oleh receiver dalam range jarak sekitar 11 meter.

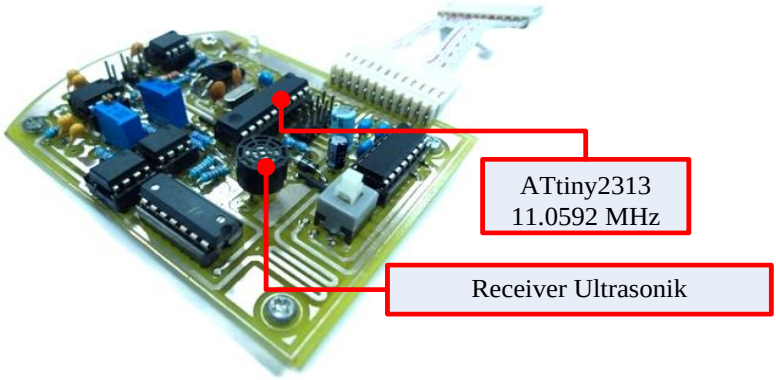
Cone reflector ini terbukti ampuh untuk memantulkan sinyal akustik tersebut namun untuk jarak yang jauh sinyal tersebut mengalami pelemahan. Untuk mengantisipasi melemahnya sinyal ultrasonik yang sampai pada sisi penerima maka rangkaian penguatan dua tahap diharapkan dapat membuat data input yang kecil dapat diolah atau diproses oleh rangkaian kontroller. Sensor ini juga dilengkapi dengan rangkaian komparator yang berguna untuk mengatur sensitivitas sensor terhadap error. Terdapat dua resistor multitone yang masing-masing digunakan untuk mensetting batas threshold positif dan negatifnya.



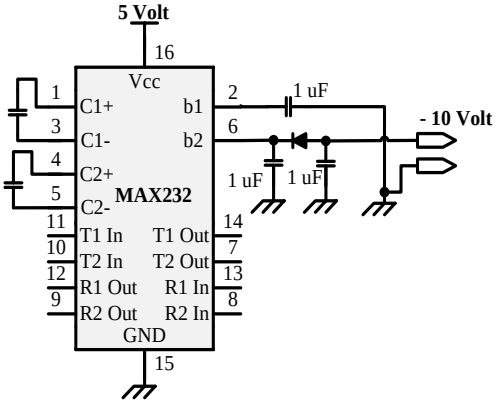
Gambar 3.14 *: Skema Rangkaian Receiver Ultrasonik*

Pada rangkaian transmitter sinyal burst dihasilkan oleh 2 buah pin mikrokontroler dengan frekwensi 40KHz. Sinyal tersebut perlu dikuatkan menggunakan IC konverter RS232 (MAX232). Seperti yang diketahui bahwa level tegangan yang ditentukan untuk RS232 tidak ada hubungannya dengan level tegangan TTL. Dalam standard RS232, tegangan antara +3 sampai +15 Volt pada input Line Receiver dianggap sebagai level tegangan '0', dan tegangan antara –3 sampai –15 Volt dianggap sebagai level tegangan '1' sedangkan tegangan output Line Driver berkisar antara +5 sampai +15 Volt untuk menyatakan level

tegangan ‘0’, dan berkisar antara –5 sampai –15 Volt untuk menyatakan level tegangan ‘1’. Demikian pula pada rangkaian receiver diperlukan IC converter RS232 untuk menyediakan sumber tegangan -10Volt yang digunakan pada IC OpAmp TL08

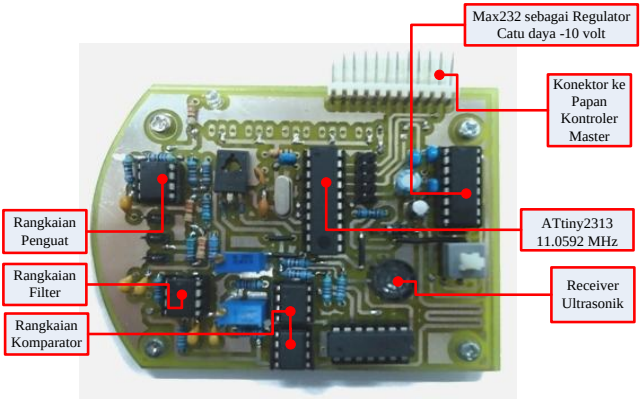


Gambar 3.15 *: Rangkaian Receiver Ultrasonik*

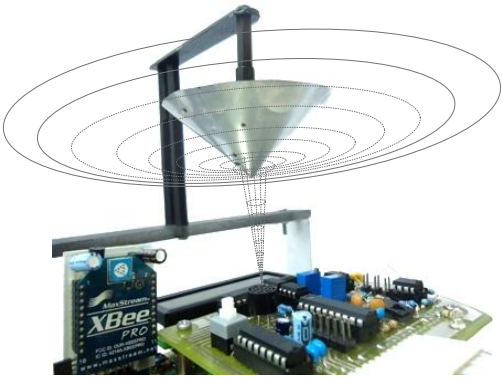


Gambar 3.16 *: IC Konverter TTL – 232 digunakan untuk menghasilkan tegangan level -10V*

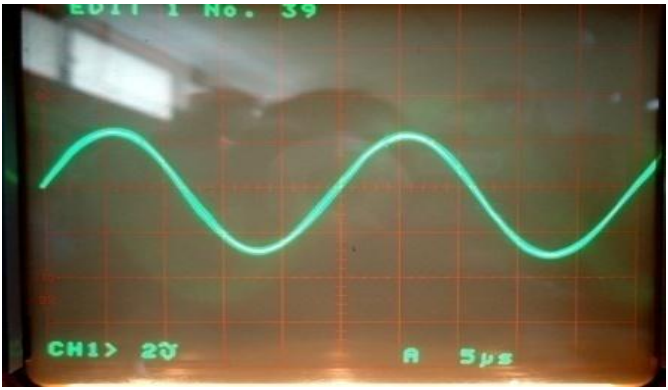
Sinyal ultrasonik yang diterima receiver perlu dikuatkan terlebih dahulu. Secara default penguatan cukup dilakukan dalam 1 tahap dengan besar penguatan: 32x. Namun dalam kasus ini sinyal ultrasonik yang dipancarkan mengalami dua kali pemantulan sehingga dikhawatirkan sinyal menjadi lemah maka penguatan sinyal akan dilakukan dalam 2 tahap dengan besar penguatan 964x



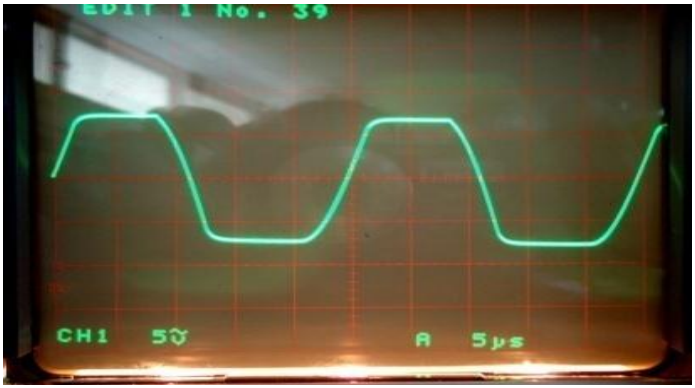
Gambar 3.17 *: Rangkaian Pengkondisi sinyal untuk Penerima Ultrasonik*



Gambar 3.18 *: Ilustrasi Penyebaran gelombang Ultrasonik menggunakan reflektor*

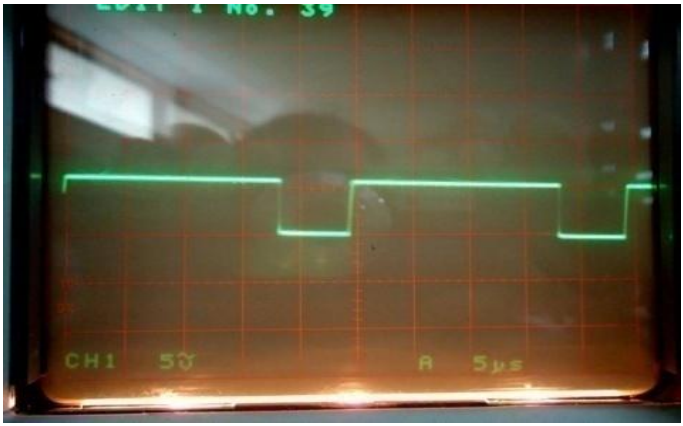


Gambar 3.19 *: Sinyal Keluaran Pada Rangkaian Penguat lapis kedua*



Gambar 3.20 *: sinyal keluaran Pada Rangkaian Band Pass Filter*

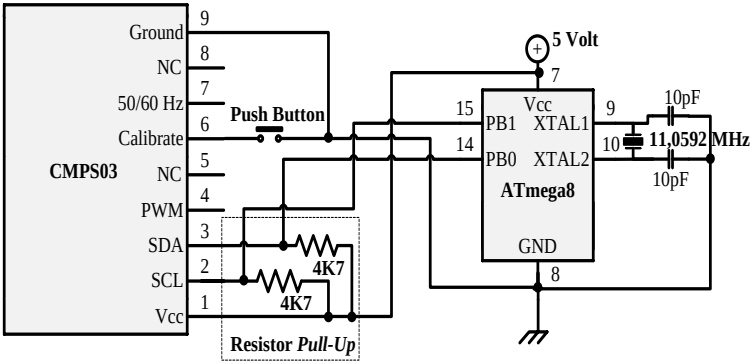
Pembacaan data sensor kompas elektronik CMPS03 menggunakan fitur I2C dengan menambahkan resistor *pull-up* pada masing-masing pin SDA dan SCL. Dengan melakukan pengalamatan pada modul tersebut data akan disajikan langsung dalam format hexadesimal. Nilai 0 merepresentasikan sudut 0º dan nilai 255 merepresentasikan sudut 360º.



Gambar 3.21 *: Sinyal Keluaran Pada Rangkaian komparator*

Berikut langkah-langkah komunikasi modul cmps03 dengan mikrokontroler menggunakan protokol I2C dalam kode program:

```
unsigned int kompas_read()
{
unsigned int data;
i2c_start();
i2c_write(0xC0);
i2c_write(0x01);
i2c_start();
i2c_write(0xC1);
data = i2c_read(0);
 i2c_stop();
}
```



Gambar 3.22 *: Antarmuka CMPS03 dengan mikrokontroller*

Agar modul cmps03 dapat bekerja dengan baik dan memberikan data yang akurat maka perlu dilakukan pengkalibrasian modul. Pada pin 6 (calibrate) terdapat pull-up resistor yang terpasang secara on-board. Yang perlu dilakukan dalam pengkalibrasian adalah memberikan logika 0 pada pin calibrate saat modul diarahkan pada empat arah mata angin (Utara, Selatan, Barat, dan Timur). Seringkali pengguna modul ini mengabaikan prosedur pengkalibrasian padahal simpangan error yang dihasilkan cukup mengganggu sistem secara keseluruhan. Pengkalibrasian perlu dilakukan setiap akan digunakan.

Tabel 3.1 : Data pembacaan kompas cmps03 sebelum kalibrasi

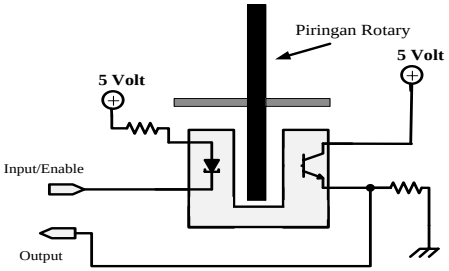
Data Kompas (integer)	Arah hadap
155	Utara
186	Barat
249	Timur
217	Selatan

Untuk mendapat linieritas data yang lebih baik dapat dilakukan kalibrasi pada kompas. Setelah dilakukan kalibrasi, diperoleh data pada tabel 3.2.

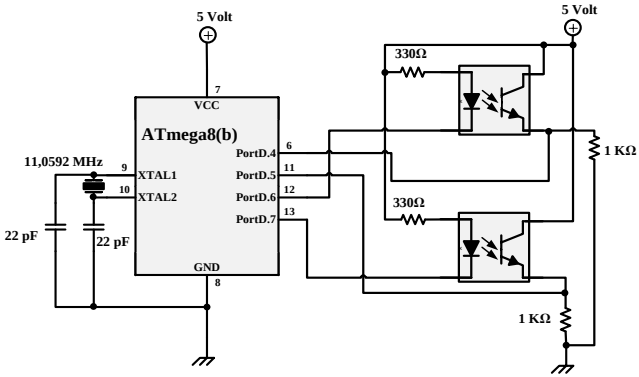
Tabel 3.2 : Data pembacaan kompas cmps03 setelah kalibrasi

Data Kompas (integer)	Arah hadap
0	Utara
63	Barat
191	Timur
127	Selatan

Untuk mengetahui kecepatan aktual motor DC diperlukan sensor rotary encoder pada masing-masing roda. Skema rangkaian rotary encoder dijelaskan pada gambar 3.23



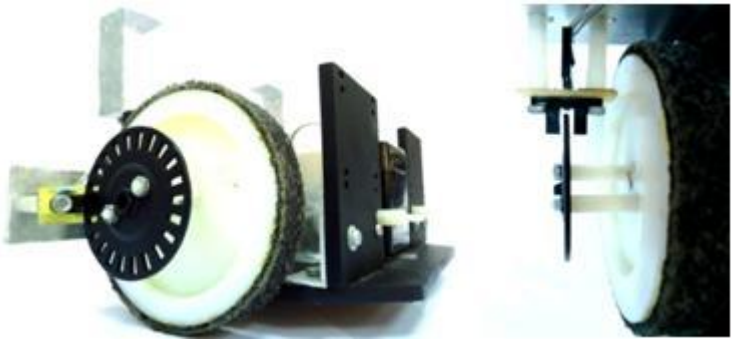
Gambar 3.23 *: Skema Rangkaian Rotary Encoder*



Gambar 3.24 *: Antarmuka rotary encoder dengan mikrokontroller*

Port output pada skema rangkaian rotary encoder dihubungkan dengan pin 16 (PORTD.2 / INT0) dan pin 17 (PORTD.3 / INT1) sebagai sumber sinyal interrupt eksternal. Cahaya photodiode yang mengenai rangkaian phototransistor akan membangkitkan sinyal intterupt. Pada mikrokontroler terdapat rutine interrupt yang melakukan *counter* setiap sinyal intterupt yang diterima kemudian menyimpan data hasil *counter* tersebut dalam sebuah variabel. Data tersebut dapat digunakan untuk mengukur jarak yang telah ditempuh selama robot berjalan dengan formula sebagai berikut

$$Jarak = \frac{Jumlah\ counter}{Jumlah\ pulsa\ satu\ putaran\ penuh} \times Keliling\ roda \qquad (3.2)$$



Gambar 3.25 *: Rotary Encoder*

Jika jarak tersebut diukur dalam satu satuan waktu maka didapatkan data kecepatan roda. Fitur mikrokontroler yang mendukung keperluan tersebut adalah dengan interrupt timer0. Berikut ini adalah langkah-langkah untuk mengatur konfigurasi register pada timer0 dengan overflow presisi 1 detik.

Membagi frekwensi clock dengan prescaler

$\frac{system\ clock}{1024} = 10800$ Hz

Periode → $\frac{1}{10800Hz} = 92.59$ mikrodetik

$92.59\ us\ x\ 256 = 23\ ms$ atau $92.59\ us\ x\ 216 = 20$ milidetik

$256 - 216 = 40$ maka register TCNT = 40

Sinyal interrupt akibat overflow akan terjadi setiap 20 milidetik sekali. Untuk membangkitkan sinyal interrupt tiap 1 detik dapat dimanipulasi dengan variabel tambahan

Konter overflow → 50 x 20 milidetik = 1 detik

Berikut ini adalah contoh kode program untuk mengukur kecepatan roda menggunakan rotary encoder.

```
#include <stdio.h>
unsigned char kanan= 0,kiri=0, keliling_roda=27;
// External Interrupt 0 service routine
interrupt [EXT_INT0] void ext_int0_isr(void)
{   kanan++; }
// External Interrupt 1 service routine
interrupt [EXT_INT1] void ext_int1_isr(void)
{ kiri++;}
interrupt [TIM0_OVF] void timer0_ovf_isr(void)
{
   TCNT0 = 40;
  ++konter_overflow;
   if (konter_overflow == 200)
            {v_kanan =(kanan/22)*keliling_roda;
             v_kiri=(kiri/22)*keliling_roda;
            }
}
void main(void)
{
PORTD=0xF0;
DDRD=0xF0;
MCUCR=0x00;
MCUCSR=0x00;
TCCR0=0x05;
TCNT0=0x00;
OCR0=0x00;
}
```

3.2.4 Perangkat Komunikasi

Kehandalan sistem Perilaku Mengikuti Pemimpin sangat bergantung pada kesuksesan proses estimasi jarak relatif antara robot

pemimpin dan robot pengikut yang menggunakan dasar konsep Time Of Flight (TOF). Elemen terpenting dari konsep tersebut adalah adanya sistem komunikasi antara kedua robot. Antara kedua perangkat keras (robot) tidak diizinkan menggunakan kabel sebagai media komunikasi dikarenakan kedua perangkat keras tersebut selalu mobile (bergerak) sehingga satu-satunya solusi adalah menggunakan media komunikasi tanpa kabel (wireless).

Media komunikasi nirkabel yang akan kami gunakan adalah Xbee Pro RF Module yang memenuhi standard IEEE dan sangat mendukung untuk keperluan sistem yang murah dan sistem jaringan sensor berdaya rendah. Meskipun murah dan berdaya rendah namun modul ini terbukti cukup tangguh dalam pengiriman data antar perangkat keras.

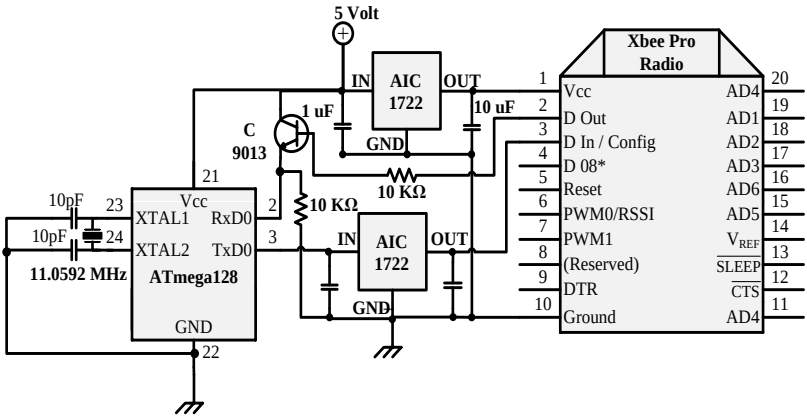
Gambar 3.26 *: Modul Komunikasi Frekuensi Radio*

Modul ini bekerja dengan sumber tegangan antara 2,8 – 3,4 Volt. Konsumsi arus saat transmit adalah 45 mA dan saat receive data adalah 50 mA. Robot pengikut dan robot pemimpin direncanakan saling berkomunikasi dalam *range* yang relatif sempit yakni ±30 cm sedangkan kemampuan modul ini dalam melakukan pengiriman data adalah lebih dari 30 m dalam ruangan dan lebih dari 100 m apabila dilakukan di luar ruangan. Untuk keperluan antarmuka (interface), antarmuka perangkat ini ke mikrokontroler juga tidak terlalu rumit. Desain minimum sistem dari modul ini dengan menggunakan 4 pin.

1. Pin 1(Vcc)

2. Pin 2 (Data_Out) dihubungkan dengan pin 14 (Receive) pada mikrokontroler.
3. Pin 3 (Data_In) dihubungkan dengan pin 15 (Transmit) pada mikrokontroler.
4. Pin 10 (Ground)

Dikarenakan modul ini tidak bekerja dalam level TTL maka diperlukan sebuah konverter untuk merubah level TTL ke dalam level yang diizinkan (3,3 Volt). Untuk merubah level tegangan 5 volt ke level tegangan ke 3.3 volt dapat dilakuka dengan menggunakan IC regulator AIC 1722.



Gambar 3.27 *: Rangkaian Antarmuka Xbee-Pro dengan mikrokontroller*

AIC 1722 merupakan regulator tegangan 3 pin yang memiliki akurasi tegangan output sebesar 2%. Untuk menjaga stabilitas perlu ditambahkan kapasitor sebesar 1µF pada sisi input dan 10µF pada sisi output. AIC 1722 sering digunakan sebagai regulator tegangan untuk CD-ROM Drivers, LAN Cards, Mikroprosesor dan sistem komunikasi wireless.

Untuk menguji modul X-Bee Pro dapat menggunakan aplikasi X-CTU yang merupakan software dari MaxStream untuk melakukan interface dan konfigurasi modul X-Bee Pro. Aplikasi ini di desain untuk melakukan pengaturan konfigurasi modul X-Bee Pro dan parameter

parameter lain dengan AT Commands. Berikut ini adalah langkah-langkah pengujian modul menggunakan AT Commands:

- Ketik pada panel terminal untuk memulai command mode: + + + . Apabila terdapat respon OK maka modul siap melakukan konfigurasi.
- Ketik ATDL<Enter> → Membaca alamat tujuan.
- Ketik ATDL 1A0D <Enter> → Merubah alamat tujuan.
- Ketik ATWR <Enter> →Menuliskan memori non-volatile
- Ketik ATCN <Enter> → Keluar dari command mode.

Komunikasi dan pengiriman data dapat dilakukan seperti halnya melakukan komunikasi serial biasa. Untuk catatan bahwa, komunikasi dengan modul ini membutuhkan konsumsi daya yang cukup besar sehingga untuk menghemat energi komunikasi dengan RF module perlu diatur pewaktuannya.

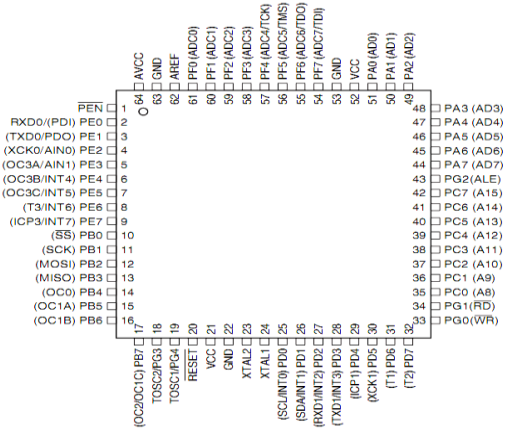
3.2.5 Sistem Kontroler

Perancangan sistem kontroler selalu disesuaikan perintah yang dibebankan. Semakin berat dan rumit perintah yang dibebankan maka semakin tinggi spesifikasi dari kontroler tersebut. Bahkan apabila tugas dari sistem terlalu berat maka diperlukan beberapa kontroler tambahan untuk menjalankan sebagian tugas sehingga pekerjaan dapat diselesaikan dengan lebih mudah dan cepat. Mikrokontroler yang berbasis arsitektur RISC (*Reduced Instruction Set Computer*) didukung dengan sumber clock berfrekuensi tinggi juga mutlak diperhitungkan berkaitan dengan isu real-time sistem dimana kecepatan eksekusi juga sangat diperlukan.

Dari seluruh mikrokontroler produksi Atmel, ATmega128 dinilai paling cocok untuk memenuhi kebutuhan-kebutuhan yang telah dijelaskan. Berikut ini beberapa fitur-fitur yang dimiliki mikrokontroler ATmega128,

- 133 Instruksi dengan waktu Eksekusi Satu Clock Cycle per instruksi
- 32 x 8 Register *General Purpose* dan Register Kontrol Peripheral
- Mampu mengeksekusi lebih dari 16 MIPS (*Million Instruction Per Second*) pada frekuensi crystal 16 MHz
- 128K Bytes of In-System Self-programmable Flash program memory

- 4K Bytes EEPROM
- 4K Bytes Internal SRAM
- Lebih dari 64K Bytes Memori eksternal tambahan
- Dua Timer/Counter 8-bit bit dengan Prescaler independen
- Dua Timer/Counter 16-bit dengan Prescaler independen– Real Time Counter with Separate Oscillator
- Dua channel PWM 8-bit
- 6 Channel PWM dengan Resolusi terprogram dari 2 sampai 16 Bit
- 8 Channel, 10-bit ADC
- Antarmuka serial *Two-wire* berorientasi byte
- Dual Programmable Serial USARTs
- Master/Slave SPI Serial Interface
- Watchdog Timer terprogram
- Komparator Analog
- Internal Calibrated RC Oscillator
- External and Internal Interrupt Sources
- Frekwensi Clock yang dapat dipilih secara terpogram
- Kompatibel dengan ATmega103.
- 53 Jalur Input Output yang terprogram

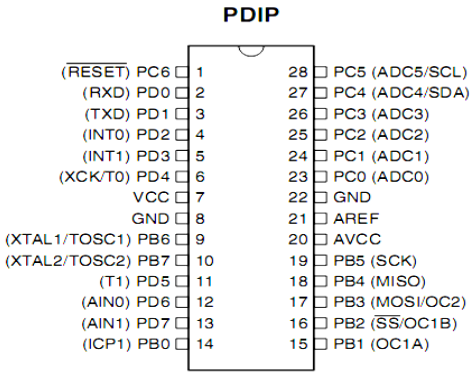


Gambar 3.28 *: Mikrokontroller ATMega128*

Tabel 3.6 menyajikan alokasi pin pada mikrokontroler Atmega128

Tabel 3.6 : Alokasi Pin ATmega128

Pin	Keterangan	Koneksi
22,53	Ground	Ground catu daya
21,52	VCC	Catu daya +5 Volt
23	Xtal1	Crystal 11.0592 MHz
24	Xtall2	Crystal 11.0592 MHz
35-38 4-5	PortC.0 – 3, PortE.2-3	74xx125 controller
28	Tx usart 1	Atmega 8 untuk rotary
27	Rx usart 1	Atmega 8 untuk rotary
2	Tx usart 0	Xb-proo, sensor
3	Rx usart 0	Xb-proo, sensor
6-9	PortE.4-7	Switch setting
44-51	PORTA	LCD
15	OCR1A	Motor DC parameter
16	OCR1B	Motor DC parameter



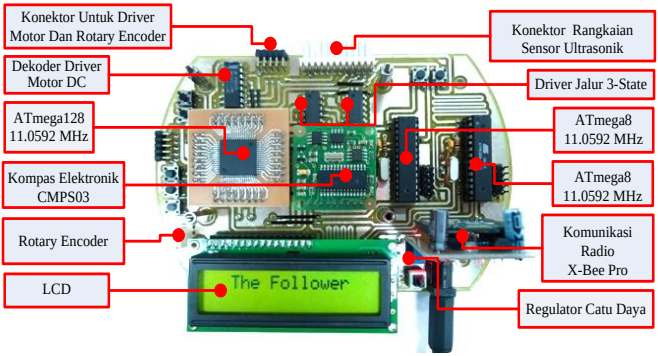
Gambar 3.28 *: Mikrokontroler ATmega8*

Mikrokontroler ATmega128 akan difungsikan sebagai kontroler *master* untuk mengerjakan perintah yang berkaitan dengan perhitungan matematis yang rumit, sedangkan perintah ringan yang berkaitan

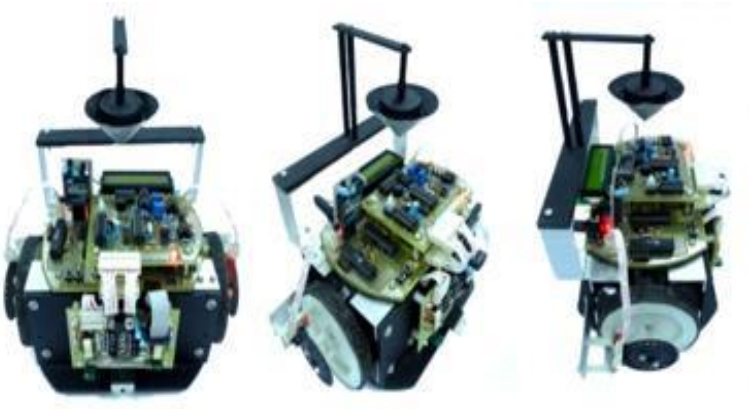
dengan pembacaan sensor dan pembacaan kecepatan motor dibebankan pada mikrokontroler ATmega8 sebagai kontroler *slave*.
Tabel 3.7 menyajikan alokasi pin pada mikrokontroler Atmega8.

Tabel 3.7 : Alokasi Pin Atmega8

Pin	Keterangan	Koneksi
23	ADC0	Sensor GPD-12
24	ADC1	Sensor GPD-12
25	ADC2	Sensor GPD-12
26	ADC3	Sensor GPD-12
27	ADC4	Sensor GPD-12
11	T1	Masukan dari rotary
6	T0	Masukan dari rotary
10	VCC	+5 Volt
8,22	Ground	Ground
9	XTAL2	Crystal 11.0592 MHz
10	XTAL1	Crystal 11.0592 MHz
12	Port D.6	Rotary Encoder Kiri aktif
13	Port D.7	Rotary Encoder Kanan aktif
14	PortB.0	Sensor Kompas (SCL)
15	PortB.1	Sensor Kompas (SDA)



Gambar 3.29 *: Rangkaian controller Mobile Robot*



Gambar 3.30 *: Fisik Robot Pengikut*

3.3 Kesimpulan

1. Reflektor gelombang ultrasonik yang dibuat mampu menyebarkan gelombang ultrasonik dengan baik demikian pula reflektor pada sisi penerima (robot pengikut) mampu memantulkan gelombang ultrasonik mengarah ke membran transduser.
2. Rangkaian pengkondisi sinyal yang telah dibuat mampu menguatkan sinyal ultrasonik yang mengalami pelemahan sehingga rangkaian kontroler mampu mendeteksi sinyal ultrasonik. Dari hasil pengujian didapatkan bahwa sinyal masih dapat terdeteksi dengan baik dalam radius 7 meter. Permasalahan yang sering muncul adalah pada saat melakukan *tuning* resistor multitone untuk rangkaian penguat dan filter karena sangat sensitif mengalami perubahan sehingga setelah mendapatkan hasil tuning yang terbaik dianjurkan untuk menjaga agar nilai multitone tidak berubah sama sekali.
3. Pada rangkaian komparator juga diperlukan tuning batas sinyal keluaran dari filter untuk menghalangi masuknya noise ke sistem kontroller. Hasil tuning yang baik mampu mendeteksi sinyal ultrasonik tanpa menghiraukan sinyal gangguan.

BAB IV
PERENCANAAN DAN EKSPERIMEN PERGERAKAN ROBOT BERDASARKAN ALGORITMA PERILAKU MENGIKUTI PEMIMPIN

Pada bagian ini akan dibahas begaimana merencanakan rute pergerakan robot yang berupa rute aman dan tercepat pada model yang mengandung deskripsi lingkungan atau tempat robot bekerja dengan menggunakan algoritma yang sederhana. Perencanaan rute ini berkaiatn dengan kinematika robot dan penghindaran halangan (*Obstacle Avoidance*) sebagai dasar pengambilan data eksperimen yang nantinya akan dilakukan analisa terhadap data-data tersebut.

4.1 Tujuan dan Metodologi

Perhintungan posisi relatif antara robot pemimpin dengan robot pengikut dilakukan untuk mendapatkan trajektori referensi berupa lintasan yang akan dilalui oleh robot, trajektori referensi ini selanjutnya akan dipakai sebagai masukan untuk pergerakan robot.

Metodologi yang dilakukan meliputi *tuning* terhadap setiap parameter atau dengan menggunakan metode heuristik, terutama parameter kontrol yang diajukan. Ini dilakukan untuk menghindari perhitungan matematis yang rumit dan cenderung ideal, padahal kondisi yang sesungguhnya menghendaki proses yang cepat walaupun tidak ideal.

4.2 Mengukur Jarak Robot Pemimpin

Untuk mengukur jarak antar robot pemimpin dengan robot pengikut Dilakukan dengan menggunakan gabungan sinyal ultrasonik dan gelombang radio (*rf*) yang sudah ditumpangi dengan data perubahan gerak roda pada robot pemimpin. Pengerjaan pengukuran jarak dilakukan dalam beberapa langkah sebagai berikut:

- Perancangan dan penentuan aturan komunikasi
- Perancangan dan pembuatan program perhitungan jarak.
- Implementasi program kedalam system pengukuran jarak.

4.2.1 Perancangan dan penentuan aturan komunikasi.

Dalam perancangan untuk komunikasi antar robot, hal pertama yang harus dikerjakan adalah menentukan perangkat yang digunakan, spesifikasi kerja dan aturan yang dipakai, entah itu berupa penambahan header, delimeter dan lain lain.

Pada proyek akhir ini digunakan xb-proo produk buatan max-boot dengan spesifikasi frekuensi kerja pada area 2.4GHz[8]. Dan untuk spesifikasi kerja xb-proo yang digunakan di set pada kecepatan transfer data sebesar 9600bps, dengan penambahan spesifikasi yaitu xb-proo diset pada channel E dengan id=100. hal ini bertujuan untuk menghindari tabrakan data antar perangkat karena banyaknya mahasiswa yang menggunakan xbpro sebagai perangkat untuk keperluan proyek akhir masing-masing mahasiswa.

Sedang untuk format data kiriman, rencana awal adalah dengan menggunakan data sederhana berupa kiriman data angka 0-255, hal ini bertujuan untuk kecepatan transfer data karena total data kirim per kiriman adalah sebesar 11Byte, namun terjadi kekurangan yaitu pada sisi PC dimana visual basic 6.0 tidak dapat menerima data karakter diluar karakter ascii.

Tabel 4.1 *format data terkirim ke PC versi lama*

255	5	u	H xpos	L xpos	U	H Ypos	L ypos	u	α	254

Sehingga untuk menyiasatainya maka data terkirim bukan berupa data angka 0-255 namun data karakter yang mewakili angka-angka yang akan terkirim.

Tabel 4.2 *format data terkirim ke PC versi baru*

'@'	'5'	4 character yang mewakili posisi x	4 character yang mewakili posisi y	3character yang mewakili sudut	'#'

Jika tiap karakter terdiri dari 8 byte maka total data kirim pada cara kedua yaitu sebesar 14 Byte. Pada dasarnya tidak terdapat perbedaan antara data kirim dari robot pemimpin dan robot pengikut yang

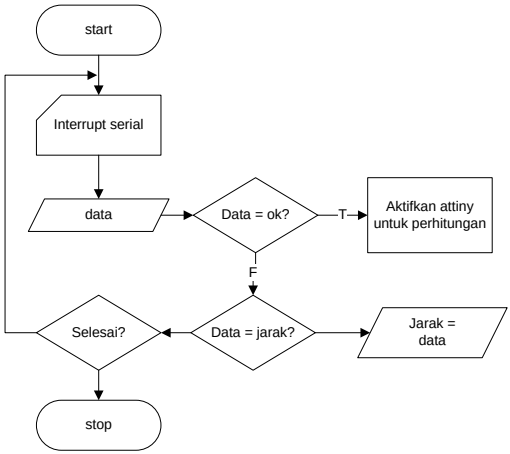
membedakan hanya pada initial id, ditabel 4.2 ditunjukkan dengan karakter ‘5’.

4.2.2 Perancangan dan pembuatan program penghitung jarak.

secara garis besar langkah penghitungan jarak adalah sebagai berikut:

- Tunggu sinyal picu dari robot pemimpin.
- Setelah sinyal picu dating, maka lakukan aktifkan *counter/timer*.
- Tunggu sampai sinyal ultrasonik tiba.
- Begitu sinyal ultrasonik tiba, maka hentikan *counter/timer*. Dan kemudian kalikan dengan faktor pengali, maka akan didapatkan data jarak.

Gambar 4.1 adalah diagram alir yang menjelaskan proses perhitungan.

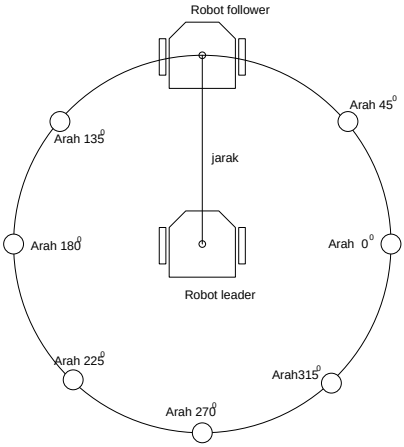


Gambar 4.1 *diagram alir perhitungan jarak*

Dapat dilihat dari diagram alir pada gambar 4.1 semua proses perhitungan dilakukan dengan memanfaatkan fungsi interrupt hal ini bertujuan agar system tidak terganggu oleh proses waktu tunggu.

4.2.3 Implementasi Program Kedalam System Pengukuran Jarak

Setelah merancang dan mendesain perangkat keras terutama pada system sensor ultrasonik yang dijelaskan pada BAB III, maka langkah selanjutnya adalah menggabungkan perangkat keras yang sudah dibuat dengan program yang sudah dirancang. Metode pengukuran jarak yang dilakukan adalah dengan mengatur dan merubah jarak robot pada 8 mata arah angin. Memang dalam pengujian jarak yang kami lakukan belum sampai pada pengukuran jarak ketika robot bergerak, baru sebatas pada kondisi robot ketika diam. Konfigurasi penyusunan robot dan pengukuran adalah seperti tampak pada gambar 4.2.



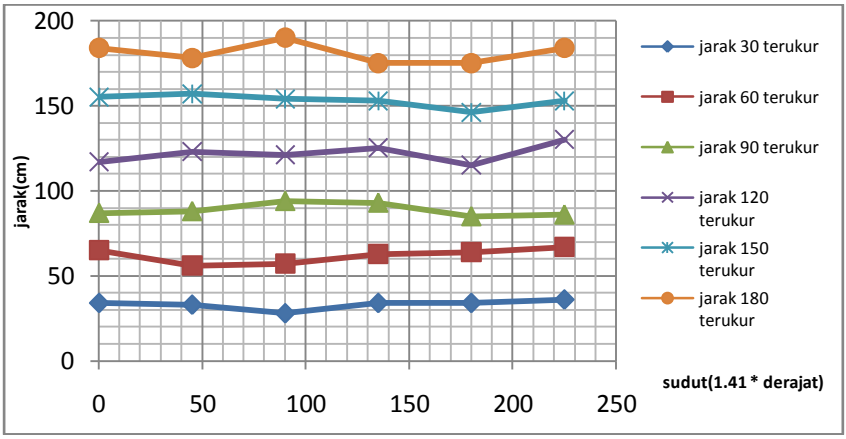
Gambar 4.2 *: posisi pengukuran robot*

Dari pengaturan robot seperti tampak pada gambar 4.2, dengan jarak jarak dan sudut yang dirubah-rubah, didapatkan data-data seperti terlihat pada table 4.3 dan pada lampiran. Dari pengujian didapatkan walaupun rangkaian penguat dan sensor sudah mampu menghasilkan dan menguatkan sinyal sesuai yang diinginkan namun data jarak yang didapatkan masih jauh dari presisi. Sedangkan dari sisi perangkat lunak semua perubahan sudah menggunakan fungsi interrupt, kemungkinan kesalahan yang ada adalah adanya gangguan dari alam seperti medan magenet bumi yang tidak menentu disuatu tempat. Salah satu indikasi kearah sana adalah jarak hasil pengukuran mempunyai tingkat

kesalahan terbesar pada arah mata angin tertentu. Dengan adanya medan magnet disekitar transducer maka akan mempengaruhi frekuensi resonansi dari transducer yang bersangkutan, namun data ini juga masih belum bisa dijadikan pegangan karena pengukuran masih dilakukan pada satu area (lapangan basket D4).

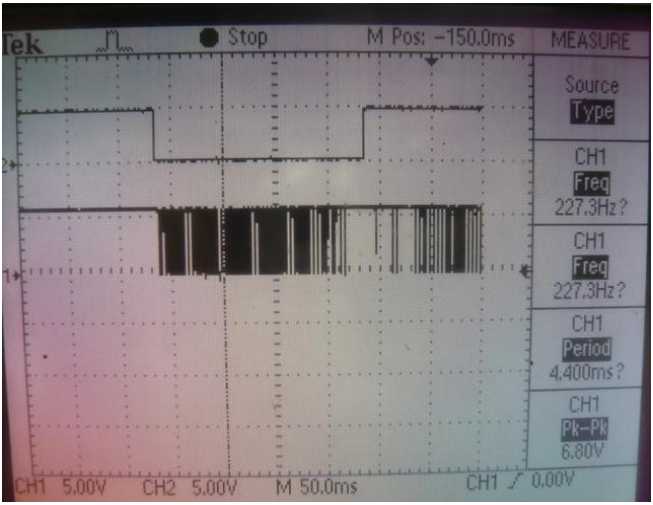
Tabel 4.3 : Data pengujian sensor jarak

Sudut (derajat)	Jarak (cm)					
	30	**60**	**90**	**120**	**150**	**180**
0	34	65	87	117	155	184
45	33	56	88	123	157	178
90	28	57	94	121	154	190
135	34	63	93	125	153	175
180	34	64	85	115	146	175
225	36	67	86	130	153	184



***Gambar 4.3** : grafik pengukuran jarak*

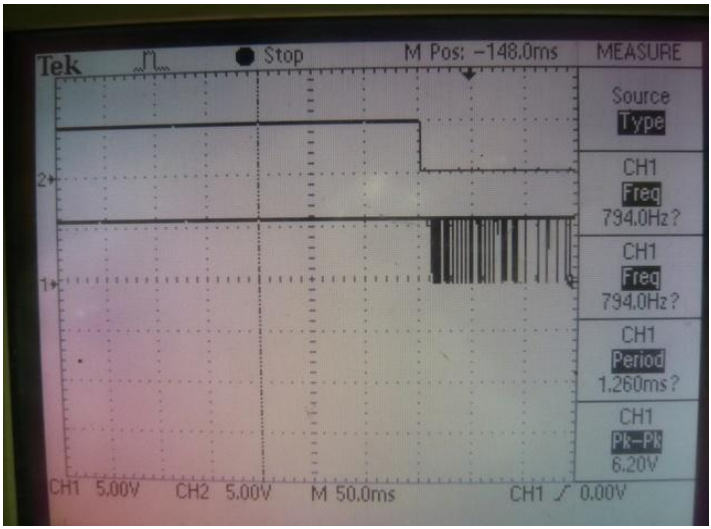
Selain karena faktor magnet bumi paling tidak masih terdapat empat kemungkinan penyebab kesalahan ukur jarak. Pertama adalah faktor suhu indikasinya adalah ketika angin berhembus kencang maka tingkat kesalahan data pengukuran jarak pun semakin besar. Yang kedua adalah desain fisik, dari fisik robot seperti tampak pada gambar 3.30, terlihat pada bagin belakang robot terdapat penyangga yang berguna untuk meletakkan reflector, ukuran penyangga yang dibuat adalah cukup besar untuk tujuan kekohan, namun mengingat ukuran dan letaknya maka sinyal pancar yang diharapkan kemungkinan juga tidak optimal karena adanya pantulan dan halangan yang tidak diinginkan. Faktor ketiga adalah transducer ultrasonik yang digunakan, adanya berbagai macam pilihan tranduser ultrasonik dipasaran dengan harga yang berfariasi, namun kelemahan indikasi kearah ini adalah tidak adanya data pendukung (datasheet) komponen bersangkutan di toko penjual.



Gambar 4.4 *: Sinyal picu dan sinyal waktu pada jarak 10cm*

Faktor keempat adalah lambatnya respon dari perangkat nirkable yang digunakan (xb-pro). mengingat xb-pro adalah modul wireless yang kompleks dimana didalamnya terdapat berbagai macam protocol dalam beberapa level layer, mulai dari pengturan IP/ID, tujuan

kirim, frekuensi kerja, baudrate dan masih banyak lagi, sehingga yang terjadi adalah ketika data keluar dan masuk modul xb-pro harus melewati beberapa tahapan yang cukup panjang. Dalam percobaan kami mencoba mengirim data dari pc ke pc lewat serial dengan mengunakan xb-pro, semisal pc kesatu adalah pcA dan pc kedua adalah pcB. Salah satu indicator adanya sinyal masuk adalah berubahnya nilai dari pin RSSI, pin ke-6 kaki xb-pro. nilai awal untuk pin ini jika tidak ada sinyal adalah logika 0 (0v), sedang ketika ada data maka pin berubah menjadi logika 1(3.3V). namun dalam pengujian ketika pcA berhenti mengirim data ke pcB, logika pada pin RSSI tenyata masih menunjukkan logika 1 untuk selang beberapa detik. Sehingga dalam pengiriman sinyal picu dan sinyal ultrasonik tidak dapat dilakukan dengan waktu yang hamper bersamaan namun harus dengan menanmbahkan jeda waktu tertentu antara pengiriman sinyal picu dengan sinyal ultrasonik. Gambar 4.4 dan 4.5 menunjukkan keluaran rangkaian penerima dengan jarak 10 cm dan jarak 125cm.



Gambar 4.5 *: Sinyal picu dan sinyal waktu pada jarak 125cm*

4.3 Uji Coba dan pengukuran arus motor DC

Tujuan dari pengukuran ini adalah untuk mendapatkan data konsumsi arus dari motor DC yang digunakan dengan mengatur dutycicle. Data pengujian dapat dilihat pada table 4.4

Tabel 4.3 : data arus motor DC

Duty cicle	Arus(A)
0	0.000
5	0.003
10	0.005
20	0.012
25	0.015
50	0.025
100	0.05

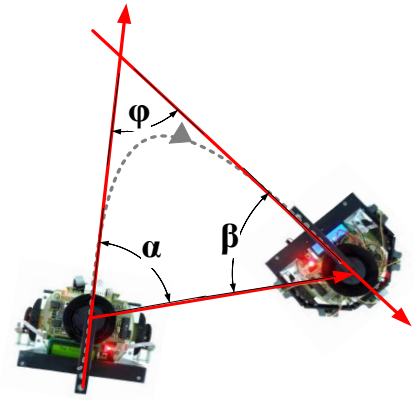
Keterangan : tegangan 12V, frekuensi : 500Hz

Dengan arus maksimal yang hanya sekitar 0.05A maka rangkaian driver yang digunakan (datasheet : IC L298[9]) sudah aman dan layak digunakan untuk keperluan robot.

4.4 Uji Coba Sistem

Karena pada pengukuran jarak masih terdapat kesalahan cukup besar. Maka metode mengikuti robot yang diujicobakan dibagi dalam dua macam, yaitu tanpa melibatkan sensor jarak dan dengan melibatkan sensor jarak. Namun pada kedua macam pengukuran peletakan posisi awal robot adalah sama yaitu, robot pemimpin pada koordinat (30,30) sedang robot pengikut berada pada koordinat (30,0)

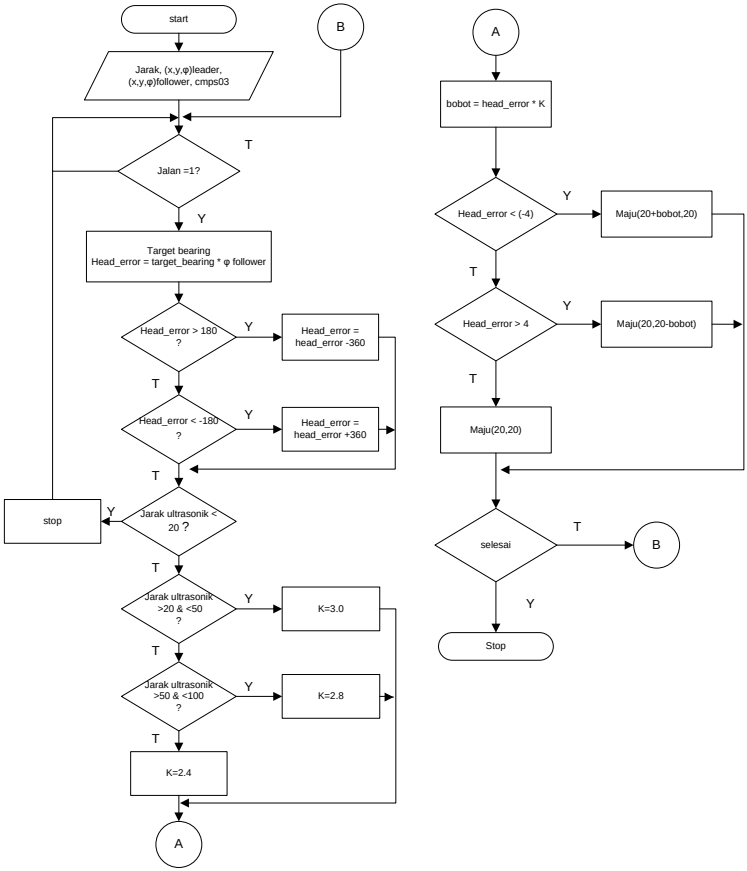
Gambar 4.6 *: mode lintasan*



Gambar 4.7 *: sudut lintasan yang menjadi acuan gerakan robot*

4.4.1 Uji Coba Sistem *Follow The Leader* tanpa halangan/obstacle.

Diagram alir untuk follow the leader dengan sensor jarak ultrasonik tampak seperti pada gambar 4.9. sedang tabel 4.4 adalah data hasil pengukuran dan



***Gambar 4.9** : Diagram Alir Kerja Robot Pengikut Dengan Menggunakan Sensor Jarak Ultrasonik*

Tabel 4.4: data posisi robot follower dan posisi robot leader

no	Robot pengikut				Robot pemimpin			
	Posisi robot rotary (x,y)		Posisi robot pengukuran (x,y)		Posisi robot rotary (x,y)		Posisi robot pengukuran (x,y)	
	X	y	x	y	x	y	x	y
1								
2								
3								
4								
5								
6								
7								
8								
9								
10								
11								
12								
13								
14								
15								
16								
17								
18								
19								
20								

Dari data tabel 4.4 didapatkan perbedaan mencolok dari rute/trajectory yang seharusnya dilewati. Jika diplotkan dalam grafik excel hasilnya tampak seperti pada gambar 4.11.

Gambar 4.11: *foto pengujian tanpa halangan / obstacle*

4.4.1 Uji Coba Sistem *Follow The Leader* dengan halangan / obstacle.

Tabel 4.4: data posisi robot follower dan posisi robot leader

no	Robot pengikut				Robot pemimpin			
	Posisi robot rotary (x,y)		Posisi robot pengukuran (x,y)		Posisi robot rotary (x,y)		Posisi robot pengukuran (x,y)	
	x	y	x	y	x	y	x	y
1								
2								
3								
4								
5								
6								
7								
8								
9								
10								
11								
12								

13								
14								
15								
16								
17								
18								
19								
20								

Dari data tabel 4.4 didapatkan perbedaan mencolok dari rute/trajectory yang seharusnya dilewati. Jika diplotkan dalam.

BAB V
PENUTUP

5.1 Kesimpulan

Setelah melakukan tahap perancangan dan pembuatan sistem yang kemudian dilanjutkan dengan tahap pengujian dan analisa maka dapat diambil kesimpulan sebagai berikut :

Pengukuran jarak dengan kondisi robot tidak terdapat halangan mempunyai error yang berfariasi mulai dari yang terendah 1% sampai dengan yang paling tinggi 20%. Hal terjadi karena banyak sebab mulai dari kurang optimalnya rangkaian sensor yang dibuat, gangguan dari alam yang tidak dapat dihindarkan seperti pengaruh medan magnet bumi yang mempengaruhi pergerakan membran pada tranduser ultrasonic walaupun kemungkinan pengaruhnya kecil, pengaruh suhu yang tidak merata dan stabil yang disebabkan pergerakan angin, pengaruh redaman pancaran gelombang ultrasonik oleh bumi, seperti yang terjadi pada gelombang dengan frekuensi rendah dimana semakin rendah frekuensi maka semakin besar pula redaman bumi terhadap gelombang tersebut, juga bisa disebabkan karena pengaruh material-material yang dibawa angin. Dan pengaruh disisi perangkat lunak seperti kurangnya optimasi program dan lain sebagainya. Sedangkan dari system *follow the leader*, robot sudah cukup mampu mengikuti robot pemimpin dengan rata-rata kesalahan sebesar

5.2 Saran

Pada pengerjaan Tugas Akhir ini, masih banyak terdapat berbagai macam kekurangan yang perlu untuk diperbaiki. Kekurangan yang ada meliputi kekurangan pada pengukuran jarak, dan metode gerakan robot. Sehingga perlu perancangan yang tepat dan cermat agar jarak pengukuran yang diperoleh tepat, cepat dan akurat. Dan harapan penulis kedepannya dapat dikembangkan swarm robot dengan sistem follow the leader yang mempunyai fariasi formasi yang lebih banyak. Semisal formasi gerakan diamond, lingkaran dan segitiga.

DAFTAR PUSTAKA

[1] Pitowarno, Endra, (2006). Robotika: Desain, Kontrol, Dan Kecerdasan Buatan. Buku Teks. Yogyakarta: ANDI. 1-3.

[2] http://en.wikipedia.org/wiki/Swarm_robotics.

[3] http://people.csail.mit.edu/jamesm/swarm.php.

[4]

[5] Chandak, P. 2002. "Study and Implementation of Follow the Leader". in partial fulfillment of the requirements for the degree of MASTER OF SCIENCE In the Department of Mechanical, Industrial and Nuclear Engineering of the College of Engineering, August-8-2002

[6] http://en.wikipedia.org/wiki/Time_of_arrival

[7] Rodney, H. (2004). "A Trilaterative Localization System for Small Mobile Robots in Swarms". A thesis submitted to the Department of Electrical and Computer Engineering and The Graduate School of The University of Wyoming in partial ful¯llment of the requirements for the degree of Master of Science in Electrical Engineering, Laramie, Wyoming August, 2004, pp. 8-9.

[8] datasheet Xb-pro

[9] datasheet ic L298

LAMPIRAN A
Schematic Rangkaian

LAMPIRAN B
LISTING PROGRAM

A. Listing Program ATMega128

```
/*****************************************************
Chip type           : ATmega128
Clock frequency     : 11.059200 MHz
*****************************************************/

#include <mega128.h>
#include <delay.h>
#include <math.h>
#include <stdlib.h>
#include <stdio.h>

// Alphanumeric LCD Module functions
#asm
   .equ   lcd port=0x1B ;PORTA
#endasm
#include <lcd.h>
// =====================================================
#define maju_ok    0x1E
#define stop ok    0x00
#define belki_ok   0x1A
#define belka_ok   0x0E
#define mundur_ok  0x0A

#define gerak   PORTF
#define velkan  OCR1A
#define velkir  OCR1B
#define phi     3.14159265358979323846264338327795
#define rads    57.29578

#define m8_en   PORTE.2
#define xb_en   PORTE.3
#define ti_en   PORTE.4

#define mulai   0xDF
#define henti   0x20
#define kirim  0xBF
#define enkir  0x40

#define sc45    0.70710678118654752440084436210485
// =====================================================
unsigned char xdata[5];
```

```
float target_bearing;           /* bearing in radians from
current position */
float target_distance;               /* distance in inches
from position */
float heading_error;            /* heading error in degrees
*/

float  bes_vek,res_sudut;

// ======================== usart0 data
============================
unsigned char lead_theta;
unsigned int kompas;
unsigned int waktu;
unsigned int dataH;
unsigned char ir1;
unsigned char ir2;
unsigned char ir3;
unsigned char ir4;
unsigned char ir5;
// ======================== usart1 data
============================
unsigned int data_H;
float z;
float x pos;
float y_pos;
float theta;
struct targ_param{
   float x;
   float y;
}target;
// ====================================================

#define RXB8 1
#define TXB8 0
#define UPE 2
#define OVR 3
#define FE 4
#define UDRE 5
#define RXC 7

#define FRAMING ERROR (1<<FE)
#define PARITY_ERROR (1<<UPE)
#define DATA_OVERRUN (1<<OVR)
#define DATA_REGISTER_EMPTY (1<<UDRE)
#define RX_COMPLETE (1<<RXC)

// USART0 Receiver buffer
#define RX_BUFFER_SIZE0 8
char rx_buffer0[RX_BUFFER_SIZE0];
```

```
#if RX_BUFFER_SIZE0<256
unsigned char rx_wr_index0,rx_rd_index0,rx_counter0;
#else
unsigned int rx_wr_index0,rx_rd_index0,rx_counter0;
#endif

// This flag is set on USART0 Receiver buffer overflow
bit rx_buffer_overflow0;
//------------------------
bit he_ok=0;
char fl_pos=0;
bit fl bcsens=0;
unsigned char tulis[10];
//------------------------
// usart0 adalah data untuk xbpro, dan data dari sensor
kompas, sensor luar
// USART0 Receiver interrupt service routine
interrupt [USART0_RXC] void usart0_rx_isr(void)
{
char status,data;
status=UCSR0A;
data=UDR0;
if ((status & (FRAMING_ERROR | PARITY_ERROR |
DATA OVERRUN))==0)
   {
   if (data==255){
        he_ok=1;
        rx_wr_index0=0;
   }
   if (he_ok){
        rx_buffer0[rx_wr_index0]=data;
        // ==================== panjang data 8 byte
================================
        if (++rx_wr_index0 == RX_BUFFER_SIZE0){
               if (rx_buffer0[1]==1){
                              fl_pos=0;
                      PORTF &= mulai;        // nutuk tiny
itung
               }
               else if (rx_buffer0[1]==2) fl_pos=1;
               else if (rx buffer0[1]==3) fl pos=2;

            switch(fl_pos){
               case 0:
                      lead_theta =rx_buffer0[2];
                      xb_en=1;               // xb  rakeno
                      ti_en=0;               // tiny keno
                      delay_ms(65);           // enteni 65
ms
```

```
                         PORTF |= henti;        // ucul tiny-
itung
                         PORTF &= kirim;         // nutuk tiny-
kirim
                         break;
                case 1:
                         ir1 = rx_buffer0[2];
                         ir2 = rx_buffer0[3];
                         ir3 = rx_buffer0[4];
                         ir4 = rx_buffer0[5];
                         ir5 = rx_buffer0[6];
                         if(rx_buffer0[7]!=0) kompas =
rx_buffer0[7];
                         sprintf(tulis,"LC=%3d",kompas);
                         lcd_gotoxy(9,0);
                         lcd_puts(tulis);
                         m8_en=1;               // m8 rakeno
                         xb_en=0;               // xb keno
                         break;
                 case 2:
                         dataH = rx_buffer0[2]; dataH <<= 8;
                         waktu = dataH | rx_buffer0[3];
                         PORTF |= enkir;                // ucul
tiny-kirim;
                         ti_en=1;               // tiny rakeno
                         m8_en=0;               // m8 keno
                         //--------------------
                         waktu = (float)waktu *
0.003101490162037037037037037037;
                         sprintf(tulis,"j=%5ld",(unsigned
int)waktu);
                         lcd_gotoxy(0,0);
                         lcd_puts(tulis);
                         //--------------------
                         fl_bcsens=1;
                         break;
                default:
                         break;
        }
             he_ok=0;
             rx_wr_index0=0;
        }
    }

  };
}

#ifndef _DEBUG_TERMINAL_IO_
// Get a character from the USART0 Receiver buffer
#define _ALTERNATE_GETCHAR_
```

```
#pragma used+
char getchar(void)
{
char data;
while (rx_counter0==0);
data=rx_buffer0[rx_rd_index0];
if (++rx_rd_index0 == RX_BUFFER_SIZE0) rx_rd_index0=0;
#asm("cli")
--rx_counter0;
#asm("sei")
return data;
}
#pragma used-
#endif
// ===================== USART1 Receiver buffer
================================
#define RX_BUFFER_SIZE1 9
char rx_buffer1[RX_BUFFER_SIZE1];

#if RX_BUFFER_SIZE1<256
unsigned char rx wr index1,rx rd index1,rx counter1;
#else
unsigned int rx_wr_index1,rx_rd_index1,rx_counter1;
#endif

// This flag is set on USART1 Receiver buffer overflow
bit rx_buffer_overflow1;
//---------------------
bit rot_ok=0;
unsigned char tulis2[16];
//---------------------
// USART1 Receiver interrupt service routine
interrupt [USART1_RXC] void usart1_rx_isr(void)
{
char status,data;
status=UCSR1A;
data=UDR1;
if ((status & (FRAMING_ERROR | PARITY_ERROR |
DATA_OVERRUN))==0)
   {
   if (data == 255){
        rot ok=1;
        rx_wr_index1=0;
   }

   if (rot_ok){
        rx_buffer1[rx_wr_index1]=data;
        if(++rx_wr_index1==RX_BUFFER_SIZE1){
               // xpos
               if(rx_buffer1[1]==1)z=-1.0;else z=1.0;
```

```
                data_H=rx_buffer1[3]<<8;
                x_pos=(data_H | rx_buffer1[2])*z;
                // ypos
                if(rx buffer1[4]==1)z=-1.0;else z=1.0;
                data_H=rx_buffer1[6]<<8;
                y_pos=(data_H | rx_buffer1[5])*z;
                // theta
                if(rx_buffer1[7]==1)z=-1.0;else z=1.0;
                theta=rx_buffer1[8]*z/40;
                sprintf(tulis2,"x=%5d y=%5d",(unsigned
int)x_pos,(unsigned int)y_pos);
                lcd_gotoxy(0,1);
                lcd puts(tulis2);
        }
   }
 };
}

// Get a character from the USART1 Receiver buffer
#pragma used+
char getchar1(void)
{
char data;
while (rx_counter1==0);
data=rx buffer1[rx rd index1];
if (++rx_rd_index1 == RX_BUFFER_SIZE1) rx_rd_index1=0;
#asm("cli")
--rx_counter1;
#asm("sei")
return data;
}
#pragma used-
// Write a character to the USART1 Transmitter
#pragma used+
void putchar1(char c)
{
while ((UCSR1A & DATA_REGISTER_EMPTY)==0);
UDR1=c;
}
#pragma used-
// ======================== umune pekerjaan
==================================

void maju(unsigned char pwmL, unsigned char pwmR);
void stop(void);
void belka(unsigned char pwmL, unsigned char pwmR);
void belki(unsigned char pwmL, unsigned char pwmR);
void rst_rotL(void);
void rst_rotR(void);
void start_odo(void);
```

```
void stop_odo(void);
float ICC(float jarak, float sudut);
void resultanx();
void avoidance();
void disall(void);
void cari_target(float xG,float yG);
void baca_sensor(void);
void cari_induk(void);

// ========================= Timer 0 overflow interrupt
service routine ==================================
unsigned char kir=0;
interrupt [TIM0 OVF] void timer0 ovf isr(void)
{
    TCNT0 = 40;               // .02s
    kir++;
    if (kir == 15){    // .3s
        TCCR0=0x00;
        if(fl_bcsens==1){
               baca_sensor();
               fl bcsens=0;
        }
        TCCR0=0x07;
        kir=0;
   }
}

// ========================= pekerjaan utomo
==================================

void main(void)
{
// Declare your local variables here
PORTA=0x00;
DDRA=0x00;

PORTB=0x00;
DDRB=0x60;

PORTC=0xFF;
DDRC=0xFF;

PORTD=0x00;
DDRD=0x00;

PORTE=0xFC;
DDRE=0x1C;

PORTF=0xE0;
DDRF=0xFF;
```

```
PORTG=0x00;
DDRG=0x00;

// Timer/Counter 0 initialization
ASSR=0x00;
TCCR0=0x07;
TCNT0=0x00;
OCR0=0x00;

// Timer/Counter 1 initialization
TCCR1A=0xA2;
TCCR1B=0x1D;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x36;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;
OCR1CH=0x00;
OCR1CL=0x00;

// Timer/Counter 2 initialization
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;

// Timer/Counter 3 initialization
TCCR3A=0x00;
TCCR3B=0x00;
TCNT3H=0x00;
TCNT3L=0x00;
ICR3H=0x00;
ICR3L=0x00;
OCR3AH=0x00;
OCR3AL=0x00;
OCR3BH=0x00;
OCR3BL=0x00;
OCR3CH=0x00;
OCR3CL=0x00;

// External Interrupt(s) initialization
EICRA=0x00;
EICRB=0x00;
EIMSK=0x00;

// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=0x01;
```

```
ETIMSK=0x00;

// USART0 initialization
UCSR0A=0x00;
UCSR0B=0x98;
UCSR0C=0x06;
UBRR0H=0x00;
UBRR0L=0x47;

// USART1 initialization
UCSR1A=0x00;
UCSR1B=0x98;
UCSR1C=0x06;
UBRR1H=0x00;
UBRR1L=0x47;

// Analog Comparator initialization
ACSR=0x80;
SFIOR=0x00;

// LCD module initialization
lcd_init(16);

// Global enable interrupts

stop();
disall();
xb_en=0;
stop_odo();
delay_ms(500);
rst_rotL();rst_rotR();
lcd_gotoxy(3,0);
lcd_putsf("The Follower");
delay_ms(2000);
lcd_clear();
#asm("sei")

cari_induk();

while (1)
    {

    }
}

//==================== fungsi gerak====================
void maju(unsigned char pwmL, unsigned char pwmR){
    gerak &=0xE0;
    gerak |= maju_ok;
    velkan = 50-(pwmR/2);
```

```
    velkir = 50-(pwmL/2);
}
void stop(void){
    gerak &=0xE0;
    gerak |= stop_ok;
    velkan = 0;
    velkir = 0;
}
void belka(unsigned char pwmL, unsigned char pwmR){
    gerak &=0xE0;
    gerak |= belka_ok;
    velkan = 50-(pwmR/2);
    velkir = 50-(pwmL/2);
}
void belki(unsigned char pwmL, unsigned char pwmR){
    gerak &=0xE0;
    gerak |= belki_ok;
    velkan = 50-(pwmR/2);
    velkir = 50-(pwmL/2);
}
//=====================================================

//******************odometri function to slave
uc**********************
void rst rotL(void){
        unsigned int j;
        for(j=0;j<16;j++)putchar1(0);
}
void rst_rotR(void){
       unsigned int j;
        for(j=0;j<16;j++)putchar1(1);
}
void start_odo(void){
       unsigned int j;
        for(j=0;j<16;j++)putchar1(255);
}
void stop_odo(void){
       unsigned int j;
        for(j=0;j<16;j++)putchar1(254);
}
//=============================================================
=============

/******************* utama
*************************************/
void cari_induk(){
       unsigned int sel_theta;
       float jarak;
       float Ls, alfa;
```

```
        while(1){
                sel_theta = lead_theta - theta;
                if(sel_theta < 0) sel_theta += 255;
                if (sel_theta > 127 ) sel_theta = 255-
sel_theta;
                jarak =  waktu;

                // proses pencarian
                if (jarak > 10.0){
                        if (fabs(sel_theta) > 1.0 ){
                                alfa = 0.5 * sel_theta;
                                Ls = 0.5*jarak *
sin(alfa/40.45);

                        }

                }
        }


}

/*************************************************************
***************/
float ICC(float jarak, float sudut){
        float Ls=0,alfa=0;
        alfa = 3.14 - sudut;
        Ls = (0.5*jarak)/sin(0.5*alfa);
        return(Ls);
}

void disall(void){
        m8_en = 1;
        ti_en = 1;
        xb_en = 1;
}

void cari_target(float xG,float yG)
{
        float x=0,y=0,v_bobot,k;
        float target_error;

        target.x=xG;
        target.y=yG;

        target_error=3;
        heading_error=0;
        // bagi dengan 10 karena di slave data dalam mm
sedang di sini berupa cm
        x = target.x - (x_pos/10);
```

```
        y = target.y - (y_pos/10);
        target_distance = sqrt(x*x+y*y);

        start odo();
        while(1){
                x = target.x - (x_pos/10);
                y = target.y - (y_pos/10);
                target_distance = sqrt(x*x+y*y);

                if (x > 0.00001) target_bearing = 90.0 -
(rads*(atan(y/x)));
                else if (x < -0.00001) target_bearing = -90.0
- (rads*(atan(y/x)));

                heading_error = target_bearing -
(theta*rads);

                if (heading_error > 180.0) heading_error -=
360.0;
                else if (heading_error < -180.0)
heading error += 360.0;

                //isikan obstacle avoidance disini
           //
        if((ir1<25)||(ir2<40)||(ir3<55)||(ir4<40)||(ir5<25))
avoidance();
                else{
                        if(target_distance<55)k=2.8;
                        else
if((target_distance>=55)&&(target_distance<=110))k=2.6;
                        else k=2.4;
                        v_bobot=heading_error*k;

                        if(heading_error<(-4)){
                                if(v_bobot<(-200))v_bobot=(-
200);
                                maju((50+(v_bobot/4)),50);
                        }
                        else if(heading_error>4){
                                if(v_bobot>200)v_bobot=200;
                                maju(50,(50-(v_bobot/4)));
                        }
                        else maju(50,50);
                }

                if((fabs(x) < target_error)&&(fabs(y) <
target_error))
                        while(1) stop();
        }
}
```

```
void baca_sensor(void){
        delay_us(10);
        putchar(255);
        putchar(2);    // m8 id
        putchar(0);
        putchar(0);
        putchar(0);
}
```

Halaman ini sengaja dikosongkan

B. Listing Program ATMega8 rotari

```
/*****************************************************
Chip type           : ATmega8
Program type        : Application
Clock frequency     : 11.059200 MHz
*****************************************************/

#include <mega8.h>
#include <math.h>
#include <delay.h>
#include <stdlib.h>

#define rotA_en PORTD.6
#define rotB_en PORTD.7
/*
       robot_data.diameter = 8.6;
       robot_data.lebar = 20.0;
       robot_data.rodaacuan = 5.5;
       robot_data.jmlrot = 22;
       robot_data.keliling =
27.017696820872221850778733096204;
*/
#define phi    3.14159265358979323846264338327 95
#define Rads   57.2958
#define Degs    0.01745329251994329576923690768 4886
#define jrkrot  1.22807712822146462958085150437 29
#define lebar   20.0
#define enable 0
#define disable       1
//=============================================================
==
float kanan=0.0;
float kiri=0.0;
//=============================================================
==
void rst_countL(void);
void rst_countR(void);
void stop_timer2(void);
void start_timer2(void);
//=============================================================
==
#define RXB8 1
#define TXB8 0
#define UPE 2
#define OVR 3
#define FE 4
#define UDRE 5
#define RXC 7
```

```
#define FRAMING_ERROR (1<<FE)
#define PARITY_ERROR (1<<UPE)
#define DATA_OVERRUN (1<<OVR)
#define DATA_REGISTER_EMPTY (1<<UDRE)
#define RX_COMPLETE (1<<RXC)

// USART Receiver buffer
#define RX_BUFFER_SIZE 8
char rx_buffer[RX_BUFFER_SIZE];

#if RX_BUFFER_SIZE<256
unsigned char rx_wr_index,rx_rd_index,rx_counter;
#else
unsigned int rx_wr_index,rx_rd_index,rx_counter;
#endif

// This flag is set on USART Receiver buffer overflow
bit rx_buffer_overflow;

// USART Receiver interrupt service routine
interrupt [USART_RXC] void usart_rx_isr(void)
{
char status,data;
status=UCSRA;
data=UDR;
if ((status & (FRAMING_ERROR | PARITY_ERROR |
DATA_OVERRUN))==0)
   {
   rx_buffer[rx_wr_index]=data;
   switch(data)
    {
        case 0: rst_countL(); break;
        case 1: rst_countR(); break;
        case 254: stop_timer2(); break;
        case 255: start_timer2(); break;
        default : break;
    }

   if (++rx_wr_index == RX_BUFFER_SIZE) rx_wr_index=0;
   if (++rx_counter == RX_BUFFER_SIZE)
      {
      rx_counter=0;
      rx_buffer_overflow=1;
      };
   };
}

#ifndef _DEBUG_TERMINAL_IO_
// Get a character from the USART Receiver buffer
#define _ALTERNATE_GETCHAR_
```

```
#pragma used+
char getchar(void)
{
char data;
while (rx_counter==0);
data=rx_buffer[rx_rd_index];
if (++rx_rd_index == RX_BUFFER_SIZE) rx_rd_index=0;
#asm("cli")
--rx_counter;
#asm("sei")
return data;
}
#pragma used-
#endif

// Standard Input/Output functions
#include <stdio.h>

// Timer 0 overflow interrupt service routine
interrupt [TIM0_OVF] void timer0_ovf_isr(void)
{
// Place your code here
    kanan += 255.0;
}
// Timer 1 overflow interrupt service routine
interrupt [TIM1_OVF] void timer1_ovf_isr(void)
{
// Place your code here
   kiri += 255.0;
}

unsigned char kirim=0;
unsigned int a,data_outL,data_outH;
float Rdist, Ldist, theta, jarak, x_pos, y_pos;
float total_jarak;
char z;
// Timer 2 overflow interrupt service routine
interrupt [TIM2_OVF] void timer2_ovf_isr(void)
{
// Place your code here
 TCNT2=40;     // .01s

 if(++kirim == 20){
       TCCR2=0x00;

       Rdist=jrkrot*((float)TCNT1L + kanan);
       Ldist=jrkrot*((float)TCNT0 + kiri);

       if (kanan != 0.0) kanan=0.0;
       if (kiri != 0.0)  kiri=0.0;
```

```
        TCNT1L=0x00;
        TCNT1H=0xFF;
        TCNT0=0x00;

        theta += ((Rdist-Ldist)/lebar);
        jarak  = (Ldist+Rdist)/2.0;
        total_jarak += jarak;

        x_pos+= (jarak/theta) * sin(theta);
        y_pos+= (-(jarak/theta)) * cos(theta)+ jarak/theta;

        if(x pos==255)x pos=254;
        if(y_pos==255)y_pos=254;
        if(theta==255)theta=254;
        if(jarak==255)total_jarak=254;

        putchar(255);                           // head
         if(x_pos<0) z=1; else z=0;             // x_pos
        a=fabs(x_pos);
        putchar(z);
         data_outL = a & 0x00FF;
         if(data_outL==255)data_outL=254;
        putchar(data_outL);
         data outH= a>>8;
         if(data_outH==255)data_outH=254;
        putchar(data_outH);
         if(y_pos<0) z=1; else z=0;             // y_pos
         a=fabs(y_pos);
        putchar(z);
         data_outL=a & 0x00FF;
         if(data_outL==255)data_outL=254;
        putchar(data_outL);
         data_outH= a>>8;
         if(data_outH==255)data_outH=254;
        putchar(data_outH);
         if(theta<0) z=1; else z=0;             // theta
        a=fabs(theta);
        putchar(z);
         a=a*40;
         if(a==255)a=254;
        putchar(a);

        TCCR2=0x07;
        kirim=0;
 }

}
// Declare your global variables here
```

```
void main(void)
{
// Declare your local variables here

PORTB=0x00;
DDRB=0x00;

PORTC=0x00;
DDRC=0x00;

PORTD=0x00;
DDRD=0xC0;

// Timer/Counter 0 initialization
// Clock source: T0 pin Falling Edge
TCCR0=0x06;
TCNT0=0x00;

// Timer/Counter 1 initialization
// Clock source: T1 pin Falling Edge
// Mode: Ph. correct PWM top=00FFh
// OC1A output: Discon.
// OC1B output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer 1 Overflow Interrupt: On
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
TCCR1A=0x00;
TCCR1B=0x06;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;

// Timer/Counter 2 initialization
// Clock source: System Clock
// Clock value: 10.800 kHz
// Mode: Normal top=FFh
// OC2 output: Disconnected
ASSR=0x00;
TCCR2=0x07;
TCNT2=0x00;
OCR2=0x00;
```

```
// External Interrupt(s) initialization
// INT0: Off
// INT1: Off
MCUCR=0x00;

// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=0x45;

// USART initialization
// Communication Parameters: 8 Data, 1 Stop, No Parity
// USART Receiver: On
// USART Transmitter: On
// USART Mode: Asynchronous
// USART Baud Rate: 9600
UCSRA=0x00;
UCSRB=0x98;
UCSRC=0x86;
UBRRH=0x00;
UBRRL=0x47;

// Analog Comparator initialization
// Analog Comparator: Off
// Analog Comparator Input Capture by Timer/Counter 1: Off
ACSR=0x80;
SFIOR=0x00;

// Global enable interrupts
#asm("sei")
// aktifkan rotari
rotA_en = enable;
rotB_en = enable;
while (1)
      {
      // Place your code here

      };
}
// *****************************************************
void rst_countL(void){
    TCNT1L=0x00;
    TCNT1H=0xFF;
    kiri=0;
}
void rst_countR(void){
    TCNT0=0xFF;
    kanan=0;
}
void stop_timer2(void){
    TCCR2=0x00;
}
```

```
void start_timer2(void){
    TCCR2=0x07;
}
```

Halaman ini sengaja dikosongkan

C. Listing Program ATMega8 sensor

```
/*****************************************************
Chip type           : ATmega8
Clock frequency     : 11.059200 MHz
*****************************************************/

#include <mega8.h>

// I2C Bus functions
#asm
   .equ __i2c_port=0x18 ;PORTB
   .equ __sda_bit=0
   .equ __scl_bit=1
#endasm
#include <i2c.h>

#define RXB8 1
#define TXB8 0
#define UPE 2
#define OVR 3
#define FE 4
#define UDRE 5
#define RXC 7

#define FRAMING_ERROR (1<<FE)
#define PARITY_ERROR (1<<UPE)
#define DATA_OVERRUN (1<<OVR)
#define DATA_REGISTER_EMPTY (1<<UDRE)
#define RX_COMPLETE (1<<RXC)

// USART Receiver buffer
#define RX_BUFFER_SIZE 5
char rx_buffer[RX_BUFFER_SIZE];

#if RX_BUFFER_SIZE<256
unsigned char rx_wr_index,rx_rd_index,rx_counter;
#else
unsigned int rx_wr_index,rx_rd_index,rx_counter;
#endif

// This flag is set on USART Receiver buffer overflow
bit rx_buffer_overflow;
bit head_ok=0;
bit kirim_ok;
// USART Receiver interrupt service routine
interrupt [USART_RXC] void usart_rx_isr(void)
{
char status,data;
status=UCSRA;
```

```
data=UDR;
if ((status & (FRAMING_ERROR | PARITY_ERROR |
DATA_OVERRUN))==0)
   {
    if (data==255){
        rx_wr_index=0;
        head_ok = 1;
    }

    if (head_ok){
        rx_buffer[rx_wr_index]=data;
        if(++rx_wr_index== RX_BUFFER_SIZE){
            if (rx buffer[1]==2 && rx buffer[4]==0){
              kirim_ok=1;
              rx_wr_index=0;
              rx_counter=0;
              rx_buffer_overflow=1;
            }
        }
    }
   };
}

#ifndef _DEBUG_TERMINAL_IO_
// Get a character from the USART Receiver buffer
#define _ALTERNATE_GETCHAR_
#pragma used+
char getchar(void)
{
char data;
while (rx_counter==0);
data=rx_buffer[rx_rd_index];
if (++rx_rd_index == RX_BUFFER_SIZE) rx_rd_index=0;
#asm("cli")
--rx_counter;
#asm("sei")
return data;
}
#pragma used-
#endif

// Standard Input/Output functions
#include <stdio.h>
#include <delay.h>

#define ADC_VREF_TYPE 0x20

// Read the 8 most significant bits
// of the AD conversion result
unsigned char read_adc(unsigned char adc_input)
```

```
{
ADMUX=adc_input | (ADC_VREF_TYPE & 0xff);
// Delay needed for the stabilization of the ADC input
voltage
delay_us(10);
// Start the AD conversion
ADCSRA|=0x40;
// Wait for the AD conversion to complete
while ((ADCSRA & 0x10)==0);
ADCSRA|=0x10;
return ADCH;
}

unsigned char kompas;

unsigned char baca_kompas(void){
        char read_data=0;
        i2c_start();
       i2c_write(0xC0);
       i2c_write(1);
       i2c_start();
       i2c_write(0xC1);
       delay_ms(1);
       read_data = i2c_read(0);
       i2c_stop();
       return read_data;
}


char count=0;
unsigned char ir[5];
// Timer 0 overflow interrupt service routine
interrupt [TIM0_OVF] void timer0_ovf_isr(void)
{
    char i,j;
    unsigned int adc_rate=0;
    TCNT0 = 40;               // .02s
    if(++count == 10){ // .2s
        TCCR0=0x00;
        for(i=0;i<6;i++){
          for (j=0;j<6;j++)
            adc_rate += read_adc(i);
          ir[i]=adc_rate/6;
          if(ir[i]==255)ir[i]=254;
        }
        kompas=baca_kompas();
        count=0;
        TCCR0=0x05;
    }
}
```

```
// Declare your global variables here

void main(void)
{
// Declare your local variables here

PORTB=0x00;
DDRB=0x00;

PORTC=0x00;
DDRC=0x00;

PORTD=0x00;
DDRD=0x00;

// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: 10.800 kHz
TCCR0=0x05;
TCNT0=0x00;

TCCR1A=0x00;
TCCR1B=0x00;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;

ASSR=0x00;
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;

MCUCR=0x00;

// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=0x01;

// USART initialization
// Communication Parameters: 8 Data, 1 Stop, No Parity
// USART Receiver: On
// USART Transmitter: On
// USART Mode: Asynchronous
// USART Baud Rate: 9600
UCSRA=0x00;
```

```
UCSRB=0x98;
UCSRC=0x86;
UBRRH=0x00;
UBRRL=0x47;

// Analog Comparator initialization
// Analog Comparator: Off
// Analog Comparator Input Capture by Timer/Counter 1: Off
ACSR=0x80;
SFIOR=0x00;

// ADC initialization
// ADC Clock frequency: 86.400 kHz
// ADC Voltage Reference: AREF pin
// Only the 8 most significant bits of
// the AD conversion result are used
ADMUX=ADC_VREF_TYPE & 0xff;
ADCSRA=0x87;

// I2C Bus initialization
i2c_init();

// Global enable interrupts
#asm("sei")

while (1)
      {
      // Place your code here

        if (kirim_ok==1){
            putchar(255);
            putchar(2);
            putchar(ir[0]);
            putchar(ir[1]);
            putchar(ir[2]);
            putchar(ir[3]);
            putchar(ir[4]);
            putchar(kompas);
            kirim_ok=0;
        }
      };
}
```

Halaman ini sengaja dikosongkan

D. Listing Program attiny2313

```
#include <tiny2313.h>
#include <delay.h>

#define max232 PORTB.4
#define aktif  0
#define mati   1
//bit mulai =0;
//bit kirim =0;
unsigned int waktu=0, wL=0;
bit fungsi =0;

/*

// External Interrupt 0 service routine
interrupt [EXT_INT0] void ext_int0_isr(void)
{
// Place your code here
        mulai=1;
}

// External Interrupt 1 service routine
interrupt [EXT_INT1] void ext_int1_isr(void)
{
// Place your code here
        kirim=1;
}
*/
// Standard Input/Output functions
#include <stdio.h>

// Declare your global variables here

void main(void)
{
// Declare your local variables here
char i;
// Crystal Oscillator division factor: 1
CLKPR=0x80;
CLKPR=0x00;

// Input/Output Ports initialization
// Port A initialization
// Func0=In Func1=In Func2=In
// State0=T State1=T State2=T
PORTA=0x00;
DDRA=0x00;

// Port B initialization
```

```
// Func0=In Func1=In Func2=out Func3=Out Func4=Out Func5=In
Func6=In Func7=In
// State0=T State1=T State2=0 State3=0 State4=1 State5=T
State6=T State7=T
PORTB=0x10;
DDRB=0x1C;

// Port D initialization
// Func0=In Func1=In Func2=In Func3=In Func4=In Func5=In
Func6=In
// State0=P State1=P State2=P State3=P State4=P State5=P
State6=P
PORTD=0xFF;
DDRD=0x00;

// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: Timer 0 Stopped
// Mode: Normal top=FFh
// OC0A output: Disconnected
// OC0B output: Disconnected
TCCR0A=0x00;
TCCR0B=0x00;
TCNT0=0x00;
OCR0A=0x00;
OCR0B=0x00;

// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: 11059.200 kHz
// Mode: Normal top=FFFFh
// OC1A output: Discon.
// OC1B output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
TCCR1A=0x00;
TCCR1B=0x01;
TCNT1H=0x00;
TCNT1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;


// External Interrupt(s) initialization
// INT0: Off
// INT1: Off
// Interrupt on any change on pins PCINT0-7: Off
GIMSK=0x00;
```

```
MCUCR=0x00;


// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=0x00;

// Universal Serial Interface initialization
// Mode: Disabled
// Clock source: Register & Counter=no clk.
// USI Counter Overflow Interrupt: Off
USICR=0x00;

// USART initialization
// Communication Parameters: 8 Data, 1 Stop, No Parity
// USART Receiver: Off
// USART Transmitter: On
// USART Mode: Asynchronous
// USART Baud rate: 9600
UCSRB=0x08;
UCSRC=0x06;
UBRRH=0x00;
UBRRL=0x47;

// Analog Comparator initialization
// Analog Comparator: Off
// Analog Comparator Input Capture by Timer/Counter 1: Off
// Analog Comparator Output: Off
ACSR=0x80;

// Global enable interrupts
//#asm("sei")
max232 = aktif;
while (1)
      {
      // Place your code here
      if (fungsi==0){
        while(PIND.2==1);
        TCNT1=0;
        while(PIND.4==1);
        waktu=TCNT1;
        while(PIND.3==1);
        wL=waktu & 0x00FF;
        waktu &= 0xFF00;
        waktu >>= 8;
        putchar(255);
        putchar(3);
        putchar(waktu);
        putchar(wL);
        putchar(0);
        putchar(0);
```

```
        putchar(0);
        putchar(0);
        PORTD=0xFC;
      }
      else
      {
           for(i=0;i<8;i++){
                PORTB.3 = 0;
                PORTB.2 = 1;
                delay_us(12);
                PORTB.3 = 1;
                PORTB.2 = 0;
                delay us(12);
            }
      }

      };
}
```

TENTANG PENULIS

Nama : hakim s
Tempat lahir : Kediri
Alamat : Kediri
mail : hqm_saad@yahoo.co.id

Sesuai kode pada gambar IC diatas. IC 74HC125 adalah IC quad bus buffer Tristate. Tristate melambangkan keadaan dimana kondisi pada saat itu jalur yang terhubung dalam keadaan ambang/tidak jelas. Begitu juga dengan harapan orang tua penulis, agar penulis menjadi orang yang berguna bagi agama, masyarakat, bangsa dan negara dan tidak menjadi sampah masyarakat atau orang yang bingung memilih diantara keduanya.

Sekilas tentang penulis : dilahirkan puluhan tahun yang lalu. Tepatnya di sekitar daerah pemakaman auliya' Tambak-Ngadi-Mojo-Kediri, atau secara geografis dibawah lereng gunung wilis yang cukup panas karena sudah cukup gundul dan ditepi sungai Brantas yang sering meluap bila turun hujan.

Tidak ada catatan istimewa dan prestasi yang membanggakan dalam sejarah kependidikan formal penulis, sekolah dari SD dan SMP didaerah impress tertinggal. Dan dengan keberuntungan karena nilai sekolah yang pas-pas an akhirnya penulis mampu sekolah di SMK terbaik dikota Kediri, SMK N 1 KEDIRI. Namun, lagi-lagi walaupun sekolah disekolah yang cukup mentereng dalam prestasinya, penulis juga lulus dengan nilai yang pas-pas an, ini dapat dilihat dari peringkat kelas penulis yang selalu menghuni sepuluh besar dari bawah. Setelah lulus dari SMK N 1 Kediri dengan nilai pas-pas an juga. Penulis memberanikan diri untuk mendaftar di salah satu perguruan tinggi kiblatnya robot di Indonesis. PENS-ITS. Dan dengan keberuntungan yang besar karena nilai sekolahnya yang pas-pas an juga, Alhamdulillah diterima. Dan akhirnya setelah tahunan menimba, mempompa ilmu di PENS-ITS, penulis pun menyabet gelar sarjana nya dengan keberuntungan dan nilai IPK yang pas-pas an jua. Namun diluar semua itu penulis sangat bangga terhadap para guru, dosen dan teman-teman penulis yang mempunyai semangat tholabul ilmi yang sangat tinggi. Semoga Allah selalu menancapkan semangat tersebut pada jiwa-jiwa mereka.

Ki Enthus Susmono: "nek wani ojo wedi-wedi, nek wedi ojo wani-wani"