Institut
Teknologi
Sepuluh Nopember

TUGAS AKHIR – TE141599

SISTEM *AUTODOCKING MOBILE ROBOT* BERBASIS SUARA UNTUK PENGISIAN ULANG BATERAI

Ari Hidayanto
NRP. 07111645000023

Dosen Pembimbing
Dr. Muhammad Rivai, ST., MT.
Astria Nur Irfansyah, ST., M.Eng., Ph.D.

DEPARTEMEN TEKNIK ELEKTRO
Fakultas Teknologi Elektro
Institut Teknologi Sepuluh Nopember
Surabaya 2018

TUGAS AKHIR – TE141599

SISTEM *AUTODOCKING MOBILE ROBOT* BERBASIS SUARA UNTUK PENGISIAN ULANG BATERAI

Ari Hidayanto
NRP. 07111645000023

Dosen Pembimbing
Dr. Muhammad Rivai, ST., MT.
Astria Nur Irfansyah, ST., M.Eng., Ph.D.

DEPARTEMEN TEKNIK ELEKTRO
Fakultas Teknologi Elektro
Institut Teknologi Sepuluh Nopember
Surabaya 2018

FINAL PROJECT – TE141599

AUTODOCKING MOBILE ROBOT SYSTEM BASED ON SOUND FOR BATTERY RECHARGING

Ari Hidayanto
NRP. 07111645000023

Supervisor
Dr. Muhammad Rivai, ST., MT.
Astria Nur Irfansyah, ST., M.Eng., Ph.D.

ELECTRICAL ENGINEERING DEPARTMENT
Faculty Of Electrical Technology
Institut Teknologi Sepuluh Nopember
Surabaya 2018

PERNYATAAN KEASLIAN
TUGAS AKHIR

Dengan ini saya menyatakan bahwa isi sebagian maupun keseluruhan Tugas Akhir saya dengan judul "**Sistem *Autodocking Mobile Robot* Berbasis Suara Untuk Pengisian Ulang Baterai**" adalah benar-benar hasil karya intelektual mandiri, diselesaikan tanpa menggunakan bahan-bahan yang tidak diijinkan dan bukan merupakan karya pihak lain yang saya akui sebagai karya sendiri.

Semua referensi yang dikutip maupun dirujuk telah ditulis secara lengkap pada daftar pustaka.

Apabila ternyata pernyataan ini tidak benar, saya bersedia menerima sanksi sesuai peraturan yang berlaku.

Surabaya, 16 Juli 2018

Ari Hidayanto
NRP. 07111645000023

----- Halaman ini sengaja dikosongkan -----

SISTEM *AUTODOCKING MOBILE ROBOT* BERBASIS SUARA UNTUK PENGISIAN ULANG BATERAI

TUGAS AKHIR

Diajukan Guna Memenuhi Sebagian Persyaratan
Untuk Memperoleh Gelar Sarjana Teknik
Pada
Bidang Studi Elektronika
Departemen Teknik Elektro
Fakultas Teknologi Elektro
Institut Teknologi Sepuluh Nopember

Menyetujui :

Dosen Pembimbing I	Dosen Pembimbing II
Dr. Muhammad Rivai, ST., MT.	**Astria Nur Irfansyah, ST., M.Eng., Ph.D.**
NIP. 196904261994031003	**NIP. 198103252010121002**

SURABAYA
JULI 2018

----- Halaman ini sengaja dikosongkan -----

Sistem *Autodocking Mobile Robot* Berbasis Suara Untuk Pengisian Ulang Baterai

Nama : Ari Hidayanto
Dosen Pembimbing 1 : Dr. Muhammad Rivai, ST., MT.
Dosen Pembimbing 2 : Astria Nur Irfansyah, ST., M.Eng., Ph.D.

ABSTRAK

Siklus kerja *autonomous mobile robot* dirancang penuh untuk mampu berjalan dan bekerja secara kontinyu dengan lintasan yang sudah ditentukan oleh *waypoint Global Positioning System* (GPS) tanpa ada campur tangan manusia. Salah satu sistem yang cukup penting pada *mobile robot* untuk masalah tersebut adalah sistem *autodocking* yang digunakan ketika baterai dalam level rendah atau akan habis supaya kembali ke *power station* untuk mengisi ulang baterai. Dalam proses menuju *power station* diperlukan keakuratan *mobile robot* agar posisi antara *transmitter* dengan *receiver* baterai berada pada jarak transfer *Wireless Power Transmision* (WPT). Penelitian ini mengimplementasikan *mobile robot* yang dirancang menggunakan Arduino Mega 2560 sebagai kontrol utama sistem, GPS Ublox Neo M8N, modul kompas HMC5883L, modul mikrofon kondenser, modul HC-SR04, motor *driver* L298N dengan 4 buah motor DC, dan LCD. Sedangkan untuk sumber suara menggunakan *speaker* Polytron Muze (PSP B1) dengan suara sonar yang memiliki rentang frekuensi 900 Hz hingga 1100 Hz. Modul mikrofon ditempatkan pada sisi kanan dan kiri *mobile* robot dengan metode mencari arah sumber suara pada WPT berdasarkan *different level intensity*. Hasil pengujian penelitian ini menunjukkan *mobile robot* yang dirancang memiliki kemampuan menuju titik *waypoint* dengan *error* jarak posisi mencapai 6 meter. Penggunaan corong pengarah pada sensor suara memiliki efektifitas sudut *directivity* 90° dan -90° membuat sensor suara menjadi lebih sensitif. Sedangkan *mobile robot* dapat mendeteksi suara dan menuju ke sumber suara dengan efektifitas jarak kurang dari 100 cm yang memiliki waktu tempuh kurang dari 60 detik dengan tingkat keberhasilan 56,25%.

Kata Kunci: *Autodocking, Mobile robot, Power Station,* Sensor Suara

----- Halaman ini sengaja dikosongkan -----

Autodocking Mobile Robot System Based On Sound For Battery Recharging

Name **: Ari Hidayanto**
Supervisor **: Dr. Muhammad Rivai, ST., MT.**
Co-Supervisor **: Astria Nur Irfansyah, ST., M.Eng., Ph.D.**

ABSTRACT

The autonomous mobile robot work cycle is fully designed to be able to run and work continuously with paths defined by the Global Positioning System (GPS) waypoint without any human intervention. One of the most important systems in the mobile robot for this problem is the autodocking system used when the battery is low or will run out to return to the power station to recharge the battery. In the process of going to the power station the accuracy of the mobile robot is required so that the position between the transmitter and the battery receiver is at the transfer distance of Wireless Power Transmission (WPT). This research implements mobile robot designed using Arduino Mega 2560 as the main control system, GPS Ublox Neo M8N, HMC5883L compass module, condenser microphone module, HC-SR04 module, L298N motor driver with 4 DC motor and LCD. As for the sound source using Polytron Muze speakers (PSP B1) with sonar sound that has a frequency range of 900 Hz to 1100 Hz. The microphone module is placed on the right and left sides of the mobile robot by the method of searching the direction of the sound source on the WPT based on different intensity levels. The result of this research shows that mobile robot designed has ability to point waypoint with error distance of position reaches 6 meters. The use of the pointing funnel of the sound sensor has an effectivity of 90° and -90° directivity angle makes the sound sensor more sensitive. While the mobile robot can detect sound and go to the sound source with the effectiveness of distance less than 100 cm which has travel time less than 60 seconds with success rate of 56,25%.

Keywords : Autodocking, Mobile Robot, Power Station, Sound Sensor

----- Halaman ini sengaja dikosongkan -----

KATA PENGANTAR

Segala puji syukur penulis panjatkan atas kehadirat Allah SWT yang selalu memberikan rahmat serta hidayah-Nya sehingga penelitian Tugas Akhir ini dapat terselesaikan dengan baik. Shalawat serta salam selalu dilimpahkan kepada Rasulullah Muhammad SAW, keluarga, sahabat, dan umat muslim yang senantiasa meneladani beliau.

Banyak pihak yang berperan dalam penyelesaian penelitian Tugas Akhir ini. Oleh karena itu, pada kesempatan ini penulis ingin menyampaikan terima kasih kepada pihak-pihak yang terkait antara lain:

1. Dr. Muhammad Rivai, ST., MT. dan Astria Nur Irfansyah, ST., M.Eng., Ph.D. selaku dosen pembimbing yang telah memberikan bimbingan, saran, pengarahan, dan motivasi yang diberikan selama pengerjaan penelitian Tugas Akhir ini.
2. Dr. Ir. Hendra Kusuma, M.Eng.Sc., Dr. Ir. Totok Mujiono, M.Kom., Ir. Haris Pringadi, MT., dan Muhammad Attamimi, B.Eng., M.Eng., Ph.D. selaku dosen penguji sidang ujian Tugas Akhir.
3. Ketua Departemen Teknik Elektro Dr. Eng. Ardyono Priyadi, ST., M.Eng.
4. Seluruh dosen bidang studi Elektronika Departemen Teknik Elektro FTE ITS.
5. Seluruh keluarga penulis terutama ayahanda Si'iswanto, ibunda Sih Setyaning Hastuti, dan kakak-kakakku yang selalu memberikan doa, dukungan, motivasi, semangat, perhatian dan kasih sayangnya.
6. Semua pihak yang telah banyak membantu dalam penyelesaian Tugas Akhir ini yang tidak dapat penulis sebutkan satu-persatu.

Penulis menyadari bahwa dalam penelitian Tugas Akhir ini terdapat banyak kekurangan. Saran, kritik, dan masukkan dari berbagai pihak sangat membantu untuk pengembangan lebih lanjut. Semoga melalui tulisan ini dapat bermanfaat dan dapat berbagi ilmu bagi pembacanya.

Surabaya, 16 Juli 2018

Penulis

----- Halaman ini sengaja dikosongkan -----

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

----- Halaman ini sengaja dikosongkan -----

BAB I
PENDAHULUAN

1.1. Latar Belakang

Dalam proses kegiatan eksplorasi hingga produksi gas pastinya menghasilkan temuan berbagai senyawa kimia yang belum diketahui secara pasti kadar maupun tingkat keamanannya. Ketika manusia secara langsung mengecek area dan kondisi senyawa kimia tersebut, kemungkinan resiko terkena dampak negatif bagi keamanan dan kesehatan diri seperti sesak nafas hingga kemandulan akan lebih besar.

Misalnya, terdapat gas hidrogen sulfida yang merupakan suatu gas tidak berwarna, sangat beracun, mudah terbakar dan memiliki karakteristik bau telur busuk. Gas ini dapat menyebabkan dampak yang buruk bagi kesehatan. Paparan hidrogen sulfida dengan konsentrasi rendah dalam jangka waktu yang lama dapat menyebabkan efek yang cukup parah seperti gangguan saluran pernafasan, gangguan pencernaan, iritasi mata, sakit kepala, insomnia, dan batuk kronis [1].

Berdasarkan data Kementerian Energi dan Sumber Daya (ESDM) tercatat angka kecelakaan kerja pada kegiatan hulu minyak dan gas bumi (migas) di tahun 2014 mencapai 159 kejadian. Dari angka tersebut, 106 diantaranya merupakan kecelakaan ringan, 32 kecelakaan sedang, 16 kecelakaan berkategori berat, dan 6 lainnya kecelakaan fatal. Sementara di tahun sebelumnya, kecelakan kerja tercatat mencapai 183 kecelakaan [2].

Sebagai upaya untuk mencegah terjadinya kecelakaan dalam kegiatan eksplorasi dan produksi dalam dunia gas salah satunya dapat menggunakan *mobile robot* yang akan secara kontinyu memetakkan titik-titik lokasi keberadaan senyawa kimia minyak atau gas. *Mobile robot* ini akan berjalan secara *autonomous* dan mandiri dimana robot mampu terus berjalan dan bekerja tanpa ada campur tangan manusia, kecuali robot mengalami kerusakan.

Salah satu sistem yang cukup penting pada *mobile robot* tersebut adalah sistem autodocking yang digunakan ketika baterai dalam level rendah atau akan habis supaya kembali ke *power station* untuk mengisi ulang baterai. Yang populer digunakan sebagai penentuan titik kembali untuk *autodocking* adalah *waypoint* GPS, namun *waypoint* GPS memiliki *error* jarak yang cukup jauh sehingga tidak dapat secara tepat masuk ke dalam *power station*. Dalam kesempatan ini akan dirancang sistem

autodocking dengan kombinasi 2 metode yaitu ketika *mobile robot* ada di jarak yang jauh akan menggunakan *waypoint* GPS, kemudian ketika GPS telah sampai pada titik yang dituju, metode jarak dekat dengan sensor suara akan aktif.

Dalam proses menuju *power station* diperlukan ketepatan posisi *mobile robot* agar posisi antara *transmitter* dengan receiver *charger* baterai sesuai. Maka dari itu pada *mobile robot* menggunakan sensor audio sebagai telinga dari robot untuk memposisikan secara tepat dan sesuai. Kemampuan *mobile robot* untuk dapat berjalan menelusuri area dan kembali ke *power station* (*dock*) merupakan salah satu sistem otomasi yang sangat dibutuhkan agar robot dapat terus bekerja secara normal.

1.2. Perumusan Masalah

Berdasarkan latar belakang di atas, dapat dirumuskan beberapa masalah. Antara lain:

1. Bagaimana memandu *mobile robot* menuju ke *power station* untuk pengisian baterai.
2. Bagaimana *mobile robot* dapat menuju ke antena *transmitter* pengisi baterai secara tepat.
3. Bagaimana mengimplementasikan sistem *autodocking* dalam suatu mikrokontroler.

1.3. Tujuan Penelitian

Tujuan dari penelitian pada tugas akhir ini adalah sebagai berikut:

1. Membuat metode pada saat sistem *autodocking* jarak jauh menggunakan metode GPS *waypoint*.
2. Membuat *power station* atau *docking station* yang dilengkapi dengan sumber suara agar *mobile robot* mampu mencari titik terdekat menuju transmitter pengisi baterai.
3. Rancang bangun sistem *autodocking* menggunakan mikrokontroler Arduino Mega 2560.

1.4. Batasan Masalah

Untuk batasan masalah dari tugas akhir ini sebagai berikut:

1. Robot dijalankan pada tempat yang datar, tidak terdapat halangan dan tidak tergenang air.
2. Sensor suara sebagai telinga robot yang digunakan sebanyak dua buah dan mampu mendeteksi sumber suara dengan radius ±2 m.

3. Frekuensi dari sumber suara harus ditetapkan dengan interval tertentu.
4. *Mobile robot* mampu bergerak menuju sumber suara dengan tepat.

1.5. Metodologi Penelitian

Dalam penyelesaian tugas akhir ini digunakan metodologi sebagai berikut:

1. Studi literatur

   Pada tahap ini dilakukan pengumpulan dasar teori yang menunjang dalam penulisan tugas akhir. Berikut studi literatur yang dilakukan:

   a. Studi tentang protokol NMEA pada GPS UBLOX M8
   b. Studi tentang akuisisi data serial protokol NMEA.
   c. Studi tentang akuisisi data ADC pada sensor suara analog.
   d. Studi tentang metode penentuan arah sumber suara.

   Dasar teori ini dapat diambil dari buku-buku, jurnal, *proceeding*, dan artikel-artikel di internet.
2. Perancangan Sistem

   Setelah mempelajari literatur yang ada, selanjutnya akan dilakukan perancangan sistem. Sistem yang akan dirancang meliputi integrasi modul-modul sensor, *driver* motor, dan telemetri.
3. Pengujian Sistem

   Pengujian dilakukan pada *software* dan *hardware* yang telah dibuat untuk menguji keakuratan alat dengan mencoba menjalankannya pada lapangan terbuka, disertai memberikan beberapa lokasi titik koordinat kerja *mobile robot* dan titik koordinat *power station* yang terdapat sumber suara.
4. Pengolahan Data

   Data yang telah diperoleh dari komputer *server*, modul sensor kompas, sensor GPS, dan sensor suara diakuisisi oleh mikrokontroler pada minimum sistem mikrokontroler. Data koordinat *waypoint* diolah untuk digunakan sebagai acuan tujuan, data dari sensor kompas dan GPS diolah sebagai navigasi menuju tujuan, dan data dari sensor suara analog agar dapat menentukan sumber suara sehingga *mobile robot* dapat mengarah ke sumber suara
5. Penulisan Laporan Tugas Akhir

   Tahap penulisan laporan tugas akhir dilakukan pada saat pengujian sistem dimulai dan setelahnya.

1.6. Sistematika Penulisan

Laporan Tugas Akhir ini terdiri dari lima bab dengan sistematika penulisan sebagai berikut:

- Bab 1: Pendahuluan

  Bab ini meliputi latar belakang, perumusan masalah, tujuan penelitian, batasan masalah, metodologi, sistematika penulisan, dan relevansi.
- Bab 2: Dasar Teori

  Bab ini menjelaskan tentang dasar-dasar teori tentang sensor suara analog, metode penentuan sumber suara, kompas digital, GPS, protokol NMEA yang digunakan untuk GPS, sistem navigasi, papan mikrokontroler.
- Bab 3: Perancangan Alat

  Bab ini menjelaskan tentang perancangan perangkat keras dan perangkat lunak untuk pengaplikasian penelitian.
- Bab 4: Pengujian Alat

  Bab ini menjelaskan tentang hasil yang didapat dalam pengujian dari tiap blok sistem dan subsistem serta hasil evaluasi sistem tersebut.
- Bab 5: Penutup

  Bab ini menjelaskan tentang kesimpulan meliputi hasil yang dicapai, dan kekurangan-kekurangan pada kerja alat dari hasil analisa serta saran untuk pengembangan ke depan.

1.7. Relevansi

Hasil dari tugas akhir ini diharapkan dapat memberikan manfaat sebagai berikut:

1. Dapat mendukung penelitian khususnya pada bidang studi elektronika tentang penggunaan sensor suara dalam aplikasi pengukuran dan penelitian selanjutnya.
2. Mendukung penelitian tentang robotika dengan navigasi otomatis yang dapat dimanfaatkan untuk keperluan lebih luas.
3. Dihasilkan aplikasi metode pengisian ulang baterai pada *mobile robot* yang bekerja secara *unmanned.*

BAB II
TEORI PENUNJANG

2.1. *Mobile Ro*bot

Robot adalah perangkat mekanik yang dapat dikendalikan oleh perangkat lunak yang menggunakan sensor untuk memandu satu atau lebih efektor melalui gerakan terprogram dalam suatu ruang kerja dalam hal untuk manipulasi obyek fisik [3]. Salah satu jenis robot adalah *mobile robot*. *Mobile robot* adalah robot yang memiliki mekanisme penggerak berupa roda (*wheel*) dan atau kaki (*leg*), untuk dapat berpindah tempat dari suatu tempat ke tempat yang lain. *Mobile robot* digunakan secara luas sebagai aktuator dalam berbagai bidang aplikasi dengan dilengkapi sensor untuk mengendalikan pergerakan robot tersebut [4].

Mobile robot diklasifikasikan menjadi dua yaitu menurut lingkungan tempat robot tersebut bekerja dan alat yang digunakan untuk bergerak. Berdasarkan lingkungan tempat robot tersebut bekerja, *mobile robot* terbagi menjadi empat macam:

- Robot yang bekerja di atas permukaan tanah (*land robot*),
- Robot udara yang biasa disebut *unmanned aerial vehicle* (UAV),
- *Autonomous underwater vehicles* (AUVs),
- Robot yang bekerja pada lingkungan kutub, dimana robot tersebut berkerja pada kondisi permukaan tanah yang dilapisi es (*polar robots*).

Sedangkan berdasarkan alat yang digunakan untuk bergerak, robot mobil terbagi menjadi robot berlengan atau berkaki dan robot beroda. Untuk robot berlengan, lengan atau kaki berbentuk menyerupai manusia (android) ataupun hewan, robot beroda atau *Wheeled Mobile Robot* (WMR).

Gambar 2.1 Contoh *mobile robot* beroda [5]

2.2. Sistem *Autodocking*

Salah satu contoh penggunaan *mobile robot* yaitu robot navigasi yang dapat difungsikan dalam berbagai hal, antara lain memetakan lokasi dan memonitoring kondisi lingkungan. *Mobile robot* navigasi ini memiliki komponen utama sebagai penentuan titik geraknya robot yaitu *Global Positioning System* (GPS).

Dalam perkembangan dunia robotika, *mobile robot* navigasi dirancang secara autonomous yang terus berjalan dan bekerja tanpa ada campur tangan manusia, kecuali robot mengalami kerusakan. Dalam siklus kerjanya tersebut, kebutuhan power suplai dari baterai agar tetap dalam level yang normal merupakan hal penting. Sistem *autodocking* merupakan sistem kemandirian robot agar mampu kembali ke *docking station* atau *power station* sehingga dapat mengisi ulang baterai.

Integrasi antara sensor pada *mobile robot* dengan *docking station* akan sangat dibutuhkan agar robot mampu kembali ke posisi yang tepat dan memposisikan badan robot dengan sesuai. Dibutuhkan desain *docking station* yang memiliki pemancar suara dengan frekuensi tertentu agar sensor suara dapat mendeteksinya. Penempatan sumber suara itu juga harus ditempatkan pada posisi yang sesuai dengan *transmitter* pada *power station* dan *receiver* pada *mobile robot* sehingga pengisian ulang baterai dapat optimal.

2.3. Navigasi *Waypoint*

Navigasi merupakan teknik untuk membaca kedudukan (posisi) dan arah benda terhadap kondisi di sekitarnya. Pada umumnya menemukan keduduan dan posisi dari benda dapat dilakukan dengan melihat beberapa penanda dengan sensor [6]. Metode umum untuk mewujudkan navigasi otomatis dapat melalui navigasi visual, model peta navigasi dan navigasi *waypoint* [7]. Metode inilah yang akan digunakan sebagai kemampuan kemampuan robot untuk berjalan secara otomatis menuju suatu titik tujuan pada bidang x-y tanpa bantuan manusia yang juga disebut sebagai *autonavigation robot.*

Koordinat yang digunakan dapat bervariasi tergantung pada aplikasi. Misalnya yang berkenan dengan navigasi pada permukaan bumi, koordinat ini dapat mencakup *lattitude* dan *longitude* [8]. *Waypoint* memiliki cakupan yang luas dalam bidang navigasi oleh orang awam, semenjak dikembangkannya sistem navigasi lanjutan. *Waypoint* yang ditempatkan di atas permukaan bumi, biasanya didefinisikan dalam dua dimensi (*lattitude* dan *longitude*) [9]. Pada navigasi waypoint dalam dua

dimensi, penentuan posisi target terhadap suatu objek atau benda dapat dilihat pada gambar 2.2 dan gambar 2.3.

Pada gambar di atas, dapat dilihat bahwa jarak antara posisi benda dengan target dapat dihitung dengan persamaan 2.1.

$$R_{target} = \sqrt{X_{target}^2 + Y_{target}^2} \quad (2.1)$$

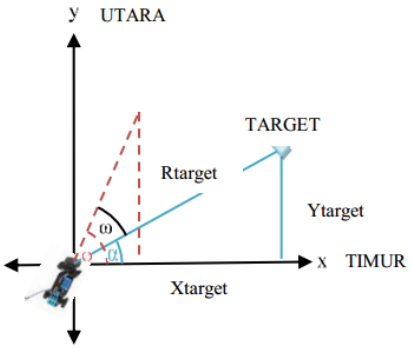
Variable Xtarget dan Ytarget merupakan koordinat lattitude dan longitude yang akan dituju. Sehingga arah yang harus dituju oleh benda (ω) dapat dihitung dengan persamaan 2.2.

$$\alpha = \tan^{-1}\left(\frac{Ytarget}{Xtarget}\right) \quad (2.2)$$

$$\omega = \alpha - \theta \quad (2.3)$$

Sudut α pada persamaan 2.3 merupakan sudut yang dibentuk setelah menentukan titik target, dan sudut θ merupakan sudut yang didapat dari kompas.

Gambar 2.2 Titik navigasi *waypoint* [8]



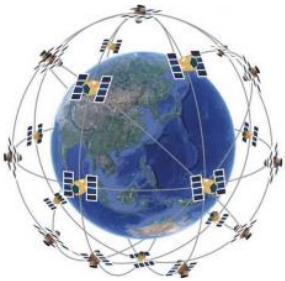
Gambar 2.3 Penggambaran titik target terhadap objek [10]

2.3.1. *Global Position System* (GPS)

Dalam Buku *Location Based Service*, GPS adalah sistem untuk menentukan letak di permukaan bumi dengan bantuan penyelarasan sinyal satelit. Secara umum GPS adalah sistem navigasi yang menggunakan satelit yang didesain agar dapat menyediakan posisi secara instan, kecepatan dan informasi waktu di hampir semua tempat di muka bumi, setiap saat dan dalam kondisi cuaca apapun. GPS memberikan informasi lokasi dengan koordinat lintang *(latitude)* dan bujur *(longitude)* berdasarkan koordinat bumi.

Untuk dapat mengetahui posisi seseorang maka diperlukan alat yang diberi nama GPS receiver yang berfungsi untuk menerima sinyal yang dikirim dari satelit GPS. Posisi diubah menjadi titik yang dikenal dengan nama *Way-point* nantinya akan berupa titik-titik koordinat lintang dan bujur dari posisi seseorang atau suatu lokasi kemudian di layar pada peta elektronik. Sejak tahun 1980, layanan GPS yang dulunya hanya untuk keperluan militer mulai terbuka untuk publik. Uniknya, walau satelit-satelit tersebut berharga sangatlah mahal, namun setiap orang dapat menggunakannya dengan gratis.

Satelit-satelit ini mengorbit pada ketinggian sekitar 12.000 mil dari permukaan bumi. Posisi ini sangat ideal karena satelit dapat menjangkau area *coverage* yang lebih luas. Satelit-satelit ini akan selalu berada posisi yang bisa menjangkau semua area di ataspermukaan bumi sehingga dapat meminimalkan terjadinya *blank spot* (area yang tidak terjangkau oleh satelit). Gambar 2.4 merupakan ilustrasi dari banyaknya satelit GPS dengan orbit masing-masing yang beredar mengelilingi bumi.



Gambar 2.4 Ilustrasi jalur orbit satelit GPS [11]

Setiap satelit mampu mengelilingi bumi hanya dalam waktu 12 jam. Sangat cepat, sehingga mereka selalu bisa menjangkau dimana pun posisi semua orang dari atas permukaan bumi. GPS *receiver* sendiri berisi beberapa *integrated circuit* (IC) sehingga murah dan teknologinya mudah untuk digunakan oleh semua orang. GPS dapat digunakan untuk berbagai kepentingan, misalnya untuk mengetahui lokasi sebuah mobil, kapal, pesawat terbang, pemetaan pertanian, dan diintegrasikan dengan mudah menggunakan komputer maupun laptop.

Format data keluaran GPS ditetapkan oleh *National Maritime Electronic Association* (NMEA) ada lima jenis, yaitu NMEA 0180, NMEA 0812, NMEA 0813, AVIATON, dan PLOTTING. Format data tersebut dapat dibaca oleh komputer melalui komunikasi *serial*. Data keluaran dalam format NMEA 0183 berupa kalimat (*string*) yang merupakan karakter ASCII 8 bit. Setiap awal kalimat diawali dengan karakter ”$”, dua karakter *Talker* ID, tiga karakter *Sentence* ID, dan diikuti oleh data fields yang masing-masing dipisahkan oleh koma (,) serta diakhiri oleh *optional checksum* dan karakter *carriage return/line feed* (CR/LF).

Sebuah GPS mempunyai *Talker* ID berupa GP. Jenis kalimat yang dihasilkan ada beberapa macam, diantaranya adalah kalimat *Recommended Minimum Specific* (RMC), dan *Global Positioning Fix Data* (GGA) dapat dilihat pada tabel 2.1. Bentuk format umum untuk data keluaran dari GPS sebagai berikut:

$<Alamat>,<Data>*<Checksum>,<CR><LF>

$: Karakter Awal

<alamat> : Bagian alamat. “aa” adalah *talker identifier*, “ccc” identitas tipe.
, : Bagian *delimiter* atau pembatas
<Data> : Blok data
* : *Checksum Delimiter*
<Checksum> : Bagian Checksum
<CR><LF> : Akhir dari barisan data (*Carriage Return, Line Feed*)

Tabel 2.1 Format kalimat ID data GPS

$GNGGA	Waktu, posisi, dan perbaikan data yang terkait penerima
$GNGLL	Waktu, posisi, dan status perbaikan
$GNGSA	Digunakan untuk merepresentasikan posisi yang tetap ketika kedua satelit dan Beidou
$GPGSA	Digunakan dalam solusi posisi. Kalimat $GNGSA digunakan untuk satelit GPS, dan $GNGSA yang lain digunakan untuk satelit Beidou
$BDGSA	Ketika hanya satelit GPS yang digunakan untuk penetapan posisi, $GPGSA sebagai keluaran. Jika satelit Beidou, maka $BDGSA sebagai keluaran
$GPGSV	Informasi dari satelit terkait azimuth, elevasi, dan CNR, $GPGSA digunakan untuk satelit GPS
$BDGSV	Dan $BDGSV untuk satelit Beidou
$GNRMC	Waktu, tanggal, dan kecepatan perjalanan
$GNVTG	Perjalanan dan kecepatan relatif terhadap tanah
$GNZDA	UTC, hari, tanggal, dan tahun

Format NMEA juga mendukung pesan lainnya, yang pertama *Global Positioning Fix Data* (GGA) waktu, posisi, dan tetapan penerima data GPS. Format ini dijelaskan pada tabel 2.2. Yang kedua adalah *Recommended Minimum Specific GNSS Data* (RMC). Data waktu, tanggal, posisi, perjalanan dan kecepatan diperoleh dari penerima GNSS. Format ini dijelaskan pada tabel 2.3.

Format data GGA sebagai berikut:

$--GGA,hhmmss.ss,llll.lll,a,yyyyy.yy,a,x,uu,xxxxxx,,,v*hh<CR><LF>

Tabel 2.2 Format pesan GGA

Bagian	Nama	Deskripsi
hhmmss.ss	Waktu UTC	Posisi UTC dalam format hhmmss.ss (000000.00 ~235959.99)
llll.lll	Latitude	Format latitude dalam ddmm.mmmm (derajat, menit.menit)
A	Indikator	N = Utara, S = Selatan
yyyyy.yyy	Longitude	Format longitude dddmm.mmmm (derajat, menit.menit)
A	Indikator	E = Timur, W = Barat
x	Indikator kualitas GPS	0 : Belum terkunci tepat
		1 : Posisi valid, mode SPS
		2 : Posisi valid, mode DGPS
uu	Penggunaan satelit	Jumlah satelit yang digunakan (0~24)
v.v	HDOP	Horizontal Delution of Precission (00.0~99.9)
w.w	Altitude	Ketinggian dari permukaan laut
x.x	Geoidal Separation	Meter
zzzz	ID Stasiun DGPS	0000~1023, null jika tidak ada
hh	Checksum	

Format data RMC sebagai berikut:

$--RMC,hhmmss.ss,x,llll.ll,a,yyyyy.yy,a,x.x,u.u,xxxxxx,,,v*hh<CR><LF>

2.3.2. *Global Positioning System* (GPS) Ublox Neo M8N

Ublox M8 merupakan modul GPS dengan satelit GNSS dan memiliki fitur lengkap yang tersedia dalam standar industri, mudah untuk diintegrasikan dan dikombinasikan dengan fleksibilitas daya, desain dan pilihan. Bentuk fisik dari modul GPS ini ditunjukkan pada gambar 2.5. Sumber catu daya Neo M8N dapat menggunakan 3,3 volt atau dapat menggunakan 5 volt.

Modul GPS Neo M8N ini menggunakan komunikasi *Universal Asynchronous Receiver Transmitter* (UART) Transistor-Transistor Logic (TTL) Rx/Tx. Dengan *baudrate* awal 4800 dan dapat diatur dalam

konfigurasi. Pada jalur komunikasi Rx/Tx GPS Neo M8N perlu diberi pengaman, untuk menghindari interferensi elektromagnet. Karena jalur sinyal I/O dengan panjang melebihi 3 mm dapat menjadi sebuah antena dan mungkin membawa sinyal radio frekuensi menjadi derau pada GPS. Untuk menghalau interferensi sinyal, direkomendasikan menggunakan resistor > 20 Ω, *ferrite beads* seperti BLM15HD102SN1, atau induktor seperti LQG15HS47NJ02 yang dipasang secara seri pada jalur I/O seperti yang ditunjukkan pada gambar 2.6. Spesifikasi dari modul GPS Ublox M8 ini ditunjukkan pada tabel 2.4.

Tabel 2.3 Format pesan RMC

Bagian	Nama	Deskripsi
hhmmss.ss	Waktu UTC	Waktu UTC dengan format hhmmss.ss (00000.00~235959.99)
x	Status	V = Peringatan penerima navigasi
		A = Data valid
llll.lll	Latitude	Format latitud dalam ddmm.mmmm (derajat, menit.menit)
a	Indikator	N = Utara, S = Selatan
yyyyy.yyy	Longitude	Format longitude dddmm.mmmm (derajat, menit.menit)
a	Indikatot	E = Timur, W = Barat
x.x	Kecepatan Permukaan	Knot (000.0~999.9)
u.u	Perjalanan Permukaan	Sudut perjalanan (000.00~359.99)
xxxxxx	Tanggal UTC	Posisi tanggal UTC ddmmyy
v	Indikator Mode	N = Data tidak valid
		A = Mode Autonomous
		D = Mode Diferensial
		E = Estimasi
hh	Checksum	

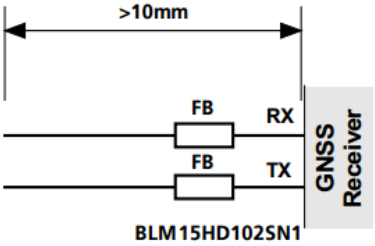
Konfigurasi dari pin pada gambar 2.5 adalah sebagai berikut:

VCC : Pin supply +5 Volt
Rx : Pin data serial penerima (*Receiver*).
Tx : Pin data serial pengirim (*Transmitter*)
GND : Pin supply 0 Volt.

Gambar 2.5 Modul GPS Ublox Neo-M8N [12]

Tabel 2.4 Spesifikasi GPS Ublox Neo M8

Parameter	Spesifikasi
Receiver type	72 *Channels* GPS L1 *frequency*, C/A Code SBAS: WAAS, EGNOS, MSAS, GLONASS, BEIDOU, QZSS
Time-To-First-Fix	*Cold start* : 26 s *Warm start* : 4 s *Hot Start* : 1 s
Sensitivity	*Tracking & Navigation* : -167 dBm *Reacquisition* : -160 dBm *Cold Start (without aiding)* : -148 dBm *Hot Start* : -156 dBm
Maximum Navigation update rate	5Hz
Horizontal position accuracy	GPS : 2.5 m SBAS : 2.0 m
Heading accuracy	0.3 *degrees*
Operational Limits	*Dynamics* : ≤ 4 g *Altitude* : 50,000 m *Velocity* : 500 m/s



Gambar 2.6 Perlindungan terhadap interferensi radio frekuensi [12]

Format protokol data yang digunakan oleh modul ini adalah NMEA0183. Pada protokol NMEA, data *latitude* dan *longitude* menggunakan format sumbu derajat, menit, dan detik. Untuk itu diperlukan konversi data dari nilai derajat menjadi radian yang ditunjukkan pada persamaan 2.4 hingga 2.5.

Konversi dari ddmm.mmm ke dd.dddd:

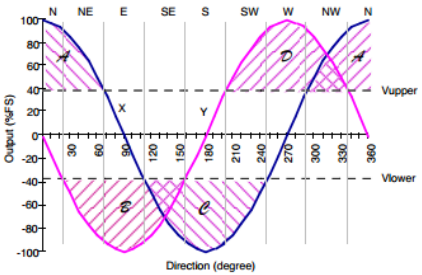
$$0.\,dddd = mm.\,mmmm/60 \quad (2.4)$$

$$dd.\,dddd = dd + 0.\,dddd \quad (2.5)$$

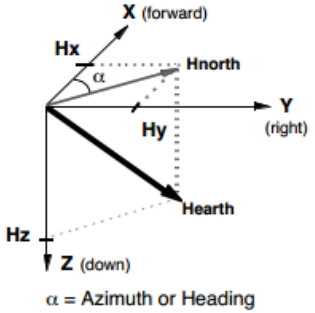
2.3.3. *Heading*

Ada dua bentuk metode kompas yang digunakan dalam sistem navigasi yang menggunakan sensor magnetoresistif adalah *the eight point compass* dan *the one-degree compass*. *Eight Point Compass* merupakan metode *heading* menggunakan penggambaran sederhana delapan titik kardinal kompas (U, S, B, T) dan titik tengah (TL, TG, BD, BL) [10].

Tipe kompas ini banyak digunakan dalam otomotif sederhana di mana pengendara ingin mengetahui arah perjalanan secara umum. Untuk aplikasi ini, sensor magnetik dapat direduksi menjadi hanya dua sumbu. Hanya menggunakan sumbu X, dan Y. Untuk menganalisa respon sensor magnetoresistif, gambar keluaran X, dan Y saat berjalan memutar yang ditunjukkan pada gambar 2.7. Dengan mengetahui medan magnet bumi selalu menunjuk arah utara, mulai analisa dengan sumbu X (perjalanan) lurus ke utara. Keluaran sumbu X akan berada pada nilai maksimumnya ketika keluaran Y berada pada 0. Saat perjalan belok searah jarum jam menuju arah timur, sumbu X akan berkurang nilainya hingga 0, dan sumbu Y akan berkurang hingga nilai keluaran pada negatif maksimal.



Gambar 2.7 Sumbu keluaran arah metode *eight point compass* [13]



Gambar 2.8 Vektor sumbu kompas [13]

Lengkungan X dan Y pada gambar 2.8 dapat disusun menjadi delapan daerah yang merepresentasikan empat titik kardinal dan titik tengah kompas.

Yang kedua adalah *One Degree Compass.*Yaitu metode *heading* yang mencapai ketelitian 1º. Kompas membutuhkan sensor magnet yang dapat secara nyata memperbaiki perubahan *angular* hingga 0.1º. Sensor juga harus menunjukkan histerisis yang rendah (<0.05%FS), linearitas derajat yang tinggi (<0.5%FS *error*) dan dapat dipakai berulang-ulang (konsisten). Medan magnet pada bidang X dan Y biasanya dapat mencapai sekitar 200 hingga 300 miligauss pada bagian ekuator dan kurang pada bagian sumbu. Sehingga berlaku hubungan pada persamaan 2.6. Secara vektor ditunjukkan pada gambar 2.8.

$$Azimuth = tan^{-1}\frac{y}{x} \tag{2.6}$$

2.3.4. Kompas Digital HMC5883L

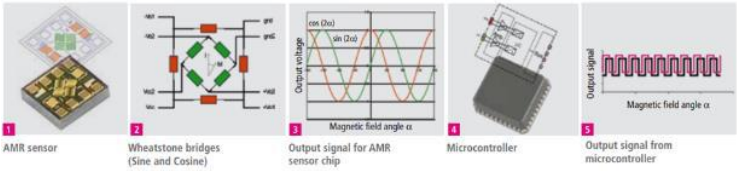
HMC5883L merupakan jenis sensor magnetoresistif yang dapat mengukur medan magnet hingga mencapai 2 mili-Gauss pada resolusi medan +/- 8 Gauss. Saat sumber daya dipasang, sensor akan merubah tiap pengaruh medan magnet dalam arah sumbu yang peka menjadi perbedaan tegangan. Sensor magnetoresistif dibuat dari bahan film tipis *nickel-iron* (*Permalloy*) yang bermotif sebagai elemen jalur resistif. Adanya medan magnet mengubah elemen jembatan resistif yang menyebabkan perubahan tegangan sesuai tegangan yang melintasi keluaran jembatan [16].

Sumber arus internal menghasilkan arus DC sebesar 10 mA dari sumber VDD. Arus DC ini yang diterapkan menjadi ikatan seimbang pada sensor magnetoresistif, yang mana menghasilkan bias medan magnet buatan sensor.

Perbedaan pengukuran dan keadaan medan ini diambil untuk menjadi tiga sumbu register keluaran X, Y, dan Z. Proses ini ditunjukkan pada gambar 2.9. Spesifikasi dari sensor ini ditunjukkan pada tabel 2.5

Pin dan bentuk fisik dari modul kompas digital HMC5883L ini ditunjukkan pada gambar 2.10. Konfigurasi pin dari kompas digital dari gambar 2.10 adalah sebagai berikut:

1. Vcc : Pin supply +5 Volt.
2. GND : Pin *ground*.
3. SDA : Pin masukan data SDA.
4. SCL : Pin masukan *clock* SCL.
5. DRDY : *Not Connected*.



Gambar 2.9 Proses menghasilkan data sumbu HMC5883L [14]

Gambar 2.10 Pin kompas digital HMC5883L [15]

Tabel 2.5 Spesifikasi Sensor Kompas HMC5883L

Karakteristik	Kondisi	Min	Typ	Max	Unit
Supply	VDD Referensi pada AGND	2.6	2.5	3.6	Volts
	VDD Referensi pada DGND	1.71	1.8	VDD+0.1	Volts
Arus Rata-rata	Idle	-	2	-	µA
	Pengukuran	-	100	-	µA
Jangkauan Medan	Skala penuh	-8		8	gauss
Sensitivitas	VDD=3V, GN=0 sampai 7, 12-bit ADC	230		1370	LSb/gauss
Resolusi	VDD=3V, GN=0 sampai 7, 12-bit ADC	0.73		4.35	mili-gauss
Derau dasar			2		mili-gauss
Linieritas	2.0 gauss jangkauan masukan8-bit alamat baca			0.1	%FS
Alamat I2C	8-bit alamat tulis		0x3 D		hex
			0x3 C		hex
I2C rate				400	kHz

2.4. Sensor Suara

Sensor suara adalah sebuah alat yang mampu mengubah gelombang Sinusoidal suara menjadi gelombang sinus energi listrik. Sensor suara bekerja berdasarkan besar/kecilnya kekuatan gelombang suara yang mengenai membran sensoryang sering disebut dengan microphone. Hal tersebut menyebabkan bergeraknya membran sensor (*microphone*) yang juga terdapat sebuah kumparan kecil di balik membran sensor bergerak naik dan turun. Kecepatan gerak kumparan menentukan kuat-lemahnya gelombang listrik yang dihasilkannya [16].

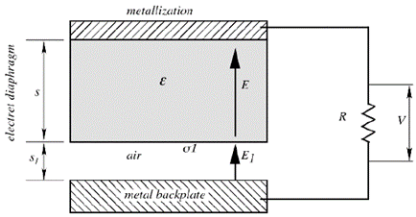
2.4.1. *Electret Microphone*

Microphone (mikrofon) adalah suatu alat atau komponen elektronika yang dapat mengubah atau mengkonversikan energi akustik (gelombang suara) ke energi listrik (sinyal audio). *Microphone* merupakan keluarga transduser yang berfungsi sebagai komponen atau alat pengubah satu bentuk energi ke bentuk energi lainnya. Setiap jenis Mikrofon memiliki cara yang berbeda dalam mengubah bentuk energinya, tetapi mereka semua memiliki persamaan yaitu semua jenis *microphone* memiliki suatu bagian utama yang disebut dengan diafragma. Jenis diafragma yang dipakai sangat beragam, salah satunya adalah material elektret atau yang sering disebut *electret microphone*.

Electret merupakan kerabat dekat bahan piezoelektrik sehingga bahan electret dikembangkan dengan metode piezoelektrik. *Electret* adalah bahan dielektrik kristal terpolarisasi elektrik permanen. Aplikasi pertama dari elektret ke mikrofon dan earphone dimana dijelaskan pada tahun 1928. Sebuah mikrofon electret adalah transduser elektrostatik yang terdiri dari electret metal dan *backplate* yang dipisahkan dari diafragma oleh celah udara dengan struktur seperti gambar 2.11.

Metallization bagian atas dan backplate logam dihubungkan melalui resistor R dengan V yang melaluinya, dimana ia dapat diamplifikasi dan digunakan sebagai sinyal output. Karena *electret* adalah dielektrik yang terpolarisasi secara permanen, kerapatan muatan σ1 pada permukaannya konstan dan menetapkan medan listrik E1 di celah udara. Bila gelombang akustik menempel pada diafragma, yang terakhir membelok ke bawah, mengurangi ketebalan celah udara s1 untuk nilai s. Pada kondisi sirkuit terbuka, amplitudo bagian variabel dari tegangan keluaran dapat dicari dengan persamaan 2.7.

$$V = \frac{s\,\Delta s}{\varepsilon_0(s+\varepsilon s_1)} \tag{2.7}$$



Gambar 2.11 Struktur umum diafragma e*lectret microphone* [16]

Mikrophone dengan diafragma electret memiliki fitur yang lebih sering digunakan daripada jenis mikrofon lainnya. Keaunggulan yang ditonjolkan diantaranya rentang frekuensinya sangat lebar mulai dari 10^{-3} Hz hingga ratusan megahertz. Mereka juga menampilkan respons frekuensi datar (dalam ± 1 dB), distorsi harmonis rendah, sensitivitas getaran rendah, respon impuls yang baik, dan ketidakpekaan terhadap medan magnet. Sensitivitas mikrofon electret berada dalam kisaran beberapa milimeter per microbar [16].

2.5. Sensor Ultrasonik

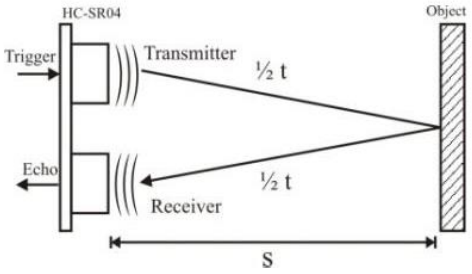
Sensor ultrasonik adalah komponen yang mampu mengubah energi listrik menjadi energi mekanik berupa gelombang ultrasonik. Sensor ultrasonik ini bekerja berdasarkan prinsip pantulan gelombang ultrasonik dan digunakan untuk mendeteksi keberadaan suatu objek dengan jarak tertentu di depannya, frekuensi kerjanya pada daerah di atas gelombang suara dari 40 KHz hingga 400 KHz. Sensor ini terdiri dari rangkaian pemancar ultrasonik yang dinamakan transmitter dan penerima yang disebut receiver seperti terlihat pada gambar 2.11.

Prinsip pengukuran jarak menggunakan sensor ultrasonik HC-SR04 dengan spesifikasi pada tabel 2.6 berawal ketika pulsa trigger diberikan pada sensor, transmitter akan mulai memancarkan gelombang ultrasonik, pada saat yang sama sensor akan menghasilkan output TTL rsing edge menandakan sensor mulai menghitung waktu pengukuran, setelah receiver menerima gelombang pantulan maka pengukuran waktu akan dihentikan dengan menghasilkan output TTL falling edge. Jika waktu pengukuran adalah t dan kecepatan suara adalah 344 m/s, maka jarak antara sensor ultrasonik dengan objek dapat dihitung dengan menggunakan persamaan berikut.

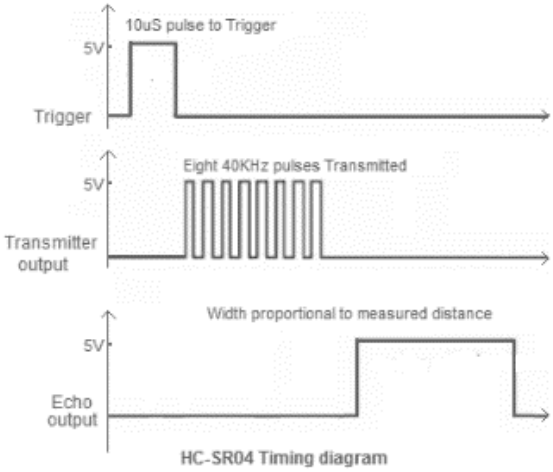
$$s = t \times 344 / 2 \tag{2.8}$$

Tabel 2.6 Spesifikasi sensor ultrasonik HC-SR04

Tegangan Suplai	5 VDC
Arus Kerja	15 mA
Sudut Efektif	<15º
Rentang Jarak	2 cm – 400 cm
Resolusi	0,3 cm



Gambar 2.12 Prinsip kerja sensor ultrasonik [17]



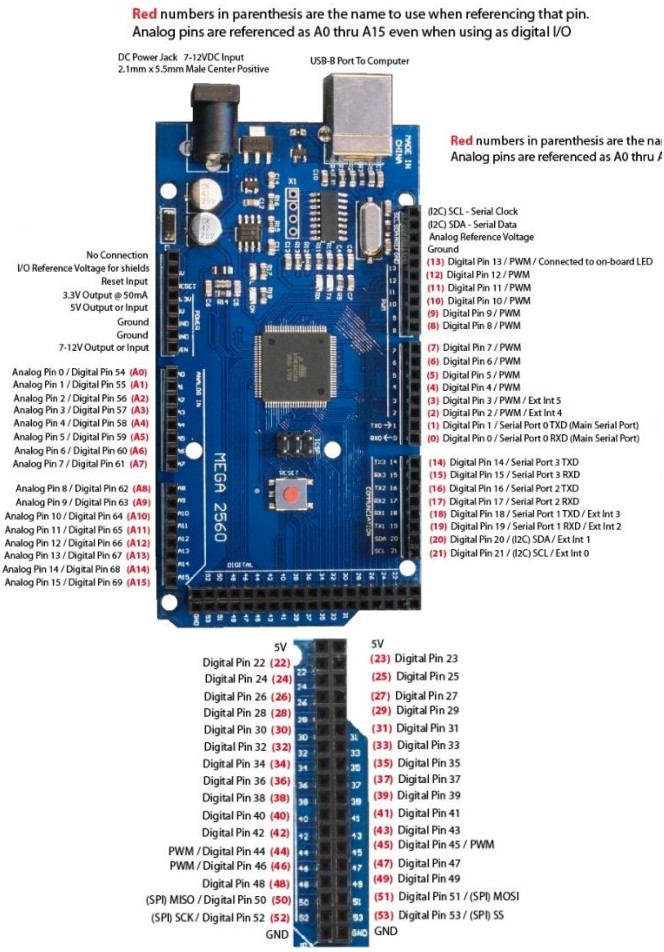
Gambar 2.13 *Timing diagram* dari sensor HC-SR04 [18]

2.6. Mikrokontroler Arduino Mega 2560

Arduino Mega 2560 adalah papan kit elektronik atau papan rangkaian elektronik *open source* yang di dalamnya terdapat komponen utama yaitu sebuah chip mikrokontroler Atmega2560 dengan jenis AVR dari perusahaan Atmel.

Mikrokontroler itu sendiri adalah chip atau IC (*integrated circuit*) yang bisa diprogram menggunakan komputer. Tujuan menanamkan program pada mikrokontroler adalah agar rangkaian elektronik dapat membaca input, memproses input tersebut dan kemudian menghasilkan output sesuai yang diinginkan. Jadi mikrokontroler bertugas sebagai otak

yang mengendalikan input, proses dan output sebuah rangkaian elektronik.



Gambar 2.14 Bentuk fisik dan *pinout* Arduino Mega 2560 [19]

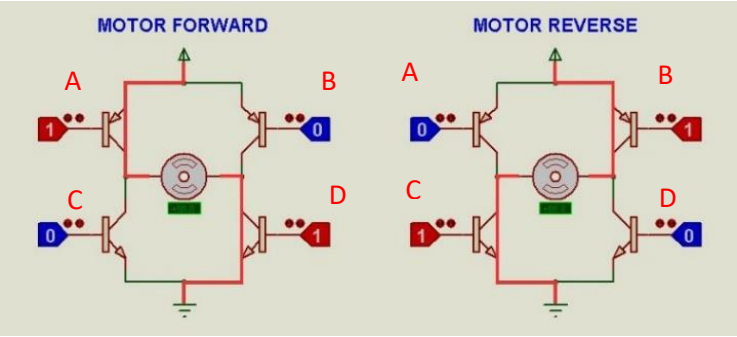
Mikrokontroler Arduino Mega 2560 ditunjukkan pada gambar 2.14 adalah papan mikrokontroler berbasiskan ATmega2560. Arduino Mega 2560 memiliki 54 pin digital *input/output*, dimana 15 pin dapat digunakan sebagai *output* PWM, 16 pin sebagai *input* analog, dan 4 pin sebagai *Universal Asynchronous Receiver Transmitter* (UART), 16 MHz kristal osilator, koneksi USB, *jack power*, *header* ICSP, dan tombol reset. ATMega 2560 sendiri memiliki 11 *port* dan memiliki jumlah total pin sebanyak 100 pin. Chip tersebut memiliki pin *Serial Data* (SDA) dan *serial Clock* (SCL), 4 komunikasi *serial* dan *pin change interupt* yang sangat memungkinkan jika digunakan untuk keperluan sistem kendali yang banyak menggunakan sensor dan jalur komunikasi *serial*.

Tegangan yang direkomendasikan untuk men-*supply* Arduino ini berkisar antara 7 Volt hingga 12 Volt. Sedangkan tegangan yang diijinkan antara 6 Volt hingga 20 Volt.

2.7. Motor *Driver*

Driver motor merupakan suatu rangkaian khusus yang memiliki fungsi untuk mengatur arah ataupun kecepatan pada motor DC. Perlunya rangkaian driver motor ini dikarenakan pada umumnya suatu motor DC membutuhkan arus lebih dari 250 mA untuk beberapa IC contohnya NE555, ATMEGA 16 dan IC seri 74 tidak bisa memberikan arus lebih dari nilai tersebut. Jika motor langsung dihubungkan ke IC, maka hal ini akan menyebabkan kerusakan pada IC tersebut.

2.7.1. Prinsip Kerja *Driver* Motor



Gambar 2.15 Prinsip motor *driver* H-*bridge* [20]

Bentuk rangkaian *driver* motor yang umum digunakan yaitu H-*Bridge*. Berbentuk seperti huruf H yang memiliki perbedaan fungsi di setiap sisinya. Prinsip sederhana dari pergerakan rangkaian *driver* motor DC ini ditunjukkan pada gambar 2.15.

Pada gambar di atas *driver* motor yang digunakan yaitu Transistor Bipolar (BJT). A dan B merupakan trnsistor PNP, C dan D merupakan transistor NPN namun ada juga H-*bridge* yang memakai transistor NPN semuanya. Berputarnya motor juga dipengaruhi jenis transistornya dan mengkopel antar transistornya. Apabila pada gambar ini titik A dikopel dengan C, titik B dengan titik D. Motor akan bergerak *forward* atau searah jarum jam apabila transistor pada sebelah kiri atas dan kanan bawah aktif (*high*) serta transistor kiri bawah dan kanan atas tidak aktif (*low*). Pada kondisi ini kutub positif pada motor DC mendapatkan tegangan sumber dan kutub negatifnya terhubung dengan *ground* sehingga ada perbedaan potensial yang menyebabkan motor berputar. Untuk pergerakan berlawanan jarum jam (*reverse*) kebalikan dari seluruh kondisi pada keadaan *forward*. Jangan sekali-kali mencoba untuk mengaktifkan seluruh transistor pada bagian kiri saja atau kanan saja, hal ini dapat menyebabkan hubung singkat atau *short circuit* yang dapat berakibat rusaknya komponen.

2.7.2. Motor *Driver* L298N

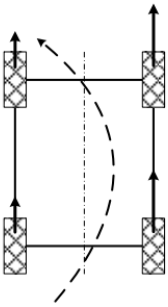
Driver motor L298N merupakan *driver* motor yang cukup populer digunakan dalam pengaturan kecepatan dan arah putar motor DC. Kelebihan dari *driver* motor L298N ini adalah cukup presisi dalam mengontrol motor. Selain itu, kelebihan *driver* motor L298N adalah mudah untuk dikontrol. Untuk mengontrol *driver* L298N ini dibutuhkan 6 buah pin mikrokontroler. Dua buah untuk pin Enable (satu buah untuk motor pertama dan satu buah yang lain untuk motor kedua, karena *driver* L298N ini dapat mengontrol dua buah motor DC), 4 buah untuk mengatur kecepatan motor-motor tersebut. Output dari rangkaian ini sudah berupa dua pin untuk masing masing motor. Pada prinsipnya rangkaian *driver* motor L298N ini dapat mengatur tegangan dan arus sehingga kecepatan dan arah motor dapat diatur. Untuk rangkaian modul motor *driver* L298N dengan voltage regulator 5V guna menstabilkan tegangan input yang umum digunakan ditunjukkan pada gambar 2.16.

Gambar 2.16 Modul motor *driver* L298N [21]

2.8. *Differential Speed Steering*

Differential speed steering merupakan metode kendali manuver kendaraan dengan memanfaatkan selisih kecepatan roda kanan dan roda kiri. Mekanisme *steering* dengan metode ini seperti yang ditunjukkan pada gambar 2.17 membuat kendaraan secara mekanik menjadi sederhana yang mana menghasilkan jalur baru untuk merealisasikan kendali arah untuk kendaraan elektrik [22].

Untuk mudahnya, dua roda pada kendaraan yang masing-masing diasumsikan berotasi dengan kecepatan yang sama. Untuk *differential speed steering*, arah belokan ditentukan oleh sisi yang mempunyai kecepatan lebih tinggi dari lainnya. Dapat didefinisikan dengan kendaraan akan berbelok ke kiri jika $\Delta\omega > 0$ dan akan berbelok ke kanan jika $\Delta\omega < 0$. Untuk mendapatkan selisih perbedaan kecepatan, strategi kendali dapat menggunakan tiga cara. Yang pertama adalah menambah kecepatan roda terluar, yang kedua mengurangi kecepatan roda terdalam, dan yang ketiga adalah menambah kecepatan pada roda terluar dan mengurangi kecepatan roda terdalam. Persamaan kecepatan sudut yang dihasilkan dari perbedaan kecepatan roda ditunjukkan pada persamaan 2.8 dan 2.9.



Gambar 2.17 Prinsip *Differential Speed Steering* [22]

$$\omega_r(t) = \frac{\omega_r(t)}{r} \quad (2.8)$$

$$\omega_l(t) = \frac{\omega_l(t)}{r} \quad (2.9)$$

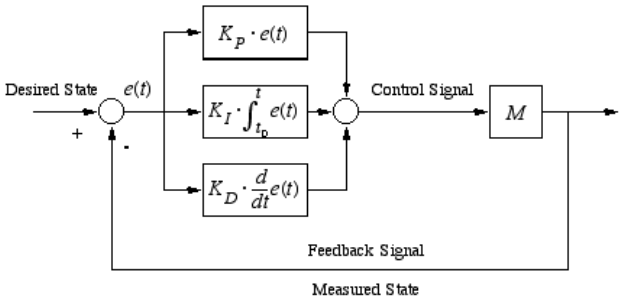
2.9. Kendali Proporsional Integral Derivatif

Kendali proporsional integral derivatif adalah suatu sistem kendali yang merupakan gabungan dari kendali proporsional, kendali integral dan kendali derivatif dimana kendali proporsinal akan menghasilkan keluaran kendali yang sebanding dengan nilai eror sehingga diperoleh nilai *steady state* [16]. Sedangkan kendali integral berfungsi untuk meminimalisir nilai *error steady state* terhadap *setpoint*.

Untuk mengurangi osilasi atau *overshoot* respon maka ditambahkan kendali derivatif yang merupakan fungsi derivatif dari nilai *error* dikalikan dengan konstanta derivatif sehingga kendali PID ditunjukkan pada persamaan 2.10, dan diagram bloknya ditunjukkan pada gambar 2.18.

$$u(t) = P(t) = K_p e(t) + K_i \int_0^t e(\tau)d\tau + K_d \frac{de(t)}{dt} \quad (2.10)$$

Dari setiap kendali yang menyusun kendali PID mempunyai karakteristik masing-masing.



Gambar 2.18 Kendali proporsional integral derivatif [23]

2.9.1. Kendali Proporsional

Kedali proposional adalah kendali yang menghasilkan keluaran kendali yang sebanding dengan *error* masukan yang dikalikan dengan konstanta proposional. Kendali ini berfungsi untuk memperkuat sinyal *error*, sehingga mempercepat keluaran sistem mencapai titik referensi. Persamaan dari kendali proposional ini ditunjukkan pada persamaan 2.11.

$$u(t) = K_p e(t) \tag{2.11}$$

2.9.2. Kendali Integral

Kendali integral adalah kendali yang menghasilkan keluaran berbanding lurus dengan besar dan lamanya *error*. Integral dalam kendali PID adalah jumlah *error* tiap waktu dan mengakumulasi *offset* yang sebelumnya telah dikoreksi. *Error* terakumulasi dikalikan dengan *gain* integral (Ki) dan menjadi keluaran kendali. Persamaan kendali integral ini ditunjukkan pada persamaan 2.12.

$$u(t) = K_i \int_0^t e(t)dt \tag{2.12}$$

Kendali integral mempercepat perpindahan proses menuju *setpoint* dengan menghilangkan *error stady state* yang muncul pada kendali proposional. Namun, karena integral mengakumulasi error sebelumnya, maka dapat menyebabkan *overshoot*.

2.9.3. Kendali Derivatif

Kendali derifatif adalah kendali yang menghasilkan keluaran sebanding dengan nilai selisih dari masukan. Kendali derivatif digabungkan dengan kendali proposional, akan meredam nilai *overshoot* dari keluaran kendali. Persamaan kendali derivatif ditunjukkan pada persamaan 2.13.

$$u(t) = K_d \tau_d \left(\frac{d_e(t)}{dt}\right) \tag{2.13}$$

2.10. Metode Penentuan Arah Sumber Suara

Manusia dapat mengenali lokasi suara secara 3 dimensi dan bahkan dapat mengenali keberadaan sumber suara pada posisi belakang dimana tidak terdapat informasi visual. Sudah banyak dilakukan penelitian mengenai pengenalan lokasi sumber suara menggunakan

beberapa *array* mikrofon yang memakai lebih dari 3 buah mikrofon, akan tetapi penelitian sistem pengenalan lokasi sumber suara yang menggunakan 2 buah mikrofon seperti pendengaran manusia masih jarang dilakukan [25].

Penelitian pada sistem lokalisasi suara pada manusia sudah dilakukan selama beberapa dekade dan mekanisme dari lokalisasi sumber suara pada manusia sudah diketahui. Dari penelitian itu terlihat bahwa manusia menggunakan beberapa petunjuk untuk mendapatkan lokasi sumber suara yaitu membedakan level atau intensitas suara dan waktu sampai suara dari sumber ke telinga [26].

Pada penelitian tugas akhir ini akan dirancang suatu sistem untuk membedakan intensitas dan level suara dengan cara menangkap suara atau bunyi dari sumber dengan 2 buah mikrofon. Dimana intensitas suara adalah energi yang dibawa oleh gelombang bunyi per satuan waktu melalui perubahan setiap satuan luas. Intensitas dapat dibandingkan dengan amplitudo gelombang suara & memiliki satuan SI Watt/ meter^2 (W/m^2) seperti yang ditunjukkan persamaan 2.14.

$$\mathrm{I} = \frac{P}{A} \tag{2.14}$$

Diketahui bahwa telinga manusia sensitif dengan berbagai intensitas suara. Jadi skala intensitas logaritmik (β) digunakan, yang didefinisikan oleh persamaan berikut.

$$\beta = (10\ dB) log I0 \tag{2.15}$$

Pada persamaan di atas, I0 adalah intensitas kebisingan referensi yang telah ditentukan yaitu 10 - 12 W/m^2, berdasarkan ambang batas pendengaran manusia minimum 1000 Hz. Tingkat intensitas suara dinyatakan dalam desibel (dB).

----- Halaman ini sengaja dikosongkan -----

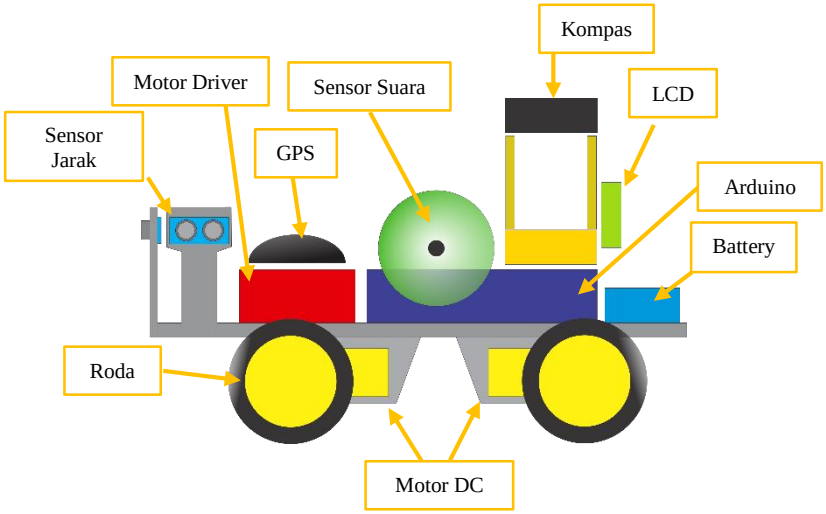
BAB III
PERANCANGAN SISTEM

Pada bab ini menjelaskan tentang perancangan sistem yang meliputi perancangan perangkat keras, arsitektur sistem, perancangan perangkat lunak, desain kendali dan persiapan robot.

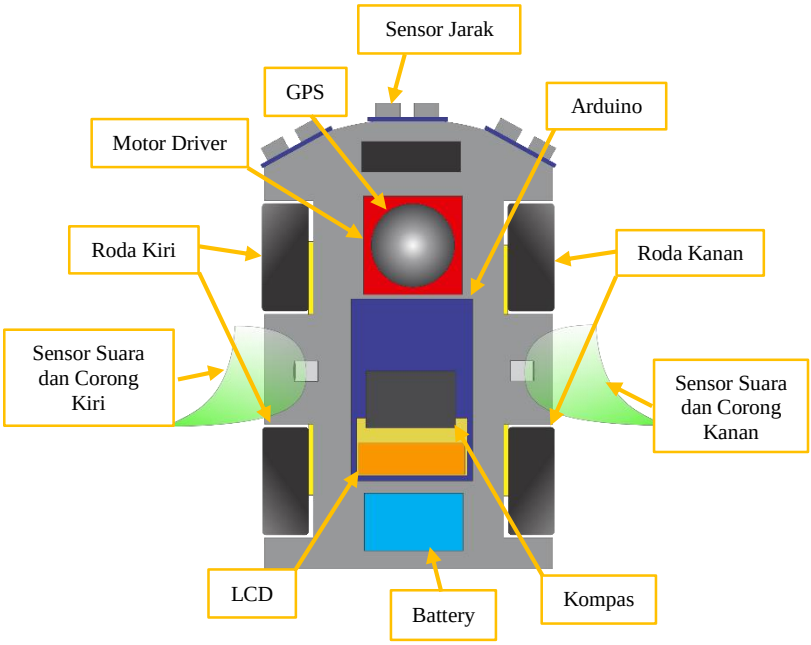
3.1 Perancangan Perangkat Keras

Desain robot ini dimulai dengan perancangan perangkat keras sistem kendali *Autonomous Navigation Robot* penggabungan fungsi beberapa komponen seperti ditunjukkan pada gambar 3.4 yaitu *driver motor*, sensor-sensor, mikrokontroler, dan aktuator. Semua komponen penyusun dirangkai sedemikian rupa seperti yang ditunjukkan pada gambar 3.1 dan gambar 3.2 sebagai desain robot keseluruhan.

Pembuatan mekanik robot menggunakan desain yang dibuat oleh penulis. Model robot yang digunakan adalah mobil robot berpenggerak motor DC 4 buah dengan masing-masing menggunakan satu roda pada tiap motor seperti konsep pada mobil 2 WD.



Gambar 3.1 Desain mekanik tampak samping



Gambar 3.2 Desain mekanik tampak atas

3.2.1. Arsitektur Sistem

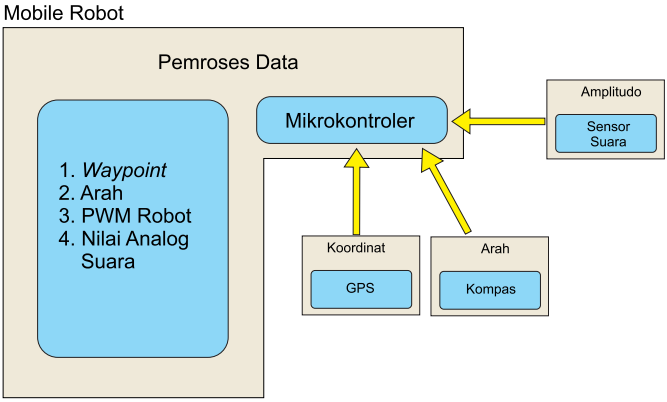
Blok diagram dari gambar 3.3 merupakan penyederhanaan dari kombinasi sistem *Autonomous Navigation Robot* menggunakan GPS dengan sistem *sound localization* sebagai Tugas Akhir ini. Perencanaan sistem ini bertujuan agar robot dapat bekerja sesuai dengan perencanaan. Yaitu dapat berjalan sesuai arah dari koordinat yang diberikan secara otomatis, mendeteksi sumber suara berada, dan menuju ke arah sumber suara tersebut.

Aplikasi dari adanya sumber suara diharapkan mampu memperbaiki toleransi dari kesalahan pembacaan koordinat GPS yang diberikan. Hal tersebut berpengaruh kepada optimalisasi pengisian baterai

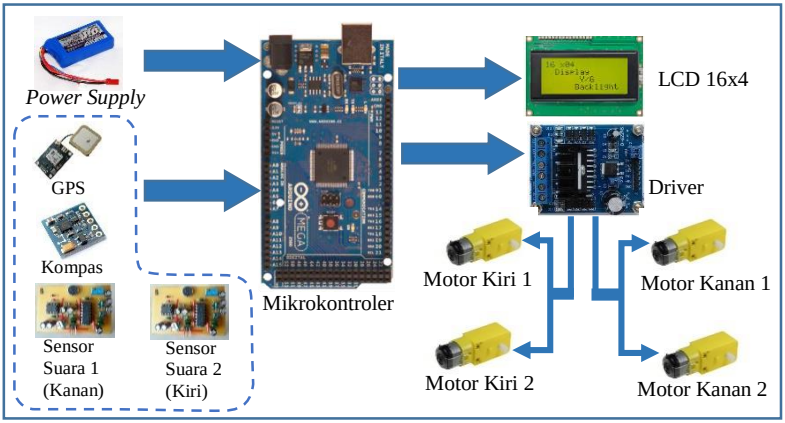
yang dilakukan secara wireless, dimana radiasi pengisian dapat terjadi pada jarak tertentu dari area *power station.*

Pada bagian robot, terdapat beberapa bagian seperti sensor, pemroses data, dan kendali. Pada bagian sensor, terdapat tiga bagian komponen yaitu GPS untuk data koordinat, kompas untuk data arah, dan sensor suara untuk mengetahui sumber suara. Pada bagian pemroses data terdapat mikrokontroler untuk mengolah data masukan dari sensor dan komputer, yang nantinya akan digunakan untuk mengendalikan bagian kendali. Di mana bagian kendali ini terdapat aktuator yang berupa motor DC.

Secara keseluruhan sistem bekerja dengan memasukkan nilai *latitude* dan *longitude* pada program Arduino sebagai *waypoint*. Ketika robot berjalan, data *waypoint* yang telah masuk, secara urut akan menjadi titik-titik tujuan yang harus dilalui. Sensor GPS secara periodik memberikan informasi data lokasi robot berada, dengan bantuan sensor kompas yang memberikan arah robot. Informasi data tersebut masuk ke dalam pemroses data, selanjutnya data tersebut digunakan sebagai acuan. Koordinat lokasi menjadi data jarak dan arah yang harus dituju antara *waypoint* dan robot, data kompas menjadi arah *heading* robot ketika menuju *waypoint*. Data kompas juga memberikan nilai selisih sudut yang digunakan kendali untuk mengatur PWM motor, agar robot dapat bermanuver kearah sudut yang harus dituju. Data lokasi juga digunakan kendali untuk memutuskan robot harus menuju ke *waypoint* ke berapa. Skematik dari sistem yang ditunjukkan pada gambar 3.4 ada di lampiran.



Gambar 3.3 Diagram blok arsitektur sistem



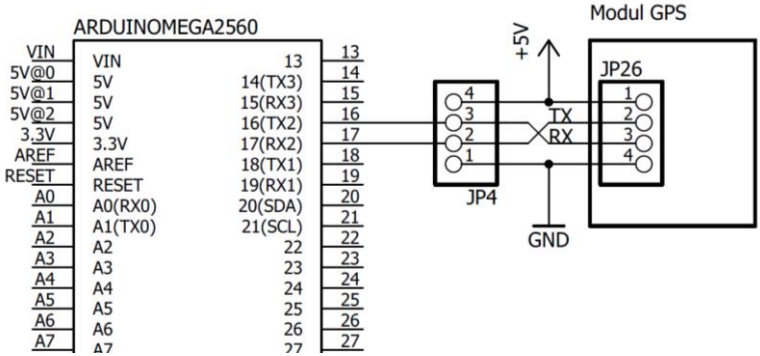
Gambar 3.4 Perancangan sistem perangkat keras

Adanya sumber suara pada *wapoint* tertentu yang telah diatur menunjukkan bahwa pada koordinat *waypoint* tersebut merupakan sebuah *power station.* Robot yang telah sampai pada titik koordinat *waypoint power station* akan mengarah ke sumber suara berdasarkan data yang diambil dari sensor suara berupa amplitudo.

3.2.2. Perancangan Sensor GPS Ublox Neo M8N

GPS Ublox M8 menggunakan komunikasi serial UART TTL yang dihubungkan pada pin serial2 arduino yaitu Tx2 dan Rx2. Dengan model *cross connection* dimana Rx GPS terhubung dengan Tx2 arduino karena pin Tx2 arduino merupakan pin *transmitter* data yang kemudian diterima oleh pin *receiver* data pada GPS, dan Tx GPS yang merupakan pin *transmiter* data GPS terhubung Rx2 arduino yang merupakan pin *receiver* pada arduino.

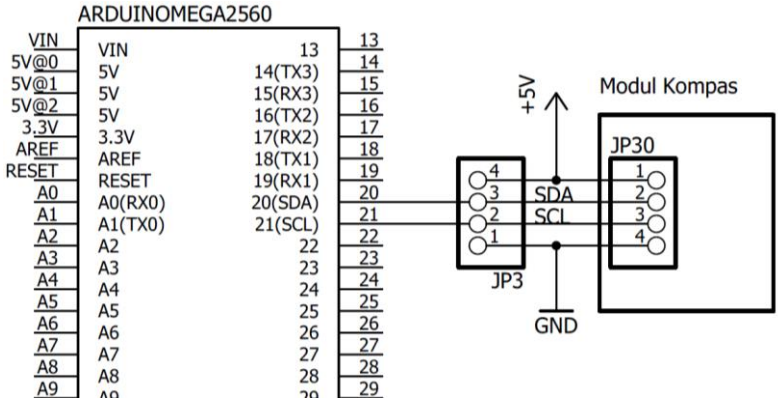
Skema jalur konfigurasi rangkaian GPS Ublox M8 dengan Arduino Mega dapat dilihat pada gambar 3.5.



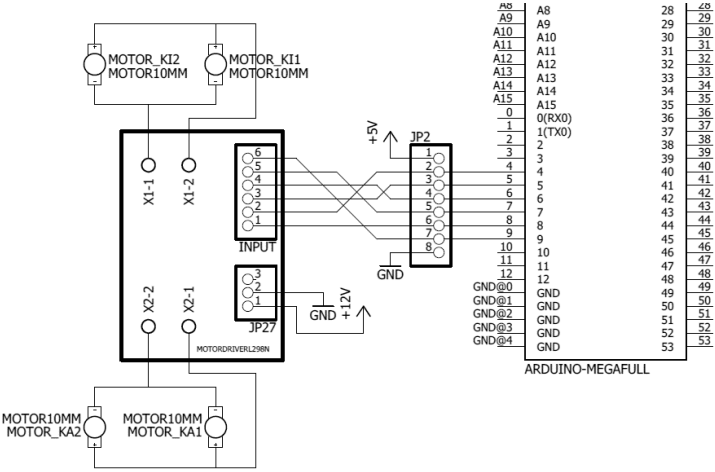
Gambar 3.5 Koneksi modul GPS dengan Arduino Mega 2560

3.2.3. Perancangan Sensor Kompas HMC5883L

Sensor kompas digunakan sebagai arah *heading* robot. Sensor ini menggunakan komunikasi *Inter-Integrated Communication* (I2C). Keunggunlan dari komunikasi I2C ini adalah dapat menampung banyak perangkat hanya dengan menggunakan dua buah kabel komunikasi. Kelemahannya adalah kecepatan transfer data yang cukup lambat, mulai dari 100 kHz hingga 400 kHz. Konfigurasi rangkaian sensor kompas HMC5883L dengan Arduino Mega ditunjukkan pada gambar 3.6.



Gambar 3.6 Konfigurasi modul sensor kompas HMC5883L



Gambar 3.7 Konfigurasi rangkaian motor *driver*

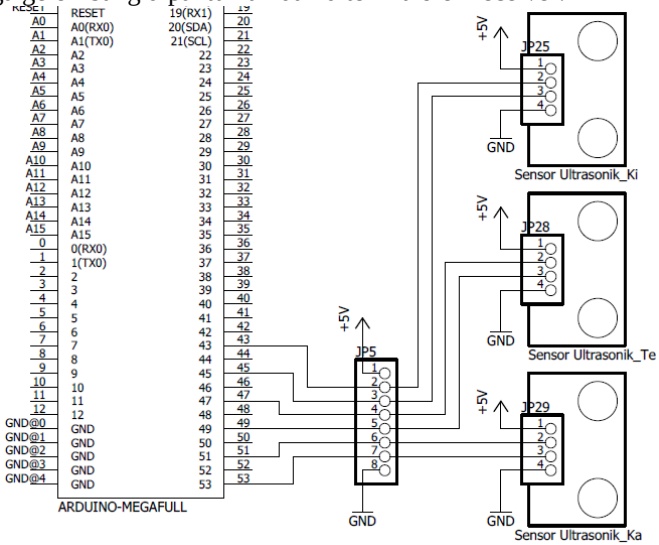
3.2.4. Perancangan *Driver* Motor L298N

Modul motor *driver* L298N dipilih untuk dipakai karena selain mudah dipakai dan digunakan sebagai pengatur arah dan kecepatan putaran motor DC juga karena pada sistem memakai 4 motor DC dengan tiap 2 motor DC *gearbox* yang diparalel. Modul motor *driver* L298N ini mampu mengontrol 2 motor DC dengan arus masing-masing 4 A sehingga mencukupi kebutuhan. Konfigurasi dari *driver* motor L298N ini dengan Arduino ditunjukkan pada gambar 3.7.

3.2.5. Perancangan Perangkat Pendeteksi Jarak

Perangkat pendeteksi jarak yang digunakan pada *mobile robot* adalah tiga sensor ultrasonik HC-SR04 yang ditempatkan pada depan robot dengan posisi di depan kiri, depan tengah, dan depan kanan robot, sedangkan posisi sudut antara satu sensor ultrasonik dengan sensor ultrasonik yang lain sebesar ±30° dengan konfigurasi pin seperti pada gambar 3.8. Hal tersebut difungsikan untuk memperbesar sudut pembacaan 3 sensor ultrasonik yang dipasang. Sensor ultrasonik dapat mengukur dapat mengukur jarak dengan cara menghitung waktu dari

gelombang ultrasonik sebesar 40kHz yang dikirim oleh transmitter hingga gelombang dipantulkan dan diterima oleh receiver.



Gambar 3.8 Konfiguarsi rangkaian sensor ultrasonik

Namun ada batasan yang dimiliki oleh sensor HC-SR04 dimana sudut efektif pengukuran maksimum adalah 30° dari posisi tegak lurus dengan objek. Apabila lebih dari sudut tersebut maka pembacaan sensor akan mendapatkan banyak error sehingga kurang akurat pembacaannya.

3.2.6. Perancangan Sensor Akustik (Sensor Suara)

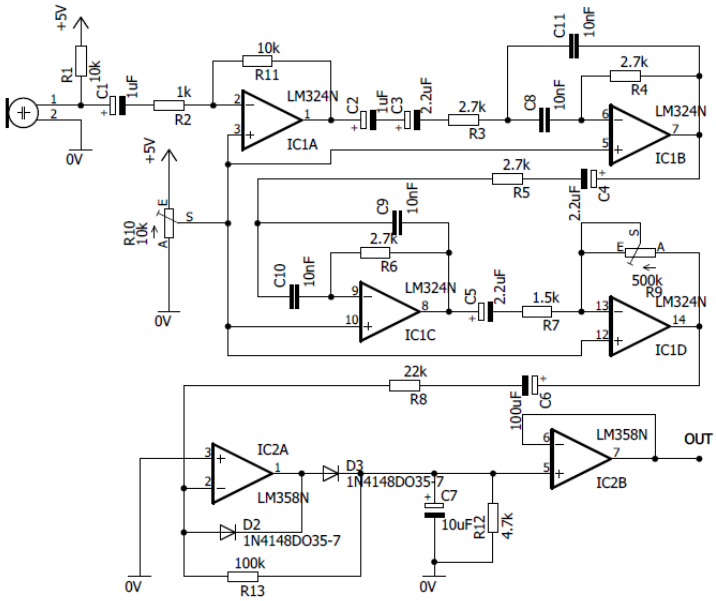
Perangkat penerima merupakan rangkaian pendeteksi suara dengan komponen utama mikrofon elektret (kondenser) yang dilengkapi dengan Band Pass Filter (BPF), rangkaian penguat, dan envelope detector untuk menghitung level dari suara yang ditangkap. Keluaran yang diharapkan dari rangkaian penerima ini adalah berupa sinyal envelope yang menyerupai gelombang kotak unutk mengukur level peak dari sinyal suara. Berikut adalah rangkaian skematik dari sistem.

Cara kerja dari rangkaian pada gambar 3.9 berawal ketika mikrofon menerima sebuah getaran berupa suara, getara tersebut akan dikonversi menjadi gelombang elektrik yang sangat lemah. Kemudian gelombang elektrik masuk ke rangkaian preamplifier unutk dikuatkan

sebesar -10 kali. Rangkaian preamplifier sangat sensitif terhadap sinyal yang lemah dan memiliki penguat yang sangat besar sehingga mikrofon menjadi sangat peka terhadap suara yang masuk.

Jenis IC op-amp yang digunakan merupakan single supply (tegangan input V+ dan ground) dan sinyal yang diolah merupakan sinyal AC, oleh karena itu input non-inverting (+) dari op-amp diberi tegangan bias DC agar sinyal yang diproses tidak terpotong pada tegangan negatifnya. Nilai bias 1,75 V dipilih berdasarkan pengaruh karakteristik dari op-amp yang digunakan adalah mampu menghasilkan keluaran tegangan maksimum sebesar 3,5 V (Vcc – 1,5 V). apabila penguatannya melebihi dari batas tegangan output tegangan akan mengakibatkan sinyal terpotong pada nilai saturasi op-amp [26].

Tahap berikutnya merupakan BPF orde 4 dimana rangkaian ini berfungsi untuk memilah sinyal dengan rentang frekuensi tertentu yang dapat lewat. BPF orde 4 dibentuk dari dua buah BPF orde 2 yang disusun secara seri. Karakteristik dari filter BPF di bawah ditunjukkan oleh persamaan 3.1 dan persamaan 3.2.



Gambar 3.9 Skematik rangkaian sensor suara [26]

$$f_{r} = \frac{1}{2\pi\sqrt{R_1 R_2 C_1 C_2}} \quad (3.1)$$

$$Penguatan\ maksimal\ (Av) = -\frac{R_2}{2R_2} \quad (3.2)$$

Untuk mendapatkan frekuensi resonan f_r sebesar 5 kHz, dengan nilai $C_1 = C_2 = 10$ nF maka nilai dari resistor yang dibutuhkan :

$$R_1 = R_2 = \frac{1}{2\pi \times 5000 \times 10 \times 10^{-9}} \approx 3{,}1\ kOhm$$

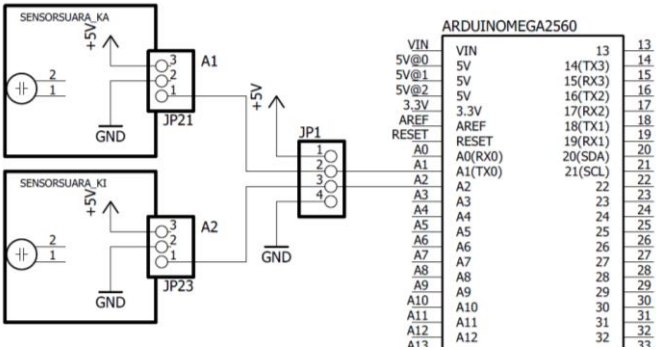
Karena nilia resistor 3,1 k**Ω** tidak ada di pasaran, dipilih nilai yang mendekati yakni 2,7 k**Ω** sehingga nilai f_r bergeser menjadi 5,8 kHz. Alasannya adalah untuk semakin menjauhi frekuensi 500 Hz – 2 kHz. Dengan begitu total penguatan dari BPF orde 4 adalah (-0,5)×(-0,5) = 0,25 kali.

Sinyal yang telah diseleksi oleh BPF kemudian dikuatkan oleh rangkaian penguat. Rangkaian penguat berupa inverting amplifier dengan ri = 1,5 k**Ω** dan Rf berupa resistor variabel 500 k**Ω** yang nilainya dapat diubah-ubah. Besar penguatan dari rangkaian penguat apabila resistor variabel diatur pada nilai 300 k**Ω** adalah -200 kali.

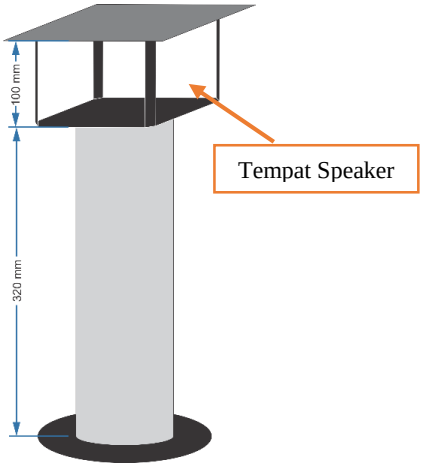
Sinyal yang telah dikuatkan masuk ke rangkaian enveloope detector yang bertujuan untuk memberi selubung ke tiap deret gelombang untuk memudahkan proses Arduino dalam menghitung tegangan peak yang masuk. Rangkaian envelope detector terdiri dari rangkaian penyearah setengah gelombang, kapasitor, dan resistor. Prinsip kerjanya hampir sama dengan rangkaian detektor puncak, dimana gelombang sinyal disearahkan oleh dioda D1 dan D2 untuk memperoleh gelombang positif dan mengisi kapasitor C4. Kapasitor akan terus mengisi dan perlahan tegangan kapasitor akan sama dengna tegangan puncak dari sinyal.

Saat rangkaian tidak menerima sinyal, kapasitor C4 akan membuang muatan menuju ke resistor R10 hingga tegangan kapasitor sama dengan nol. Nilia resistor dan kapasitor sangat mempengaruhi bentuk sinyal yang dihasilkan. Apabila nilai kapasitor atau resistor terlalu kecil, sinyal yang dihasilkan membentuk ripple, namun sebaliknya ketika nilai resistor terlalu besar, proses discharging dari kapasitor akan lambat dan mempengaruhi pembacaan pada Arduino.

Perangkat sensor suara ini dipasang pada sisi kanan dan kiri *mobile robot* sebagai telinga dari robot dengan metode menghitung besar amplitudo yang ditangkap dar masing-masing penerima, dapat diperoleh informasi arah datangnya suara. Apabila tegangan puncak yang diterima mikrofon kanan lebih besar daripada mikrofon kiri maka sumber suara cenderung berada pada daerah kanan dari *mobile robot*. Jika tegangan puncak mikrofon kiri lebih besar, maka sumber suara berada pada daerah kiri dari *mobile robot*. Untuk konfigurasi rangkaian dari sensor suara yang digunakan dengan Arduino ditunjukkan oleh gambar 3.10.



Gambar 3.10 Konfigurasi rangkaian sensor suara



Gambar 3.11 Penyangga speaker

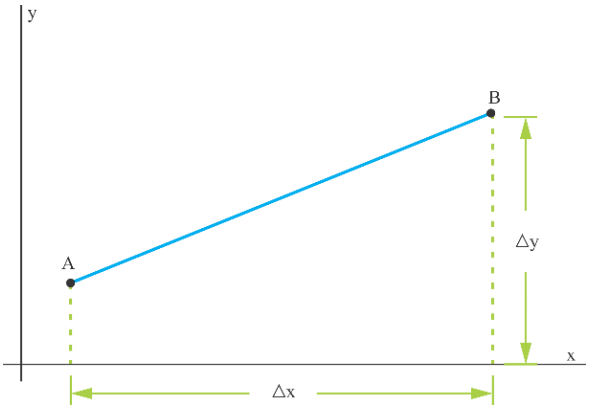
3.2.7. Perancangan Penyangga Sumber Suara

Sumber suara yang digunakan robot sebagai acuan dari sebuah *power station* untuk mengisi ulang baterai merupakan sebuah speaker audio. Spesifikasi speaker yang digunakan adalah Polytron Muze (PSP B1) dengan 2 speaker yang memiliki diameter 1,5 inchi, 4 Ohm, 3 Wrms, dan frekuensi keluaran dari 60 Hz hingga 20.000 Hz. Spesifikasi tersebut sangat mendukung sensor suara yang digunakan, yaitu sensor suara yang mengkombinasikan penguat LM324 dengan LM358.

Speaker yang memiliki dimensi 11 cm x 7 cm x 7 cm nantinya ditempatkan di atas sebuah penyangga sebagai bentuk simulasi *power station* dengan mekanik penyangga seperti yang ditunjukkan Gambar 3.11.

3.3. Perancangan Perangkat Lunak

Perancangan perangkat lunak ini digunakan untuk mengintegrasikan dan memfungsikan modul-modul perangkat keras sehingga sistem mampu bekerja sesuai aturan-aturan yang diinginkan. Perancangan ini meliputi penunjukkan arah dari GPS dan kompas HMC5883L, navigasi *waypoint* yang dituju, kendali PID untuk *Differential Speed Steering* berdasarkan navigasi *waypoint* dan amplitudo sumber suara.



Gambar 3.12 Ilustrasi jarak dari titik A ke titik B

3.3.1. Navigasi ke *Waypoint*

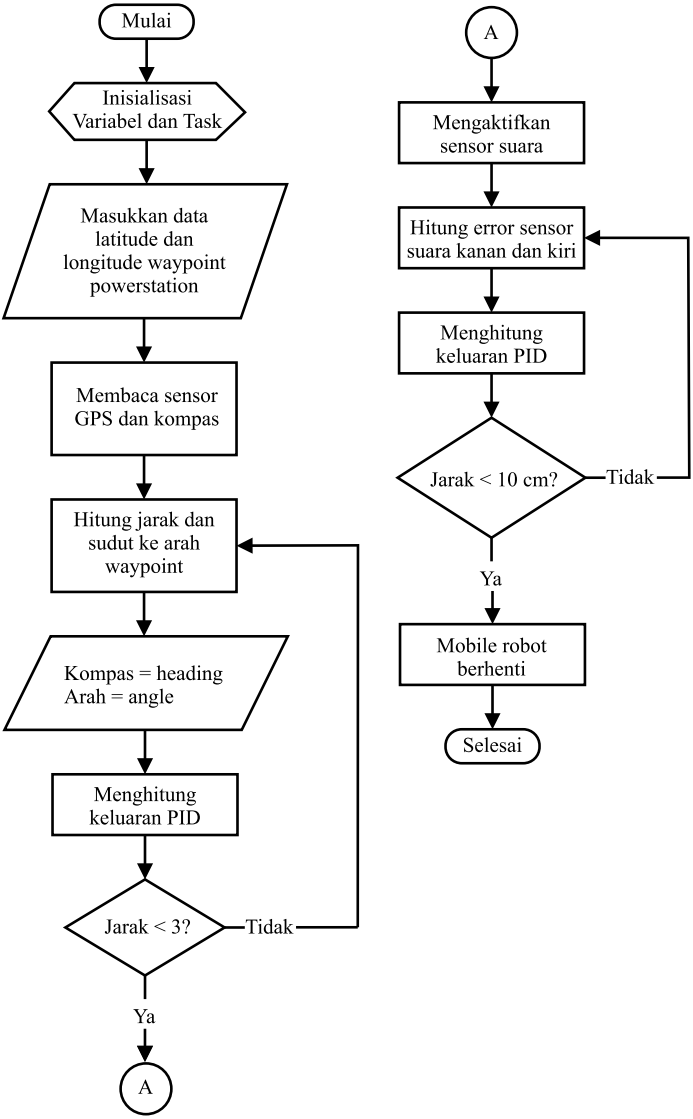
Titik koordinat *waypoint* didapatkan dari hasil pemetaan koordinat pada sebuah aplikasi PC seperti *Google Map*. Kemudian memasukkan data *lattitude* dan *longitude* yang diperoleh ke dalam program Arduino sebagai instruksi koordinat *waypoint* tujuan.

Dengan menggunakan *formula phytagoras*, akan didapatkan jarak antara 2 titik. Di mana titik A dapat direpresentasikan sebagai titik asal, dan titik B sebagai titik tujuan. Jarak titik A ke B dapat dicari menggunakan formula $\sqrt{\Delta x^2 + \Delta y^2}$ di mana Δx adalah selisih antara lintang sekarang dikurangi lintang tujuan, dan Δy adalah selisih antara bujur sekarang dikurangi bujur tujuan.

Jika sudah diketahui jarak, maka dapat disusun algoritma *waypoint*. Sebagai ilustrasi, jarak titik A dan B ditunjukkan pada gambar 3.12. Selanjutnya menyusun algoritma diagram alir yang ditunjukkan gambar 3.13.

Pada diagram alir gambar 3.13, ketika robot mulai, pertama kali menginisialisasi data masukan berupa koordinat *waypoint* yang telah diset. Setelah *task* dari tiap tugas aktif, maka program akan mulai membaca nilai sensor GPS, dan kompas. *Task* navigasi ke *waypoint* mengukur jarak dan arah robot terhadap *heading* dan tujuan. Dalam urutan tersebut, terdapat kondisi pemeriksaan untuk menghentikan segala aktifitas navigasi *waypoint*. Dimana jika sensor jaraktelah mendeteksi jarak objek sumber suara kurang dari 3 meter maka cara berjalan berdasarkan suara akan aktif.

Jika nilai sensor suara kanan lebih besar daripada sensor suara kiri maka robot akan bergerak ke kanan, jika nilai sensor suara kiri lebih besar daripada sensor suara kanan makan robot akan berbelok ke kiri, dan jika nilai antara sensor kanan dengan sensor kiri sama maka robot bergerak maju. Hal tersebut menjadi acuan untuk mengatur PID pada pergerakan robot dengan berdasarkan sensor suara. Robot akan berhenti ketika telah mencapai nilai maksimal yang telah diinstruksikan dalam program.



Gambar 3.13 Gambar diagram alir navigasi ke *waypoint*

Untuk menjalankan algoritma diagram alir dari gambar 3.13, setiap perangkat modul yang terpasang pada sistem mempunyai cara kerja dan akses yang berbeda-beda. Akses tersebut digunakan sebagai tahap untuk memperoleh data dari tiap-tiap perangkat melalui pemrograman yang ditanamkan pada Arduino. Cara akses dari masing-masing modul dalam navigasi ke *waypoint* adalah sebagai berikut:

a. Mengakses GPS Ublox Neo M8N

Untuk dapat mengakses dan mendapatkan data GPS, pada sistem *autodocking mobile robot* ini digunakan *library* yang cocok dengan GPS Ublox Neo M8N yaitu menggunakan *library* Ublox.h. Pengaturan baudrate antara modul dengan penerima harus sama, GPS disini menggunakan baudrate 9600. Kode dasar untuk membaca data masukkan dari Serial2 yang digunakan sebagai pin GPS adalah sebagai berikut:

```
#include “Ublox.h”

void setup()          {
      Serial.begin(9600);
      Serial2.begin(9600);
}
```

Kemudian dari data GPS yang diterima, perlu dilakukan pemilahan menggunakan intstruksi dari *library* Ublox.h karena data yang diterima masih berbentuk data mentah yang terdiri dari beberapa informasi. Hasil proses parsing data yang dilakukan oleh Ublox.h akan ditampilkan dalam program utama sistem *autodocking* berupa *altitude*, *latitude*, *longitude*, dan jumlah satelit yang sedang digunakan untuk mengumpulkan data GPS. Instruksi tersebut ditunjukkan oleh program sebagai berikut:

```
if(!Serial2.available())
  return;

while(Serial2.available()){
     char c = Serial2.read();
      if (M8_Gps.encode(c)) {
       gpsArray[0] = M8_Gps.altitude;
       gpsArray[1] = M8_Gps.latitude;
       gpsArray[2] = M8_Gps.longitude;
       gpsArray[3] = M8_Gps.sats_in_use;
      }
}
```

```
 for(byte i = 0; i < 4; i++) {
 Serial.print(gpsArray[i], 6);Serial.print(" ");
}

 Serial.println("");
```

b. Mengakses Modul Kompas HMC5883L

Modul kompas HMC5883L menggunakan komunikasi I2C (Inter Integrated Circuit) sebagai protokolnya. Sehingga untuk membaca modul ini, harus mengetahui alamat dari setiap register yang akan diakses. Alamat register HMC5883L 0x1E, alamat register tulis 0x3C, alamat register baca 0x3D, alamat register data 0x03. Prosedur kode pembacaan sensor kompas HMC5883L adalah sebagai berikut:

```
void kompas(){
  Wire.beginTransmission(0x1E);
  Wire.write(0x03);
  Wire.endTransmission();
  Wire.requestFrom(0x1E, 6);
  if(6 <= Wire.available()){
      x = Wire.read()<<8;    //X msb
      x |= Wire.read();      //X lsb
      z = Wire.read()<<8;    //Z msb
      z |= Wire.read();      //Z lsb
      y = Wire.read()<<8;    //Y msb
      y |= Wire.read();      //Y lsb
  }

  heading = atan2(y,x);
  float sudutDeklinasi = 0.022;
  heading += sudutDeklinasi;
  //koreksi arah Sudut ketika tanda berubah
  if(heading < 0) heading += 2*PI;
  //memeriksa arah ketika terselimuti karena penambahan
  deklinasi
  if(heading > 2*PI) heading -= 2*PI;
  sudutHeading = heading * 180/M_PI;
```

Data dari masing-masing sumbu mempunyai panjang 16 bit. Pembacaan dibagi menjadi dua, yaitu pembacaan MSB sepanjang 8 bit pertama, kemudian LSB pada 8 bit kedua. Data dari sumbu X dan sumbu Y kemudian dimasukkan dalam formula *atan2(y, x)* untuk menghasilkan nilai *arc tangen* dalam empat kuadran pada sumbu X atau Y tertentu. Sumbu Z tidak digunakan dalam formula, karena dianggap kompas selalu berada dalam keadaan tegak terhadap garis normal. Hasil formula tersebut

menghasilkan nilai sudut dengan satuan *radian* dengan nilai antara -3,14/2 hingga 3,14/2. Untuk merubahnya ke bentuk derajat hasil dari formula tersebut dikalikan dengan hasil 180/π.

c. Mengakses Sensor Suara

Modul sensor suara yang digunakan mengkombinasikan beberapa penguat amplifier mengeluarkan hasil keluaran berupa nilai analog yang dimasukkan ke dalam pin analog Arduino. Program untuk pengaturan awal dari pin analog yang digunakan adalah sebagai berikut:

```
void setup() {
  pinMode(A1,INPUT);
  pinMode(A2,INPUT);
  Serial.begin(9600);
}
```

Arduino yang digunakan memiliki ADC 10 bit sehingga data analog yang diperoleh perlu diakuisisi dan dikalibrasi, yaitu dengan mengkalikan nilai analog yang dibaca dengan hasil bagi tegangan referensi yang digunakan Arduino dengan nilai 1023 (10 bit). Bentuk program tersebut adalah sebagai berikut:

```
  float real_ka = analogRead(A1);
  float amp_ka = real_ka *(5.0/1023.0);
  float real_ki = analogRead(A2);
  float amp_ki = real_ki *(5.0/1023.0);
```

d. Mengakses *Driver* Motor L28N

Untuk dapat mengakses perancangan modul *driver* motor L298N pada Arduino seperti yang ditunjukkan pada gambar 3.7, secara sederhana kode tersebut ditulis sebagai berikut:

```
int in1 = 4;
int in2 = 5;
int in3 = 7;
int in4 = 6;
int ena = 8;
int enb = 9;

void setup() {

  pinMode(ena, OUTPUT);
  pinMode(enb, OUTPUT);
  pinMode(in1, OUTPUT);
  pinMode(in2, OUTPUT);
```

```
  pinMode(in3, OUTPUT);
  pinMode(in4, OUTPUT);
}
```

Instruksi Arduino yang dimasukkan ke dalam *driver* motor untuk manuver dari jalannya *mobile robot* ini adalah sebagai berikut:

```
void maju(){
  analogWrite(ena, pwm_ka);
  digitalWrite(in1, LOW);
  digitalWrite(in2, HIGH);

  analogWrite(enb, pwm_ki);
  digitalWrite(in3, LOW);
  digitalWrite(in4, HIGH);
}

void belok_kanan(){
  analogWrite(ena, pwm_ka);
  digitalWrite(in1, HIGH);
  digitalWrite(in2, LOW);

  analogWrite(enb, pwm_ki);
  digitalWrite(in3, LOW);
  digitalWrite(in4, HIGH);
}

void belok_kiri(){
  analogWrite(ena, pwm_ka);
  digitalWrite(in1, LOW);
  digitalWrite(in2, HIGH);

  analogWrite(enb, pwm_ki);
  digitalWrite(in3, HIGH);
  digitalWrite(in4, LOW);
}

void Stop(){
  digitalWrite(in1, LOW);
  digitalWrite(in2, LOW);

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

3.3.2. Desain Kendali PID untuk *Differential Speed Steering* Berdasarkan *Waypoint*

Kendali ini digunakan robot dalam bermanuver mengikuti sudut acuan dari hasil pengolahan data koordinat yang diberikan oleh *user* pada robot. Diagram blok dari kendali ini ditunjukkan pada gambar 3.12. Nilai *setpoint* didapatkan dari fungsi tan-1(y/x) selisih dua titik antara koordinat lokasi robot dengan koordinat *waypoint*. Dimana y = bujur tujuan – bujur sekarang, dan x = lintang tujuan – lintang sekarang.

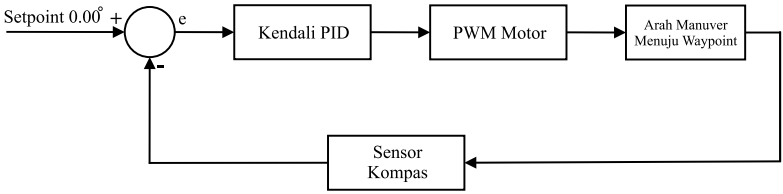
Fungsi tersebut akan menghasilkan nilai antara -3,14/2 hingga 3,14/2. Nilai tersebut kemudian dikonversi ke sudut dengan cara mengalikannya dengan 180/π. Kode dari penjelasan diagram blok adalah sebagai berikut:

```
double lat_tujuan;
double lon_tujuan;

switch(point){
 case 1: lat_tujuan =  -7.285047; //-7.285215;
        lon_tujuan = 112.795927; //112.795799;
        break;
 case 2: lat_tujuan =  -7.284955;
        lon_tujuan = 112.796821;
        break;
 case 3: lat_tujuan =  -7.284934;
        lon_tujuan = 112.796714;
        break;
}
if(!Serial2.available())
  return;

kompas();

while(Serial2.available()){
     char c = Serial2.read();
      if (M8_Gps.encode(c)) {
       gpsArray[0] = M8_Gps.altitude;
       gpsArray[1] = M8_Gps.latitude;
       gpsArray[2] = M8_Gps.longitude;
       gpsArray[3] = M8_Gps.sats_in_use;
      }
}
 for(byte i = 0; i < 4; i++) {
  Serial.print(gpsArray[i], 6);Serial.print(" ");
}
delmil(0);
Serial.println("");
```

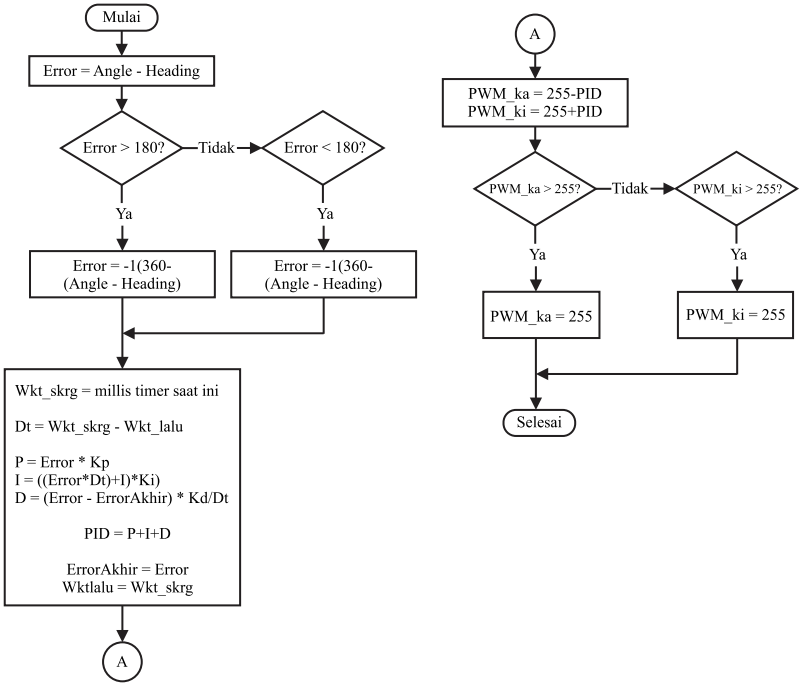
```
float hitungSudut = atan2((lon_tujuan - M8_Gps.longitude),(lat_tujuan -
M8_Gps.latitude));
//sudut alfa, arah angle set point
  float rubahKeRadian = 57.29577951;
//ubah radian ke sudut
       hitungSudut *= rubahKeRadian;
  if (hitungSudut < 0){
     hitungSudut = 360 + hitungSudut;
     }
  else if (hitungSudut > 0){
     hitungSudut = hitungSudut;
     }
  else if (hitungSudut > 360){
     hitungSudut = 0;
     }
   Angle = hitungSudut;
  Serial.println(Angle);
```

Setelah nilai *angle* sebagai nilai *setpoint* didapatkan, kemudian menyusun algoritma agar robot dapat bermanuver ke arah sudut *angle* tersebut. Algoritma yang digunakan diambil dari dasar kendali PID agar robot mampu berjalan dengan kecepatan yang sesuai dengan perbedaan sudut terhadap arah tujuan. Besar kecil kecilnya nilai PID tergantung dari nilai kP, kI, dan kD yang telah dideklarasikan pada program, tidak hanya itu, perbedaan sudut kompas dengan sudut titik tujuan juga akan menentukan nilai PID.

Nilai PID ini kemudian akan dikombinasikan dengan nilai PWM untuk menggerakkan motor DC. Hal tersebut sangat berguna agar robot tidak berjalan dengan kecepatan yang kurang atau kecepatan yang lebih sehingga mengakibatkan robot tidak akurat dalam menuju titik yang dituju. Algoritma itu disusun menjadi diagram alir yang ditunjukkan pada gambar 3.14.



Gambar 3.14 Diagram blok kendali arah motor



Gambar 3.15 Diagram alir algoritma kendali manuver *mobile robot* berdasarkan *waypoint*

Algoritma yang telah disusun menjadi diagram blok, kemudian diterjemahkan ke dalam kode program. Kode program dari algoritma pada gambar 3.15 adalah sebagai berikut:

```
float Error = sudutHeading - Angle;
  if (Error > 180) {
    Error = -1 * (360.00 - (sudutHeading - Angle));
  }
  else if (Error < -180) {
    Error = (360.00 + (sudutHeading - Angle));
  }

waktuSekarang = millis();
Dt = (float)(waktuSekarang - waktuLalu); //Delta waktu,waktuSkrg - waktuYgLalu
  float P = Error * Kp;
```

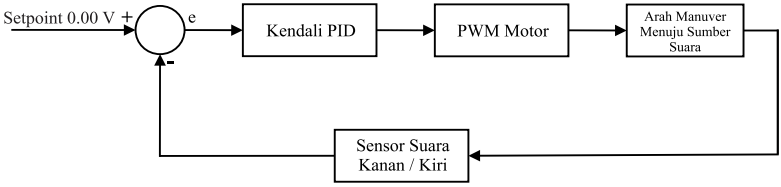
```
  float I = ((Error*Dt) + I) * Ki;
  float D = ((Error - lastError) * Kd)/Dt;
       PID = P + I + D;
       lastError = Error;

Serial.println("Error=" + String(Error) + " " + "LastError=" + String(lastError) + " "
+ "PID=" + String(PID));

      waktuLalu = waktuSekarang; //jadi waktuYgLalu

      pwm_ka = (255 - PID);
      pwm_ki = (255 + PID);
    if(pwm_ka > pwm_max){
      pwm_ka = pwm_max;
    if (pwm_ki < 0){
      belok_kiri();
    }
    else {
      maju();
    }
  }
    if(pwm_ki > pwm_max){
      pwm_ki = pwm_max;
    if(pwm_ka < 0){
      belok_kanan();
    }
    else{
      maju();
    }
  }

  #ifdef _DEBUG_
    Serial.println("PID = " + String(millis()));
  #endif

X = ((lon_tujuan - M8_Gps.longitude) * 2 * PI * 6371000.0 *
cos(M8_Gps.latitude))/360.0; //bujur24901--40074275
Y = ((lat_tujuan - M8_Gps.latitude) * 2 * PI * 6371000.0)/360.0; //lintang24860--
40008292

X2 = X * X;
Y2 = Y * Y;

  jarak = sqrt(X2 + Y2);
  Serial.println(X);
  delmil(0);

  if(jarak <= 9.0){
    //point++;
```

```
  Stop();
}

if(point > 3){
  point = 1;
}
Display();
```

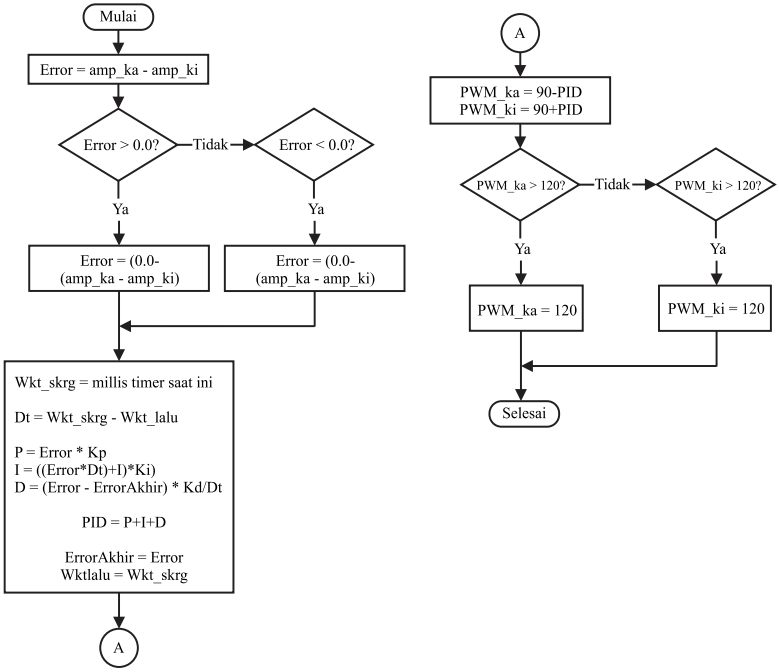
3.3.3. Desain Kendali PID untuk *Differential Speed Steering* Berdasarkan Sumber Suara

Mengadopsi dari kendali untuk mengatur menuver pergerakan robot berdasarkan suara digunakan kendali PID mengikuti tinggi rendahnya amplitudo yang dibaca oleh sensor suara. Kendali PID ini digunakan agar nantinya robot bergerak dengan kecepatan yang tidak berlebihan yang mengakibatkan respon antara sensor suara dengan jalannya robot tidak tepat. Diagram blok dari kendali ini ditunjukkan pada gambar 3.16. Dari nilai analog yang telah diterima, akan diolah untuk mencari *setpoint* dimana merupakan error dengan membandingkan nilai analog sensor suara kanan dengan sensor suara kiri. Logika yang dipakai adalah jika nilai sensor kanan lebih besar daripada nilai dari sensor kiri maka error akan bernilai positif, maka robot akan bermanuver ke kanan. Sebaliknya, jika nilai pembacaan sensor suara kiri lebih besar daripada nilai sensor kanan, error akan bernilai negatif dan *mobile robot* akan bermanuver ke kiri.

Setelah nilai *error* sebagai nilai *setpoint* didapatkan, kemudian menyusun algoritma agar robot dapat bermanuver ke arah sumer suara berasal. Algoritma itu disusun menjadi diagram alir yang ditunjukkan pada gambar 3.17.



Gambar 3.16 Diagram blok kendali manuver *mobile robot* berdasarkan sensor suara



Gambar 3.17 Diagram alir algoritma kendali manuver *mobile robot* berdasarkan sumber suara

Algoritma yang telah disusun menjadi diagram blok, kemudian diterjemahkan ke dalam kode program. Kode program dari algoritma pada gambar 3.15 adalah sebagai berikut:

```
waktuSekarang = millis();
Dt = (float)(waktuSekarang - waktuLalu);
//Delta waktu,waktuSkrg - waktuYgLalu
  float P = error * Kp;
  float I = ((error*Dt) + I) * Ki;
  float D = ((error - lastError) * Kd)/Dt;
       PID = P + I + D;
       lastError = error;

 Serial.println("Error=" + String(error) + " " + "LastError=" + String(lastError) + " "
+ "PID=" + String(PID));
     waktuLalu = waktuSekarang; //jadi waktuYgLalu
```

```
    pwm_ka = (90 - PID);
    pwm_ki = (90 + PID);
  if(pwm_ka > pwm_max){
    pwm_ka = pwm_max;
  if (pwm_ki < 0){
    belok_kiri();
  }
  else {
    maju();
  }
}
  if(pwm_ki > pwm_max){
    pwm_ki = pwm_max;
  if(pwm_ka < 0){
    belok_kanan();
  }
  else{
    maju();
  }
}
#ifdef _DEBUG_
  Serial.println("PID = " + String(millis()));
#endif

if (amp_ka == 2.88 && amp_ki == 2.88){
        Stop();
   }
```

BAB IV
PENGUJIAN DAN ANALISA DATA

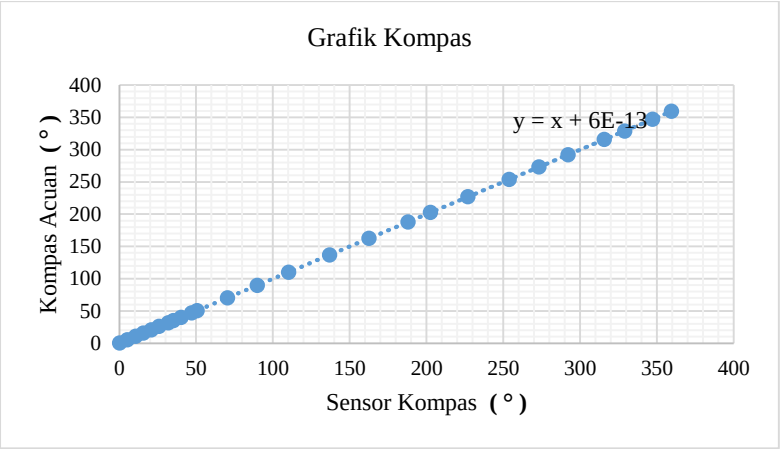
Bab ini menjelaskan tentang pengujian dan analisa data sistem dari hasil rancangan yang telah dijelaskan pada bab sebelumnya. Pengujian yang dilakukan adalah pengujian sensor kompas, sensor GPS, sensor suara, PWM kanan, PWM kiri, pengujian PWM terhadap kepekaan suara, *directivity* sensor suara, dan tingkat keberhasilan.

4.1. Pengujian Sudut Kompas

Pengujian sensor kompas ini dilakukan dengan cara membandingkan sensor kompas HMC5883L dengan sensor kompas acuan. Sensor kompas HMC5883L dan sensor kompas acuan dipasang dengan sudut *heading* awal 0º dan diputar searah jarum jam.

Tabel 4.1 Hasil Pengujian Sensor Kompas HMC5883L

No.	Kompas Acuan (°)	Kompas HMC5883L (°)	*Error %*
1	0	0.1	1%
2	5	5.1	2%
3	10	10.55	5%
4	15	15.46	3%
5	20	20.77	3%
6	25	25.82	3%
7	30	31.69	5%
8	35	35.2	0%
9	40	40.15	0%
10	45	46.05	2%
11	50	50.5	0%
12	70	70.3	0%
13	90	89.78	0%
14	110	110.15	0%
15	135	136.9	1%
16	160	162.53	1%
17	180	187.86	4%
18	200	202.5	1%
19	225	226.88	1%
20	250	253.79	1%
21	270	273.05	1%
22	290	292.04	1%
23	315	315.66	0%
24	330	328.9	0%
25	345	347.12	1%
26	360	359.44	0%



Gambar 4.1 Grafik nilai linier sensor kompas HMC5883L

Dari tabel 4.1 didapatkan *error* rata-rata pengujian sebesar 1,76% dan *error* tiap sudut antara hasil pengukuran kompas acuan dengan hasil pengukuran kompas HC5883L memiliki presentasi *error* yang kecil. Kecuali dalam beberapa titik sudut yang menunjukkan *error* cukup besar pada sudut 10º, 30º, dan 180º. Ini dapat disebabkan karena pada sudut-sudut tersebut terdapat benda-benda logam dan medan magnet pengganggu di sekitar sensor kompas sehingga dapat mempengaruhi hasil pembacaan.

Error yang ditunjukkan juga cenderung memiliki nilai lebih, hal tersebut sangat dipengaruhi oleh keberadaan benda di sekitar kompas berada, jenis kompas, dan nilai sudut deklinasi yang diberikan. Pada gambar 4.1 menunjukkan hasil dari sensor kompas mendekati nilai linier, ini dibuktikan dengan hasil persamaan y linier dengan nilai x ditambah 6E-13 adalah 6×10^{-13} yang terbilang nilainya sangat kecil mendekati 0 sehingga nilai persamaan mendekati nilai y.

4.2. Pengujian GPS

Pengujian GPS dilakukan dengan membandingkan nilai GPS acuan dari aplikasi peta seperti *Google Maps* pada ponsel dengan nilai GPS yang dihasilkan oleh sensor GPS. Pengujian mengambil beberapa sampel lokasi berbeda dalam lingkungan Institut Teknologi Sepuluh Nopember (ITS) dan sekitarnya.

Tabel 4.2 Pengujian Sensor GPS Ublox Neo M8N dengan GPS Acuan

Sensor GPS		**GPS Ponsel**		***Error***		
Latitude	**Longitude**	**Latitude**	**Longitude**	**Latitude**	**Longitude**	**Selisih (Meter)**
-7.292755	112.798423	-7.292752	112.79843	0.0000411%	0.0000062%	1.07
-6.916728	110.190719	-6.916732	110.190689	0.0000578%	0.0000272%	3.24
-7.293731	112.801407	-7.293693	112.801453	0.0005210%	0.0000408%	6.66
-7.284862	112.795916	-7.284912	112.7958135	0.0006864%	0.0000909%	3.49
-7.285001	112.7954539	-7.285008	112.7954561	0.0000961%	0.0000020%	0.86

Dari tabel 4.2 didapatkan selisih rata-rata sebesar 3,064 meter dan menunjukkan selisih antara sensor GPS dengan GPS acuan cukup kecil, kecuali pada nomor 3. Selisih terlihat cukup besar mencapai 6 meter. Ini dapat disebabkan karena faktor lingkungan. Di mana saat pengujian, langit tertutup awan mendung pekat. Sehingga sinyal satelit tidak diterima sensor GPS dengan baik.

4.3. Pengujiian Kendali PWM terhadap *Error* Sudut

Nilai *error* sudut didapatkan dari selisih antara *setpoint* dengan nilai sudut *heading*. Nilai *error* ini kemudian dimasukkan dalam persamaan kendali PID, dimana nilai proposional sebagai *gain* yang didapat dari perkalian nilai *error* dengan kP. Nilai integral didapatkan dari penjumlahan nilai *error* tiap waktu yang dikalikan dengan kI. Untuk nilai *derivative* didapatkan dari selisih *error* tiap waktu dikalikan dengan kD untuk menghasilkan nilai pengaturan PWM.

Pada pengujian ini nilai *setpoint* diatur pada 0º dengan hasil PWM bagian kiri (sinyal kuning) dengan PWM kanan (sinyal hijau) hampir berhimpitan seperti yang ditunjukkan gambar 4.2 dimana robot akan bergerak maju. Kemudian robot diputar ke arah kanan dan ke arah kiri secara bergantian. *Error* bernilai negatif ketika *heading* diarahkan ke kanan, dan bernilai positif ketika diarahkan ke kiri terhadap nilai *setpoint*.

a. Robot dalam kondisi *error* mendekati 0º (Maju)

Gambar 4.2 PWM ketika *error* mendekati 0

Pada hasil pengukuran osiloskop terlihat bahwa sinyal PWM memiliki frekuensi sebesar 490 Hz dengan periode bernilai 4 kotak dalam sumbu x yaitu 2 ms dan tampilan hasil *duty cycle* yang berubah berdasarkan *error* yang dihasilkan. Sedangkan untuk mengetahui nilai PWM dari sinyal yang dihasilkan dapat dicari dengan mengalikan 8 bitoutput PWM yang bernilai 255 dengan t_{on} dan kemudian dibagi dengan periode total dari sinyal PWM, dimana periode 2 ms, proses tersebut dapat ditulis dalam persamaan 4.1.

$$PWM = \frac{255 \times t_{on}}{T} \tag{4.1}$$

$$PWM\ ka = \frac{255 \times 2}{2} = 255$$

$$PWM\ ki = \frac{255 \times 1{,}9}{2} = 242{,}25 = 242$$

b. Robot diarahkan ke kiri

Ketika *heading* diarahkan ke kiri, sudut bergeser dari 0º hingga 45º. Nilai PWM bagian kiri (sinyal kuning) akan tetap pada 255, dan nilai PWM kanan (sinyal hijau) mengalami penurunan seperti yang ditunjukkan gambar 4.3. Semakin besar nilai *error* yang dihasilkan, maka semakin besar penurunan nilai PWM bagian kanan.

Kuning : PWM Kiri
Hijau : PWM Kanan

Gambar 4.3 PWM ketika *error* 45

$$PWM\ ka = \frac{255 \times 2}{2} = 255$$

$$PWM\ ki = \frac{255 \times 1}{2} = 127{,}5 = 127$$

c. Robot diarahkan ke kanan

Ketika heading diarahkan ke kanan, sudut bergeser dari 0º hingga 20º. Nilai PWM bagian kanan akan tetap pada 255, dan nilai PWM kiri mengalami penurunan seperti yang ditunjukkan gambar 4.4. Semakin besar nilai error yang dihasilkan, maka semakin besar penurunan nilai PWM bagian kiri.

$$PWM\ ka = \frac{255 \times 1{,}25}{2} = 159{,}375 = 159$$

$$PWM\ ki = \frac{255 \times 2}{2} = 255$$

Kuning : PWM Kiri
Hijau : PWM Kanan

Gambar 4.4 PWM ketika *error* -20

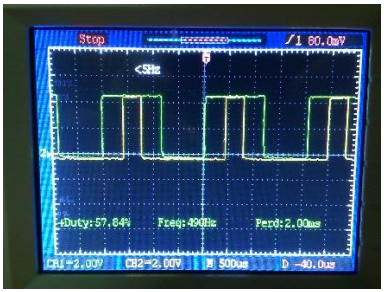
Nilai *error* yang dihasilkan dari pergeseran sudut antara *setpoint* dan heading merubah nilai PWM. Nilai PWM ini akan mengendalikan kecepatan putaran roda robot. Dengan menggunakan prinsip *differential speed steering*, robot akan dapat bermanuver mengikuti arah *setpoint angle* dengan memanfaatkan perbedaan kecepatan putaran roda yang dihasilkan dari perbedaan nilai PWM.

4.4. Pengujian Kendali PWM terhadap Kepekaan Suara

Pada pengujian ini nilai *setpoint* diatur pada 0.00 dengan cara memberikan suara pada sisi kanan kiri dan depan *mobile robot*. Hasil yang ditunjukkan pada gambar 4.5, gambar 4.6, dan gambar 4.7 adalah bentuk sinyal PWM bagian kiri (sinyal kuning) dengan PWM kanan (sinyal hijau).

Kuning : PWM Kiri
Hijau : PWM Kanan

Gambar 4.5 PWM ketika intensitas sensor kanan lebih tinggi



Kuning : PWM Kiri
Hijau : PWM Kanan

Gambar 4.6 PWM ketika intensitas sensor kiri lebih tinggi

Kuning : PWM Kiri
Hijau : PWM Kanan

Gambar 4.7 PWM ketika intensitas sensor kanan sama dengan sensor kiri

Gambar 4.5 adalah pengujian dimana sumber suara diletakkan pada sisi kanan *mobile robot* sehingga *mobile robot* diharuskan belok atau mengarah ke kanan dimana sumber suara berada. Hal tersebut terlihat dari gambar 4.15 bahwa besar PWM kuning (motor kiri) lebih besar daripada besar PWM hijau (kanan) sehingga *mobile robot* akan berbelok ke kanan.

Untuk gambar 4.6 adalah pengujian dimana sumber suara diletakkan pada sisi kiri *mobile robot* sehingga *mobile robot* diharuskan belok atau mengarah ke kiri dimana sumber suara berada. Hal tersebut terlihat dari gambar 4.6 bahwa besar PWM hijau (motor kanan) lebih besar daripada besar PWM kuning (kiri) sehingga *mobile robot* akan berbelok ke kanan.

Sedangkan gambar 4.7 adalah pengujian dimana sumber suara diletakkan pada depan *mobile robot* sehingga *mobile robot* diharuskan lurus atau maju ke sumber suara berada. Hal tersebut terlihat dari gambar 4.7 bahwa besar PWM kuning (motor kiri) berhimpitan dengan PWM hijau (kanan) sehingga *mobile robot* akan maju menuju sumber suara.

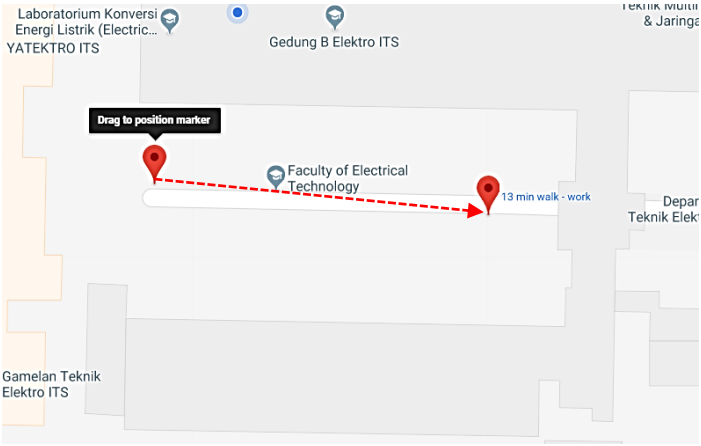
4.5. Pengujian Navigasi *Waypoint*

Pengujian navigasi *waypoint* dilakukan untuk mengetahui seberapa besar ketepatan robot dalam bernavigasi menuju titik-titik *waypoint* yang ditentukan.

Titik-titik *waypoint* yang ditunjukkan pada tabel 4.3 didapat dengan menentukan koordinat yang terlihat pada *Google Maps* dengan hasil plot koordinat seperti pada gambar 4.8.

Tabel 4.3 Daftar Titik Waypoint

No. WP.	*Lattitude*	*Longitude*
1	-7.285123	112.796288
2	-7. 285123	112. 796288
3	-7. 285088	112. 795898
4	-7.285088	112.795898



Gambar 4.8 Plot arah jalan *waypoint*

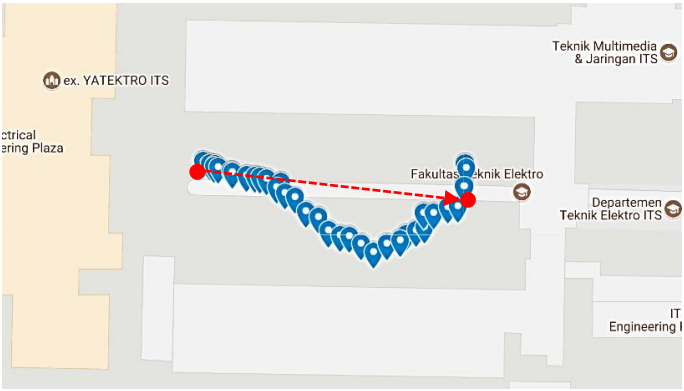
Setelah waypoint dimasukkan, robot dijalankan dan mengirimkan koordinat perjalanan setiap 2 detik. Hasil data pengiriman koordinat ditunjukkan pada tabel 4.4.

Data hasil perjalanan dan data waypoint kemudian di-plot seperti pada gambar 4.9 untuk menunjukkan jalur yang telah dilalui oleh robot dengan jalur yang seharusnya dilalui. Tanda lokasi biru merupakan garis perjalanan, sedangkan garis merah merupakan garis jalur waypoint. Pada beberapa titik terjadi banyak simpangan antara jalur yang seharusnya dengan jalur yang dilewati oleh robot. Pada titik 1 pergeseran sebesar 1,85 meter, pada titik 3 pergeseran sebesar 3,5 meter. Hal ini dapat dikarenakan simpangan sensor kompas. Kemudian jumlah satelit yang mengunci GPS juga sangat berpengaruh. Semakin banyak jumlah satelit yang mengunci, semakin tinggi keakuratan dari GPS dan juga kondisi cuaca yang kurang baik.

Tabel 4.4 Data Koordinat Perjalanan

Lattitude	*Longitude*
-7.285123	112.796288
-7.285087	112.795907
-7.285091	112.795917
-7.285092	112.795925
-7.285193	112.7962037
-7.285094	112.795923
-7.285095	112.79593
-7.285101	112.795948
-7.285102	112.79595
-7.285107	112.79597
-7.285108	112.795983
-7.285109	112.795981
-7.285112	112.796
-7.285117	112.796021
-7.285124	112.796014
-7.285133	112.796027
-7.285139	112.796041
-7.285159	112.796057

Lattitude	*Longitude*
-7.2851673	112.7960756
-7.2851873	112.7960899
-7.285194	112.7961074
-7.2851956	112.7961199
-7.2852066	112.7961369
-7.2852185	112.7961552
-7.2852062	112.7961749
-7.2852009	112.7961949
-7.2851872	112.7962176
-7.2851819	112.7962295
-7.2851619	112.7962294
-7.2851615	112.7962438
-7.285091	112.796289
-7.2851532	112.796264
-7.2851515	112.7962786
-7.285097	112.796291
-7.285123	112.796288



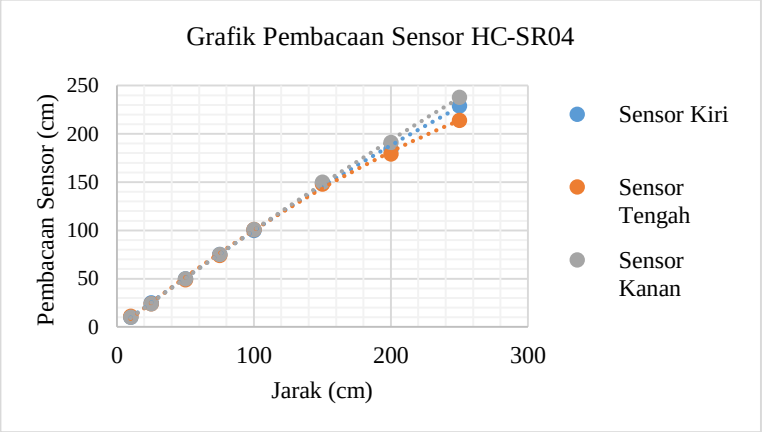
Gambar 4.9 Hasil *plotting* jalur waypoint yang dilalui oleh robot

4.6. Pengujian Perangkat Pendeteksi Jarak

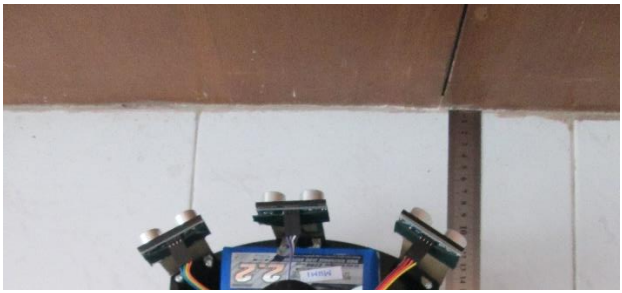
Pengujian sensor jarak HC-SR04 dilakukan dengan menguji kesesuaian pembacaan sensor dengan nilai pembacaan langsung menggunakan penggaris. Tujuan daripada pengujian ini adalah untuk mengetahui karakteristik dari sensor HC-SR04 yang kegunaannya pada *mobile robot* ini adalah untuk mendeteksi jarak sumber suara agar terhindar dari tabrakan dan benturan serta menghentikan *mobile robot* jika jarak sudah mencapai 10 cm dari penyangga sumber suara. Hasil pengujian disajikan pada tabel 4.5 berikut.

Tabel 4.5 Pembacaan sensor ultrasonik pada *mobile robot*

Jarak (cm)	Pembacaan Sensor		
	Kiri (cm)	Tengah (cm)	Kanan (cm)
10	10	11	10
25	25	24	24
50	50	49	50
75	75	74	75
100	100	101	101
150	149	148	150
200	184	179	191
250	229	214	238



Gambar 4.10 Grafik uji pembacaan sensor HC-SR04



Gambar 4.11 Pengujian sensor tengah dengan pembacaan 10 cm

Gambar 4.12 Pengujian sensor kiri dengan pembacaan 50 cm

Gambar 4.13 Pengujian sensor tengah dengan pembacaan 100 cm

Hasil yang tertera pada tabel 4.5 untuk pembacaan sensor secara tegak lurus dengan objek seperti yang ditunjukakan gambar 4.11, gambar 4.12, dan gambar 4.13, nilai pembacaanya sudah sesuai dengan pembacaan langsung dengan penggaris. Namun pada beberapa pembacaan terdapat sedikit perbedaan yang merupakan toleransi kesalahan pembacaan oleh sensor HC-SR04 seerti yang tertera pada spesifikasi sensor HC-SR04 yakni resolusinya mencapai 0,3 cm.

Dari gambar grafik uji 4.10, semakin jauh pembacaan sensor, semakin besar kesalahan yang diperoleh. Hal tersebut dikarenakan pengaruh posisi sudut pembacaan sensor satu dengan yang lainnya kurang dari sudut efektif pembacaan sensor HC-SR04 seperti yang disebutkan pada *datasheet* HC-SR04, sehingga gelombang pantul dari sensor lainnya mempengaruhi pembacaan sensor yang sedang diuji.

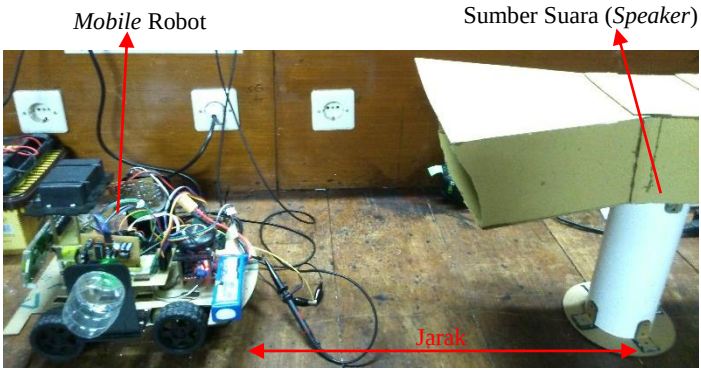
4.7. Pengujian Kepekaan Sensor Suara

Pengujian dari dua sensor suara yang digunakan dan ditempatkan pada sisi kanan dan kiri dari *mobile robot* merupakan modul sensor suara analog yang mengokmbinasikan beberapa IC penguat LM324 dengan LM358. Untuk suara yang digunakan adalah suara sonar dengan rentang frekuensi 900 Hz hingga 1100 Hz. Spesifikasi speaker yang digunakan adalah Polytron Muze (PSP B1) dengan 2 speaker yang memiliki diameter 1,5 inchi, 4 Ohm, 3 Wrms, dan frekuensi keluaran dari 60 Hz hingga 20.000 Hz. Spesifikasi tersebut sangat mendukung sensor suara yang digunakan, yaitu sensor suara yang mengkombinasikan penguat LM324 dengan LM358.

Pengujian sensor suara dilakukan dengan memperhatikan nilai tegangan keluaran sensor terhadap jarak dengan posisi sumber suara yang berbeda-beda seperti yang ditunjukkan gambar 4.14, gambar 4.15, gambar 4.16, dan gambar 4.17, hal ini dilakukan untuk mengetahui seberapa peka sensor yang digunakan.

Dari hasil pengujian yang tertera pada tabel 4.6 hingga tabel 4.9 menunjukkan adanya perbedaan yang signifikan dari pembacaaan nilai tegangan pada saat tanpa corong pengarah dengan menggunakan corong pengarah. Hal tersebut membuktikan bahwa penggunaan corong pengarah sangat membantu dalam mengumpulkan intensitas suara yang diperoleh. Dimana gelombang suara dapat terkumpul dan fokus masuk menuju mikrofon sehingga intensitas lebih tinggi dan tegangan yang dapat dibaca oleh arduino juga lebih besar.

1. Posisi *speaker* di sebelah depan *mobile robot*



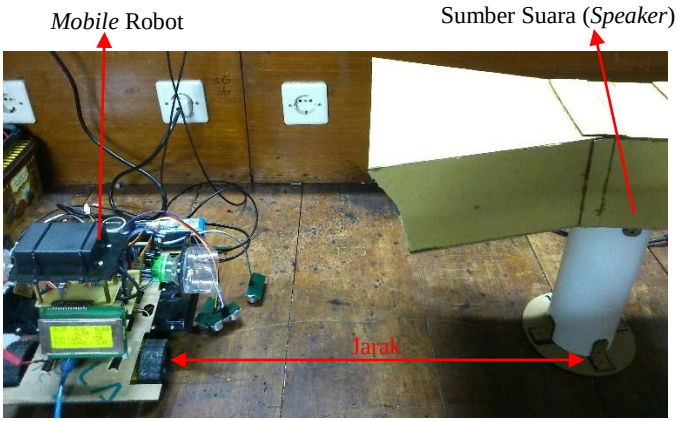
Gambar 4.14 Posisi *speaker* di depan *mobile robot*

Hasil pengujian dari gambar 4.6 ditunjukkan oleh tabel 4.4.

Tabel 4.6 Pengujian Posisi *Speaker* di Sebelah Depan *Mobile Robot*

Jarak	Tanpa Corong Pengarah		Selisih	Dengan Corong Pengarah		Selisih
	Vmax (mV)			Vmax (mV)		
	Kanan	Kiri		Kanan	Kiri	
10	2630	2670	40	2780	2790	10
20	2460	2490	30	2750	2770	20
30	2110	2190	80	2710	2710	0
40	1760	1770	10	2700	2700	0
50	1460	1480	20	2610	2670	60
60	1110	1180	70	2560	2580	20
70	730	735	5	2380	2390	10
80	670	695	25	1820	1870	50
90	560	600	40	1700	1770	70
100	500	520	20	1680	1710	30
120	310	405	95	1610	1660	50
140	180	250	70	1570	1590	20
160	150	180	30	1500	1540	40
180	100	140	40	1230	1280	50
200	70	100	30	990	1120	130

2. Posisi *speaker* di sebelah kanan *mobile robot*



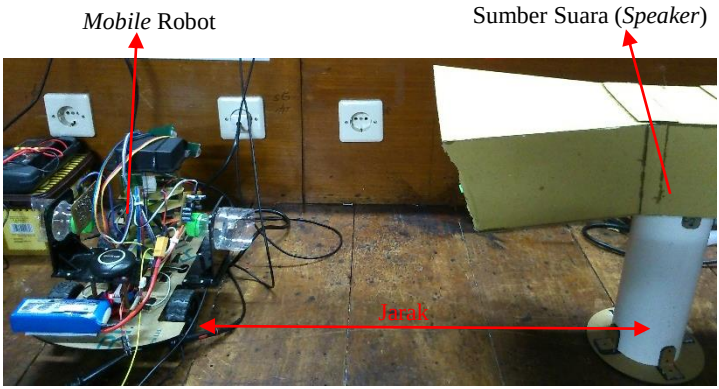
Gambar 4.15 Posisi *speaker* di samping kanan *mobile robot*

Hasil pengujian dari gambar 4.7 ditunjukkan oleh tabel 4.5.

Tabel 4.7 Pengujian Posisi *Speaker* di Sebelah Kanan *Mobile Robot*

Jarak	Tanpa Corong Pengarah		Selisih	Dengan Corong Pengarah		Selisih
	Vmax (mV)			Vmax (mV)		
	Kanan	Kiri		Kanan	Kiri	
10	2750	2570	180	2880	2800	80
20	2500	2430	70	2850	2790	60
30	2210	1980	230	2800	2770	30
40	1750	1580	170	2740	2700	40
50	1560	1460	100	2660	2520	140
60	1140	880	260	2590	2440	150
70	760	520	240	2360	2100	260
80	730	455	275	1820	1750	70
90	680	310	370	1700	1590	110
100	650	240	410	1680	1350	330
120	520	190	330	1610	1200	410
140	440	145	295	1570	1090	380
160	230	110	120	1500	890	610
180	170	75	95	1250	570	680
200	140	50	90	1080	305	703

3. Posisi *speaker* di sebelah kiri *mobile robot*



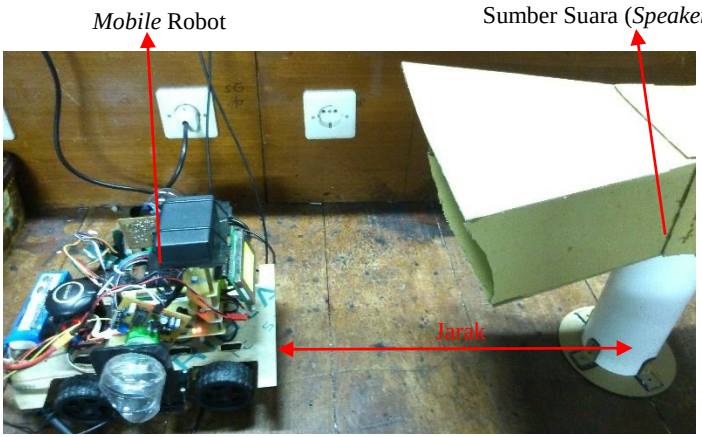
Gambar 4.16 Posisi *speaker* di samping kiri *mobile robot*

Hasil pengujian dari gambar 4.8 ditunjukkan oleh tabel 4.6.

Tabel 4.8 Pengujian Posisi *Speaker* di Sebelah Kiri *Mobile Robot*

Jarak	Tanpa Corong Pengarah		Selisih	Dengan Corong Pengarah		Selisih
	Vmax (mV)			Vmax (mV)		
	Kanan	Kiri		Kanan	Kiri	
10	2630	2790	160	2810	2890	80
20	2490	2580	90	2800	2850	50
30	2060	2260	200	2780	2800	20
40	1710	1800	90	2700	2780	80
50	1350	1590	240	2600	2670	70
60	990	1210	220	2450	2610	160
70	630	780	150	2230	2400	170
80	430	738	308	1770	1870	100
90	305	670	365	1650	1770	120
100	240	640	400	1400	1710	310
120	190	530	340	1260	1650	390
140	150	450	300	1120	1590	470
160	115	240	125	990	1540	550
180	60	170	110	550	1350	800
200	40	150	110	400	1180	780

4. Posisi *speaker* di sebelah belakang *mobile robot*



Gambar 4.17 Posisi *speaker* di belakang *mobile robot*

Hasil pengujian dari gambar 4.9 ditunjukkan oleh tabel 4.7.

Tabel 4.9 Pengujian Posisi *Speaker* di Sebelah Belakang *Mobile Robot*

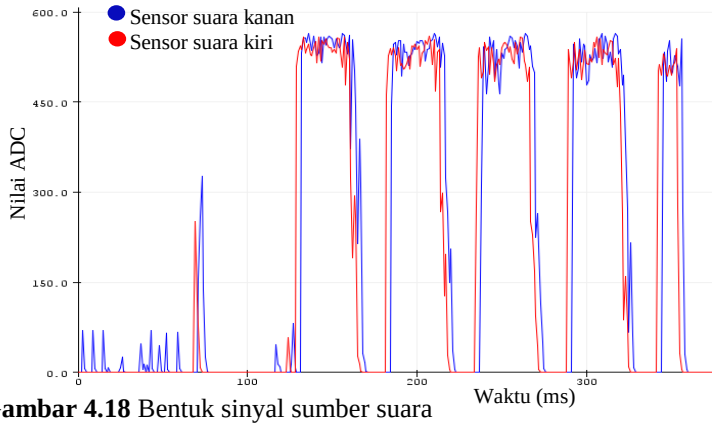
Jarak	Tanpa Corong Pengarah		Selisih	Dengan Corong Pengarah		Selisih
	Vmax (mV)			Vmax (mV)		
	Kanan	Kiri		Kanan	Kiri	
10	2590	2570	20	2680	2620	60
20	2420	2410	10	2600	2600	0
30	1990	2000	10	2570	2560	10
40	1730	1710	20	2530	2520	10
50	1380	1380	0	2500	2490	10
60	1100	1050	50	2490	2450	40
70	680	670	10	2410	2390	20
80	620	600	20	1780	1790	10
90	510	460	50	1690	1700	10
100	430	430	0	1600	1550	50
120	275	220	55	1520	1470	50
140	140	130	10	1400	1340	60
160	100	60	40	1310	1250	60
180	60	60	0	1050	1020	30
200	20	20	0	830	800	30

Pada hasil pengujian pada tabel 4.6 dan tabel 4.9 menunjukkan perbedaan yang signifikan ketika mikrofon kanan atau kiri tegak lurus dengan posisi sumber suara. Dimana ketika sensor kanan pada posisi tegak lurus dengan sumber suara, nilai pembacaan tegangannya akan lebih besar dari pada pembacaan sensor suara kiri dengan perbedaan mencapai 410 mV, sedangkan ketika sensor kiri pada posisi tegak lurus dengan sumber suara, nilai pembacaan tegangannya akan lebih besar dari pada pembacaan sensor suara kanan dengan perbedaan mencapai 800 mV.

4.8. Pengujian *Directivity* Sensor Suara pada *Mobile Robot*

Untuk mengetahui karakteristik kombinasi dari sensor suara yang digunakan dengan *hardware mobile robot* dilakukan pengujian dimana memperhatikan jarak radius dari titik tengah dari *mobile robot* dengan speaker dan sudut dari sumber suara. Pengukuran dilakukan membandingkan nilai pengukuran suara menggunakan *sound level meter* dengan pembacaan nilai ADC. Dalam hal ini *sound level meter* yang digunakan adalah aplikasi ukur tingkat suara pada ponsel yang diukur dalam satuan dB dan nilai ADC 10 bit pada Arduino.

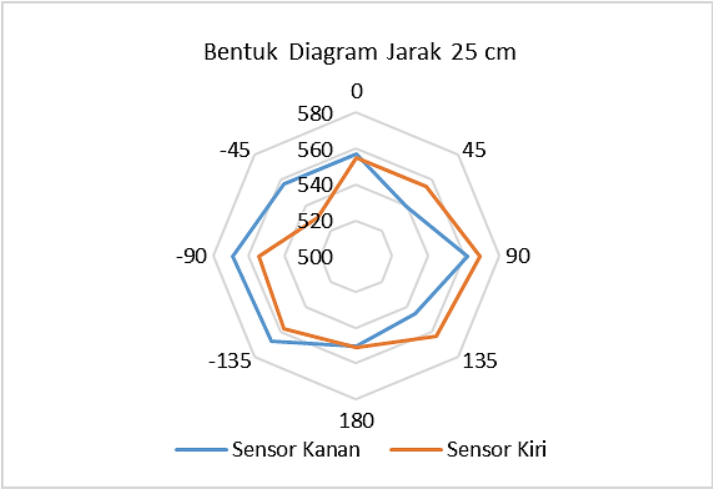
Dengan memperhatikan jarak dan sudut dari sumber suara akan dicari *directivity* yang merupakan ukuran karakteristik arah dari sumber suara berasal. Karakteristik arah sumber suara sangat dipengaruhi oleh permukaan pantulan di dekatnya.



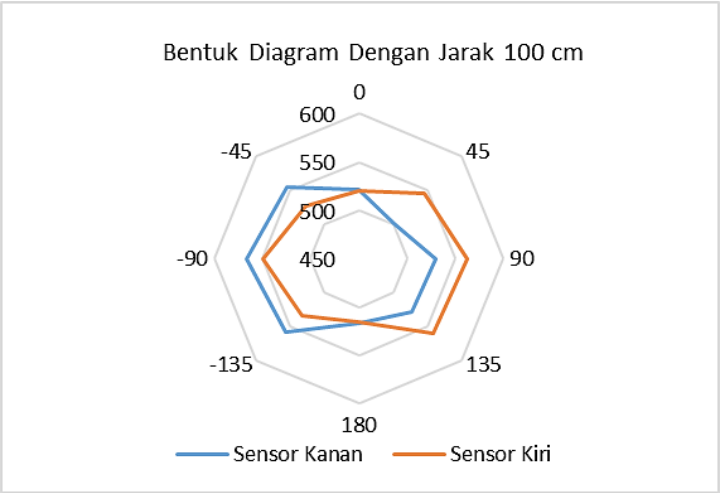
Gambar 4.18 Bentuk sinyal sumber suara

Tabel 4.10 Pembandingan *Directivity*

Sudut	Pengukuran	Jarak (cm)								
		25	**50**	**70**	**100**	**120**	**150**	**170**	**200**	**250**
0°	ADC Kanan	557	536	523	522	521	518	510	506	476
	ADC Kiri	555	546	546	520	518	515	503	486	482
	Level (dB)	59	55	53	52	50	49	48	47	43
45°	ADC Kanan	539	538	519	501	493	486	485	483	472
	ADC Kiri	555	550	548	546	539	534	531	530	527
	Level (dB)	59	55	53	52	50	49	48	47	43
90°	ADC Kanan	562	542	535	530	524	511	488	465	451
	ADC Kiri	569	565	564	562	559	540	526	520	502
	Level (dB)	59	55	53	52	50	49	48	47	43
135°	ADC Kanan	546	536	530	528	523	521	509	507	488
	ADC Kiri	563	562	560	559	555	551	550	548	531
	Level (dB)	59	55	53	52	50	49	48	47	43
180°	ADC Kanan	550	534	520	517	516	511	501	492	465
	ADC Kiri	551	541	539	516	514	507	499	480	469
	Level (dB)	59	55	53	52	50	49	48	47	43
-135°	ADC Kanan	567	564	559	557	550	550	547	540	528
	ADC Kiri	560	557	549	533	528	524	515	503	485
	Level (dB)	59	55	53	52	50	49	48	47	43
-90°	ADC Kanan	569	568	566	566	560	554	540	537	512
	ADC Kiri	563	554	553	549	539	532	519	502	485
	Level (dB)	59	55	53	52	50	49	48	47	43
-45°	ADC Kanan	557	556	556	555	550	549	541	538	535
	ADC Kiri	532	530	530	528	524	520	520	515	498
	Level (dB)	59	55	53	52	50	49	48	47	43



Gambar 4.19 Bentuk Diagram Radar Directivity Jarak 25 cm



Gambar 4.20 Bentuk Diagram Radar Directivity Jarak 100 cm

Berdasarkan tabel 4.10, kemudian akan dibuat menjadi diagram radar seperti gambar 4.19 dan gambar 4.20. Dengan melihat dan membandingkan gambar 4.19 dengan gambar 4.20 terlihat bahwa sudut pembacaan *directivity* sumber suara sangat baik pada sudut 90° untuk sensor suara kanan dan -90° untuk sensor suara kiri dengan nilai selalu lebih dari 550. Hasil tersebut membuktikan bahwa suara sangat efektif dengan arah tegak lurus antara pancaran suara dengan media terima.

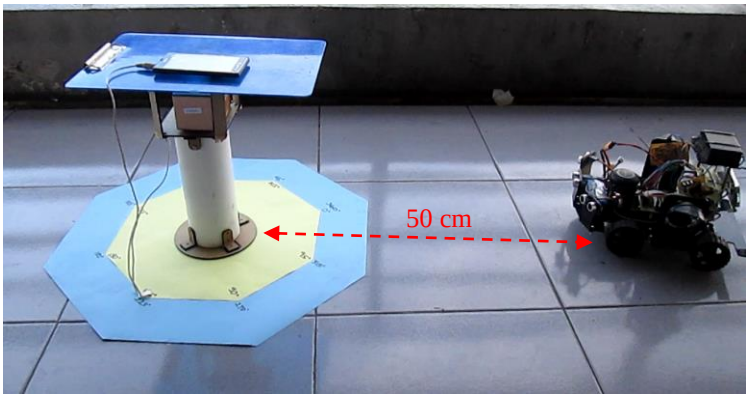
4.9. Pengujian Waktu Tempuh Sistem *Autodocking*

Dalam proses menuju sumber suara atau proses *autodocking*, *mobile robot* perlu diuji dan diukur waktu tempuhnya, hal tersebut nantinya berpengaruh pada efektifitas sistem yang telah dibuat pada aplikasi pengisian baterai *mobile robot*. Efektifitas yang diambil dalam pengujian adalah cepat atau lambatnya *mobile robot* menuju sumber suara sebagai *power station*.

Dalam pengujian ini, ukuran dari *power station* disimulasikan dengan daerah berwarna kuning dan warna biru sebagai daerah radiasi dari sistem *wireless charging*. Radiasi sistem *wireless charging* tercatat mampu memberikan pengisian sebuah baterai hingga radius 10 cm hingga 20 cm dari fisik terluar sistem *wireless charging*.

Pengujian akan dimulai dengan meletakkan *mobile robot* dalam jarak yang telah diatur dengan arah yang tidak tentu (tidak mengarah ke sumber suara) seperti yan ditunjukkan gambar 4.21. Waktu akan mulai dihitung ketika sistem *autodocking* berbasis suara aktif menggunakan *stopwatch*, kemudian waktu sampai *mobile robot* pada daerah radiasi *wireless charging* yang berwarna biru atau berada pada jarak terdekat dengan simulasi *transmitter wireless charging* seperti posisi yang ditunjukkan oleh gambar 4.26, kemudian akan dicatat. Dari hasil pengukuran tersebut akan diperoleh nilai dari waktu tempuh *mobile robot* dalam melaksanakan sistem *autodocking* berbasis suara seperti yang ditunjukkan pada tabel 4.11.

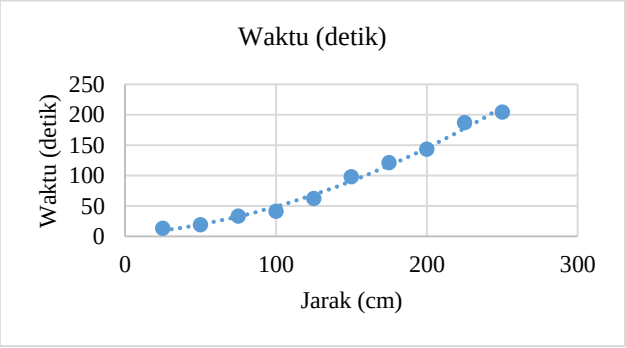
Hasil yang ditunjukkan oleh tabel 4.11 dari percobaan 1, 2, dan 3 yang merupakan 3 hasil terbaik dari beberapa percobaan memiliki hasil berbeda-beda. Hasil yang berbeda tersebut dikarenakan faktor cuaca pada saat percobaan berangin, dimana angin yang berhemus tidak stabil dan akan membuat kabur suara dari sumber suara. Dengan gambar grafik 4.22 memperkuat bahwa perubahan waktu tempuh berbanding lurus dengan jarak, semakin jauh jarak yang ditempuh, semakin lama pula waktu yang dibutuhkan untuk menuju sumber suara, suara yang terdeteksi juga akan lebih rendah, begitu juga *noise* yang terjadi akan semakin banyak. *Mobile robot* yang dirancang cukup efisien dalam jarak 100 cm dengan waktu tempuh kurang dari 1 menit (60 detik).



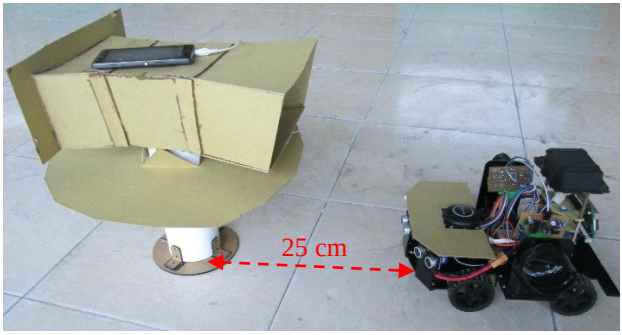
Gambar 4.21 Pengujian waktu tempuh robot dengan jarak 50 cm

Tabel 4.11 Waktu Tempuh Sistem *Autodocking*

Jarak (cm)	Waktu Tempuh (detik)			Waktu Tempuh Tercepat
	Percobaan 1	Percobaan 2	Percobaan 3	
25	18	13	15	13
50	32	21	19	19
75	36	35	33	33
100	41	45	64	41
125	62	78	75	62
150	98	101	100	98
175	135	122	121	121
200	143	168	149	143
225	190	187	201	187
250	204	245	204	204

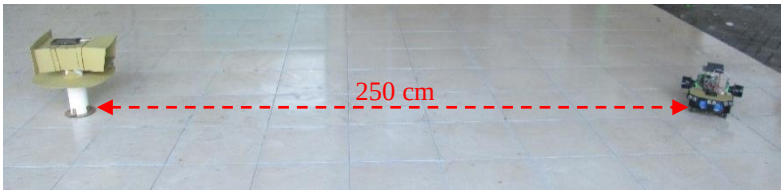


Gambar 4.22 Grafik waktu tempuh *mobile robot*

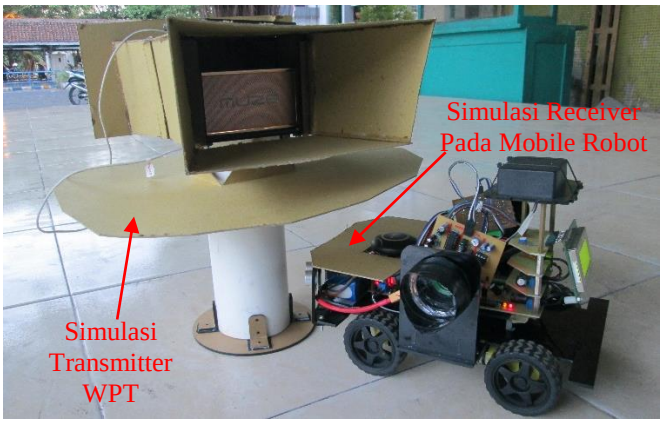


Gambar 4.23 Pengujian jarak 25 cm dengan sudut 0°

Gambar 4.24 Pengujian jarak 150 cm dengan sudut -45°



Gambar 4.25 Pengujian jarak 250 cm dengan sudut 0°



Gambar 4.26 Capaian posisi yang diharapkan

4.10. Pengujian Tingkat Keberhasilan

Untuk mengetahhui nilai persentase nilai tingkat keberhasilan atau yang disebut *success rate* dilakukan dengan cara menguji *mobile robot* secara kontinyu dengan jarak dan sudut yang berbeda. Dalam hal ini, jarak yang diujikan dari 25 cm hingga 250 cm dengan sudut masih-masing tiap jarak 0° hingga -45° seperti halnya pada pengujian *directivity*. *Mobile*

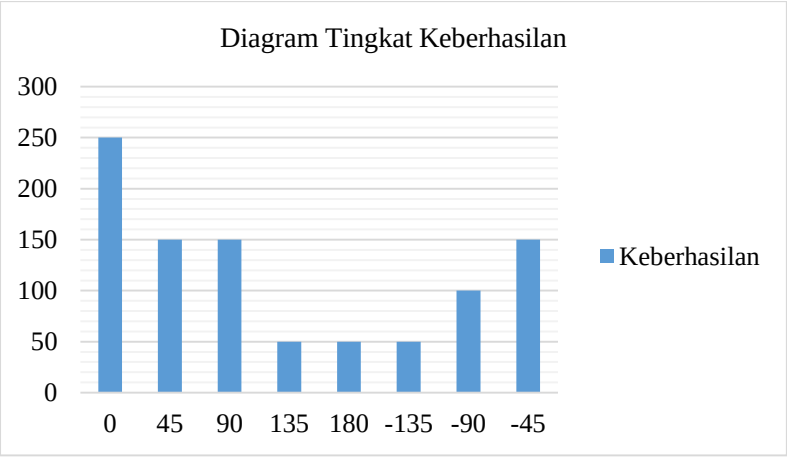
robot akan diuji menuju ke titik lokasi dimana sumber suara berada dengan simulasi *transmitter Wireless Power Transmission* (WPT) berupa kertas karton pada sumber suara dan simulasi *receiver* WPT yang juga berupa kertas ditempatkan pada *mobile robot.* Hasil pengujian tingkat keberhasilan ini ditunjukkan oleh tabel 4.12 dengan tanda centang yang berarti berhasil dan tanda strip untuk percobaan yang gagal.

Berdasarkan data dari tabel 4.12 terlihat bahwa sistem *autodocking mobile robot* berbasis suara yang telah dibuat cenderung berhasil menuju ke *power station* dengan tingkat keberhasilan sebesar 56,25%. Faktor yang mempengaruhi keberhasilan dari sistem *autodocking* berbasis suara ini antara lain adalah kondisi lingkungan dimana tidak banyak media pantul, tingkat kebisingan (*noise*), kondisi cuaca dimana cuaca yang cerah dan tidak berangin akan menghasilkan hasil yang bail karena angin dan temperature sangat mempengaruhi kecepatanrambat dari sumber suara menuju ke *mobile robot.* Pada percobaan ini dilakukan pada kondisi di luar ruangan dengan cuaca cerah namun angin bertiup cukup kencang yang membuat sebagian suara terbawa angin sehingga suara yang terdengar *mobile robot* tidak terlalu jelas.

Gambar 4.27 merupakan konversi dari tabel 4.12 yang diubah menjadi sebuah diagram untuk memperjelas sudut mana yang paling efektif ketika *mobile robot* dalam perjalanan. Terlihat pada gambar 4.27 robot dengan sutu 0° robot mampu menuju ke sumber suara dengan baik hal tersebut menunjukkan bahwa pancaran sumber suara ada pada sudut pancar ±45° hal tersebut cukup dipengaruhi oleh penggunaan corong pengarah pada sumber suara. Pada penelitian ini, corong pengarah smuber suara terbuat dari kertas karton, dimana kertas merupakan material yang dapat meredam suara.

Tabel 4.12 Tabel Pengujian Tingkat Keberhasilan

Jarak	**Sudut**							
(cm)	**0°**	**45°**	**90°**	**135°**	**180°**	**-135°**	**-90°**	**-45°**
25	√	√	√	√	√	√	√	√
50	√	√	√	√	√	√	√	√
100	√	√	√	-	-	-	√	√
150	√	√	√	-	-	-	-	√
200	√	-	-	-	-	-	-	-
250	√	-	-	-	-	-	-	-



Gambar 4.27 Diagram Capaian Tingkat Keberhasilan

Hasil akhir yang diharapkan dari tabel 4.12 tingkat keberhasilan yang disajikan dalam sebuah nilai persentase dapat dicari dengan persamaan 4.2 berikut.

$$SR = \frac{data\ berhasil}{data\ keseluruhan} \times 100\% \quad (4.2)$$
$$= \frac{27}{48} \times 100\%$$
$$= 56{,}25\%$$

4.11. Perbandingan Dengan Sistem Lain

Sistem *autodocking* pada robot telah diteliti dan dikembangkan sejak lama, namun dalam hal penelitian dengan berbagai metode menjadi dasar sebagai cara untuk memperoleh kerja sistem yang terbaik. Pada bagian ini beberapa perbandingan akan dilakukan untuk mengetahui kelebihan dan kekurangan sistem sehingga dapat menjadi referensi guna pengembangan sistem lebih lanjut. Sistem *autodocking* yang akan dibandingkan adalah sistem *autodocking* berbasis suara dengan sistem *autodocking* berbasis citra visual berdasarkan fungsi, biaya, dan kompleksitasnya yang disajikan dalam tabel 4.13.

Berdasarkan tabel 4.13, setiap metode yang digunakan pada sistem *autodocking* memiliki kelebihan dan kekurangan masing-masing. Dari

fungsinya sistem *autodocking* berbasis suara dapat mendeteksi suara dengan frekuensi tertentu dalam kondisi gelap atau rendah cahaya, namun rentan terhadap gangguan *noise.* Sedangkan sistem *autodocking* berbasis citra visual dapat mendeteksi gambar, warna, dan atau pola dengan syarat lingkungan memberikan pencahayaan yang baik kepada objek yang dideteksi.

Untuk biaya yang digunakan dalam membuat sistem *autodocking* berbasis suara relatif lebih murah dibanding sistem *autodocking* berbasis cita visual. Harga komponen utama sensor suara dari sistem *autodocking* berbasis suara lebih murah karena masih menggunakan teknologi analog, sedangkan kamera untuk pengolahan citra visual menggunakan teknologi digital dengan harga toko yang cukup mahal.

Sedangkan dilihat dari kompleksitas sistem, sistem *autodocking* berbasis suara memiliki tingkat kesulitan pada perancangan, analisa, dan pembuatan sensor suara yang menggunakan komponen analog sehingga mampu mendeteksi suara dengan rentang yang diinginkan. Sistem berbasis citra visual menggunakan kamera pabrikan, sehingga cara mengaksesnya lebih mudah, namun untuk dapat mengolah suatu objek diperlukan integrasi kode program yang cukup kompleks sehingga sistem *autodocking* berbasis citra ini sangat tergantung dari kualitas kamera dan kodeprogram yang sematkan pada mikrokontroler.

Tabel 4.13 Tabel Perbandingan Sistem *Autodocking*

Perbedaan	**Sistem Autodocking**	
	Berbasis Suara	**Berbasis Citra**
Teknik	Mendeteksi suara dengan frekuensi tertentu	Mendeteksi gambar, warna, dan atau pola
Biaya	Murah	Mahal
Kompleksitas	Perancangan dan analisa komponen analog pada sensor suara	Perancangan kode program pengolahan citra
Presisi	Cukup akurat	Akurat

----- Halaman ini sengaja dikosongkan -----

BAB V
PENUTUP

5.1. Kesimpulan

Kesimpulan yang dapat diambil dari penelitian tugas akhir ini adalah diperoleh *mobile robot* dengan sistem *autodocking* berbasis suara menggunakan Arduino Mega 2560, GPS Ublox Neo M8N, sensor kompas HMC5883L, mikrofon kondenser sebagai sensor suara, sensor jarak HC-SR04, dan motor *driver* dengan 4 buah motor DC. *Mobile robot* yang dirancang memiliki kemampuan menuju titik *waypoint* dengan *error* jarak posisi mencapai 6 meter. Hal tersebut dibuktikan dari hasil pengujian navigasi *waypoint* GPS yang tidak tegak lurus dalam menuju titik *waypoint.* Penerapan penggunaan corong pengarah pada sensor suara memiliki efektifitas sudut *directivity* 90° dan -90° sangat membantu mengurangi kerugian pembacaan level sumber suara yang memiliki suara audio dengan rentang frekuensi 900 Hz – 1100 Hz sehingga sensor suara menjadi lebih sensitif. Sedangkan *mobile robot* dapat mendeteksi suara dan menuju ke sumber suara dengan efektifitas jarak kurang dari 100 cm yang memiliki waktu tempuh kurang dari 60 detik dengan tingkat keberhasilan 56,25%. Dengan kombinasi metode penentuan arah sumber suara menggunakan dua sensor suara kanan dan kiri diibaratkan sebagai telinga bagi *mobile robot*, dengan membaca kekerasan dari sumber suara dalam nilai tegangan sangat dipengaruhi oleh kondisi kebisingan dari lingkungan sekitar.

5.2. Saran

Saran yang dapat diambil agar tujuan utama dari sistem yang dibuat pada tugas akhir ini tercapai maka pada mobile robot dapat menggunakan sensor suara dengan frekuensi yang berbeda antara sensor suara kanan dengan sensor suara kiri sehingga robot dapat dengan baik membedakan intensitas. Pada sumber suara menggunakan 2 sumber suara dengan frekuensi yang berbeda. Disamping itu pemilihan dan penggunaan motor yang memiliki kecepatan dan torsi yang tinggi dengan suara yang lebih halus dapat meningkatkan kontrol terhadap cara berjalannya *mobile robot*.

----- Halaman ini sengaja dikosongkan -----

DAFTAR PUSTAKA

[1] OSHA, "Hiydrogen Sulfide Fact." U.S. Department of Labor, Oct-2005.

[2] Diemas Kresna Duta, "2014, Angka Kecelakaan Kerja Hulu Migas Capai 159 kejadian," *CNN Indonesia*, Jakarta, 18-Mar-2015.

[3] Robert J. Schilling, *Fundamentals Of Robotics: Analysis and Control*, 5th ed. New Delhi: Asoke K. Ghosh, 2003.

[4] Dewi Indah Pertiwi, Muhammad Rivai, and Fajar Budiman, "Rancang Bangun Deteksi Jalur Pipa Terpendam Menggunakan Mobile Robot dengan Metal Detector," *Inst. Teknol. Sepuluh Nop.*, vol. 6, p. A-176, Mar. 2017.

[5] Admin Geek Studio, "4WD Aluminum Mobile Robot Platform," *4WD Aluminum Mobile Robot Platform*, 2015. [Online]. Available: https://www.geeker.co.nz/robot/chassis/4wd-aluminum-mobile-robot-platform.html. [Accessed: 12-Jun-2018].

[6] D. Reyes, G. Millan, R. Osorio-Corparan, and G. Lefranc, "Mobile Robot Navigation Assisted by GPS," *IEEE Lat. Am. Trans.*, vol. 13, no. 6, pp. 1915–1920, Jun. 2015.

[7] S. Rady, A. A. Kandil, and E. Badreddin, "A hybrid localization approach for UAV in GPS denied areas," in *2011 IEEE/SICE International Symposium on System Integration (SII)*, 2011, pp. 1269–1274.

[8] Rengarajan M. and Anitha G, "Algorithm Development and Testing of Low Cost Waypoint Navigation System," vol. 3, Apr. 2013.

[9] W. Lee and W. Chung, "Position estimation using multiple low-cost GPS receivers for outdoor mobile robots," in *2015 12th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI)*, 2015, pp. 460–461.

[10] Roby Adi Wibowo, "Implementasi Autonomous Navigation Robot Menggunakan Global Positioning System (GPS) Untuk Pemetaan Kadar Gas Berbahaya," Institut Teknologi Sepuluh Nopember, Surabaya, Tugas Akhir, Jan. 2017.

[11] Aleksandar Ristic, D. Petrovacki, and Miro Govedarica, "GNSS: Status and perspective," *ResearchGate*, vol. 14, p. 2, Jan. 2010.

[12] Admin Tronixlabs, "U-blox NEO-M8N GPS Module," *Tronixlabs Australia*. [Online]. Available: https://tronixlabs.com.au/wireless/gps/u-blox-neo-m8n-gps-module-australia/. [Accessed: 12-Jun-2018].

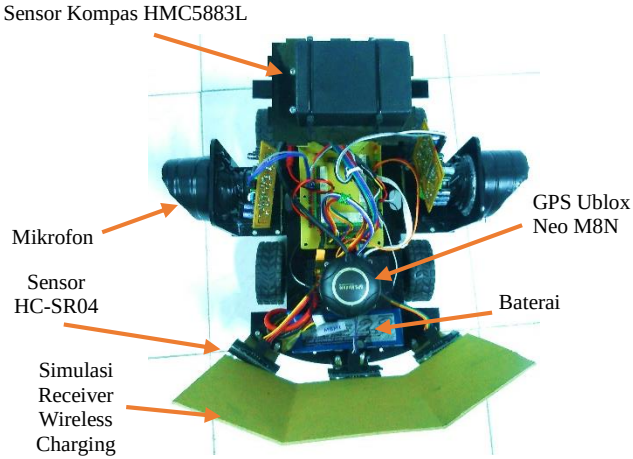
[13] M. J. Caruso, “Applications of Magnetoresistive Sensors in Navigation Systems,” 1997.
[14] Admin Honeywell, “3-Axis Digital Compass IC HMC5883L.” Adafruit, Feb-2013.
[15] Admin Amazon.com, “WINGONEER GY-271 QMC5883L Digital Compass Module 3-Axis Magnetic Sensor Module: Industrial & Scientific,” *WINGONEER GY-271 QMC5883L Digital Compass Module 3-Axis Magnetic Sensor Module: Industrial & Scientific*. [Online]. Available: https://www.amazon.com/WINGONEER-QMC5883L-Digital-Compass-Magnetic/dp/B06XHJBCC2. [Accessed: 12-Jun-2018].
[16] J. Fraden, *Handbook Of Modern Sensor : Physics, Designs, and Applications*, 3rd ed. United States Of America: AIP Press, Springer, 2004.
[17] I Gede Surya Adi Pranata and Anak Agung Ngurah Gunawan, “Application of Compass and Range Sensors on Eyeglass for Blind People Based on Microcontroller AT89S52,” *Dep. Phys. Univ. Udayana Bali Indones.*, vol. 3, p. 12, 2015.
[18] Praveen, “Ultrasonic water level controller using 8051. Measures water level,” *Ultrasonic water level controller using 8051. Measures water level*, 2015. [Online]. Available: http://www.circuitstoday.com/ultrasonic-water-level-controller-using-8051. [Accessed: 12-Jun-2018].
[19] Admin Protosupplies, “Arduino MEGA 2560 R3 with USB Cable – ProtoSupplies,” *Arduino MEGA 2560 R3 with USB Cable*, 2017. .
[20] I Putu Giovani, “Merancang Driver Motor DC,” *Merancang Driver Motor DC*, 2014. [Online]. Available: http://www.geyosoft.com/2014/merancang-driver-motor-dc.
[21] Admin Indoware, “L298N Driver Modul,” *L298N Driver Modul*, 2012. [Online]. Available: https://www.indo-ware.com/produk-2242-l298n--driver-modul.html. [Accessed: 12-Jun-2018].
[22] X. Wu, M. Xu, and L. Wang, “Differential speed steering control for four-wheel independent driving electric vehicle,” in *2013 IEEE International Symposium on Industrial Electronics*, 2013, pp. 1–6.
[23] Katsuhiko Ogata and Edi Laksono (Penterjemah), *Teknik Kontrol Automatik (Sistem Pengaturan) Jilid 2*, 2nd ed. Jakarta: Erlangga, 1991.
[24] Muhammad Agung Nursyeha, *Pengenalan Suara Burung menggunakan Mel Frequency Cepstrum Coefficient dan Jaringan*

Syaraf Tiruan pada Sistem Pengusir Hama Burung, vol. 5. Surabaya: Institut Teknologi Sepuluh Nopember, 2016.
[25] Meifal Rusli, John Malta, and Irsyad, *Prediksi Arah Sumber Suara untuk Perawatan Prediktif*. Palembang: Prosiding Seminar Nasional Tahunan Teknik Mesin (SNTTM), 2010.
[26] Putu Agus Antara Adiputra and Muhammad Rivai, *Swarm Robot Menggunakan Sistem Koordinasi Suara Untuk Mencari Sumber Gas*, 1st ed., vol. 1. Surabaya: Institut Teknologi Sepuluh Nopember, 2017.

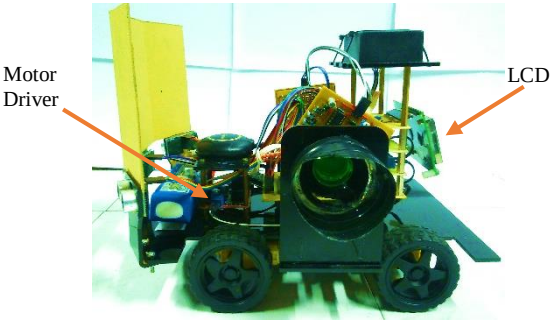
----- Halaman ini sengaja dikosongkan -----

LAMPIRAN

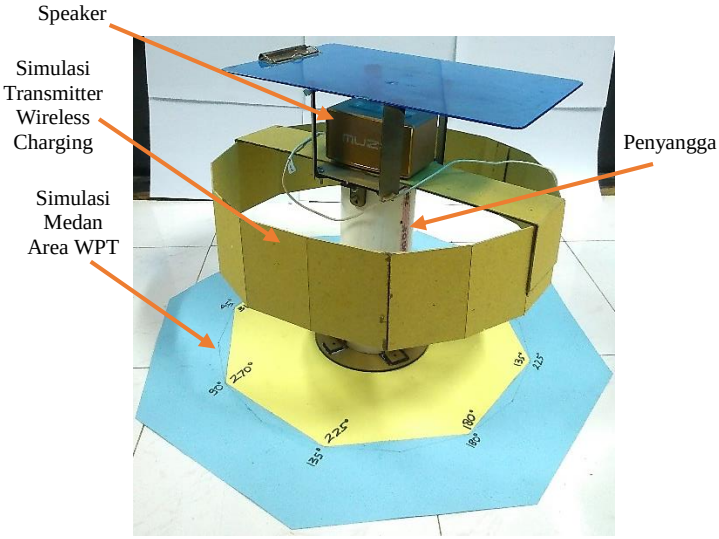
1. Gambar Realisasi *Mobile Robot*
   - Tampak Atas



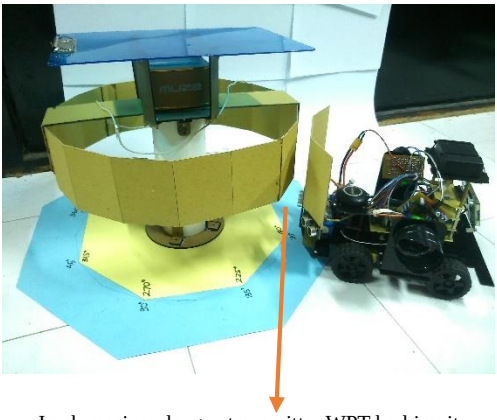
   - Tampak Samping



2. Gambar Realisasi Sumber Suara

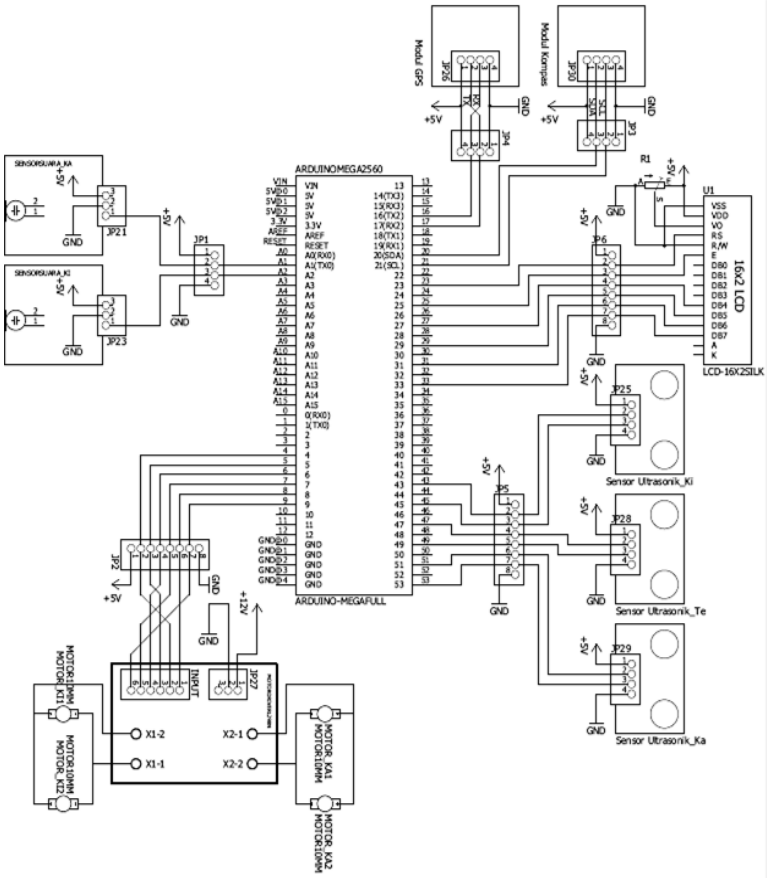


3. Gambar Capaian Posisi *Mobile Robot* Pada Area Sumber Suara



Jarak receiver dengan transmitter WPT berhimpit dengan jarak <20 cm

4. Gambar Skematik Rangkaian Keseluruhan



5. Kode Program Arduino Untuk Setup

```
void setup() {
  pinMode(ena, OUTPUT);
  pinMode(enb, OUTPUT);
  pinMode(in1, OUTPUT);
  pinMode(in2, OUTPUT);
  pinMode(in3, OUTPUT);
  pinMode(in4, OUTPUT);
  pinMode(trigPin1, OUTPUT);
```

```
  pinMode(echoPin1, INPUT);
  pinMode(trigPin2, OUTPUT);
  pinMode(echoPin2, INPUT);
  pinMode(trigPin3, OUTPUT);
  pinMode(echoPin3, INPUT);
  pinMode(A0, INPUT);
  pinMode(A1,INPUT);
  pinMode(A2,INPUT);

   Serial.begin(9600);
   Serial2.begin(9600);
   lcd.begin(16, 4);
   Wire.begin();

//Put the HMC5883 IC into the correct operating mode
  Wire.beginTransmission(address); //open communication with
HMC5883
  Wire.write(0x02); //select mode register
  Wire.write(0x00); //continuous measurement mode
  Wire.endTransmission();
}

static void delmil(unsigned long ms) {
  unsigned long start = millis();
  do {
    while (Serial2.available())
      M8_Gps.encode(Serial2.read());
  } while (millis() - start < ms);
}
```

6. Kode Program Untuk Membaca Sensor Kompas HMC5883L

```
void kompas(){
  int x,y,z; //triple axis data

//Tell the HMC5883 where to begin reading data
  Wire.beginTransmission(address);
  Wire.write(0x03); //select register 3, X MSB register
  Wire.endTransmission();
```

```
 //Read data from each axis, 2 registers per axis
  Wire.requestFrom(address, 6);
  if(6<=Wire.available()){
    x = Wire.read()<<8;   //X msb
    x |= Wire.read();     //X lsb
    z = Wire.read()<<8;   //Z msb
    z |= Wire.read();     //Z lsb
    y = Wire.read()<<8;   //Y msb
    y |= Wire.read();     //Y lsb
   }

  heading = atan2(y,x);
  float sudutDeklinasi = 0.02;
  heading += sudutDeklinasi;
  //koreksi arah Sudut ketika tanda berubah
  if(heading < 0) heading += 2*PI;
  //memeriksa arah ketika terselimuti karena penambahan
deklinasi
  if(heading > 2*PI) heading -= 2*PI;
  sudutHeading = heading * 180/M_PI;

}
```

7. Kode Program Arduino Untuk Driver Motor DC

```
void maju(){
  analogWrite(ena, pwm_ka);
  digitalWrite(in1, LOW);
  digitalWrite(in2, HIGH);

  analogWrite(enb, pwm_ki);
  digitalWrite(in3, LOW);
  digitalWrite(in4, HIGH);
}

void belok_kanan(){
  analogWrite(ena, pwm_ka);
  digitalWrite(in1, HIGH);
  digitalWrite(in2, LOW);
```

```
  analogWrite(enb, pwm_ki);
  digitalWrite(in3, LOW);
  digitalWrite(in4, HIGH);
 }

 void belok_kiri(){
  analogWrite(ena, pwm_ka);
  digitalWrite(in1, LOW);
  digitalWrite(in2, HIGH);

  analogWrite(enb, pwm_ki);
  digitalWrite(in3, HIGH);
  digitalWrite(in4, LOW);
 }

 void Stop(){
  digitalWrite(in1, LOW);
  digitalWrite(in2, LOW);

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

8. Kode Program Arduino Untuk Membaca Sensor HC-SR04

```
void SonarSensor(int trigPin,int echoPin)
{
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);
duration = pulseIn(echoPin, HIGH);
distance = (duration*0.034)/2;

}
```

9. Kode Program Navigasi Waypoint

```
void loop() {
  Vawal = analogRead(A0);
  float tegangan = Vawal * (12.52 / 1023.0);
```

```
  double lat_tujuan;
  double lon_tujuan;

  switch(point){
   case 1: lat_tujuan = -7.285155; //-7.285123; //-7.285047; //-
7.285215;
          lon_tujuan = 112.796412;//112.796288; //112.795927;
//112.795799;
          break;
  }
 if(!Serial2.available())
   return;

 kompas();

 while(Serial2.available()){
      char c = Serial2.read();
       if (M8_Gps.encode(c)) {
        gpsArray[0] = M8_Gps.altitude;
        gpsArray[1] = M8_Gps.latitude;
        gpsArray[2] = M8_Gps.longitude;
        gpsArray[3] = M8_Gps.sats_in_use;
      }
 }

 Serial.print(gpsArray[1], 6);
 Serial.print(" ");
 Serial.println(gpsArray[2], 6);
//  delmil(0);


  float hitungSudut = atan2((lon_tujuan -
M8_Gps.longitude),(lat_tujuan - M8_Gps.latitude));//sudut
alfa, arah angle set point
  float rubahKeRadian = 57.29577951; //ubah radian ke sudut
       hitungSudut *= rubahKeRadian;
  if (hitungSudut < 0){
     hitungSudut = 360 + hitungSudut;
```

```
    }
  else if (hitungSudut > 0){
     hitungSudut = hitungSudut;
     }
  else if (hitungSudut > 360){
     hitungSudut = 0;
     }
   Angle = hitungSudut;
  Serial.println(Angle);

//---------------------PID---------------------------//
  float Error = sudutHeading - Angle;
  if (Error > 180) {
    Error = -1 * (360.00 - (sudutHeading - Angle));
  }
  else if (Error < -180) {
    Error = (360.00 + (sudutHeading - Angle));
  }

  waktuSekarang = millis();
  Dt = (float)(waktuSekarang - waktuLalu); //Delta
waktu,waktuSkrg - waktuYgLalu
  float P = Error * Kp;
  float I = ((Error*Dt) + I) * Ki;
  float D = ((Error - lastError) * Kd)/Dt;
       PID = P + I + D;
       lastError = Error;

  Serial.println("Error=" + String(Error) + " " + "LastError=" +
String(lastError) + " " + "PID=" + String(PID));
     waktuLalu = waktuSekarang; //jadi waktuYgLalu
//     /*
     pwm_ka = (170 + PID);
     pwm_ki = (170 - PID);
  Serial.println("PWM Ka=" + String(pwm_ka) + " " + "PWM
Ki=" + String(pwm_ki) + " ");
    if(pwm_ka > pwm_max){
     pwm_ka = pwm_max;
    if (pwm_ki < 0){
```

```
    belok_kiri();
   }
   else {
    maju();
   }
  }
   if(pwm_ki > pwm_max){
    pwm_ki = pwm_max;
   if(pwm_ka < 0){
    belok_kanan();
   }
   else{
    maju();
   }
  }
  #ifdef _DEBUG_
//    Serial.println("PID = " + String(millis()));
  #endif
  Serial.println("PWM Ka=" + String(pwm_ka) + " " + "PWM
Ki=" + String(pwm_ki) + " ");

  X = ((lon_tujuan - M8_Gps.longitude) * 2 * PI * 6371000.0 *
cos(M8_Gps.latitude))/360.0; //bujur24901--40074275
  Y = ((lat_tujuan - M8_Gps.latitude) * 2 * PI *
6371000.0)/360.0; //lintang24860--40008292
  X2 = X * X;
  Y2 = Y * Y;
  jarak = sqrt(X2 + Y2);

  if(jarak <= 1.0){
   //point++;
   Stop();
   suara();
  }
}
```

10. Kode Program Autodocking Berbasis Suara

```
void suara()
{
  SonarSensor(trigPin1, echoPin1);
```

```
  sensor_ki = distance;
  SonarSensor(trigPin2, echoPin2);
  sensor_dep = distance;
  SonarSensor(trigPin3, echoPin3);
  sensor_ka = distance;

 if (sensor_ka >= 18 && sensor_dep >= 18 && sensor_ki >=
18)
    {
      float real_ka = analogRead(A1);
      float amp_ka = real_ka *(5.0/1023.0);
      float real_ki = analogRead(A2);
      float amp_ki = real_ki *(5.0/1023.0);

      float error = amp_ka - amp_ki;

      if (error > 0.0) {
        error = (0.00 - (amp_ka - amp_ki));
        }
      else if (error < 0.0) {
        error = (0.00 - (amp_ka - amp_ki));
      }

      waktuSekarang = millis();
      Dt = (float)(waktuSekarang - waktuLalu); //Delta
waktu,waktuSkrg - waktuYgLalu
      float P = error * Kps;
      float I = ((error*Dt) + I) * Kis;
      float D = ((error - lastError) * Kds)/Dt;
           PID = P + I + D;
           lastError = error;

      Serial.println("Error=" + String(error) + " " + "LastError="
+ String(lastError) + " " + "PID=" + String(PID));
           waktuLalu = waktuSekarang; //jadi waktuYgLalu

       pwm_kas = (140 - PID);
       pwm_kis = (140 + PID);
```

```
    if(pwm_kas > pwm_maxs){
       pwm_kas = pwm_maxs;
       if (pwm_kis < 0){
          belok_kiri();
          }
    else {
       maju();
      }
    }

    if(pwm_kis > pwm_maxs){
       pwm_kis = pwm_maxs;
       if(pwm_kas < 0){
         belok_kanan();
         }
    else{
       maju();
      }
    }
   #ifdef _DEBUG_
     Serial.println("PID = " + String(millis()));
   #endif

   delay(350);
   Stop();
   delay(270);

   Displays();
  }

 else {
  Stop();
  }
}
```

11. Kode Program Arduino Untuk Display LCD

```
void Display(){
  lcd.setCursor(0, 0);
  lcd.print("Lat ");
```

```
  lcd.setCursor(5, 0);
  lcd.print(M8_Gps.latitude, 6);
  lcd.setCursor(0, 1);
  lcd.print("Lon ");
  lcd.setCursor(5, 1);
  lcd.println(M8_Gps.longitude,6);
  lcd.setCursor(0,2);
  lcd.print(Angle);
  lcd.setCursor(8,2);
  lcd.print(sudutHeading);
  lcd.setCursor(5, 3);
  lcd.print(" m");
  lcd.setCursor(7, 3);
  lcd.print("      ");
}
```

BIOGRAFI PENULIS

Ari Hidayanto, dilahirkan pada 4 Agustus 1994 di Kota Kendal, Jawa Tengah dan merupakan anak sulung dari tiga bersaudara. Pada tahun 2006 lulus dari SD Negeri Pekauman Kendal, kemudian melanjutkan sekolah di SMP N 2 Kendal dan lulus tahun 2009. Pada tahun 2012 lulus dari SMA N 1 Kendal dan melanjutkan kuliah di Sekolah Vokasi Elektro Undip. Lulus kuliah vokasi pada tahun 2015 dan sempat mengenyam dunia kerja selama kurang lebih satu tahun. Tahun 2016 menjadi awal pendidikan strata sarjana teknik elektro melalui program Lintas Jalur Teknik Elektro Institut Teknologi Surabaya. Semasa kuliah, penulis aktif berkegiatan sebagai anggota di Laboratorium Elektronika Industri B402. Pada tahun 2018 merupakan target untuk menyelesaikan program sarjana.

Email : arihidayanto44@gmail.com

----- Halaman ini sengaja dikosongkan -----