****

**PROYEK AKHIR**

***PATH* *PLANNING* BERBASIS *INDOOR***

***POSITIONING* *SYSTEM* PADA *MOBILE ROBOT***

***PATH PLANNING BASED ON INDOOR***

***POSITIONING SYSTEM ON MOBILE ROBOT***

**Oleh :**

**Luqman Hakim**

**NRP. 1110 121 004**

**Dosen Pembimbing :**

**Akhmad Hendriawan, S.T., M.T.**

**NIP. 197501272002121003**

**Dr. Alrijadjis, Dipl.Eng., M.T**

**NIP. 197206301999031003**

**PROGRAM STUDI TEKNIK ELEKTRONIKA**

**DEPARTEMEN TEKNIK ELEKTRO**

**POLITEKNIK ELEKTRONIKA NEGERI SURABAYA**

**2016**

****

#### PROYEK AKHIR

***PATH* *PLANNING* BERBASIS *INDOOR POSITIONING SYSTEM*  PADA *MOBILE ROBOT***

*PATH PLANNING BASED ON INDOOR POSITIONING SYSTEM FOR MOBILE ROBOT*

**Oleh :**

**Luqman Hakim**

**NRP. 1110 121 004**

**Dosen Pembimbing :**

**Akhmad Hendriawan, S.T., M.T.**

**NIP. 19750127.200212.1.003**

**Dr. Alrijadjis, Dipl.Eng., M.T.**

**NIP. 19720630.199903.1.003**

**PROGRAM STUDI D4 TEKNIK ELEKTRONIKA**

**DEPARTEMEN TEKNIK ELEKTRO**

**POLITEKNIK ELEKTRONIKA NEGERI SURABAYA**

**2016**

#### PERNYATAAN ORISINILITAS

Saya selaku penulis menyatakan bahwa Proyek Akhir ini adalah

benar-benar hasil karya saya sendiri, dan semua sumber/referensi baik yang dikutip maupun dirujuk telah saya nyatakan dengan benar.

Surabaya, 12 Agustus 2016

Penulis yang menyatakan,

Luqman Hakim

NRP. 1110 121 004

#### *PATH* *PLANNING* BERBASIS *INDOOR POSITIONING SYSTEM* PADA *MOBILE ROBOT*

**Oleh:**

**Luqman Hakim**

**NRP. 1110 121 004**

**Proyek Akhir ini diajukan sebagai salah satu syarat untuk**

**memperoleh gelar Sarjana Sains Terapan (S.ST.)**

**periode wisuda Oktober 2016**

**di**

**Politeknik Elektronika Negeri Surabaya**

**Disetujui dan disahkan pada tanggal 12 Agustus 2016**

**oleh:**

**Dosen Penguji Proyek Akhir :**

**1.**

**Dr. Eng. Agus Indra Gunawan, S.T., M.Sc.**

**NIP. 19760821.200112.1.002**

**2.**

**Firman Arifin, S.T., M.T.**

**NIP. 19740925.200112.1.002**

**3.**

**Retno Sukmaningrum, S.T., M.T.**

**NIP. 19700617.199903.2.001**

**Dosen Pembimbing:**

**1.**

**Akhmad Hendriawan, S.T., M.T.**

**NIP. 19750127.200212.1.003**

**2.**

**Dr. Alrijadjis, Dipl.Eng., M.T.**

**NIP. 19720630.199903.1.003**

**Mengetahui :**

**Ketua Program Studi D4 Teknik Elektronika**

**Ardik Wijayanto, S.T., M.T.**

**NIP. 19770620.200212.1.002**

#### ABSTRAK

Sistem navigasi dan *path* *planning* dibutuhkan oleh *mobile robot* agar dapat bergerak dalam suatu lingkungan. Beberapa metode dapat diterapkan sebagai navigasi robot di dalam ruangan (*indoor*) antara lain yaitu vision based location *positioning* yang memanfaatkan *database* gambar pada suatu ruangan yang dibandingkan dengan gambar yang diambil dari sebuah kamera pada mobile PC, kekurangan dari metode ini adalah pemrosesan gambar dapat terganggu oleh pengaruh inteensitas cahaya yang berubah – ubah pada ruangan, juga membutuhkan unit pemroses yang cepat dan *powerfull* tetapi membutuhkan daya yang besar sehingga tidak efisien jika diaplikasikan pada *mobile robot* yang mempnyai fungsi yang sederhana. Metode lain adalah dengan menggunakan odometri, yang mana metode ini tidak dapat menentukan posisi absolut robot, melainkan hanya dapat menentukan posisi relatif robot yang didasarkan pada perhitungan rotasi roda robot, tentunya metode ini hanya akan bekerja dengan baik jika lintasannya berupa bidang datar dan tidak menyebabkan *rotary encoder* selip. Penelitian ini bertujan untuk membuat *indoor positioning system* sebagai navigasi pada *mobile robot*. Sistem ini menerapkan *wireless sensor network* yang dipadukan dengan metode trilaterasi untuk menentukan posisi *mobile robot*. Juga sensor *magnetometer* digunakan untuk menentukan orienteasi robot. dari hasil pengujian didapatkan bahwa *indoor positioning system*  pada *path* *planning* *mobile robot* memiliki ketepatan dengan *error* rata – rata sebesar 4,34% pada pergerakan dari titik (0,0)m ke titik (2,2)m dan *error* rata – rata sebesar 3,3% pada pergerakan dari titik (0,0)m ke titik (4,4)m.

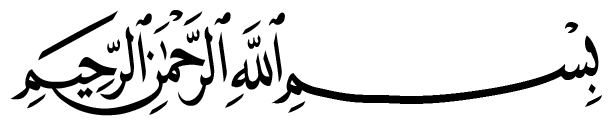
**Kata kunci :** ***mobile robot*, jaringan sensor nirkabel, navigasi*, indoor positioning***

#### ABSTRACT

*Navigation system and path path planning are needed for robot so the robot can do maneuver in its environment. Some methods can be applied as the navigation system of the robot in indoor environment, one of the method is vision based location positioning that implement image database and location model, The image sequence is obtained by a wearable mobile PC with camera, then the obtained image is compared with the image database. The disadvantage of this method are the lighting around the environment that can make the visual recognition fails. And also the processing unit demands high power input, so this method is not efficient if applied to mobile robot that has very few functionality. The other method is* odometri*, a method to determine the relative position of the robot by calculating the wheel rotation. Of course this method will not work well in environment that has slippery ground making the rotary encoder sensor does miss reading. This research presents the implementation of indoor positioning system as the navigation system of the mobile robot. This system uses wireless sensor network to do ranging measurement then do a calculation using trilateration method to determine the positioning of the mobile robot. Also magnetometer sensor used to determine the heading of the robot. From the examination we acquire that path planning using this indoor positioning system has average error of 4,34% when robot moves from initial position (0,0)m to target position (2,2)m and average error of 3,34 when robot moves from initial position (0,0)m to target position (4,4)m.*

***Keyword :******mobile robot, wireless sensor network, navigation, indoor positioning***

#### KATA PENGANTAR

****

Alhamdulillah, segala puji syukur bagi Allah SWT karena berkat rahmat dan hidayah-Nya, penulis dapat menyelesaikan proyek akhir yang berjudul :

**“*PATH* *PLANNING* BERBASIS *INDOOR POSITIONING SYSTEM*  PADA *MOBILE ROBOT*”**

Pembuatan dan penyusunan proyek akhir ini diajukan sebagai salah satu syarat untuk menyelesaikan studi Diploma-4 (D4) dan memperoleh gelar Sarjana Sains Terapan (S.ST.) di Program Studi Teknik Elektronika Politeknik Elektronika Negeri Surabaya.

Penulis berusaha secara optimal dengan segala pengetahuan dan informasi yang didapatkan dalam menyusun laporan proyek akhir ini. Namun, penulis menyadari berbagai keterbatasannya, karena itu penulis memohon maaf atas keterbatasan materi laporan proyek akhir ini. Penulis sangat mengharapkan masukan berupa saran dan kritik yang membangun demi kesempurnaan laporan proyek akhir ini.

Demikian besar harapan penulis agar laporan proyek akhir ini dapat bermanfaat bagi pembaca, khususnya dalam mempelajari sistem penghindaran halangan pada robot berkaki enam secara *otomatis*.

Surabaya, 12 Agustus 2016

**Penulis**

#### UCAPAN TERIMA KASIH

Dengan penuh rasa syukur kehadirat Allah SWT dan tanpa menghilangkan rasa hormat yang mendalam, saya selaku penyusun dan penulis mengucapkan terima kasih yang sebesar-besarnya kepada pihak-pihak yang telah membantu penulis untuk menyelesaikan proyek akhir ini, penulis mengucapkan terima kasih kepada :

* + - 1. Kedua orang tua saya tercinta, Bapak **Moch. Sholeh** dan Ibu **Siti Muchlisoh** yang selalu memberi do’a dan dukungannya, juga kakak - kakak tercinta.
      2. Bapak **Dr. Zainal Arif, S.T., M.T** selaku Direktur PENS.
      3. Bapak **Dr. I Gede Puja Astawa, S.T.,** **M.T.** selaku Kepala Departemen Teknik Elektro PENS.
      4. Bapak **Ardik Wijayanto, S.T., M.T.** selaku Ketua Program Studi D4 Teknik Elektronika PENS.
      5. Bapak **Akhmad Hendriawan, S.T., M.T.**, danBapak **Dr. Alrijadjis, Dipl.Eng., M.T** selaku dosen pembimbing proyek akhir saya.
      6. Seluruh Bapak dan Ibu dosen yang telah membimbing dan membekali ilmu kepada penulis selama penulis menempuh pendidikan di kampus tercinta, Politeknik Elektronika Negeri Surabaya.
      7. Seluruh asisten dosen dan karyawan yang telah memberi banyak bantuan untuk penyelesaian proyek akhir ini.
      8. Teman-teman kelas 4 D4 Elka A khususnya dan seluruh angkatan 2012 Teknik Elektronika PENS.
      9. Semua pihak yang telah membantu penulis hingga terselesainya proyek akhir ini yang tidak dapat penulis sebutkan.
      10. Tidak lupa ucapan terima kasih kepada teman terkasih yang selalu menemani dan mendukung saya, **Trisna Irnanti.**

Semoga Allah S.W.T. selalu memberikan perlindungan, rahmat dan nikmat-Nya bagi kita semua. Amin.

#### PERSETUJUAN PUBLIKASI TERBATAS

Sebagai Sivitas Akademika Politeknik Elektronika Negeri Surabaya, saya yang bertanda tangan di bawah ini:

Nama : **Luqman Hakim**

NRP : 1110 121 004

Program Studi : Teknik Elektronika

Departemen : Teknik Elektro

demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Politeknik Elektronika Negeri Surabaya Hak Bebas Royalti Non-eksklusif (Non-exclusive Royalty-Free Right) atas proyek akhir saya yang berjudul:

***PATH* *PLANNING* BERBASIS *INDOOR POSITIONING SYSTEM*  PADA *MOBILE ROBOT***

beserta perangkat yang ada (jika diperlukan), yang oleh karenanya Politeknik Elektronika Negeri Surabaya dengan ini berhak menyimpan, mengalih-media-kan atau mengalih-format-kan, mengelola dalam pangkalan data (database), merawat, dan memublikasikan tugas akhir saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya.

Surabaya, 12 Agustus 2016

Penulis

Luqman Hakim

NRP. 1110 121 004

#### DAFTAR ISI

[HALAMAN JUDUL i](file:///C:\Users\Luqman_Hakim\Google%20Drive\TA%20lulus%202016\dokumentasi%20TA\doc\revisi\Bismillah%20lulus%202016%20buku%20PA2%203.2.docx#_Toc458389548)

[PERNYATAAN ORISINILITAS ii](#_Toc458389549)

[LEMBAR PENGESAHAN iii](#_Toc458389550)

[ABSTRAK iv](#_Toc458389551)

[ABSTRACT v](#_Toc458389552)

[KATA PENGANTAR vi](#_Toc458389553)

[UCAPAN TERIMA KASIH vii](#_Toc458389554)

[PERSETUJUAN PUBLIKASI TERBATAS viii](#_Toc458389555)

[DAFTAR ISI ix](#_Toc458389556)

[DAFTAR GAMBAR xi](#_Toc458389557)

[DAFTAR TABEL xiv](#_Toc458389558)

[BAB I PENDAHULUAN 1](#_Toc458389559)

[1.1. Latar Belakang 1](#_Toc458389560)

[1.2. Perumusan Masalah 2](#_Toc458389561)

[1.3. Batasan Masalah 2](#_Toc458389562)

[1.4. Tujuan dan Manfaat 2](#_Toc458389563)

[1.4.1. Tujuan 2](#_Toc458389564)

[1.4.2. Manfaat 3](#_Toc458389565)

[1.5. Metodologi 3](#_Toc458389566)

[1.6. Sistematika Pembahasan 4](#_Toc458389567)

[BAB II TINJAUAN PUSTAKA 5](#_Toc458389568)

[2.1. OdometriSebagai Sistem Navigasi 5](#_Toc458389569)

[2.1.1. Jarak dan Arah Hadap 5](#_Toc458389570)

[2.1.2. Identifikasi *Error* 6](#_Toc458389571)

[2.2. Trilaterasi 7](#_Toc458389572)

[2.3. TeknikPengukuranJarakdengan Sinyal *Wireless* 9](#_Toc458389573)

[2.4. Kompas 12](#_Toc458389574)

[2.5. *Heading* 13](#_Toc458389575)

[2.6. Sensor 15](#_Toc458389576)

[2.6.1. *Magnetometer* 15](#_Toc458389577)

[2.6.2. *Rotary encoder* 16](#_Toc458389578)

[2.6.3. *Wireless sensor network* DWM1000 17](#_Toc458389579)

[2.7. Mikrokontroler ARM Cortex-M4 STM32F407 18](#_Toc458389580)

[2.8. Komunikasi *Serial* UART 20](#_Toc458389581)

[2.9. Komunikasi I2C 22](#_Toc458389582)

[2.10. *Display* (*LCD*) 24](#_Toc458389583)

[2.11. Motor DC 25](#_Toc458389584)

[BAB III PERANCANGAN DAN PEMBUATAN SISTEM 27](#_Toc458389585)

[3.1. Perancangan dan Pembuatan Mekanik 27](#_Toc458389586)

[3.2. Perancangan dan Pembuatan Perangkat Elektronik 28](#_Toc458389587)

[3.2.1. Rangkaian *Regulator* 5V dan 3,3V 31](#_Toc458389588)

[3.2.2. Rangkaian *Minimum System* STM32F407VG 32](#_Toc458389589)

[3.2.3. USART pada Mikrokontroler STM32F407VG 33](#_Toc458389590)

[3.2.4. I2C pada Mikrokontroler STM32F407VG 34](#_Toc458389591)

[3.2.5. Akses Sensor *Rotary encoder* 35](#_Toc458389592)

[3.2.6. Akses Sensor *Magnetometer* 38](#_Toc458389593)

[3.2.7. Motor DC 39](#_Toc458389594)

[3.2.8. Rangkaian *LCD* 16x4 41](#_Toc458389595)

[3.2.9. Akses Modul DWM1000 41](#_Toc458389596)

[3.3. Perancangan dan Pembuatan Perangkat Lunak 43](#_Toc458389597)

[3.3.1. Algoritma Menentukan *Heading* 43](#_Toc458389598)

[3.3.2. Menentukan Posisi Robot dengan Metode Trilaterasi 46](#_Toc458389599)

[3.3.3. Menentukan Arah *Bearing* dan Jarak Target 50](#_Toc458389600)

[3.3.4. Menentukan Gerakan Robot 53](#_Toc458389601)

[3.3.5. Kontrol Arah Hadap Robot 58](#_Toc458389602)

[BAB IV PENGUJIAN DAN ANALISIS SISTEM 61](#_Toc458389603)

[4.1. Pengujian Mekanik 61](#_Toc458389604)

[4.2. Pengujian Mikrokontroler 65](#_Toc458389605)

[4.3. Pengujian Sensor *Magnetometer* 67](#_Toc458389606)

[4.4. Pengujian Odometri 72](#_Toc458389607)

[4.5. Pengujian Modul DWM1000 77](#_Toc458389608)

[4.6. Pengujian *Ranging* Modul DWM1000 79](#_Toc458389609)

[4.7. Pembacaan Jarak Dan Penentuan Posisi Modul DWM1000 81](#_Toc458389610)

[4.8. Pengujian ketepatan *Path* *Planning* robot 86](#_Toc458389611)

[BAB V PENUTUP 95](#_Toc458389612)

[5.1. Kesimpulan 95](#_Toc458389613)

[5.2. Saran 96](#_Toc458389614)

[DAFTAR PUSTAKA 97](#_Toc458389615)

[LAMPIRAN 99](#_Toc458389616)

[A1. Gambar skematik Mikrokontroler Utama 99](#_Toc458389617)

[A2. Gambar *Board* Mikrokontroler Utama 100](#_Toc458389618)

[A3. Gambar *Board* *Driver* Motor 101](#_Toc458389619)

[BIODATA PENULIS 102](#_Toc458389620)

#### DAFTAR GAMBAR

[**Gambar 1.1.** Metode pelaksanaan kegiatan proyek akhir. 3](#_Toc458326268)

[**Gambar 2.1.** Geometry odometri[4]. 5](#_Toc458389621)

[**Gambar 2.2.** Sudut θ, jarak dan koordinat X,Y[4]. 6](#_Toc458389622)

[**Gambar 2.3.** Ilustrasi akumulasi *error* arah hadap robot[4]. 7](#_Toc458389623)

[**Gambar 2.4.** Ilustrasi model trilaterasi[7]. 8](#_Toc458389624)

[**Gambar 2.5.** Penempatan *anchor* dalam bidang[7]. 9](#_Toc458389625)

[**Gambar 2.6.** Sistem kerja TDOA[6]. 10](#_Toc458389626)

[**Gambar 2.7.** Sinyal TDOA[7]. 11](#_Toc458389627)

[**Gambar 2.8.** Karakteristik jarak terhadap RSSI[8]. 11](#_Toc458389628)

[**Gambar 2.9.** Kompas analog[18]. 13](#_Toc458389629)

[**Gambar 2.10.** Ilustrasi *heading*[17]*.* 14](#_Toc458389630)

[**Gambar 2.11.** Menentukan jarak dan titik akhir (*finish*)[17]. 14](#_Toc458389631)

[**Gambar 2.12.** Modul sensor GY-85[10]. 15](#_Toc458389632)

[**Gambar 2.13.** Blok diagram sensor *magnetometer* HMC5883L [11]. 16](#_Toc458389633)

[**Gambar 2.14.** Rangkaian *rotary encoder* [5] 16](#_Toc458389634)

[**Gambar 2.15.** Modul DWM1000[12]. 17](#_Toc458389635)

[**Gambar 2.16.** Rangkaian dalam modul DWM1000[12]. 18](#_Toc458389636)

[**Gambar 2.17.** Konfigurasi pin STM32F407VG [13]. 20](#_Toc458389637)

[**Gambar 2.18.** Sinyal komunikasi UART[19]. 21](#_Toc458389638)

[**Gambar 2.19.** Kondisi sinyal *start* dan *stop*[20]. 22](#_Toc458389639)

[**Gambar 2.20.** Sinyal ACK dan NACK[20]. 23](#_Toc458389640)

[**Gambar 2.21.** Transfer *bit* pada I2C *bus*[20]. 23](#_Toc458389641)

[**Gambar 2.22** Tampilan *LCD* 2x16[14]. 24](#_Toc458389642)

[**Gambar 2.23.** Bagian dalam motor DC[15]. 25](#_Toc458389643)

[**Gambar 3.1.** Gambar mekanik robot tampak atas. 27](#_Toc458389644)

[**Gambar 3.2.** Gambar mekanik roda robot tampak samping. 28](#_Toc458389645)

[**Gambar 3.3.** Blok diagram sistem elektronik. 29](#_Toc458389646)

[**Gambar 3.4.** *Board* sistem kontrol utama. 29](#_Toc458389647)

[**Gambar 3.5.** *Board* *driver* motor DC. 30](#_Toc458389648)

[**Gambar 3.6.** *Board* sistem *Node.* 30](#_Toc458389649)

[**Gambar 3.7.** Integrasi mekanik, *board* sistem kontrol, dan *tag* *node.* 31](#_Toc458389650)

[**Gambar 3.8.** Rangkaian *regulator* 5V dan 3,3V. 31](#_Toc458389651)

[**Gambar 3.9.** Minimum sistem STM32F407VG. 32](#_Toc458389652)

[**Gambar 3.10.** Raingkaian *interfacing* Arduino Pro Mini. 33](#_Toc458389653)

[**Gambar 3.11.** Raingkaian *interfacing* modul Blueatooth HC-05. 34](#_Toc458389654)

[**Gambar 3.12.** Rangkaian *interfacing* modul sensor GY-85. 35](#_Toc458389655)

[**Gambar 3.13.** Rangkaian *rotary encoder.* 35](#_Toc458389656)

[**Gambar 3.14.** Ilustrasi sinyal *rotary encoder.* 36](#_Toc458389657)

[**Gambar 3.15.** Rangkaian *driver* motor DC. 39](#_Toc458389658)

[**Gambar 3.16.** Rangkaian *LCD* 16x2. 41](#_Toc458389659)

[**Gambar 3.17.** Rangkaian *interface* Modul DWM1000[12]. 42](#_Toc458389660)

[**Gambar 3.18.** Rangkaian *interface* DWM1000 dengan Arduino[16]. 42](#_Toc458389661)

[**Gambar 3.19.** Menentukan *heading* dengan *magnetometer*. 43](#_Toc458389662)

[**Gambar 3.20.** *Flowchart*algoritma menentukan *heading.* 44](#_Toc458389663)

[**Gambar 3.21.** Peletakkan *anchor* *node.* 46](#_Toc458389664)

[**Gambar 3.22.** Penentuan letak *anchor* *node* dan robot 48](#_Toc458389665)

[**Gambar 3.23.** Arah *bearing* terhadap sudut 0° arah kompas 50](#_Toc458389666)

[**Gambar 3.24.** Ilustrasi penghitungan jarak. 51](#_Toc458389667)

[**Gambar 3.25.** Ilustrasi penentuan arah *bearing*. 54](#_Toc458389668)

[**Gambar 3.26.** Ilustrasi penentuan arah rotasi robot. 55](#_Toc458389669)

[**Gambar 3.27.** Selisih sudut antara *heading* dengan arah *bearing*. 56](#_Toc458389670)

[**Gambar 3.28.** Selisih sudut antara *heading* dengan arah *bearing*. 57](#_Toc458389671)

[**Gambar 3.29.**Ilustrasi penentuan *error* *heading* robot. 58](#_Toc458389672)

[**Gambar 3.30.** *Flowchart* kontrol pergerakan robot. 60](#_Toc458389673)

[**Gambar 4.1.** Grafik rata – rata simpangan mekanik pada sumbu y. 62](#_Toc458389674)

[**Gambar 4.2.** Grafik rata – rata simpangan mekanik pada sumbu x. 63](#_Toc458389675)

[**Gambar 4.3.** Ilustrasi pengujian kesalahan simpangan mekanik, 64](#_Toc458389676)

[**Gambar 4.4.** Tampilan pada *software* Hterm. 66](#_Toc458389677)

[**Gambar 4.5.** *Plotting* data *magnetometer*. 71](#_Toc458389678)

[**Gambar 4.6.** Ilustrasi pengujian sensor *magnetometer* 72](#_Toc458389679)

[**Gambar 4.7.** Grafik pengujian odometri pada sumbu y. 75](#_Toc458389680)

[**Gambar 4.8.** Grafik pengujian odometri pada sumbu x. 75](#_Toc458389681)

[**Gambar 4.9.** Ilustrasi pengujian odometri. 77](#_Toc458389682)

[**Gambar 4.10.** Komunikasi Arduino dengan DWM1000. 78](#_Toc458389683)

[**Gambar 4.11.** Pengujian konektifitas Arduino dengan DWM1000. 78](#_Toc458389684)

[**Gambar 4.12.** *Plotting* pengujian jarak modul DWM1000. 80](#_Toc458389685)

[**Gambar 4.13.** Proses pengujian *ranging* modul DWM1000. 81](#_Toc458389686)

[**Gambar 4.14.** Grafik pengukuran jarak DWM1000. 83](#_Toc458389687)

[**Gambar 4.15.** Perbandingan DWM1000 dengan RSSI Bluetooth. 85](#_Toc458389688)

[**Gambar 4.16.** Grafik pengujian posisi dengan modul DWM1000. 86](#_Toc458389689)

[**Gambar 4.17.** Peletakkan robot dan *anchor* *node.* 87](#_Toc458389690)

[**Gambar 4.18.** Grafik pergerakan robot terhadap garis acuan. 90](#_Toc458389691)

[**Gambar 4.19.** Akurasirobot mencapai titik akhir (2,2)m. 92](#_Toc458389692)

[**Gambar 4.20.** Akurasirobot mencapai titik akhir (4,4)m. 93](#_Toc458389693)

#### DAFTAR TABEL

[**Tabel 2.1.** Estimasi jarak RSSI Bluetooth pemodelan 1[9]. 12](#_Toc458326554)

[**Tabel 2.2.** Estimasi jarak RSSI Bluetooth pemodelan 2[9]. 12](#_Toc458326555)

[**Tabel 2.3.** Susunan kaki *LCD* 16 X 2. 24](#_Toc458326556)

[**Tabel 3.1.** *Read* / *write* *address* dari HMC5883L. 38](#_Toc458326557)

[**Tabel 3.2.** Alamat *register* HMC5883L. 38](#_Toc458326558)

[**Tabel 3.3.** *Frame* data komunikasi. 47](#_Toc458326559)

[**Tabel 4.1.** Pengukuran Perpindahan Robot pada Sumbu Y. 62](#_Toc458389694)

[**Tabel 4.2.** Pengukuran perpindahan robot pada sumbu x. 63](#_Toc458389695)

[**Tabel 4.3.** Pengujian interupsi *timer* dan komunkasi USART. 66](#_Toc458389696)

[**Tabel 4.4.** Pengujian sensor *magnetometer* sebelum dikalibrasi. 68](#_Toc458389697)

[**Tabel 4.5.** Pengujian sensor *magnetometer* setelah dikalibrasi. 69](#_Toc458389698)

[**Tabel 4.6.** Pengujian odometri dengan target 150cm. 73](#_Toc458389699)

[**Tabel 4.7.** Pengujian odometri dengan gangguan dengan target 150cm. 73](#_Toc458389700)

[**Tabel 4.8.** Pengujian odometri dengan target 300cm. 74](#_Toc458389701)

[**Tabel 4.9.** Pengujian odometri dengan ganguan dengan target 300cm. 74](#_Toc458389702)

[**Tabel 4.10.** Pengujian modul DWM1000. 80](#_Toc458389703)

[**Tabel 4.11.** Pegujianpengukuran jarak dengan DWM1000. 82](#_Toc458389704)

[**Tabel 4.12.** Latensi pengiriman data 3 jarak *anchor.* 83](#_Toc458389705)

[**Tabel 4.13.**Perbandingan jarak DWM1000 dengan RSSI Bluetooth. 84](#_Toc458389706)

[**Tabel 4.14.** Pengujian posisi dengan modul DWM1000. 85](#_Toc458389707)

[**Tabel 4.15.**Posisi pergerakan robot dari ttik (0,0)m ke titik (2,2)m. 88](#_Toc458389708)

[**Tabel 4.16.** Posisi pergerakan robot dari ttik (2,2)m ke titik (4,1)m. 88](#_Toc458389709)

[**Tabel 4.17.** Posisi pergerakan robot dari ttik (4,1)m ke titik (4,4)m. 89](#_Toc458389710)

[**Tabel 4.18.** Akurasi pencapaian tujuan (2,2)m dari tiik awal (0,0)m. 91](#_Toc458389711)

[**Tabel 4.19.** Akurasi pencapaian tujuan (4,4)m dari tiik awal (0,0)m. 93](#_Toc458389712)

# PENDAHULUAN

## **Latar Belakang**

Pada saat ini perkembangan robot sudah sangat berkembang. Robot banyak digunakan di dalam kehidupan manusia, membantu pekerjaan manusia. Robot sudah banyak ditemui di lingkungan industri, bahkan sudah dapat ditemui di lingkungan rumah tangga. Banyak model robot yang sudah dibuat, model – model robot robot dikelompokkan berdasarkan bentuk dan fungsinya. Salah satu model robot adalah *mobile robot*, yaitu robot yang mampu bergerak dan berpindah posisi dalam suatu lingkungan dimana robot tersebut berada.

Penelitian ini bertujuan untuk membuat sebuah sistem navigasi pada *mobile robot* yang berjalan di dalam ruangan, system yang dibuat untuk dapat menentukan posisi robot di dalam suatu rangan juga dapat menentukan arah hadap robot atau *heading.* Sistem sensor yang dibuat dapat digunakan untuk mendapatkan posisi absolut dan orientasi *mobile robot* yang tidak terpengaruh oleh gangguan yang terjadi pada sensor relatif seperti *rotary encoder* yang dipasang pada roda robot. Posisi dan orientasi robot akan digunakan untuk bernavigasi pada lingkungan tempat robot berada[1]. *Mobile robot* menggunakan *path* *planning* untuk mendapatkan jalur yang terbaik ketika bergerak dari satu posisi ke posisi lain[2].

Salah satu metode *positioning* yang digunakan adalah *vision based location positioning* yang menggunakan database gambar dan pemodelan lokasi[3]. Metode ini sangat membutuhka unit pemroses yang cepat dan membutuhkan daya yang besar sehingga tidak efisien apabila digunakan pada *mobile robot*. Dari penelitian sebelumnya tentang odometri yang dilakukan oleh Jusuf Dwi Kariyanto dan Bayu Sandi Marta, dalam kesimpulanya menyatakan bahwa, “Pencapaiantargetdenganteknikodometrisangat dipengaruhiolehseliproda.Semakinpanjangrute yangberbelok-belokmakapotensiselip semakinbesar[4]”. “Odometrimemilikibatasanyaitudapatbekerja dengan baik jika lintasan nya berupa bidang datar dan tidak menyebabkan *rotary encoder* selip[5]”. Maka dalam penelitian ini digunakan metode *global* *positioning* yang menerapkan *wireless sensor network*.

*Error* pada *local* *positioning* yang menggunakan *rotary encoder*, disebabkan oleh slip pada roda robot, berdampak pada ketidak tepatan robot dalam membuat *path* *planning* untuk dirinya. Robot tidak dapat memperbaiki *error* slip pada odometri yang menggunakan *rotary encoder*. Maka digunakan *positioning* dengan menggunakan *wireless sensor network* untuk menentukan posisi robot secara global.

Untuk membuat sebuah *indoor* *positioning* diperlukan perangkat *wireless* yang mampu menentukan posisinya terhadapa perangkat *wireless* lainya, seperti yang sudah dilakukan pada *global* *positioning* *system* atau biasa disebut GPS. Metode untuk menentukan posisi pada global *positioning* system menggunakan metode trilaterasi dimana posisi perangkat yang dicari posisinya ditentukan oleh jarak perangkat tersebut dengan satelit yang sudah diketahui posisinya. Pada penelitian ini perangkat yang digunakan untuk menentukan posisi secara global adalah jaringan sensor nirkabel yang mampu menghitung jarak antar *node*.

## **Perumusan Masalah**

Permasalahan yang dibahas dalam proyek akhir ini adalah sebagai berikut :

* Bagaimana membuat *hardware* dan *software* untuk robot dan untuk *indoor postioning system.*
* Bagaimana membuat *indoor positioning system* menggunakan jaringan sensor nirkabel.
* Bagaimana membuat *path* *tracking* dan *path* *planning* dengan menggunakan *indoor positioning system* .

## **Batasan Masalah**

Adapun batasan-batasan masalah yang dibuat agar dalam pengerjaan proyek akhir ini dapat berjalan dengan baik adalah sebagai berikut :

* Lintasan robot yang akan digunakan adalah bidang datar yang tidak licin dengan ukuran 5m x 5m.
* Posisi awal dan posisi target robot sudah ditentukan.
* Tidak ada halangan pada lintasan robot.

## **Tujuan dan Manfaat**

### **Tujuan**

Tujuan dibuatnya proyek akhir ini adalah merancang sistem navigasi pada *mobile robot* dengan meggunakan jaringan sensor nirkabel sebagai alternatf dari system navigasi yang menggunakan odometri dengan *rotary encoder* pada roda robot.

### **Manfaat**

Manfaat dibuatnya proyek akhir ini adalah Menerapkan sistem navigasi menggunakan jaringan sensor nirkabel pada *mobile robot*.

## **Metodologi**

Metodologi yang digunakan dalam proyek akhir ini adalah sebagai berikut dtunjukkan pada gambar 1.1:

Study Literatur

Perancangan Sistem

Pembuatan Mekanik

Pembuatan *Hardware*

Pembuatan *Software*

Pengujian dan Evaluasi Sistem

Apakah sudah sesuai?

Cek *Hardware* OK?

Pembuatan laporan

Ya

Ya

Ya

**Gambar 1.1.** Metode pelaksanaan kegiatan proyek akhir.

Studi literatur tentang permasalahan yang ada melalui media-media informasi (buku, internet, dll) maupun dari sumber-sumber yang terkait

Perancangan sistem, pembuatan perangkat keras (*hardware*), dan perangkat lunak (*software*) yaitu dengan membuat diagram blok, algoritma dan *state-state* dari metode yang dipakai.

Uji coba dan verifikasi yaitu pengujian perangkat keras dan perangkat lunak dengan membandingkan hasil yang diperoleh dengan apa yang ingin dicapai.

Melakukan analisis yaitu dengan melakukan perbaikan apabila terjadi kesalahan baik pada alat maupun program sejauh kesalahan tersebut dapat diperbaiki serta memperhitungkan persen *error* dari hasil data yang didapat.

Menyusun laporan untuk mengetahui apakah hasil yang didapatkan sudah sesuai dengan yang diinginkan atau tidak, setelah itu dibuat kesimpulan.

## **Sistematika Pembahasan**

Sistematika pembahasan dari penyusunan proyek akhir ini direncanakan sebagai berikut:

**BAB I PENDAHULUAN**

Pada bab ini akan dibahas tentang latar belakang, tujuan, permasalahan, batasan masalah dan metodologi yang digunakan pada proyek akhir ini.

**BAB II TINJAUAN PUSTAKA**

Bab ini membahas mengenai materi-materi yang berkaitan dengan penyelesaian proyek, yang didapatkan dari sumber-sumber terkait yang berhubungan dengan pembuatan proyek ini.

**BAB III PERENCANAAN DAN PEMBUATAN SISTEM**

Dalam tahap ini akan dibahas bagaimana system tersebut beroperasi dan komponen-komponen yang menyusun system ini.

**BAB IV PENGUJIAN DAN ANALISIS SISTEM**

Bab ini berisi mengenai pengujian terhadap alat yang telah dibuat dan fungsi dari masing-masing blok sistem. Selain itu juga dibahas mengenai kinerja dari masing-masing blok tersebut.

**BAB V PENUTUP**

Dalam bab ini berisi tentang kesimpulan dan saran-saran terhadap proyek akhir yang telah dibuat.

**DAFTAR PUSTAKA**

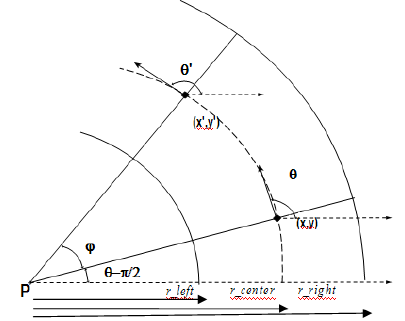
Pada bagian ini berisi tentang referensi-referensi yang telah dipakai oleh penulis sebagai acuan dan penunjang serta parameter yang mendukung penyelesaian proyek akhir ini baik secara praktis maupun teoritis.

# TINJAUAN PUSTAKA

## **OdometriSebagai Sistem Navigasi**

Odometriadalah pengambilan data dari pergerakan aktuator untuk memperkirakan perubahan posisi dari sebuah sistem penggerak. Odometri digunakan untuk memperkirakan (bukan menentukan) posisi relatif terhadap posisi awal[4].

OdometriRobot dengan Sistem penggerak *differential*  roda kanan dan roda kiri yang memiliki sistem sebagai berikut seperti tampak pada gambar 2.1 :



**Gambar 2.1.** Geometry odometri[4].

### **Jarak dan Arah Hadap**

Untuk memperkirakan posisi relatif robot, digunakan perhitungan jumlah pulsa yang dihasilkan oleh sensor *rotary encoder* setiap satuan ukuran jarak. Misalkan kita menggunakan satuan cm. Untuk mendapatkan jumlah pulsa setiap satu kali putaran roda digunakan persamaan 2.1 dan 2.2 [6] :

*K roda = 2 x x r* (2.1)

*pulsa\_per\_cm = resolusi encoder / K roda* (2.2)

Pada sistem penggerak *differential* terdapat dua roda yaitu roda kanan dan roda kiri dan dimisalkan jumlah *pulsa\_per\_cm* untuk roda kanan adalah *right\_encoder* dan roda kiri adalah *left\_encoder* dan jarak antara dua roda adalah *wheel\_base* maka didapatkan jarak tempuh (*distance*) dan sudut orientasi *( θ )*[6]*.* Rumusnya adalah persamaan 2.3 dan 2.4 :

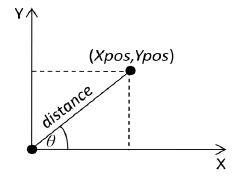
*distance = (left\_encoder+right\_encoder)/2* (2.3)

*θ= (left\_encoder-right\_encoder)/wheel\_base* (2.4)

Karena *θ* adalah sudut dalam *radian* maka untuk mengetahui sudut dalam derajat (*heading*) digunakan rumus persaaman 2.5 berikut[6]:

*heading = θ x*  (2.5)

Dari ketentuan di atas didapatkan bahwa nilai *heading* akan bernilai negative (-) ketika robot berputar melawan arah jarum jam dan akan bernilai positif (+) ketika robot berputar searah dengan jarum jam. Dengan mengetahui jarak dan sudut (*distance* dan *θ*) maka kita dapat mengetahui koordinat X dan koordinat Y dengan persamaan 2.6 dan 2.7[6]:



**Gambar 2.2.** Sudut θ, jarak dan koordinat X,Y[4].

Dari gambar 2.2, maka koordinat dari robot dapat kita ketahui dengan persamaan odometri [4]:

*Xpos = distance x sin (θ)* (2.6)

*Ypos = distance x cos (θ)* (2.7)

### **Identifikasi *Error***

Untuk menentukan *error* arah hadap dari robot (*heading error*) terhadap titik tujuan maka digunakan teorema *phytagoras* yang akan menghasilkan posisi (*path*) saat ini dalam koordinat (X,Y) danjarak terhadap titik tujuan (*target\_distance*)[6], berikut perhitungannya:

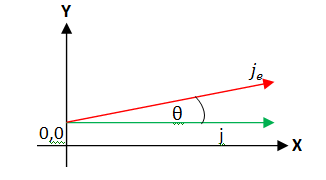
*x = Xtujuan – Xpos* (2.8)

*y = Ytujuan – Ypos* (2.9)

*target\_distance =* (2.10)

Terdapat beberapa hal yang mempengaruhi besarnya kesalahan dari perhitungan odometri antara lain adalah:

* Berjalan di atas lantai yang tidak rata.
* Terdapat benda yang tak terduga dilantai.
* Roda selip.
* Lantai yang licin.
* Kecepatan robot yang melebihi kemampuan sensor.
* Perubahan kecepatan yang mendadak.



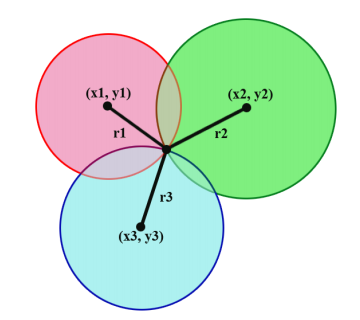
**Gambar 2.3.** Ilustrasi akumulasi *error* arah hadap robot[4].

Pada gambar 2.3 menunjukkan kesalahan yang terjadi saat perhitungan odometri akan terus diakumulasi dari waktu ke waktu sebagai kesalahan (*error*) sehingga hasil akhir perhitungan odometriadalah kesalahan yangmenumpuk, sebagian besar dari selip roda dan permukaan yang tidak rata.Tapi untuk perhitungan *theta*(θ) atau arah hadap robot, kesalahan beberapa derajat saja akan berdampak sangat besar, sehingga menghasilkan kesalahan besar di kedua posisi X dan Y. Teknik odometri lebih efektif digunakan untuk target yang posisinya berada pada garis lurus (*j*) dari robot akan tetapi apabila semakin jauh target yang dituju serta posisi target (*je*) membentuk sudut dari posisi awal robot maka kemungkinan *error* yang terjadi semakin besar ketika robot telah sampai pada posisi yang dituju tersebut[5].

## **Trilaterasi**

Trilaterasiadalahprosesmencarikoordinat sebuahtitikberdasarkanjaraktitiktersebutke minimal3buahkoordinatyangsudahdiketahui[6]. Sebuah lingkaran mempunyai persamaan seperti pada persamaan 2.11 berikut:

x2 + y2 = r2…………………………………………………..(2.11)



**Gambar 2.4.** Ilustrasi model trilaterasi[7].

Dari ketiga lingkaran pada gambar 2.4, maka didapatkan persamaan untuk masing – masing lingkaran sebagai berikut:

**(x - x1)2+(y - y1)2 = r1**………………………………...………...(2.12)

**(x - x2)2 + (y - y2)2 = r2**………………………………...………..(2.13)

**(x - x3)2 + (y - y3)2 = r3**………………………………...………..(2.14)

Dari persamaan di atas maka dapat dihitung x dan y.MisalkaningindiketahuiKoordinattitikB,dan koordinatdarititikP1,P2danP3sudahdiketahui dengan mengukur r1 (jarak antara B dengan P1), maka koordinatBpastiterletakpadakelilinglingkaran dengan jari-jari r1.

Lalu dengan mengukur r2 (jarak antara B dengan P2),makakoordinattitikpastiterletakdiAatauB, yang merupakan perpotongan antara kedua lingkaran. Ketika diukur jarak r3 (jarak antara B dengan P3), kita sudahmendapatkansebuahtitikB,yangmerupakan perpotongan antara ketiga buah lingkaran. Substitusi dua lingkaran, menghasilkan persamaan linear yang berseberangan dengan irisan yang dihasilkan.

Perhitungan dari dua persamaan linear sebelumnya menghasilkan perkiraan nilai koordinat x[7].

…………..………………………..…………………………(2.15)

Nilai *x* yang dihasilkan dari persamaan di atas disubtitusikan ke dalam persamaan di bawah sehingga menghasilkan kordinat titil *y*[7].

…………..……………………………………………………(2.16)

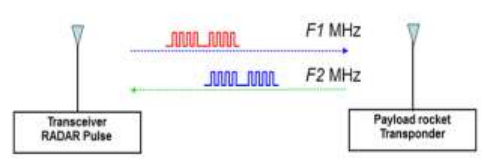
Posisi penempatan *anchor* sebagai titik koordinat yang telah diketahui sangat berpengaruh. Untuk posisi paling optimal didapatkan pada ruang berbentuk persegi dengan titik koordinat yang membentuk segitiga seperti pada gambar 2.5.



**Gambar 2.5.** Penempatan *anchor* dalam bidang[7].

## **TeknikPengukuranJarakdengan Sinyal *Wireless***

SistemRadar(*Radio Detection and Ranging*)Sekunder dalammenentukanjarakantara *transceiver* (*transmitter receiver*)dantransponder(*transmitting responder*)menggunakanteknikTDOA(*Time Difference Of Arrival*). Teknik TDOAini bekerja sesuai dengan gambaran yang ada di gambar 2.6.



**Gambar 2.6.** Sistem kerja TDOA[6].

Untukmenghitungjarakantara *Transceiver* dan *Transponder*,padamulanya *Transceiver* mengirimkansinyalsebesarF1MHzdiwaktut1 kepada Transponder. Kemudian *Transponder* menangkap sinyal F1, menguatkandanmemancarkankembalike *Transceiver* denganfrekuensiF2MHzdiwaktut2 Sinyalyangdikirim danterimapadasaatyanghampirbersamaanmemiliki perbedaanwaktuyangdapatdisimbolkandengan∆t Penundaanwaktutersebutkemudiandikalikandengan kecepatancahaya(c)sebesar299.729.458m/sdandibagi duamaka akan di dapatkan jarak obyekyang dicari.Secara matematis,untukmenghitungjarakantara *Transceiver* dan *Transponder* dapat dituliskan seperti persamaan 2.17:

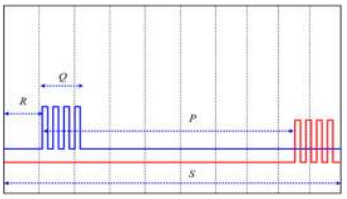
……………………………(2.17)

Dimana: T1 = Waktu data karakter dikirim

T2 = Waktu data karakter diterima

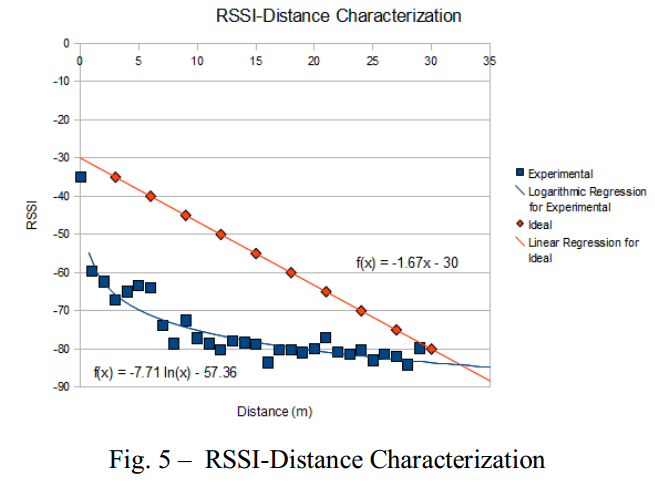
Denganmengasumsikanbahwawaktupemrosesan secaraperangkatlunak *transceiver* dan *transponder* diabaikan.Padapemrosesandanpengirimandatapada komunikasi *serial* selalumenghasilkanrugi-rugiyangakan mempengaruhibesarnyawaktutundayangdicari.Namun, pada penelitian ini rugi-rugidianggap sangatkecil sehingga dapat diabaikan.

IlustrasisinyalradarSekunderdanrentetan perjalanan sinyal radar sekunder dapat dilihat pada gambar 2.7.



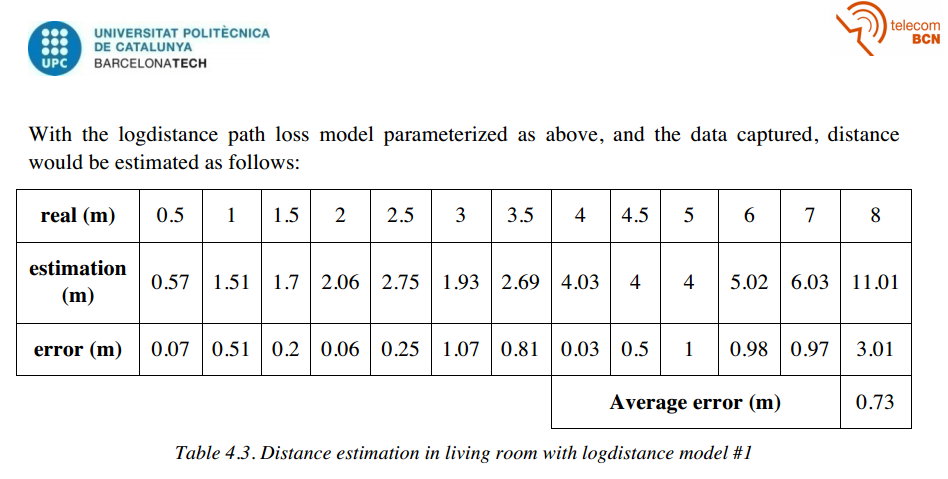
**Gambar 2.7.** Sinyal TDOA[7].

Teknik pengukuran jarak menggunakan TDOA menawarkan tingkat resolusi sampai dengan 10cm dibandingkan dengan menggunakan RSSI, Karena pengukuran jarak menggunakan TDOA lebih imun terhadap gangguan sinyal yang lain, sedangkan RSSI lebih mudah terpengaruh oleh interferensi sinyal lain sehingga menyebabkan kuat sinyal yang digunakan pada RSSI menjadi tidak valid. Gambar 2.8 adalah karakteristik RSSI terhadap jarak yang terukur.

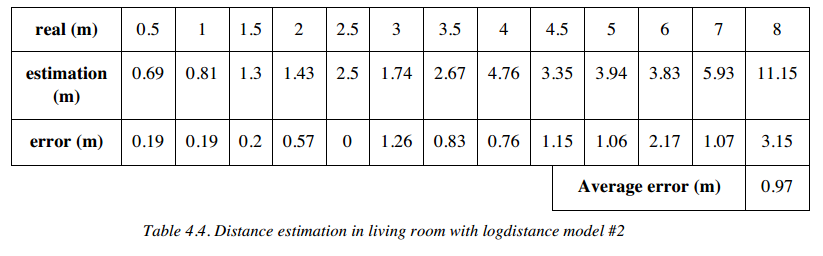


**Gambar 2.8.** Karakteristik jarak terhadap RSSI[8].

**Tabel 2.1.** Estimasi jarak RSSI Bluetooth pemodelan 1[9].



**Tabel 2.2.** Estimasi jarak RSSI Bluetooth pemodelan 2[9].



Dalam telekomunikasi, RSSI adalah sebuah ukuran kekuatan sinyal radio yang diterima oleh *receiver*. Teknologi localization *node* of *wireless sensor network* (WSN) biasanya menggunakan nilai RSSI untuk melakukan pengukuran jarak. Dengan mengumpulkan nilai RSSI, maka dapat ditentukan jarak antara *transmitter* dan *receiver*. Tabel 2.1 dan tabel 2.2 merupakann tabel estimasi pengukuran jarak menggunakan RSSI Bluetooth.

## **Kompas**

Kompas adalah alat navigasi untuk menentukan arah berupa sebuah panah penunjuk magnetis yang bebas menyelaraskan dirinya dengan medan magnet bumi secara akurat. Kompas memberikan rujukan arah tertentu, sehingga sangat membantu dalam bidang navigasi. Arah mata angin yang ditunjuknya adalah utara, selatan, timur, dan barat. Apabila digunakan bersama-sama dengan jam dan sekstan, maka kompas akan lebih akurat dalam menunjukkan arah. Penemuan bahwa jarum magnetik selalu mengarah ke utara dan selatan terjadi di Cina dan diuraikan dalam buku Loven Heng. Gambar 2.9 merupakan contoh kompas analog.



**Gambar 2.9.** Kompas analog[18].

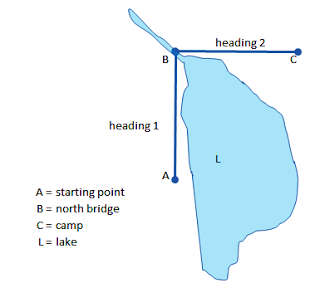
Lokasi magnet di Kutub Utara selalu bergeser dari masa ke masa. Penelitian terakhir yang dilakukan oleh The Geological Survey of Canada melaporkan bahwa posisi magnet ini bergerak kira-kira 40 km per tahun ke arah barat laut.

Kompas analog adalah kompas yang biasa kita lihat dalam kehidupan sehari-hari. Sedangkan kompas digital merupakan kompas yang telah menggunakan proses digitalisasi. Dengan kata lain cara kerja kompas ini menggunakan komputerisasi. Diciptakannya kompas digital bertujuan untuk melengkapi kebutuhan robotika yang semakin canggih. Dunia robotika ini sangat membutuhkan alat navigasi yang efektif dan efisien. Sedangkan kompas sendiri merupakan sebuah alat sistem navigasi yang efektif dengan harga lebih murah. oleh karena itu kompas digital diharapkan bisa mensubstitusi alat sistem navigasi pada robot.

## ***Heading***

*Heading* adalah arah hadap suatu objek menggunakan satuan derajat dengan utara sebagai titik 0 atau 360 derajat. Timur adalah 90 derajat, selatan 180 derajat, dan barat sebagai 270 derajat. Titik utara dapat sebagai utara magnetik (kompas), tapi bisa juga menggunakan utara absolut (kutub utara).

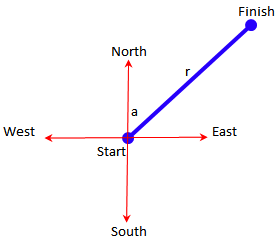
Lihat gambar 2.10, misalkan sebuah benda bergerak dari titik A menuju titik C, melalui titik B. Pada saat *start* di titik A, benda bergerak ke utara, atau disebut *heading* 0 derajat, menuju titik B sejauh 10 km. Sesampainya di titik B, benda berbelok ke kanan, atau disebut *heading* 90 derajat dan berjalan menuju titik C sejauh 10 km.



**Gambar 2.10.** Ilustrasi *heading*[17]*.*

Dari gambar 2.10 ditunjukkan bahwa *bearing* adalah sudut dari titik awal menuju titik tujuan atau akhir perjalanan, atau posisi suatu objek dari pengamat. Jadi dalam suatu *bearing*, dapat terdiri dari beberapa *heading*. Pada contoh diatas, *bearing* dari titik A menuju titik C adalah 45 derajat. *Bearing* disebut juga sebagai azimuth.

cara menghitung koordinat GPS dari titik tujuan, jika koordinat GPS di titik awal, *heading*, dan jarak tempuh diketahui.



**Gambar 2.11.** Menentukan jarak dan titik akhir (*finish*)[17].

Pada gambar 2.11, (a) adalah sudut *heading*, dimana titik 0 derajat adalah arah ke utara peta, 90 derajat adalah arah timur peta, 180 derajat adalah arah selatan peta, 270 derajat adalah arah barat peta, dan 360 derajat sama dengan 0 derajat atau arah utara peta. Jika titik awal atau *start* diketahui, maka titik akhir atau *finish* dapat dihitung. (r) adalah jarak dari *start* ke *finish*.

## **Sensor**

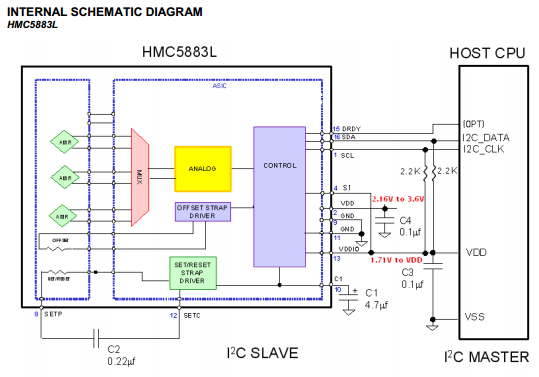
### ***Magnetometer***

Sensor *magnetometer* adalah sensor yang digunakan untuk mengukur kekuatan dan arah medan magnet pada suatu titik dalam ruangan. Sensor *magnetometer* yang digunakan termasuk dalam kategori *magneto-resistive* dan sensor ini dapat digunakan untuk mengetahui *heading* robot terhadap arah magnetik bumi.

****

**Gambar 2.12.** Modul sensor GY-85[10].

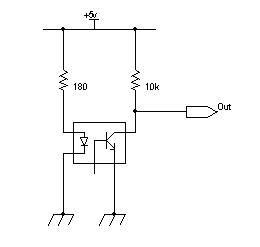
Sensor *magnetometer* tipeHMC5883L seperti gambar 2.12 memiliki *range* pengukuran antara +/-1 sampai dengan +/-8 gauss. Sedangkan untuk akses *device* itu sendiri menggunakan komunikasi antarmuka (I2C) dengan *bandwidth* data yang bisa kita atur sesuai dengan kebutuhan. *Output* modul sensor ini adalah data kekuatan dan arah medan megnet dalam bentuk digital 16 *bit*. Tegangan catu daya yang dibutuhkan modul ini adalah 5V. Blok diagram sensor ditunjukkan dalam Gambar 2.13.



**Gambar 2.13.** Blok diagram sensor *magnetometer* HMC5883L [11].

### ***Rotary encoder***

*Rotary encoder* adalah divais elektro mekanik yang dapat memonitor gerakan dan posisi. *Rotary encoder* umumnya menggunakan sensor optik untuk menghasilkan *serial* pulsa yang dapat diartikan menjadi gerakan, posisi, dan arah. Sehingga posisi sudut suatu poros benda berputar dapat diolah menjadi informasi berupa kode digital oleh *rotary encoder* untuk diteruskan oleh rangkaian kendali. *Rotary encoder* umumnya digunakan pada pengendalian robot, motor drive, dsb.

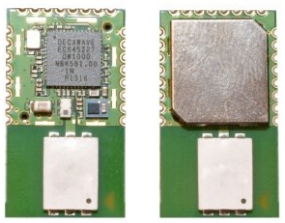
1. 

**Gambar 2.14.** Rangkaian *rotary encoder* [5]

*Rotary encoder* tersusun dari suatu piringan tipis yang memiliki lubang-lubang pada bagian lingkaran piringan. LED ditempatkan pada salah satu sisi piringan sehingga cahaya akan menuju ke piringan. Di sisi yang lain suatu photo-transistor diletakkan sehingga photo-transistor ini dapat mendeteksi cahaya dari LED yang berseberangan. Piringan tipis tadi dikopel dengan poros motor, atau divais berputar lainnya yang ingin kita ketahui posisinya, sehingga ketika motor berputar piringan juga akan ikut berputar. Apabila posisi piringan mengakibatkan cahaya dari LED dapat mencapai photo-transistor melalui lubang-lubang yang ada, maka photo-transistor akan mengalami saturasi dan akan menghasilkan suatu pulsa gelombang persegi. Gambar 2.14 menunjukkan bagan skematik sederhana dari *rotary encoder*. Semakin banyak deretan pulsa yang dihasilkan pada satu putaran menentukan akurasi *rotary encoder* tersebut, akibatnya semakin banyak jumlah lubang yang dapat dibuat pada piringan menentukan akurasi *rotary encoder* tersebut.

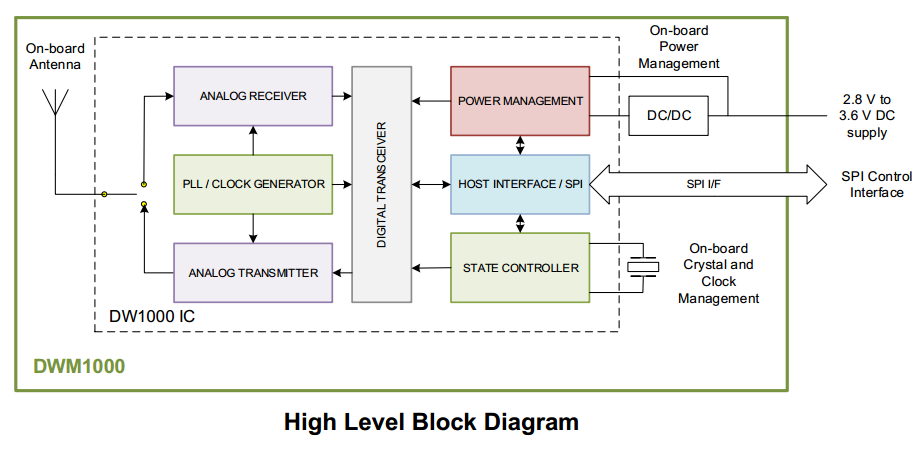
### ***Wireless sensor network* DWM1000**

Modul DWM1000 merupakan modul *transceiver* RF *Ultra Wide Band* (UWB). Di dalam modul ini sudah terintegrasi antena, rangkaian RF, rangkaian *power management* dan rangkaian pembangkit *clock.* Modul ini dapat digunakan untuk melakukan pengukuran *2-way ranging* atau *TDOA location system* dengan tingkat kepresisian mencapai 10cm dan mendukung *data rates* mencapai 6.8 Mbps. Bentuk fisik modul DWM1000 ditunjukkan oleh gambar 2.15.



**Gambar 2.15.** Modul DWM1000[12].

Modul DWM1000 menggunakan *chip* *transceiver* CMOS RF yang hemat daya, modul ini mmenuhi standar *UWB* IEE 802.15.4-2011. Modul DWM1000 memiliki kristal referensi sebesar 38.4 MHz, modul ini tidak membutuhkan desain RF tambahan karena di dalamnya sudah terintegrasi rangkaian analog RF dengan komponen – komponennya. Gambar 2.16 menunjukkan blokdiagram internal dari modul DWM1000.



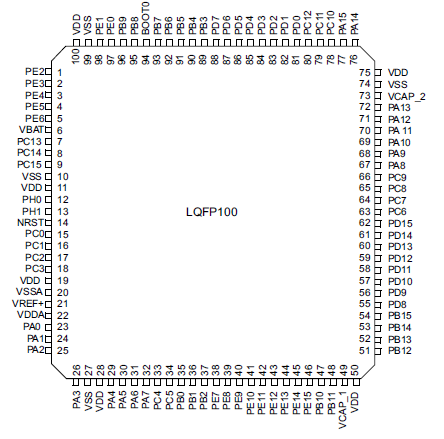
**Gambar 2.16.** Rangkaian dalam modul DWM1000[12].

## **Mikrokontroler ARM Cortex-M4 STM32F407**

Mikrokontroler yang dipilih dalam pengerjaan proyek akhir ini adalah ARM Cortex-M4 dengan kode STM32F407VG keluaran dari STMicroelectronics. Mikrokontroler ini mempunyai eksekusi memori flash sampai dengan 168 MHz .Fitur instruksi DSP dan unit *floating point* adalah kelebihan lain dari mikrokontroler ini. Fitur yang ditawarkan oleh mikrokontroler ini antara lain adalah :

* ARM 32-*bit* Cortex ™ - M4 CPU dengan FPU , Adaptive real-*time* akselerator ( ART Accelerator ™ ) frekuensi hingga 168 MHz , dan instruksi DSP
* Memori Flash sampai dengan 1 Mbyte
* Flexible Static Memory Kontroller .mendukung Compact Flash, SRAM, PSRAM , NOR dan memory NAND
* *Supply* 1,8 V sampai 3,6 V
* Daya rendah
* × 12 - *bit* , 2,4 MSPS A / D *converter* : sampai 24 *kanal* dan 7,2 MSPS di tiga mode Interval
* × 12 - *bit* D / A *converters*
* DMA : 16 DMA *stream* *controller* dengan FIFOs dan dukungan meledak
* Mempuynyai sampai dengan 17 *timer* : terdapat 16 - *bit* dan 32 -*bit* *timer* hingga 168 MHz , masing-masing dengan sampai 4IC/OC/PWM atau *pulse counter* dan quadrature(*incremental*) masukan *encoder*
* Sampai dengan 140 I / O port dengan kemampuan interupsi
* Sampai dengan 138 pin I / O dengan Toleransi 5V
* Sampai dengan 15 komunikasi antarmuka
* Sampai dengan 3 × antarmuka I2C ( SMBus / PMBus )
* Sampai dengan 4 USARTs / 2 UART ( 10,5 M*bit*/s , ISO7816 antarmuka , LIN , IrDA , kontrol modem )
* Sampai dengan 3 SPI ( 42 M*bit*s/s ) , 2 dengan *muxedfull-duplex* I2S untuk mencapai kelas audio yang akurat via PLL audio internal atau eksternal
* 2 × antarmukaCAN ( 2.0B Aktif )
* Antarmuka SDIO
* USB 2.0 *full-speed device* / *host* / OTG controller dengan on-*chip* PHY
* USB 2.0 *high-speed/full-speeddevice / host* / OTG kontroller dengan DMA , on-*chip* kecepatan penuh PHY dan ULPI
* 10/100 Ethernet MAC dengan DMA :mendukung *hardware* 1588v2 IEEE , MII / RMII
* 8 - sampai 14 - *bit* antarmuka kamera paralel hingga54 Mbytes / s
* Unit Perhitungan CRC
* RTC : akurasi *subsecond* , kalender *hardware*

Gambar 2.17 merupakan gambar konfigurasi pin dari mikrokontroler STM32F407VGT6 dengan packge LQFP100 yang terdiri atas 100 pin kaki IC.



**Gambar 2.17.** Konfigurasi pin STM32F407VG [13].

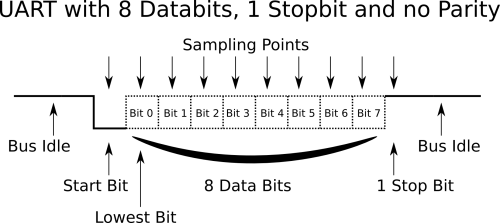
ARM adalah keluarga arsitektur *reduced instruction set computing* (RISC) yang dikembangkan oleh perusahaan Inggris ARM Holdings. ARM Cortex-M menggunakan arsitektur Harvard yang memiliki beberapa *bus* yang memungkinkan untuk melakukan operasi secara parallel, sehingga diharapkan dapat meningkatkan kinerja secara sistem yang akan dibangun keseluruhan.

## **Komunikasi *Serial* UART**

UART atau Universal *Asynchronous* *Receiver*-Transmitter adalah bagian perangkat keras komputer yang menerjemahkan antara *bit*-*bit* paralel data dan *bit*-*bit* *serial*. UART biasanya berupa sirkuit terintegrasi yang digunakan untuk komunikasi *serial* pada komputer atau port *serial* perangkat periperal. UART sekarang ini termasuk di dalam beberapa mikrokontroler (contohnya, PIC16F628).UART atau Universal *Asynchronous* *Receiver* Transmitter adalah protokol komunikasi yang umum digunakan dalam pengiriman data *serial* antara *device* satu dengan yang lainnya. Sebagai contoh komunikasi antara sesama mikrokontroler atau mikrokontroler ke PC.

Dalam pengiriman data, clock antara pengirim dan penerima harus sama karena paket data dikirim tiap *bit* mengandalkan clock tersebut. Inilah salah satu keuntungan model *asynchronous* dalam pengiriman data karena dengan hanya satu kabel transmisi maka data dapat dikirimkan. Berbeda dengan model synchronous yang terdapat pada protokol SPI (*Serial* *Peripheral* *Interface*) dan I2C (*Inter-Integrated Circuit*) karena protokol membutuhkan minimal dua kabel dalam transmisi data, yaitu transmisi clock dan data. Namun kelemahan model *Asynchronous* adalah dalam hal kecepatannya dan jarak transmisi. Karena semakin cepat dan jauhnya jarak transmisi membuat paket-paket *bit* data menjadi terdistorsi sehingga data yang dikirim atau diterima bisa mengalami *error*.*Asynchronous* memungkinkan transmisi mengirim data tanpa sang pengirim harus mengirimkan sinyal detak ke penerima. Sebaliknya, pengirim dan penerima harus mengatur parameter waktu di awal dan *bit* khusus ditambahkan untuk setiap data yang digunakan untuk mensinkronkan unit pengiriman dan penerimaan.

Saat sebuah data diberikan kepada UART untuk transmisi *Asynchronous*, "*Bit* *Start*" ditambahkan pada setiap awal data yang akan ditransmisikan. *Bit* *Start* digunakan untuk memperingatkan penerima yang kata data akan segera dikirim, dan memaksa *bit*-*bit* sinyal di *receiver* agar sinkron dengan *bit*-*bit* sinyal di pemancar. Sinyal komunikasi UART ditunjukkan pada gambar 2.18.



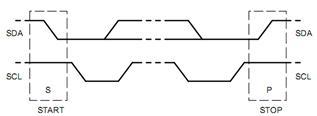
**Gambar 2.18.** Sinyal komunikasi UART[19].

Kedua *bit* ini harus akurat agar tidak memiliki penyimpangan frekuensi dengan lebih dari 10% selama transmisi *bit*-*bit* yang tersisa dalam data. (Kondisi ini ditetapkan pada zaman teleprinter mekanik dan telah dipenuhi oleh peralatan elektronik modern). Setelah *Bit* *Start*, *bit* individu dari data yang dikirim, dengan sinyal *bit* terkecil yang pertama dikirim. Setiap *bit* dalam transmisi ditransmisikan serupa dengan jumlah *bit* lainnya, dan penerima mendeteksi jalur di sekitar pertengahan periode setiap *bit* untuk menentukan apakah *bit* adalah 1 atau 0. Misalnya, jika dibutuhkan dua detik untuk mengirim setiap *bit*, penerima akan memeriksa sinyal untuk menentukan apakah itu adalah 1 atau 0 setelah satu detik telah berlalu, maka akan menunggu dua detik dan kemudian memeriksa nilai *bit* berikutnya , dan seterusnya.

## **Komunikasi I2C**

*Inter Integrated Circuit* atau sering disebut I2C adalah standar komunikasi *serial* dua arah menggunakan dua saluran yang didisain khusus untuk mengirim maupun menerima data. Sistem I2C terdiri dari saluran SCL (*Serial Clock*) dan SDA (*Serial Data*) yang membawa informasi data antara I2C dengan pengontrolnya. Piranti yang dihubungkan dengan sistem I2C *Bus* dapat dioperasikan sebagai *Master* dan *Slave*. *Master* adalah piranti yang memulai transfer data pada I2C *Bus* dengan membentuk sinyal *Start*, mengakhiri transfer data dengan membentuk sinyal *Stop*, dan membangkitkan sinyal *clock*. *Slave* adalah piranti yang dialamati *master*.

Sinyal *Start* merupakan sinyal untuk memulai semua perintah, didefinisikan sebagai perubahan tegangan SDA dari “1” menjadi “0” pada saat SCL “1”. Sinyal *Stop* merupakan sinyal untuk mengakhiri semua perintah, didefinisikan sebagai perubahan tegangan SDA dari “0” menjadi “1” pada saat SCL “1”. Kondisi sinyal *Start* dan sinyal *Stop* seperti tampak pada Gambar 2.19.



**Gambar 2.19.** Kondisi sinyal *start* dan *stop*[20].

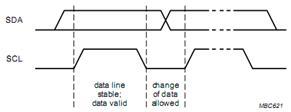
Sinyal dasar yang lain dalam I2C *Bus* adalah sinyal *acknowledge* yang disimbolkan dengan ACK Setelah transfer data oleh *master* berhasil diterima *slave*, *slave* akan menjawabnya dengan mengirim sinyal *acknowledge*, yaitu dengan membuat SDA menjadi “0” selama siklus *clock* ke 9. Ini menunjukkan bahwa *Slave* telah menerima 8 *bit* data dari *Master*. Kondisi sinyal *acknowledge* seperti tampak pada Gambar 2.20. Transfer *bit* ditunjukkan pada gambar 2.21.



**Gambar 2.20.** Sinyal ACK dan NACK[20].

Dalam melakukan transfer data pada I2C *Bus*, kita harus mengikuti tata cara yang telah ditetapkan yaitu:

* Transfer data hanya dapat dilakukan ketika *Bus* tidak dalam keadaan sibuk.
* Selama proses transfer data, keadaan data pada SDA harus stabil selama SCL dalam keadan tinggi. Keadaan perubahan “1” atau “0” pada SDA hanya dapat dilakukan selama SCL dalam keadaan rendah. Jika terjadi perubahan keadaan SDA pada saat SCL dalam keadaan tinggi, maka perubahan itu dianggap sebagai sinyal *Start* atau sinyal *Stop*.



**Gambar 2.21.** Transfer *bit* pada I2C *bus*[20].

## ***Display* (*LCD*)**

*Liquid Crystal Display* (*LCD*) merupakan sebuah teknologi layar digital yang menghasilkan citra pada sebuah permukaan yang rata (*flat*) dengan memberi sinar pada kristal cair dan filter berwarna, yang mempunyai struktur molekul polar, diapit antara dua elektroda yang transparan, bila medan listrik diberikan, molekul menyesuaikan posisinya pada medan, membentuk susunan kristalin yang mempolarisasi cahaya yang melaluinya. Bentuk fisik *LCD* 2x16 ditunjukkan pada gambar 2.22.



**Gambar 2.22** Tampilan *LCD* 2x16[14].

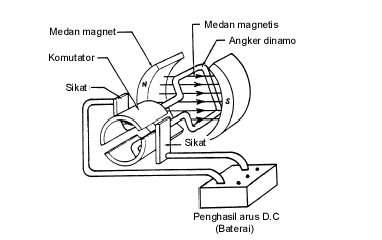
**Tabel 2.3.** Susunan kaki *LCD* 16 X 2.

|  |  |  |
| --- | --- | --- |
| Pin | Name | Comment |
| 1 | VSS | Ground |
| 2 | VDD | +3V or +5V |
| 3 | VO | *Contrast adjustmen* |
| 4 | RS | H/L *Register* select *signal* |
| 5 | R/W | H/L *read*/*write* *signal* |
| 6 | E | H-L Enable signal |
| 7 | D00 | H/L Data *bus* *line* |
| 8 | D01 | H/L Data *bus* *line* |
| 9 | D02 | H/L Data *bus* *line* |
| 10 | D03 | H/L Data *bus* *line* |
| 11 | D04 | H/L Data *bus* *line* |
| 12 | D05 | H/L Data *bus* *line* |
| 13 | D06 | H/L Data *bus* *line* |
| 14 | D07 | H/L Data *bus* *line* |
| 15 | A/Vee | 4,2V *for* Led *negative* vol*tag*e *output* |
| 16 | K | *Power* *supply* for B/l (0V) |

*LCD* berfungsiuntuk menampilkan data-data sensor, peritungan kinematika, dan keperluan *troubleshooting* lainnya. *LCD* yang akan digunakan dalam PA ini adalah jenis *LCD* M1632 dengan tampilan sebanyak 16x2 karakter seperti yang ditunjukan pada gambar 2.22 Modul tersebut dilengkapi dengan mikrokontroler yang didesain khusus untuk mengendalikan *LCD*, agar lebih mudah dalam penggunaannya. Tabel 2.3 adalah penjelasan setiap kaki dari *LCD*.

## **Motor DC**

Motor DC adalah motor listrik yang memerlukan suplai tegangan arus searah pada kumparan medan untuk diubah menjadi energi gerak mekanik. Kumparan medan pada motor dc disebut stator (bagian yang tidak berputar) dan kumparan jangkar disebut rotor (bagian yang berputar). Motor DC memerlukan suplai tegangan yang searah pada kumparan medan untuk diubah menjadi energi mekanik. Kumparan medan pada motor dc disebut stator (bagian yang tidak berputar) dan kumparan jangkar disebut rotor (bagian yang berputar). Jika terjadi putaran pada kumparan jangkar dalam pada medan magnet, maka akan timbul tegangan (GGL) yang berubah-ubah arah pada setiap setengah putaran, sehingga merupakan tegangan bolak-balik. Prinsip kerja dari arus searah adalah membalik phasa tegangan dari gelombang yang mempunyai nilai positif dengan menggunakan komutator, dengan demikian arus yang berbalik arah dengan kumparan jangkar yang berputar dalam medan magnet. Bentuk motor paling sederhana memiliki kumparan satu lilitan yang bisa berputar bebas di antara kutub-kutub magnet permanen. Bagian dalam motor DC tampak seperti gambar 2.23.



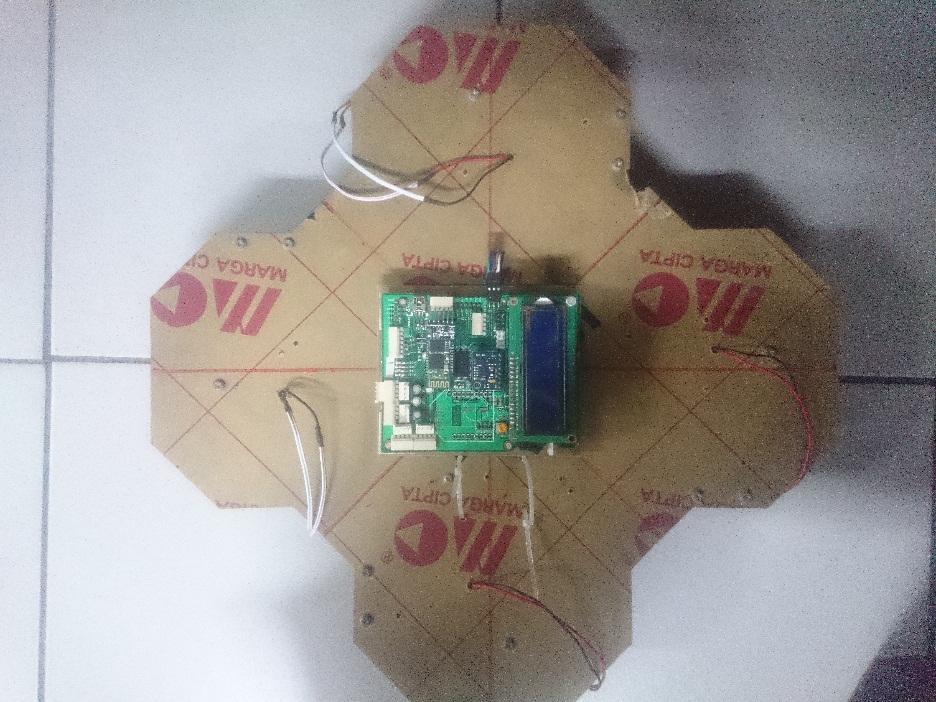
**Gambar 2.23.** Bagian dalam motor DC[15].

(*halaman ini sengaja dikosongkan*)

# PERANCANGAN DAN PEMBUATAN SISTEM

## **Perancangan dan Pembuatan Mekanik**

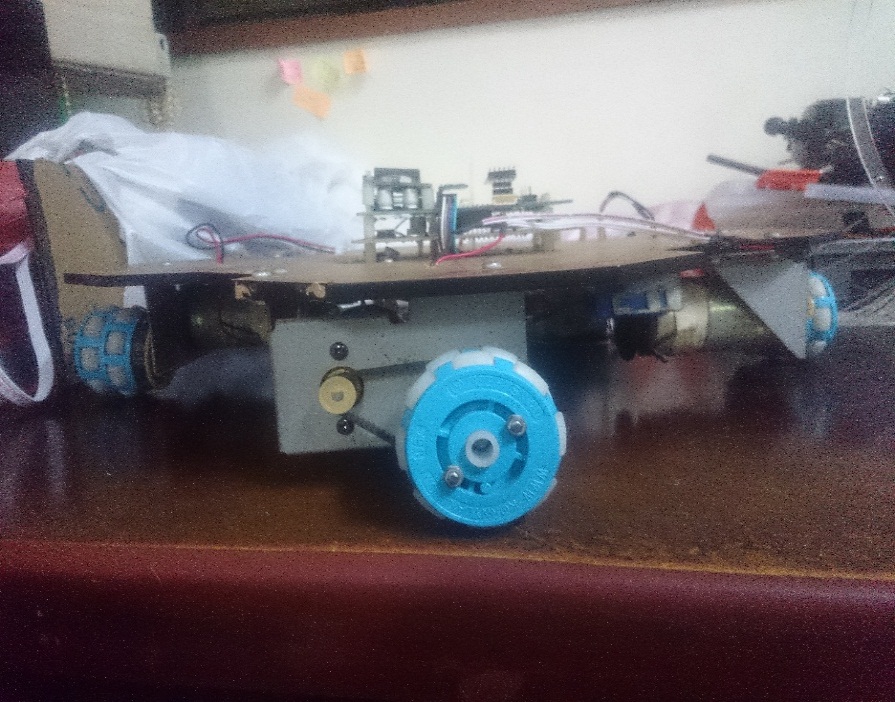
Bentuk mekanik *mobile robot* menggunakan 4 buah motor DC yan setiap motornya diubungkan dengan roda ­*bi-directional omni wheel.* Dengan desain seperti ini robot memiliki kemampuan untuk berjalan maju, mundur, berjalan ke samping, dan berotasi. Desain peletakan *hardware* elektronik diletakkan di tengah – tengah robot, hal ini bertujuan agar pembacaan *heading* atau orientasi dan posisi robot tepat berada di tengah – tengah bodi robot. Bentuk mekanik robot tampak atas ditunjukkan pada gambar 3.1 dan bentuk roda robot ditunjukkan pada gambar 3.2.



Y

X

**Gambar 3.1.** Gambar mekanik robot tampak atas.



d

**Gambar 3.2.** Gambar mekanik roda robot tampak samping.

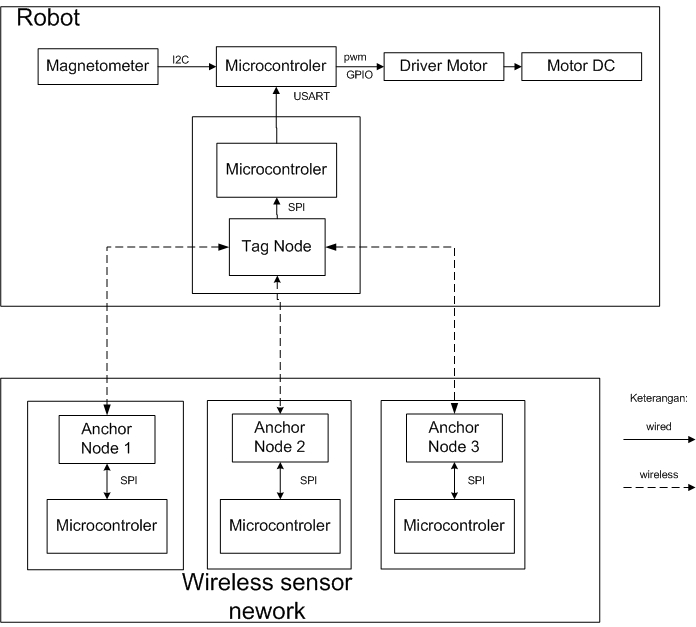
Spesifikasi robot: panjang X=35cm

panjang Y=35cm

panjang d=5cm

## **Perancangan dan Pembuatan Perangkat Elektronik**

Sistem elektronik robot terdiri dari empat bagian utama yaitu sensor, kontroler, display, dan aktuator. Sensor *Magnetometer* untuk mengetahui orientasi robot atau *heading*. Kontroler utama yang meng-*handle* seluruh sistem menggunakan mikrokontroler ARM seri STM32F407VG. Display untuk menampilkan data-data sersor, hasil komputasi, dan keperluan *troubleshooting* mengunakan sebuah *LCD* karakter 2x16.Gambar 3.3 adalah blok diagram sistem elektronik dan gambar 3.4, gambar 3.5 dan gambar 3.6 adalah *board* elektronik robot, sedangkan gambar 3.7 adalah bentuk robot secara keseluruhan.



**Gambar 3.3.** Blok diagram sistem elektronik.



***magnetometer***

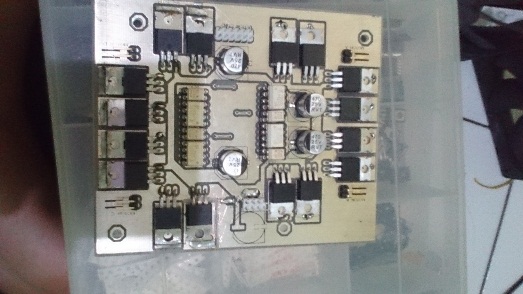
***LCD* 2X16**

**STM32F407VG**

**Interface ke *driver* motor**

**Interface ke *Tag* *Node***

**Gambar 3.4.** *Board* sistem kontrol utama.



***Driver* Motor DC A**

***Driver* Motor DC D**

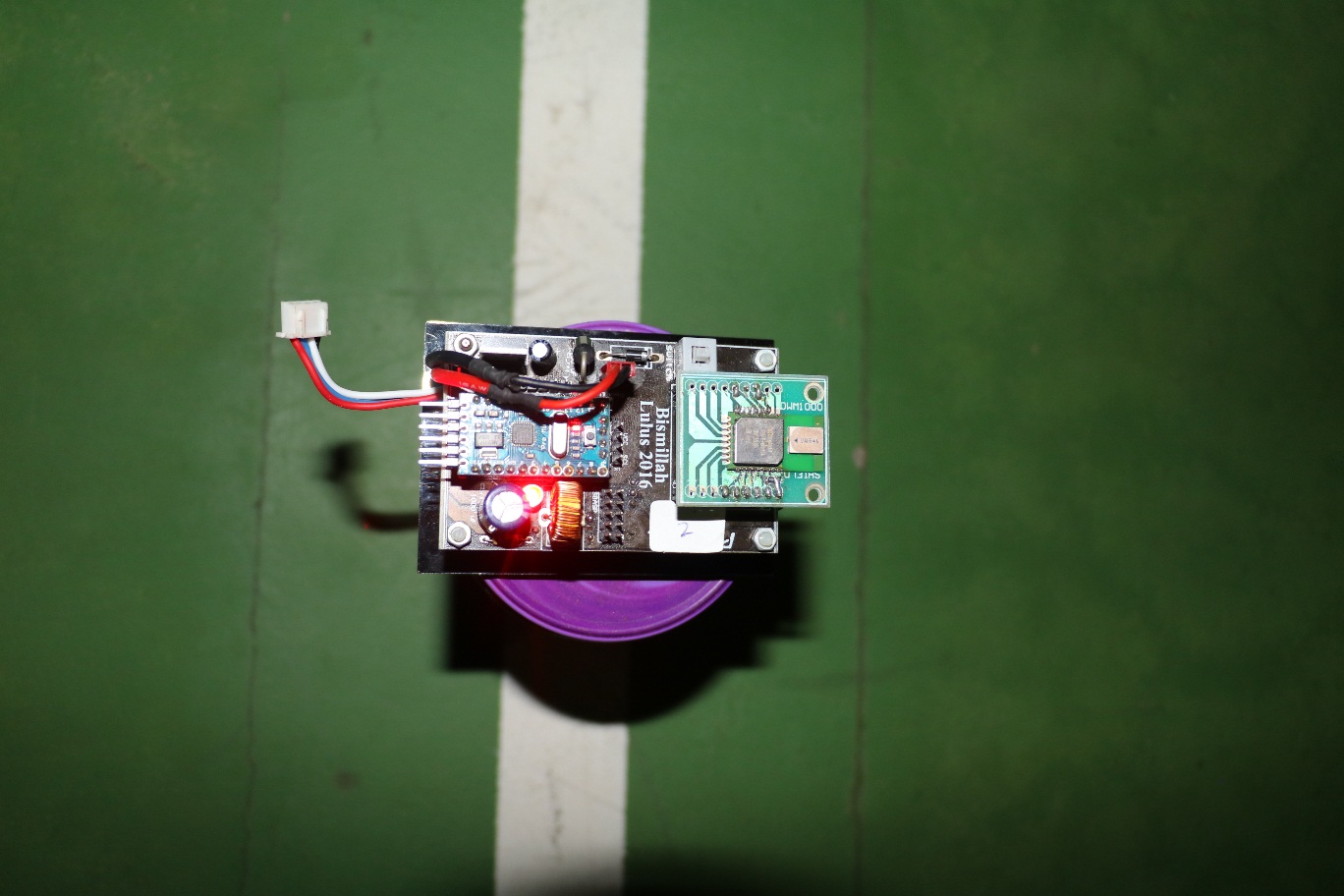
***Driver* Motor DC C**

***Driver* Motor DC A**

**Interface ke**

**Mikrokontroler**

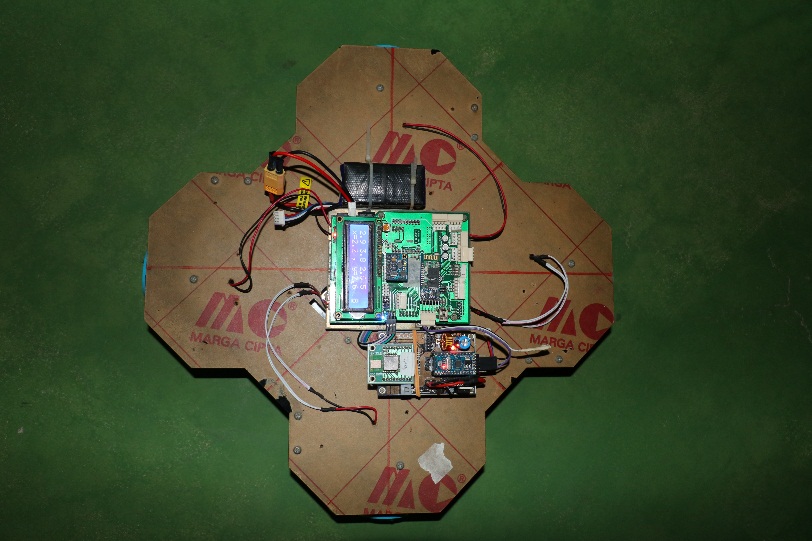
**Gambar 3.5.** *Board* *driver* motor DC.



**Modul *Ranging* DWM1000**

**ARDUINO Mini Pro**

**Gambar 3.6.** *Board* sistem *Node.*



***Board* *Tag* *Node***

***Board* Sistem Kontrol Utama**

***Power* *Supply* (Battery)**

**Gambar 3.7.** Integrasi mekanik, *board* sistem kontrol, dan *tag* *node.*

### **Rangkaian *Regulator* 5V dan 3,3V**

Rangkaian *regulator* 5v berfungsi untuk menurunkan tegangan baterai yang semula 12,6v menjadi 5v. LM2576 dipilih karena IC tersebut berdasarkan *datasheet* mampu menghasilhal regulasi tegangan 5V dengan arus keluaran mencapai 3A. Untuk rangkaian *regulator* 3,3V dipilih IC LD1117 3.3V, berdasarkan *datasheet* IC tersebut mampu menyuplai arus hingga 0,8A.

Pada rangkaian *regulator*, pada gambar 3.8 ditambahkan *fuse* untuk pengamanan ketika terjadi *overload* dan terdapat beberapa LED sebagai indikator adanya tegangan.

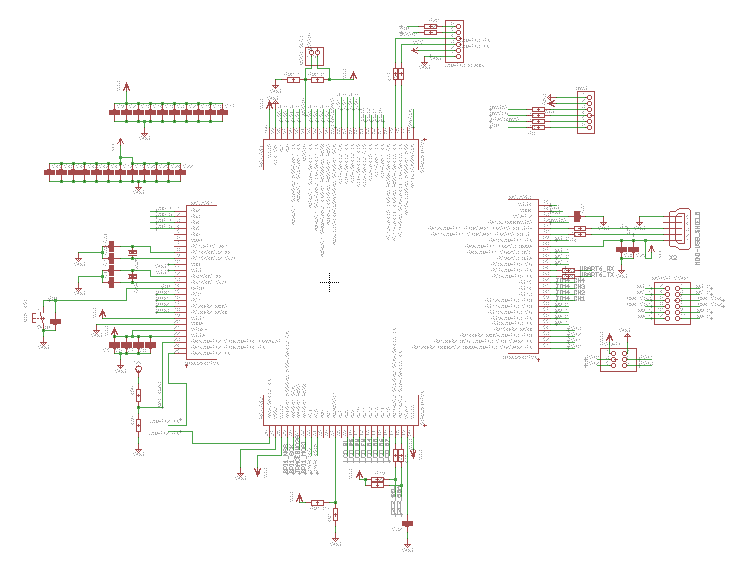


**Gambar 3.8.** Rangkaian *regulator* 5V dan 3,3V.

### **Rangkaian *Minimum System* STM32F407VG**

STM32F407VG bekerja pada level tegangan 3,3V sehingga dibutuhkan regutor 3,3V. *Regulator* yang digunakan adalah L1117 dengan arus keluaran 1A. Dari *regulator* 3,3V ditambakan induktor dan kapasitor-kapasitor pada setiap PIN VDD dan VSS yang berfungsi sebagai filter dari tegangan *input*. Minimum sistem STM32F407VG menggunakan osilator dengan kristal 8MHz yang dihubungkan pada PIN H0 dan PIN H1, selain itu untuk keperluan RESET ditambahkan *push button* pada PIN NRST yang telah di *pull* *up*.

Untuk keperluan *download* program, STM32F407 mempunyai fitur *bootloader,* dimana sistem ini akan membuat STM32F407 masuk mode *boot* yang siap di-*download* program lewat komunikasi USB OTG atau melalui USART1 dan USART3. Untuk dapat masuk mode *bootloader,* PIN BOOT0 harus mendapatkan logika “1” dan BOOT1 mendapatkan logika “0” setelah itu sistem di RESET.Gambar 3.9 adalah rangkaian dari minimum sistem STM32F407. Untuk gambar yang lebih jelas lihat lampiran A1.



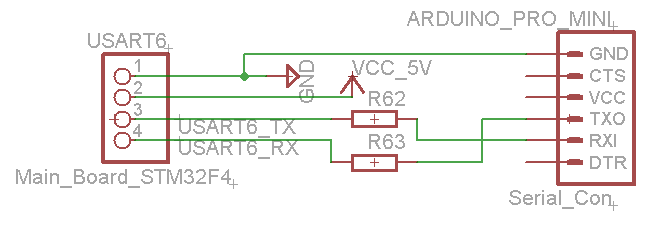
**Gambar 3.9.** Minimum sistem STM32F407VG.

### **USART pada Mikrokontroler STM32F407VG**

Sistem kontrol utama memerlukan beberapa *device* untuk membantu kinerjanya. Teknik *interfacing* digunakan untuk melakukan komunikasi antar sistem, baik itu mikrokontroler dengan dengan mikrokontroler, mikrokontroler dengan sensor digital, mikrokontroler dengan modul komunikasi seperti Bluetooth, maupun mikrokontroler dengan PC. Untuk melakukan komunikasi dengan beberapa *device* tersebut dibutuhkan teknik *interfacing* yang sesuai dengan jenis *interfacing* yang disediakan oleh pembuat *device* atau sensor untuk mengaksesnya.

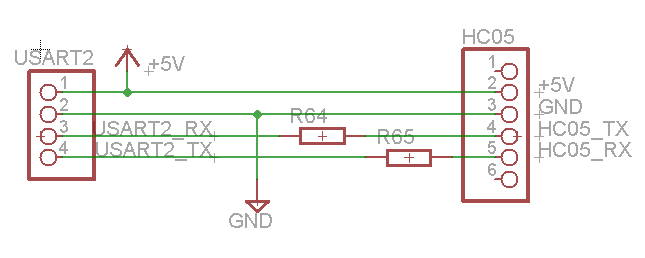
Pada proyek akhir ini ,ada beberapa jenis *data interfacing* yang digunakan pada mikrokontroler STM32F407VG, antara lainUART dan I2C. USART kali ini digunakan untuk *interfacing* dengan *Tag* *Node* untuk mendapatkan data jarak yang telah diolah oleh mikrokontroler ARDUINO Mini Pro yang digunakan untuk mengakses data yang terukur oleh modul *ranging* DWM1000. Sedangkan I2C digunakan untuk mengakses sensor *magnetometer* yang ada pada modul sensor GY-85 yang akan digunakan sebagai *heading* dari robot.

UART pada mikrokontroler STM32F407VG mempunyai level tegangan 3,3V karena sistem mikrokontroler STM32F407VG bekerja pada level tegagan 3,3V. Level tegangan tersebut sesuai dengan level tegangan yang digunakan oleh ARDUINO Mini Pro sehingga tidak dibutuhan tambahan IC level *converter* untuk menyesuaikan level tegangan yang berbeda. Salah satu kelebihan lain keluarga mikrokontroler STM32 adalah hampir semua pinout-nya dapat mentolerir tegangan sampai 5V, kecuali ketika digunakan sebagai *input* analog. Gamabar 3.10 adalah skematik *interfacing* UART antara mikrokontroler STM32F407VG dengan ARDUINO Mini Pro.



**Gambar 3.10.** Raingkaian *interfacing* Arduino Pro Mini.

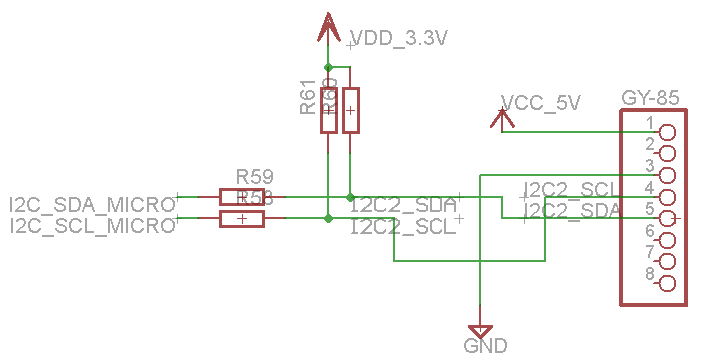
Selain digunakan untuk melakukan komunikasi dengan ARDUINO Mini Pro, UART pada proyek akhir ini juga digunakan untuk melakukan komunikasi dengan PC melalui modul Bluetooth HC-05, komunikasi antara mikrokontroler dengan PC dapat dimanfaatkan sebagai *monitoring* data dan mempermudah debugging program. Gambar 3.11 merupakan skemarik Raingkaian interfacing modul Blueatooth HC-05 dengan mikrokontroler utama.



**Gambar 3.11.** Raingkaian *interfacing* modul Blueatooth HC-05.

### **I2C pada Mikrokontroler STM32F407VG**

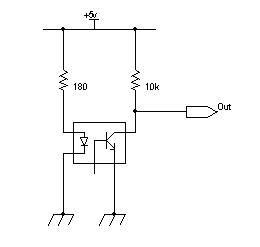
Seperti halnya UART, I2C pada mikrokontroler STM32F407VG menggunakan level tegangan 3,3V. Tidak seperti UART atau SPI, *bus* *driver* dari I2C menggunakan rangkaian “*open drain*”, artinya bahwa setiap *device* yangterkonesksi dapat men-*pull* low sinyal pada *bus*, tetapi idak dapat men-drive high sinyal pada *bus*, maka tidak akan terjadi drive sinyal high oleh satu *device* semntara *device* yang lain mem-*pull* sinyal pada *bus* menjadi low. Hal ini dapat mengurangi potensi kerusakan pada *device* akibat disipasi daya yang terlalu besar pada sistem. Maka setiap jalur sinyal ditambahi resistor *pull*-*up* untuk mengembalikan sinyal menjadi high ketika tidak ada *device* yang mem-*pull* sinyal *low*. Gambar 3.12 adalah rangkaian *interfacing* modul sensor GY-85.



**Gambar 3.12.** Rangkaian *interfacing* modul sensor GY-85.

### **Akses Sensor *Rotary encoder***

*Rotary encoder* adalah divais elektro mekanik yang dapat memonitor gerakan dan posisi. *Rotary encoder* umumnya menggunakan sensor optik untuk menghasilkan *serial* pulsa yang dapat diartikan menjadi gerakan, posisi, dan arah. Sehingga posisi sudut suatu poros benda berputar dapat diolah menjadi informasi berupa kode digital oleh *rotary encoder* untuk diteruskan oleh rangkaian kendali. *Rotary encoder* umumnya digunakan pada pengendalian robot, motor *drive*, dsb.



**Gambar 3.13.** Rangkaian *rotary encoder.*

*Rotary encoder* tersusun dari suatu piringan tipis yang memiliki lubang-lubang pada bagian lingkaran piringan. LED ditempatkan pada salah satu sisi piringan sehingga cahaya akan menuju ke piringan. Di sisi yang lain suatu *photo-transistor* diletakkan sehingga *photo-transistor* ini dapat mendeteksi cahaya dari LED yang berseberangan. Piringan tipis tadi dikopel dengan poros motor, atau divais berputar lainnya yang ingin kita ketahui posisinya, sehingga ketika motor berputar piringan juga akan ikut berputar. Apabila posisi piringan mengakibatkan cahaya dari LED dapat mencapai *photo-transistor* melalui lubang-lubang yang ada, maka photo-transistor akan mengalami saturasi dan akan menghasilkan suatu pulsa gelombang persegi. Gambar 3.13 menunjukkan bagan skematik sederhana dari *rotary encoder*. Semakin banyak deretan pulsa yang dihasilkan pada satu putaran menentukan akurasi *rotary encoder* tersebut, akibatnya semakin banyak jumlah lubang yang dapat dibuat pada piringan menentukan akurasi *rotary encoder* tersebut. Gambar 3.14 merupakan bentuk sinyal *rotary encoder*.

**Gambar 3.14.** Ilustrasi sinyal *rotary encoder.*

Untuck membaca data dari sensor *rotary encoder* digunakan external interupsi yang ada pada mikrokontroler. Pin *external interupt* padamikrokontroler STM32F407 dapat mendeteksi perubahan sinyal *rising-edge, falling edge*, maupun keduanya. Pada mikrokontroler STM32F407 semua pin GPIO dapat dijadikan sebagai pin untuk membaca *external* *interupt*. Berikut adalah culikan program untuk membaca sinyal *external* *interupt* yang dihasilkan oleh sensor *rotary encoder*:

void EXTI\_12\_15\_init()

{

RCC\_AHB1PeriphClockCmd(RCC\_AHB1Periph\_GPIOB, *ENABLE*);

/\* Enable SYSCFG clock \*/

RCC\_APB2PeriphClockCmd(RCC\_APB2Periph\_SYSCFG, *ENABLE*);

/\* Configure PA0 pin as *input* floating \*/

*GPIO\_InitStructure*.GPIO\_Mode = *GPIO\_Mode\_IN*;

*GPIO\_InitStructure*.GPIO\_PuPd = *GPIO\_PuPd\_NOPULL*;

*GPIO\_InitStructure*.GPIO\_Pin = GPIO\_Pin\_12 | GPIO\_Pin\_13 | GPIO\_Pin\_14 | GPIO\_Pin\_15;

GPIO\_Init(GPIOB, &*GPIO\_InitStructure*);

/\* Connect EXTI *Line* to PB pins \*/

SYSCFG\_EXTILineConfig(EXTI\_PortSourceGPIOB, EXTI\_PinSource12);

SYSCFG\_EXTILineConfig(EXTI\_PortSourceGPIOB, EXTI\_PinSource13);

SYSCFG\_EXTILineConfig(EXTI\_PortSourceGPIOB, EXTI\_PinSource14);

SYSCFG\_EXTILineConfig(EXTI\_PortSourceGPIOB, EXTI\_PinSource15);

/\* Configure EXTI *Lines* \*/

*EXTI\_InitStructure*.EXTI\_*Line* = EXTI\_Line12 | EXTI\_Line13| EXTI\_Line14| EXTI\_Line15;

*EXTI\_InitStructure*.EXTI\_Mode = *EXTI\_Mode\_Interupsi*;

*EXTI\_InitStructure*.EXTI\_Trigger = *EXTI\_Trigger\_Rising*;

*EXTI\_InitStructure*.EXTI\_LineCmd = *ENABLE*;

EXTI\_Init(&*EXTI\_InitStructure*);

*NVIC\_InitStructure*.NVIC\_IRQChannel = *EXTI15\_10\_IRQn*;

*NVIC\_InitStructure*.NVIC\_IRQChannelPreemptionPriority = 1;

*NVIC\_InitStructure*.NVIC\_IRQChannelSubPriority = 1;

*NVIC\_InitStructure*.NVIC\_IRQChannelCmd = *ENABLE*;

NVIC\_Init(&*NVIC\_InitStructure*);

}

void EXTI15\_10\_IRQHandler()

{

if(EXTI\_GetITStatus(EXTI\_Line12) != RESET)

{

led0\_togle;

//USART\_SendString(USART3,"exti12\n");

/\* Clear the EXTI *line* 0 pending *bit* \*/

EXTI\_ClearITPending*Bit*(EXTI\_Line12);

rot[0]++;

}

else if(EXTI\_GetITStatus(EXTI\_Line13) != RESET)

{

led1\_togle;

//USART\_SendString(USART3,"exti13\n");

/\* Clear the EXTI *line* 0 pending *bit* \*/

EXTI\_ClearITPending*Bit*(EXTI\_Line13);

rot[1]++;

}

else if(EXTI\_GetITStatus(EXTI\_Line14) != RESET)

{

led2\_togle;

//USART\_SendString(USART3,"exti14\n");

/\* Clear the EXTI *line* 0 pending *bit* \*/

EXTI\_ClearITPending*Bit*(EXTI\_Line14);

rot[2]++;

}

else if(EXTI\_GetITStatus(EXTI\_Line15) != RESET)

{

led3\_togle;

//USART\_SendString(USART3,"exti15\n");

/\* Clear the EXTI *line* 0 pending *bit* \*/

EXTI\_ClearITPending*Bit*(EXTI\_Line15);

rot[3]++;

}

}

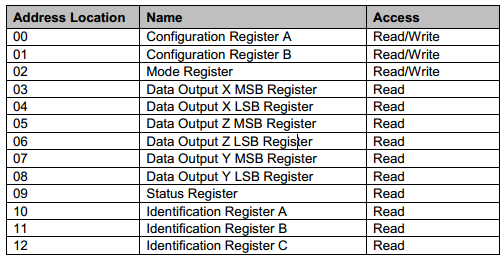
### **Akses Sensor *Magnetometer***

Sensor *magnetometer* HMC5883Lmenggunakan komunikasi I2C. Alamat *read* dari HMC5883L adalah 0x3D dan alamat *write* dari HMC5883L adalah 0x3C dan data keluaran berupa kuat medan tersimpan di dalam *register*-*register* HMC5883L. Proyek Akhir ini data yang diambil adalah kuat medan magnet sumbu X dan Y sehingga data yang diambil hanya dari *register* Data *Output* X MSB, Data *Output* X LSB, Data *Output* Y MSB, dan Data *Output* Y LSB. Tabel 3.1 merupakan tabel *read*/*write* address dari HMC5883L, dan tabel 3.2 merupakan alamat *register* yang ada pada HMC5883L.

**Tabel 3.1.** *Read* / *write* *address* dari HMC5883L.



**Tabel 3.2.** Alamat *register* HMC5883L.



Proses pengambilan data dari dengan menggunakan komunikasi I2C akan mengukuti urutan sebagai berikut ini :

* Master menulis data 1 byte ke slave



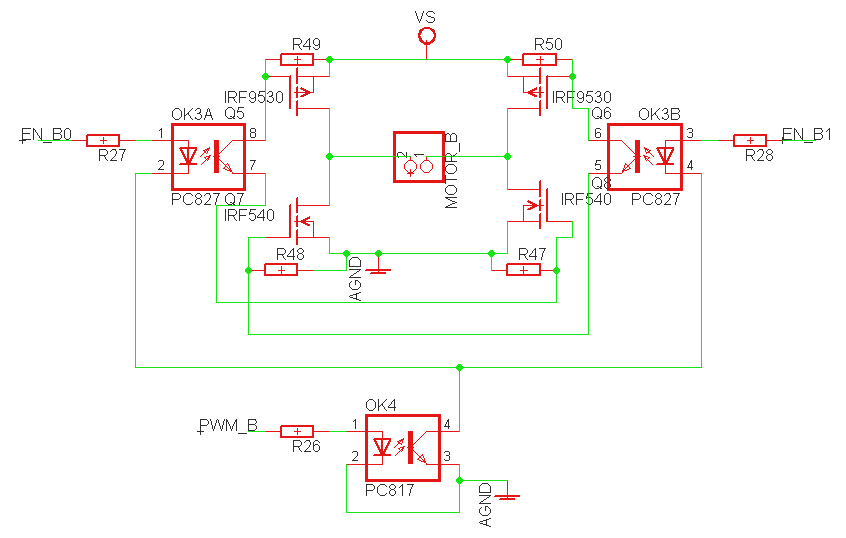
* Master membaca data 1 byte ke slave



Selurur proses pengambilan data dengan komunikasi I2C, dan penghitungan *heading* dilakukan dalam sebuah *interrupt* *timer* setiap dengan menggunakan *timer* 7 STM32F407.

### **Motor DC**

Motor DC dikontrol dengan *driver* motor dengan konfigurasi H-bridge, rangkaian *driver* motor H-bridge dibuat menggunakan MOSFET. Untuk mengontrol arah perputaran motor digunakan pin GPIO pada mikrokontroler dan untuk mengontrol kecepatan motor digunakan PWM yang dibangkitkan oleh *timer* pada mikrokontroler. Gambar 3.15 adalah gambar rangkaian untuk *driver* motor DC.



**Gambar 3.15.** Rangkaian *driver* motor DC.

PWM yang dibangkitkan memiliki frekwensi sebesar 1KHz dengan duty cycle yang dapat diubah – ubah untuk mengontrol kecepatan putaran motor DC yang dikontrol melalui *driver* motor DC. PWM dibangkitkan menggunakan *Timer* 4 STM32F407. Berikut ini cuplikan programnya:

void TIM4\_pwm\_init()

{

RCC\_AHB1PeriphClockCmd(RCC\_AHB1Periph\_GPIOD,ENABLE);

GPIO\_InitStructure.GPIO\_Pin=GPIO\_Pin\_12 | GPIO\_Pin\_13 | GPIO\_Pin\_14 | GPIO\_Pin\_15;

GPIO\_InitStructure.GPIO\_Mode=GPIO\_Mode\_AF;

GPIO\_InitStructure.GPIO\_OType=GPIO\_OType\_PP;

GPIO\_InitStructure.GPIO\_Speed=GPIO\_Speed\_50MHz;

GPIO\_Init(GPIOD,&GPIO\_InitStructure);

//Configuration

GPIO\_PinAFConfig(GPIOD,GPIO\_PinSource12,GPIO\_AF\_TIM4);

GPIO\_PinAFConfig(GPIOD,GPIO\_PinSource13,GPIO\_AF\_TIM4);

GPIO\_PinAFConfig(GPIOD,GPIO\_PinSource14,GPIO\_AF\_TIM4);

GPIO\_PinAFConfig(GPIOD,GPIO\_PinSource15,GPIO\_AF\_TIM4);

RCC\_APB1PeriphClockCmd(RCC\_APB1Periph\_TIM4,ENABLE);

TIM\_BaseInitStructure.TIM\_Prescaler=84;

TIM\_BaseInitStructure.TIM\_Period=1000;

TIM\_BaseInitStructure.TIM\_CounterMode=TIM\_CounterMode\_*Up*;

TIM\_BaseInitStructure.TIM\_ClockDivision=0;

TIM\_*Time*BaseInit(TIM4,&TIM\_BaseInitStructure);

TIM\_Cmd(TIM4, ENABLE);

//PWM channel 1

TIM\_OCInitStructure.TIM\_OCMode=TIM\_OCMode\_PWM1;

TIM\_OCInitStructure.TIM\_OutputState=TIM\_OutputState\_Enable;

TIM\_OCInitStructure.TIM\_Pulse=0;//CCR1\_Val;

TIM\_OCInitStructure.TIM\_OCPolarity=TIM\_OCPolarity\_High;

TIM\_OC1Init(TIM4,&TIM\_OCInitStructure);

TIM\_OC1PreloadConfig(TIM4,TIM\_OCPreload\_Enable);

//PWM channel 2

TIM\_OCInitStructure.TIM\_OutputState=TIM\_OutputState\_Enable;

TIM\_OCInitStructure.TIM\_Pulse = 0;

TIM\_OCInitStructure.TIM\_OCPolarity = TIM\_OCPolarity\_High;

TIM\_OC2Init(TIM4, &TIM\_OCInitStructure);

TIM\_OC2PreloadConfig(TIM4, TIM\_OCPreload\_Enable);

//PWM channel 3

TIM\_OCInitStructure.TIM\_OutputState=TIM\_OutputState\_Enable;

TIM\_OCInitStructure.TIM\_Pulse = 0;

TIM\_OCInitStructure.TIM\_OCPolarity = TIM\_OCPolarity\_High;

TIM\_OC3Init(TIM4, &TIM\_OCInitStructure);

TIM\_OC3PreloadConfig(TIM4, TIM\_OCPreload\_Enable);

//PWM channel 4

TIM\_OCInitStructure.TIM\_OutputState=TIM\_OutputState\_Enable;

TIM\_OCInitStructure.TIM\_Pulse = 0;

TIM\_OCInitStructure.TIM\_OCPolarity = TIM\_OCPolarity\_High;

TIM\_OC4Init(TIM4, &TIM\_OCInitStructure);

TIM\_OC4PreloadConfig(TIM4, TIM\_OCPreload\_Enable);

TIM\_ARRPreloadConfig(TIM4, ENABLE);

}

void TIM4\_pwm(uint8\_t chn, uint16\_t duty)

{

if(chn==1)TIM4->CCR1 = duty;else

if(chn==2)TIM4->CCR2 = duty;else

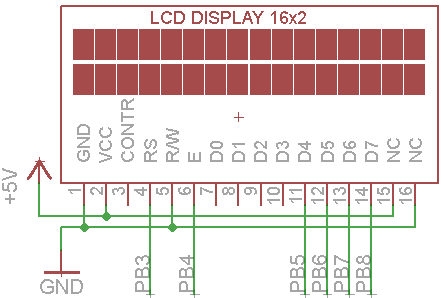
if(chn==3)TIM4->CCR3 = duty;else

if(chn==4)TIM4->CCR4 = duty;

}

### **Rangkaian *LCD* 16x4**

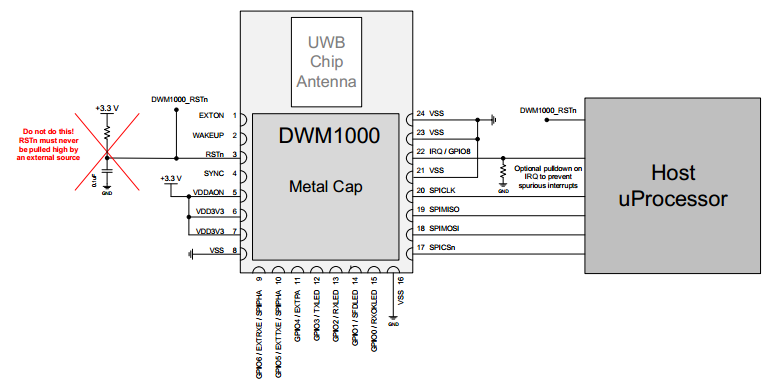
Data-data yang diolah dalam mikrokontroler akan ditampilkan dalam *LCD* 16x2. Berikut ini rangkaiannya yang ditunjukkan pada gambar 3.16.



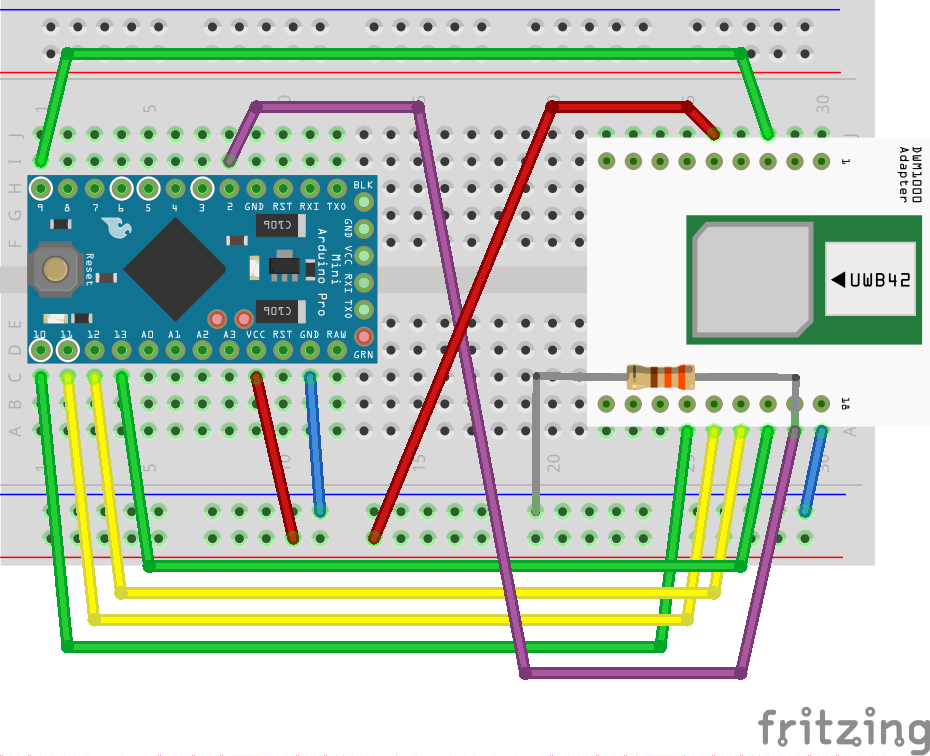
**Gambar 3.16.** Rangkaian *LCD* 16x2.

### **Akses Modul DWM1000**

Modul DWM1000 diakses lewat komunikasi SPI. Pada proyek akhir ini mikrokontroler yang digunakan untuk mengakses modul DWM1000 adalah arduino Pro mini. Dibutuhkan 8 pin untuk mengakses modul DWM1000, yaitu VDD 3.3V, Ground, MISO, MOSI, SCK, CS, Reset, dan IRQ. Untuk menggunakan modul ini pertama – tama *device* *address* dan *network* *ID* dari modul harus di-*set* terlebih dahulu. Gambar 3.17 menunjukkan rangkaian skematik *interfacing* dari modul DWM1000 dan gambar 3.18 menunjukka *wiring* antara modul DWM1000 dengan Arduino Mini Pro.



**Gambar 3.17.** Rangkaian *interface* Modul DWM1000[12].



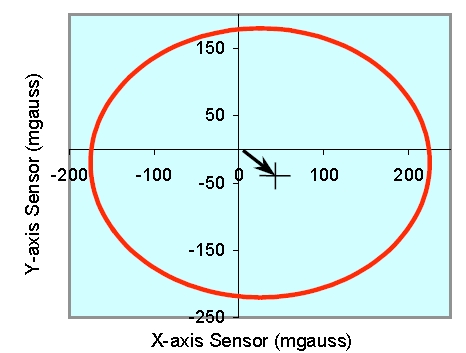
**Gambar 3.18.** Rangkaian *interface* DWM1000 dengan Arduino[16].

## **Perancangan dan Pembuatan Perangkat Lunak**

### **Algoritma Menentukan *Heading***

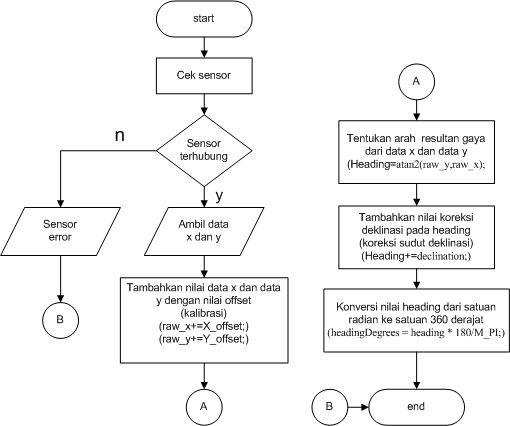
*Heading* robot ditentukan berdasarkan arah medan magnet bumi seperti halnya kompas magnetik. Untuk membaca kuat medan dan arah medan magnet bumi, digunakan sensor *magnetometer* HMC5883 yang terbenam dalam modul sensor GY-85. Untuk mengakses sensor ini, digunakan komunikasi *serial* I2C.

Untuk mendapatkan arah *heading* dari sensor, diperlukan data medan magnetik pada sumbu x dan y sensor. Data ini berupa besar medan magnet dan arah medan magnet bumi. Setelah kedua data didapatkan, selanjutnya kedua data tersebut di-vektorkan terhadap sumbu x dan sumbu y. Resultan gaya medan magnet akan membentuk resultan dengan arah tertentu pada sumbu x dan sumbu y. Arah resultan gaya ini yang akan dijadikan sebagai *heading*, seperti ditunjukkan pada gambar 3.19.



**Gambar 3.19.** Menentukan *heading* dengan *magnetometer*.

Berikut adalah *flowchart* bagaimana menetukna *heading* dengan menggunakan sensor HMC5883 ditunjukkan pada gambar 3.20:



**Gambar 3.20.** *Flowchart*algoritma menentukan *heading.*

berikut adalah *listing* program untuk menentukan *heading*:

if(hmc\_y>=32768)

{

hmc\_y=~hmc\_y;

raw\_y=-1-hmc\_y;

}else raw\_y=hmc\_y;

raw\_x+=X\_*offset*;

raw\_y+=Y\_*offset*;

*heading*=atan2(raw\_y,raw\_x);

*heading* += 0.0486174271; //declination correction

// Correct for when signs are reversed.

if(*heading* < 0)

*heading* += 2\*M\_PI;

// Check for wrap due to addition of declination.

if(*heading* > 2\*M\_PI)

*heading* -= 2\*M\_PI;

*heading*Degrees=*heading*\*180/M\_PI;

return *heading*Degrees;

}

uint16\_t get\_compass()

{

uint16\_t *heading*Degrees=0;

float *heading*=0;

uint8\_t hmc[6];

uint16\_t hmc\_x,hmc\_y,hmc\_z;

uint16\_t raw;

int32\_t raw\_x,raw\_y,raw\_z,rawe;

const int X\_*offset*=75;

const int Y\_*offset*=158;

if(!check\_i2c\_*device*(I2C2,0x3D))

{

BT\_Send("hmc disconected ");

return 0;

}

HMC5883\_*read*(hmc);

hmc\_x = hmc[1]|(hmc[0]<<8);

hmc\_y = hmc[5]|(hmc[4]<<8);

if(hmc\_x>=32768)

{

hmc\_x=~hmc\_x;

raw\_x=-1-hmc\_x;

}else raw\_x=hmc\_x;

### **Menentukan Posisi Robot dengan Metode Trilaterasi**

Untuk menerapkan metode trilaterasi, dibutuhkan setidaknya 3 *anchor* *node* dan 1 *tag* *node* untuk mendapatkan posisi dalam bidang x dan y. Untuk setiap *node* *wireless* sensor dikontrol oleh sebuah mikrokontroler berbasis Arduino Pro Mini. Hasil paling optimal untuk menerapkan metode trilaterasi adalah pada area berbentuk persegi dengan posisi *anchor* membentuk segitiga seperti ditunjukkan pada gambar 3.21.



**Gambar 3.21.** Peletakkan *anchor* *node.*

Karena metode trilaterasi yang diterapkan adalah untuk 2 dimensi, maka nilai ketinggian diabaikan. Sehingga penempatan *anchor* *node* berada pada ketinggian yang sama dengan *tag* *node*.

Untuk menentukan dimana posisi robot, digunakan *tag* *node* sebagai posisi acuan robot, maka *tag* *node* dipasang pada *body* robot. *Tag* *node* mengumpulkan data jarak antara *tag* *node* dengan ketiga *anchor* *node* yang posisinya sudah ditentukan sebelumnya. Metode pengambilan dan pengolahan data jarak antar *node* diselesaikan dengan menggunakan metode TDOA dengan cara menghitung selisih antara waktu data yang terkirim dengan waktu data diterima oleh *node* lain. Proses pegambilan data ini telah disediakan dalam bentuk *library* yang dikembangkan oleh pengembang aplikasi modul DWM1000 berbasis Arduino.

Data hasil pengukuran jarak diperoleh dengan cara melakukan *polling* terhadap *anchor* *node* yang sedang aktif. Setelah ketiga data jarak diperoleh, data data tersebut akan dikirimkan ke mikrokontroler utama melalui jalur komunikasi *serial* UART dengan protokol tertentu. Penggunaan protokol pengiriman data ini diterapkan agar mudah dilakukan parsing data atau penyeleksian data berdasarkan parameter jarak terhadap *anchor*1, *anchor*2, dan *anchor*3.

Berikut adalah *frame* dari protokol komunikasi antara *tag* *node* dengan mikrokontroler utama ditunjukkan oleh tabel 3.3:

**Tabel 3.3.** *Frame* data komunikasi.

|  |  |  |  |
| --- | --- | --- | --- |
| *header* | data | penutup | ket |
| ! | r1 | & | *anchor*1 |
| @ | r2 | & | *anhcor2* |
| ^ | r3 | & | *anchor*3 |

Dimana: ! : *header* data jarak dengan *anchor*1

r1 : data jarak *anchor*1

@ : *header* data jarak dengan *anchor*2

r2 : data jarak dengan *anchor*2

^ : *header* data jarak dengan *anchor*3

r3 : data jarak dengan *anchor*3

& : penutup

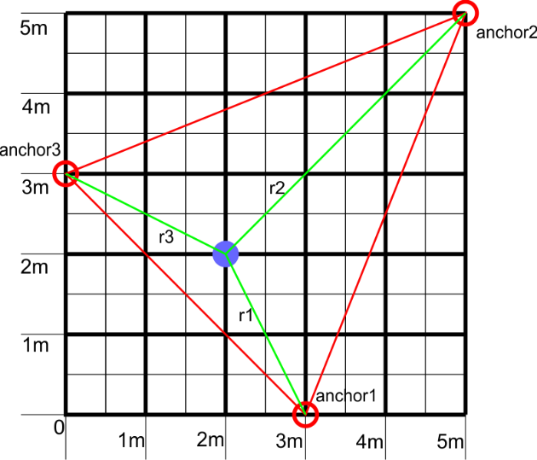
Setiap data masing – masing memiliki *frame* protokol sendiri, hal ini bertujuan untuk memudahkan proses pemisahan data, karena data jarak yang dikirim oleh *tag* *node* tidak selalu berurutan karena adanya proses *polling* pengambilan data, data jarak dari *anchor* mana yang didapat terlebih dahulu akan dikirim lebih dulu.

Perhitungan untuk menentukan posisi robot pada bidang x dan y dilakukan oleh mikrokontroler utama dengan menggunakan metode trilaterasi. Untuk melakukan perhitungan trilaterasi dibutuhkan dimana letak posisi *anchor*1(x1,y1), *anchor*2(x2,y2), dan *anchor*3(x3,y3) pada bidang x dan y, dan data jarak antara robot yang membawa *tag* *node* dengan masing - masingketiga *anchor*, yaitu r1, r2, dan r3. Untuk menentukan posisi robot digunakan persamaan 3.1 dan 3.2 [7]:

…………………………………………………………………….(3.1)

…………………………………………………………………….(3.2)

Untuk menggunakan metode trilaterasi, berikut adalah contoh perhitungan pada suati titik sampel seperti tampak pada gambar 3.22:



**Gambar 3.22.** Penentuan letak *anchor* *node* dan robot

Diketahui : r1 = 2.24m

r2 = 4.24m

r3 = 2.24m

titik *anchor*1 = (3,0)m

titik *anchor*2 = (5,5)m

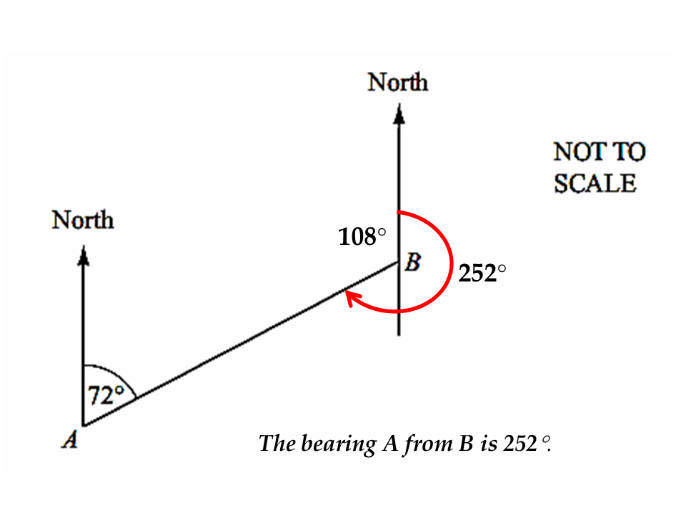
titik *anchor*3 = (0,3)m

maka perhitungan trilaterasinya adalah sebagai berikut:

Dari perhitungan di atas, hasil dari perkiraan posisi robot adalah di titik koordinat dengan nilai x = 2.003 dan y = 2.003.

### **Menentukan Arah *Bearing* dan Jarak Target**

Arah *bearing* adalah sudut dari titik awal menuju titik tujuan atau akhir perjalanan, *bearing* juga disebut *azimuth.* Untuk menentukan kemana arah *bearing* robot dibutuhkan data posisi awal robot dan posisi tujuan robot pada bidang x dan y. Penentuan arah *bearing* robot ditujukan agar robot dapat menentukan kemana dia akan bergerak menuju posisi yang dituju seperti tampak pada gambar 3.23.



**Gambar 3.23.** Arah *bearing* terhadap sudut 0° arah kompas

Untuk menentukan sudut *bearing* teradap arah utara yang digunakan sebagai arah acuan sudut 0°, digunakan persamaan *invers* trigonometri arc tan (x,y), atau dalam penulisan matematis dapat ditulis:

…………………………………………….(3.3)

Dimana: β = sudut *bearing*

∆*x* = jarak posisi awal dengan posisi akhir pada sumbu x

∆y = jarak posisi awal dengan posisi akhir pada sumbu y

Untuk menghitung jarak dari posisi awal ke posisi akhir dengan menggunakan rumus *phytagoras*, dengan menghitung jarak posisi awal dengan posisi akhir pada sumbu y, dan jarak posisi awal dengan posisi akhir pada sumbu x. Dalam penulisan matematis dapat ditulis:

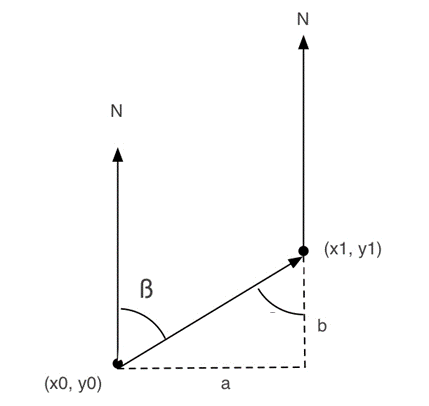
………………………………... (3.4)

……………...…..(3.5)

Dimana: r = jarak dari posisi awal ke posisi akhir

∆x = jarak posisi awal dengan posisi akhir pada sumbu x

∆y = jarak posisi awal dengan posisi akhir pada sumbu y



r

∆x

∆y

**Gambar 3.24.** Ilustrasi penghitungan jarak.

Dari ilustrasi gambar 3.24 dapat dihitung sudut *bearing* dari titik awal (x0,y0) ke titik tujuan (x1,y1) dengan menggunakan persamaan 3.3. Untuk menentukan sudut *bearing*, dengan memasukkan parameter – parameter yang ada pada gambar 3.24 ke dalam persamaan 3.6, maka penyelesainya menjadi:

…………………………………..(3.6)

…………………………….....(3.7)

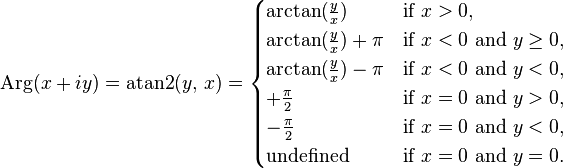
Dimana: *β* = arah *bearing*

∆x = jarak posisi awal dengan posisi akhir pada sumbu x

∆y = jarak posisi awal dengan posisi akhir pada sumbu y

Untuk mengimplementasikan sudut *bearing* dalam satuan sudut 0° - 359° maka dalam pemrograman digunakan fungsi atan2(y,x);

Dimana: atan2(y,x)adalah :



Selanjutnya nilai balik yang dihasilkan oleh fungsi atan2(y,x) yang berupa satuan *radian* dirubah ke dalam satuan derajat dengan *range* 0° - 359° sesuai dengan arah mata angin dalam satuan derajat. Berikut cuplikan programnya:

uint16\_t **calc\_heading**(**float** tgt\_x,**float** tgt\_y,**float** ori\_x,**float** ori\_y)

{

**float** heading=0;

uint16\_t headingDegrees=0;

heading=**atan2**((tgt\_x-ori\_x),(tgt\_y-ori\_y));

// Correct for when signs are reversed.

**if**(heading < 0)

heading += 2\***M\_PI**;

// Check for wrap due to addition of declination.

**if**(heading > 2\***M\_PI**)

heading -= 2\***M\_PI**;

headingDegrees = heading \* 180/**M\_PI**;

**return** headingDegrees;

}

Sedangkan untuk menghitung jarak antara titik awal dengan titik akhir, berikut adalah cuplikan programnya:

float calc\_distance(float tgt\_x, float tgt\_y,float ori\_x,float ori\_y)

{

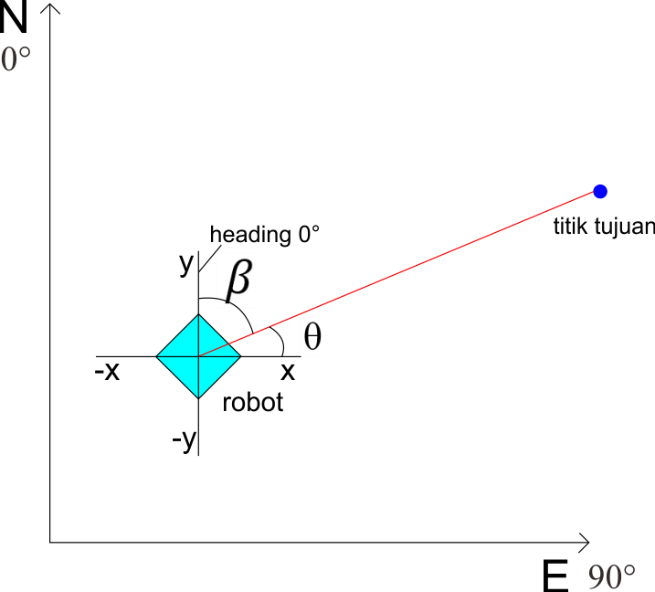
return (sqrt( pow((tgt\_y-ori\_y),2) + pow((tgt\_x-ori\_x),2)) );

}

### **Menentukan Gerakan Robot**

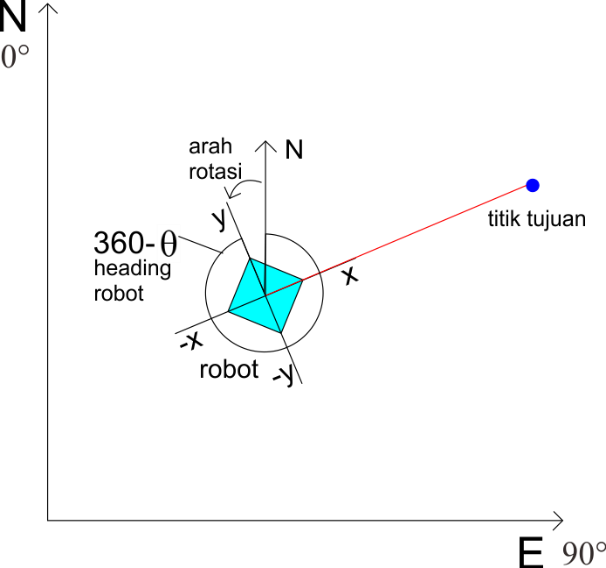
Untuk mendapatkan gerakan yang seefisien mungkin maka sebisa mungkin gerakan robot diminimalisir ketika robot bergerak ke posisi tujuanya. Desain mekanik robot memungkinkan robot untuk bergerak secara *omnidirectional*, yaitu robot dapat bergerak maju, mundur, ke samping kanan, ke samping kiri, dan memutar pada porosnya. Untuk mencapai gerakan yang seminimal mungkin ketika bergerak menuju posisi tujuanya, robot harus mengetahui terlebih dahulu di mana posisinya sekarang dan di mana posisi tujuanya. Dengan mengetahui kedua parameter tersebut maka dapat ditentukan kemana arah *heading* tujuan dari titik awal ke titik akhir, atau dapat disebut arah *bearing*.

Pertama – tama robot akan berotasi untuk mencapai arah hadap yang sesuai dengan arah *bearing* yang telah diperhitungkan. Untuk menentukan kemana robot harus berotasi, robot harus tahu kemana arah hadap atau *heading* robot sekarang. Karena robot dapat bergerak bebas maju, mundur, atau ke samping kanan – kiri maka untuk mencapai gerakan rotasi yang paling minimum adalah dengan membandingkan arah *bearing* menuju posisi akhir dengan arah hadapa robot saat ini, dan menentukan pada sisi yang mana robot akan begerak translasi (setelah gerakan rotasi terpenuhi).



**Gambar 3.25.** Ilustrasi penentuan arah *bearing*.

Dari gambar 3.25 diketahui bahwa sudut θ adalah sudut antara sisi x robot dengan arah *bearing* antara titik dimana robot berada dan titik tujuan robot. Dan sudut β merupakan sudut antara *heading* robot -yang searah dengan sisi y robot- dengan arah *bearing* menuju titik tujuan robot. Untuk menentukan kemana arah rotasi dan translasi robot agar didapat gerakan yang lebih minimum, pertama – tama yaitu membandingkan besar sudut antara keempat sisi robot dengan arah *bearing* yang sudah diketahui. Dari gambar 3.25 diketahui bahwa terdapat dua sisi robot yang paling dekat (sudut terkecil) dengan arah *bearing*, karena arah *bearing* berada di antara sisi x dan sisi y robot. Selanjutnyabesar sudut β dan θ dibandingkan mana yang lebih kecil. Dalam kasus pada gambar 3.25 besar sudut θ memiliki nilai yang lebih kecil, maka robot akan berotasi ke arah *heading* (360 – θ) sehingga sisi x robot berhimpit dengan arah *bearing*, seperti diilustrasikan pada gambar 3.26.



**Gambar 3.26.** Ilustrasi penentuan arah rotasi robot.

Untuk menghitung selisih besaran sudut yang terbentuk antara keempat sisi robot dengan arah *bearing* adalah dengan menghitung besar sudut antara *heading* robot dengan arah *bearing*, yang mana *heading* robot merupakan arah diman sisi y robot mengarah. Untuk menghitung besaran sudut pada sisi – sisi lain adalah dengan menambahkan konstanta nilai sudut sesuai dengan sisi yang akan dihitung. Berikut adalah penulisan matematisnya:

…………………………………...(3.8)

………………………………………….(3.9)

………………………………….(3.10)

…………………………………..(3.11)

Dengan batasan jika nilai dari *heading* dan arah *bearing* berada di kuadran 1 atau 4, maka:

…………………………………(3.12)

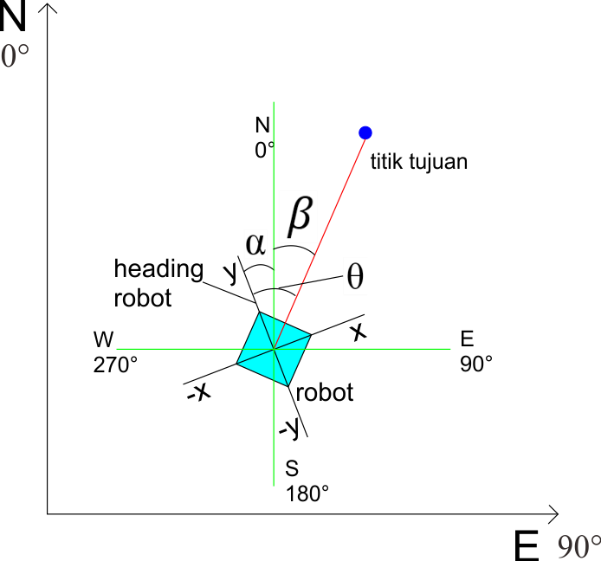
, …………………………..……………....(3.13)

jika *heading* berada di kuadran 4 dan arah *bearing* di kuadran 1, seperti tampak pada gambar 3.27, maka:

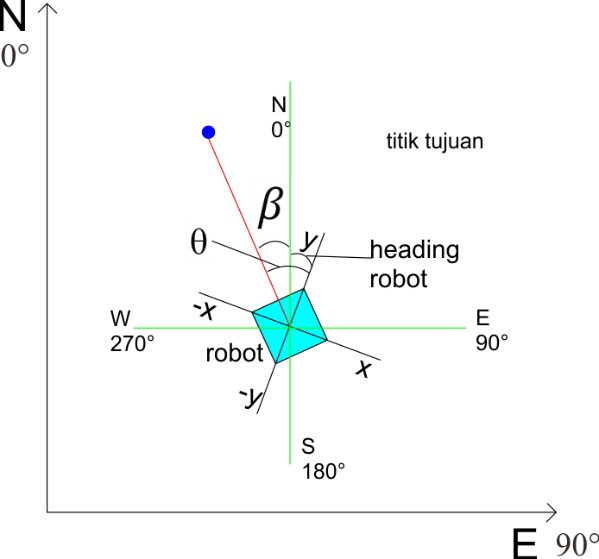
………………....……….(3.14)

jika arah *bearing* berada di kuadran 4 dan *heading* di kuadran 1, seperti tampak pada gambar 3.28, maka:

…….…………………......(3.15)



**Gambar 3.27.** Selisih sudut antara *heading* dengan arah *bearing*.



**Gambar 3.28.** Selisih sudut antara *heading* dengan arah *bearing*.

Untuk membatasi nilai dari hasil penghitungan selisih sudut ,, dan agar tetap dalam *range* sudut 0° - 359°, maka apabila penghitungan nilai ,, dan lebih besar dari 359° maka hasil penghitungan nilai ,, dan dikurangi 360°, atau dapat ditulis:

Jika (>359°), maka ;

Jika (>359°), maka ;

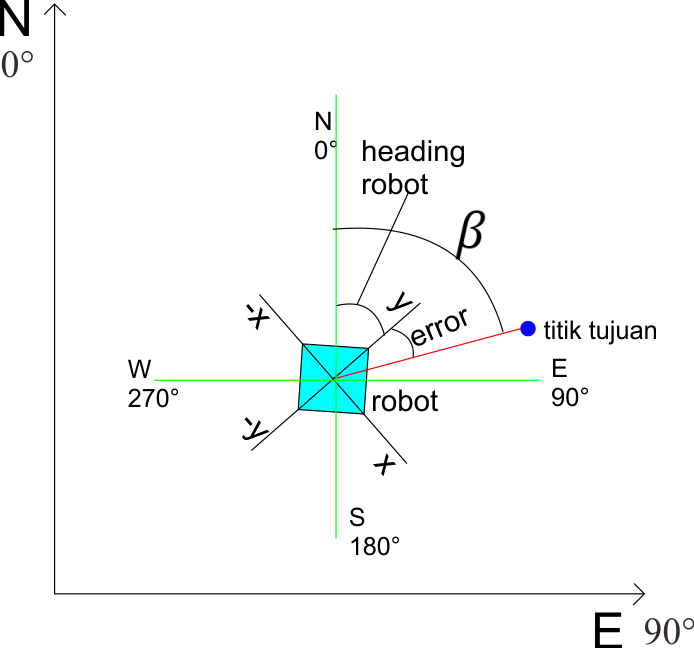
Jika (>359°), maka ;

Keempat nilai selisih sudut dibandingkan mana yang memiliki selisihsudut paling kecil untuk menentukan keputusan memilih arah gerak robot secara translasi. Apakah ke arah sisiy, x, -y, atau –x.

### **Kontrol Arah Hadap Robot**

Kontrol arah hadap robot (*heading*) digunakan agar robot tetap mengarah pada titik tujuanya, juga agar robot dapat mempertahankan *heading*-nya ketika mendapat gangguan dari luar. Untuk mendapatkan kontrol *heading* robot digunakan kontrol P (kontrol proporsional) dalam kontrol PID. Kontrol PID merupakan teknik kontrol yang umum dan banyak digunakan dalam dunia kontrol. Dalam proyek akhir ini,untuk mendapatkan tuning konsanta kP dilakukan trial and *error* untuk mendapatkan respon yang terbaik.

Untuk menjalankan kontrol tersebut tentunya dibutuhkan parameter *error* dan *time* *sampling* tertentu. Untuk mendapatakan nilai *error* adalah dengan menghitungselisih antara arah *heading* robot dengan arah *bearing* yag dijadikan *set* *point* seperti diilustrasikan pada gambar 3.29, atau dalam eksresi matematis dapat ditulis:



**Gambar 3.29.**Ilustrasi penentuan *error* *heading* robot.

Kontrol *heading* robot dijalankan bersamaan dengan gerakan robot menuju titik tujuan, kontrol P diaplikasikan pada putaran motor sehingga robot dapat berjalan lurus pada *heading* yang tepat untuk mengarah ke titik tujuan. Kontrol P yang diaplikasikan akan mempengaruhi perbedaan kecepatan dan atau perbedaan arah putar motor sehingga menyebabkan robot dapat berbelok. Kontrol ini berjalan dengan *time* *sampling* selama 0,01 detik, setiap 0,01 detik sekali system mikrokontroler akan melakukan interupsi untuk mengupdate nilai *heading* robot, posisi robot menurut perhitungan trilaterasi, arah *bearing* menuju titik tujuan, dan jarak robot menuju titik tujuan, setelah data – data terkumpulkan selanjutnya sistemakan menjalankan kontrol P agar robot tetap pada jalurnya menuju titik tujuan.

Berikut adalah cuplikan programnya:

**void** **TIM3\_IRQHandler**()

{

**if**(**TIM\_GetITStatus**(**TIM3**, **TIM\_IT\_Update**) != *RESET*)

{

**TIM\_ClearITPending*Bit***(**TIM3**, **TIM\_IT\_Update**);

***deg***=**get\_compass**();

**trilaterasi**(&***tri\_x***,&***tri\_y***,***r1***,***r2***,***r3***);

***head\_tgt***=**calc\_*heading***(***trgt\_x***,***trgt\_y***,***tri\_x***,***tri\_y***);

***distance***=**calc\_distance**(***trgt\_x***,***trgt\_y***,***tri\_x***,***tri\_y***);

**move\_decission**(***deg***,***distance***,&***head\_tgt***,&***set\_dir***);

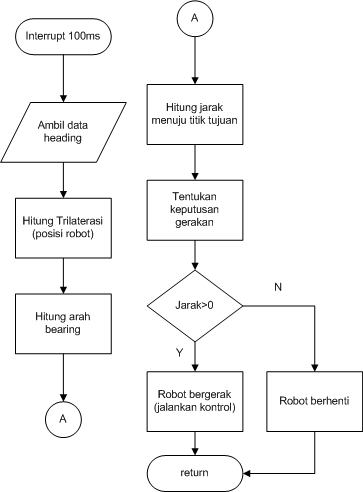
**if**(!***distance***)**move\_*set***(***stop***,***deg***,***head\_tgt***,***spd***,***kP***);

**else** **move\_*set***(***set\_dir***,***deg***,***head\_tgt***,***spd***,***kP***);

}

}

Berikut adalah gambar *flowchart-*nya yang ditunjukkan pada gambar 3.30:



**Gambar 3.30.** *Flowchart* kontrol pergerakan robot.

# PENGUJIAN DAN ANALISIS SISTEM

Pada bab ini akan dibahas tentang pengujian berdasarkan perencanaan dari sistem yang telah dibuat. Program pengujian disimulasikan di suatu sistem yang sesuai. Pengujian dilaksanakan untuk mengetahui apakah sistem sudah sesuai dengan perencanaan atau belum. Pengujian dilakukan secara terpisah, dan kemudian dilakukan pengujian sistem secara keseluruhan.

## **Pengujian Mekanik**

1. Tujuan

Mengukur besar kesalahan rangkaian mekanik akibat kesalahan pemasangan mekanik dan perbedaan kecepatan rotasi motor.

1. Peralatan

* *Mobile robot*
* Penggaris

1. Parameter

* Pengujian dilakukan di laboratorium.
* Simpangan yang diukur dalam sumbu x dan y.
* Robot dijalankan lurus pada sumbu x dan y.

1. Prosedur Percobaan

* Menandai titik referensi 0 di atas lantai.
* Meletakan robot pada titik referensi.
* Menjalankan robot tiap 5detik.
* Mencatat jarak perpindahan robot dan simpangan terhadap garis referensi.

1. Hasil dan Analisis

**Tabel 4.1.** Pengukuran Perpindahan Robot pada Sumbu Y.

|  |  |  |  |
| --- | --- | --- | --- |
| Percobaan | simpangan sumbu x (cm) | perpindahan Sumbu x (cm) | Sudut simpangan y (θy) (derajat) |
| 1 | 13 | 276 | 2.697 |
| 2 | 26 | 268 | 5.541 |
| 3 | 30 | 267 | 6.411 |
| 4 | 33 | 270 | 6.968 |
| 5 | 42 | 260 | 9.176 |
| 6 | 45 | 258 | 9.894 |
| 7 | 30 | 268 | 6.387 |
| 8 | 13 | 260 | 2.862 |
| 9 | 7 | 280 | 1.432 |
| 10 | 17 | 278 | 3.499 |
| Rata-rata | 25.6 | 268.5 | 5.487 |

**Gambar 4.1.** Grafik rata – rata simpangan mekanik pada sumbu y.

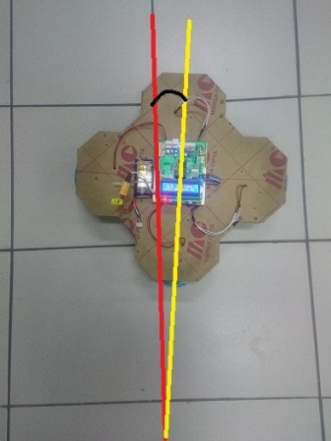
Tabel 4.1 dan gambar 4.1 menunjukkan hasil bahwa ketika robot dijalankan selama 5 detik dengan duty cycle pwm sebesar 75%, perpindahan rata – rata robot pada sumbu y sejauh 268,5 cm. Sedangkan *error* simpangan yang diukur dari garis referensi pada sumbu y didapatkan rata – rata simpangan sebesar 25,6 cm.

**Tabel 4.2.** Pengukuran perpindahan robot pada sumbu x.

|  |  |  |  |
| --- | --- | --- | --- |
| Percobaan | Perpindahan sumbu x (cm) | Simpangan Sumbu y (cm) | Sudut simpangan x (θx) (derajat) |
| 1 | 270 | 14 | 2.968 |
| 2 | 268 | 7 | 1.496 |
| 3 | 248 | 28 | 6.442 |
| 4 | 254 | 16 | 3.604 |
| 5 | 270 | 7 | 1.485 |
| 6 | 247 | 20 | 4.629 |
| 7 | 249 | 18 | 4.135 |
| 8 | 258 | 15 | 3.327 |
| 9 | 263 | 10 | 2.177 |
| 10 | 250 | 20 | 4.574 |
| Rata-rata | 257.7 | 15.5 | 3.484 |

**Gambar 4.2.** Grafik rata – rata simpangan mekanik pada sumbu x.

Untuk mengetahui seberapa besar *error* yang dihasilkan dengan menghitung sudut simpangan antara garis yang dibentuk oleh perpindahan robot dari titik awal sampai titik akhir terhadap garis referensi y. Rata – rata besar sudut simpangan yang didapat adalah 5,487o. Nilai didapatkan dengan menggunakan persamaan 4.1.

**(a) (b)**

**Gambar 4.3.** Ilustrasi pengujian kesalahan simpangan mekanik,

**(a)** simpangan mekanik sumbu y, **(b)** simpangan mekanik sumbu x.

………………………………………… (4.1)

………………………………………… (4.2)

Gambar 4.3 adalah gambar ilustrasi pengujian simpangan mekanik pada sumbu x dan sumbu y mekanik robot. Tabel 4.2 dan gambar 4.2 menunjukkan hasil bahwa ketika robot dijalankan selama 5 detik dengan duty cycle pwm sebesar 75%, perpindahan rata – rata robot pada sumbu y sejauh 257,7 cm. Sedangkan *error* simpangan yang diukur dari garis referensi pada sumbu y didapatkan rata – rata simpangan sebesar 15,5 cm. Untuk mengetahui seberapa besar *error* yang dihasilkan dengan menghitung sudut simpangan antara garis yang dibentuk oleh perpindahan robot dari titik awal sampai titik akhir terhadap garis referensi y. Rata – rata besar sudut simpangan yang didapat adalah 3,484o. Nilai didapatkan dengan menggunakan persamaan 4.2.

## **Pengujian Mikrokontroler**

1. Tujuan

Menguji apakah mikrokontroler bekerja dengan baik dan sesuai program yang dibuat.

1. Peralatan

* *Minimum system* mikrokontroler STM32F40VG
* *Software* CooCox COIDE
* *Software* FlyMcu in system programmer
* *Software* Hterm

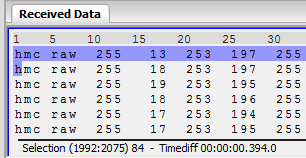
1. Parameter

* Pengujian dilakukan di laboratorium.
* Pengujian dilakukan untuk meguji ketepatan *timer* dan komunikasi UART.

1. Prosedur Percobaan

* Mambuat program untuk melakukan interupsi *timer* setiap 400ms dan mengirimkan beberapa karakter melalui USART.
* Manghubungkan mikrokontroler dengan PC lalu meng-*upload* program yang telah dibuat di-*compile*.
* Menghubungkan *software* HTerm pada PC dengan mikrokontroler dengan menggunakan USB to UART *converter*.
* Mengamati karakter yang diterima oleh *software* HTerm dan mencatat *time diff* setiap interupsi yang terjadi pada mikrokontroler.

1. Hasil dan Analisis



**Gambar 4.4.** Tampilan pada *software* Hterm.

**Tabel 4.3.** Pengujian interupsi *timer* dan komunkasi USART.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Interupsi ke- | Jeda interupsi (ms) | Jumlah karakter *error* | *Error* waktu interupsi (ms) | *Error* waktu interupsi (%) |
| 1 | 394 | 0 | -6 | 1.5 |
| 2 | 402 | 0 | 2 | 0.5 |
| 3 | 398 | 0 | -2 | 0.5 |
| 4 | 391 | 0 | -9 | 2.25 |
| 5 | 412 | 0 | 12 | 3 |
| 6 | 392 | 0 | -8 | 2 |
| 7 | 391 | 0 | -9 | 2.25 |
| 8 | 419 | 0 | 19 | 4.75 |
| 9 | 396 | 0 | -4 | 1 |
| 10 | 411 | 0 | 11 | 2.75 |
| Rata-rata | 400.6 | 0 | 15.5 | 2.05 |

Table 4.3 dan gambar 4.4 menunjukkan bahwa selama komunikasi berlangsung tidak ada karakter yang hilang atau *error*, semua karakter yang dikirim dari mikrokontroler dan yang diterima oleh PC tidak ada yang mengalami *lost.* Dari pewaktuan *timer* interupsi yang dimonitor melalui PC menunjukkan bahwa rata – rata waktu jeda interupsi adalah selama 400,6ms, hal ini menunjukkan bahwa *timer* yang dibangkitkan sudah dapat dikatakan sesuai dengan yang diinginkan yaitu 400ms.

## **Pengujian Sensor *Magnetometer***

1. Tujuan

Menguji dan melakukan kalibrasi pada sensor *magnetometer* untuk mendapatkan *heading.*

1. Peralatan

* *Minimum system* mikrokontroler STM32F40VG
* *Software* CooCox COIDE
* *Software* FlyMcu in system programmer
* *Software* Hterm
* Sensor *magnetometer* HMC5883
* *Software* Microsoft Excel

1. Parameter

* Pengujian dilakukan di laboratorium.
* Pengujian dilakukan untuk meguji ketepatan pembacaan sensor *magnetometer* dan melakukan kalibrasi sehingga didapat *heading* yang sesuai.

1. Prosedur Percobaan

* Membuat program untuk membaca nilai medan magnetik dan melakukan kalibrasi sensor *magnetometer*.
* Manghubungkan mikrokontroler dengan PC lalu meng-*upload* program yang telah dibuat di-*compile*.
* Menghubungkan *software* HTerm pada PC dengan mikrokontroler untuk mendapatkan data mentah.
* Mengamati data dan mencatat data pada *sheet* *Software* Microsoft Excel.
* Melakukan kalibrasi sensor.

1. Hasil dan Analisis

*Magnetometer* pada umumnya memiliki karakteristik dimana dalam lokasi di permukaan bumi mempunyai nilai yang berbeda – beda di setiap lokasi, maka perlu dilakukan kalibrasi untuk mendapatkan nilai *offset* agar data kuat medan yang akan divektorkan mendekati nol ketika dalam orientasi yang tegak lurus terhadap arah medan magnet bumi.

**Tabel 4.4.** Pengujian sensor *magnetometer* sebelum dikalibrasi.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| sudut referensi | raw x (mGauss) | raw y (mGauss) | *Heading* (derajat) | *error* *heading* (derajat) |
| 0 | 390 | -181 | 335 | 25 |
| 20 | 362 | -27 | 355 | 25 |
| 40 | 293 | 98 | 19 | 21 |
| 60 | 173 | 201 | 51 | 9 |
| 80 | 22 | 250 | 86 | 6 |
| 90 | -63 | 265 | 101 | 11 |
| 100 | -137 | 268 | 117 | 17 |
| 120 | -290 | 223 | 142 | 22 |
| 140 | -422 | 135 | 163 | 23 |
| 160 | -502 | 13 | 178 | 18 |
| 180 | -541 | -131 | 194 | 14 |
| 200 | -514 | -282 | 207 | 7 |
| 220 | -446 | -405 | 222 | 2 |
| 240 | -335 | -507 | 237 | 3 |
| 260 | -182 | -562 | 252 | 8 |
| 270 | -101 | -569 | 260 | 10 |
| 280 | -22 | -564 | 268 | 12 |
| 300 | 132 | -532 | 284 | 16 |
| 320 | 261 | -444 | 300 | 20 |
| 340 | 346 | -323 | 317 | 23 |
| 360 | 390 | -181 | 360 | 25 |

**Tabel 4.5.** Pengujian sensor *magnetometer* setelah dikalibrasi.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| sudut referensi | raw x (mGauss) | raw y (mGauss) | *Heading* (derajat) | *error* *heading* (derajat) |
| 0 | 456 | -18 | 0 | 0 |
| 20 | 433 | 130 | 19 | 1 |
| 40 | 356 | 262 | 38 | 2 |
| 60 | 238 | 353 | 58 | 2 |
| 80 | 91 | 404 | 80 | 0 |
| 90 | 15 | 416 | 91 | 1 |
| 100 | -63 | 413 | 101 | 1 |
| 120 | -209 | 364 | 122 | 2 |
| 140 | -338 | 276 | 143 | 3 |
| 160 | -420 | 155 | 162 | 2 |
| 180 | -456 | 14 | 180 | 0 |
| 200 | -437 | -125 | 199 | 1 |
| 220 | -365 | -252 | 218 | 2 |
| 240 | -244 | -354 | 238 | 2 |
| 260 | -94 | -404 | 259 | 1 |
| 270 | -11 | -413 | 270 | 0 |
| 280 | 60 | -412 | 281 | 1 |
| 300 | 212 | -368 | 302 | 2 |
| 320 | 337 | -282 | 322 | 2 |
| 340 | 420 | -160 | 342 | 2 |
| 360 | 456 | -18 | 0 | 0 |

Metode kalibrasi yang digunakan adalah dengan mengetahui nilai ketika sensor diorientasikan tegak lurus terhadap arah utara dan selatan yang sebelumnya ditentukan dengan kompas magnetic, pada keadaan ini data harus menujukkan nilai nol, dari sini dapat ditentukan nilai *offset* yang sesuai. Proses menentukan *offset* ini masing – masing dilakukan pada unit *magnetometer* x dan y.

(4.3)

(4.4)

Dimana : = Data *magnetometer* x sebelum ditambah *offset*

= Data *magnetometer* x setelah ditambah *offset*

= *offset* ketika x tegak lurus dengan utara – selatan

= Data *magnetometer* y sebelum ditambah *offset*

= Data *magnetometer* y setelah ditambah *offset*

= *offset* ketika y tegak lurus dengan utara – selatan

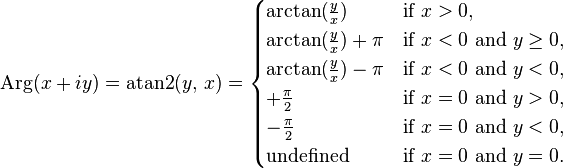
Tabel 4.4 dan tabel 4.5 ini adalah perbandingan data *magnetometer* yang belum dikalibrasi dan setelah dikalibrasi. Gambar 4.5 merupakan *plotting* data magntometer dan gambar 4.6 merupakan islustrasi pengujian sesor *magnetometer*.

Data *magnetometer* berupa kuat medan x dan y yang telah terkalibrasi selanjutnya dapat divektorkan untuk mendapatkan *heading* terhadap arah magnetik bumi. Untuk mendapatkan arah *heading* digunakan persamaan:

…………………….. (4.5)

….…(4.6)

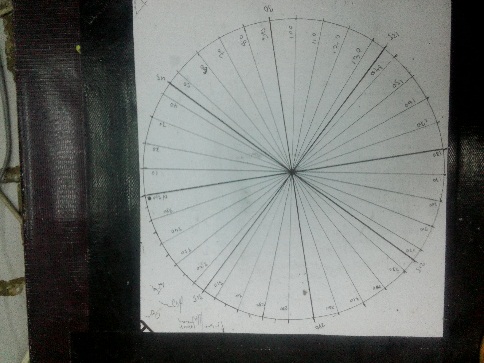
Dimana: *atan2* adalah :



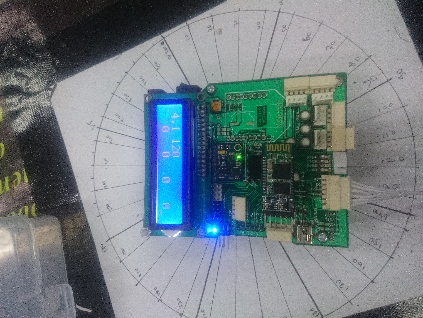
*heading =* *heading* dalam *radian*

*headingDegrees =* *heading* dalam sudut 0o – 359o

**Gambar 4.5.** *Plotting* data *magnetometer*.

(**a**) (**b**)



(**c**)

**Gambar 4.6.** Ilustrasi pengujian sensor *magnetometer*

(a)membuat referensi *heading*, (b)gambar referensi *heading*, (c)pengujian *heading.*

## **Pengujian Odometri**

1. Tujuan

Menguji odometri yang dirancang dan membandingkan hasilnya dengan perpindahan sesungguhnya.

1. Peralatan

* *Minimum system* mikrokontroler STM32F40VG
* *Software* CooCox COIDE
* *Software* FlyMcu *in system programmer*
* *Mobile robot*
* Penggaris

1. Parameter

* Pengujian dilakukan untuk meguji ketepatan odometri yang dirancanag dengan perpindahan robot yang terukur.

1. Prosedur Percobaan

* Membuat program untuk membaca pulsa dari *rotary encoder* dan mengubahnya menjadi satuan perpindahan robot.
* Manghubungkan mikrokontroler dengan PC lalu meng-*upload* program yang telah dibuat di-*compile*.
* Menjalankan robot dan menguji untuk mengetahui ketepatan odometri yang dibuat dan membandingan dengan hasil perpindahan yang sesungguhnya.
* Robot diuji dengan target odometri 60cm, 150cm, 240cm, dan 300cm, pengujian dilakukan pada odometri sumbu x dan sumbu y robot.
* Mencatat hasil perpindahan yang diperoleh dan membandingkan dengan target yang harus dicapai.

1. Hasil dan Analisis

**Tabel 4.6.** Pengujian odometri dengan target 150cm.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| percobaan | Perpindahan sumbu y(cm) | *error* (cm) | *Error* (%) | Perpindahan sumbu x(cm) | *error* (cm) | *Error* (%) |
| 1 | 150 | 0 | 0.00 | 145 | 5 | 3.33 |
| 2 | 150 | 0 | 0.00 | 143 | 7 | 4.67 |
| 3 | 151 | 1 | 0.67 | 144 | 6 | 4.00 |
| 4 | 145 | -5 | 3.33 | 143 | 7 | 4.67 |
| 5 | 155 | 5 | 3.33 | 145 | 5 | 3.33 |
| Rata-rata | 150.2 |  | 1.47 | 144 |  | 4.00 |

**Tabel 4.7.** Pengujian odometri dengan gangguan dengan target 150cm.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| percobaan | Perpindahan sumbu y(cm) | *error* (cm) | *Error* (%) | Perpindahan sumbu x(cm) | *error* (cm) | *Error* (%) |
| 1 | 115 | 35 | 23.33 | 95 | 55 | 36.67 |
| 2 | 125 | 25 | 16.67 | 124 | 26 | 17.33 |
| 3 | 107 | 43 | 28.67 | 136 | 14 | 9.33 |
| 4 | 103 | 47 | 31.33 | 104 | 46 | 30.67 |
| 5 | 106 | 44 | 29.33 | 96 | 54 | 36.00 |
| Rata-rata | 111.2 |  | 25.87 | 111 |  | 26.00 |

**Tabel 4.8.** Pengujian odometri dengan target 300cm.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| percobaan | Perpindahan sumbu y(cm) | *error* (cm) | *Error* (%) | Perpindahan sumbu x(cm) | *error* (cm) | *Error* (%) |
| 1 | 316 | 16 | 5.33 | 297 | 3 | 1.00 |
| 2 | 314 | 14 | 4.67 | 304 | 4 | 1.33 |
| 3 | 313 | 13 | 4.33 | 306 | 6 | 2.00 |
| 4 | 318 | 18 | 6.00 | 292 | 8 | 2.67 |
| 5 | 319 | 19 | 6.33 | 290 | 10 | 3.33 |
| Rata-rata | 316 |  | 5.33 | 297.8 |  | 2.07 |

**Tabel 4.9.** Pengujian odometri dengan ganguan dengan target 300cm.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| percobaan | Perpindahan sumbu y(cm) | *error* (cm) | *Error* (%) | Perpindahan sumbu x(cm) | *error* (cm) | *Error* (%) |
| 1 | 226 | 74 | 24.67 | 297 | 83 | 1.00 |
| 2 | 176 | 124 | 41.33 | 304 | 94 | 1.33 |
| 3 | 204 | 96 | 32.00 | 306 | 106 | 2.00 |
| 4 | 197 | 103 | 34.33 | 292 | 88 | 2.67 |
| 5 | 185 | 115 | 38.33 | 290 | 91 | 3.33 |
| Rata-rata | 197.6 |  | 34.13 | 207.6 |  | 30.8 |

**Gambar 4.7.** Grafik pengujian odometri pada sumbu y.

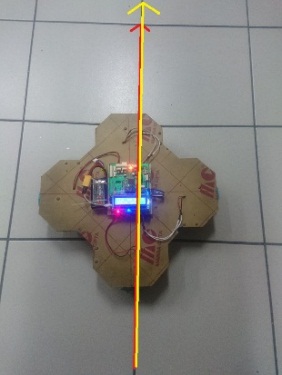
**Gambar 4.8.** Grafik pengujian odometri pada sumbu x.

Pada pengujian odometri robot nilai *error* diukur ketika robot telah mencapai titik akhir perpindahan robot, robot berjalan lurus pada sumbu x atau sumbu y dengan sekaligus membaca nilai odometri yang dihasilkan oleh *rotary encoder*. Robot akan berhenti ketika mencapai targetnya, lalu jarak perpindahan robot dikukur dan dibandingkan apakah sesuai dengan jarak perpindahan yang diinginkan. Selisih antara jarak yang diinginkan dan jarak perpindahan yang dilakukan robot akan dijadikan sebagai nilai *error*. Untuk menghitung persentase *error* digunakan persamaan 4.7:

..................(4.7)

Dari tabel 4.6, dan tabel 4.8 merupakan hasil pengujian odometri tanpa adanya gangguan pada robot. Tabel 4.7, dan tabel 4.9 merupakan hasil pengujian odometri denga adanya gangguan pada robot dan gambar 4.8 merupakan grafik untuk pengujian. Pengujian odometri pada target 150cm tanpa adanya gangguan menghasilkan *error* rata – rata sebesar 1.47% pada sumbu y dan 4% pada sumbu x. Pengujian odometri pada target 150cm dengan adanya gangguan menghasilkan *error* rata – rata sebesar 25.87% pada sumbu y dan 26% pada sumbu x. Pengujian odometri pada target 300 cm tanpa adanya gangguan menghasilkan *error* rata – rata sebesar 5.33% pada sumbu y dan 2.07% pada sumbu x. Pengujian odometri pada target 300cm dengan adanya gangguan menghasilkan *error* rata – rata sebesar 34.13% pada sumbu y dan 30.8% pada sumbu x. Gambar 4.7 adalah grafik pengujian odometri pada sumbu x dan gambar 4.8 adalah grafik pengujian pada sumbu y.

Dari pengamatan ketika dilakukan pengujian faktor yang menyebabkan ketidaktepatan odometri yang dilakukan oleh robot adalah roda robot yang beberapa kali mengalami selip, hal ini disebabkan roda robot yang terbuat dari bahan plastik sehingga daya cengkeram yang dihasilkan oleh permukaan roda idak begitu kuat pada permukaan lantai yang licin. Faktor lain yang menyebabkan kesalahan odometri adalah yaitu terjadinya sinyal *bouncing* yang dihasilkan sensor *rotary encoder* sehingga pembacaan sinyal tidak tepat. Gambar 4.9 merupakan ilustrasi pengujian odometri.



**Gambar 4.9.** Ilustrasi pengujian odometri.

## **Pengujian Modul DWM1000**

1. Tujuan

Melakukan pengujian untuk mengakses modul DWM1000.

1. Peralatan

* Modul mikrokontroler Arduino Pro Mini
* Arduino IDE
* Modul DWM1000
* *Converter* USB to TTL

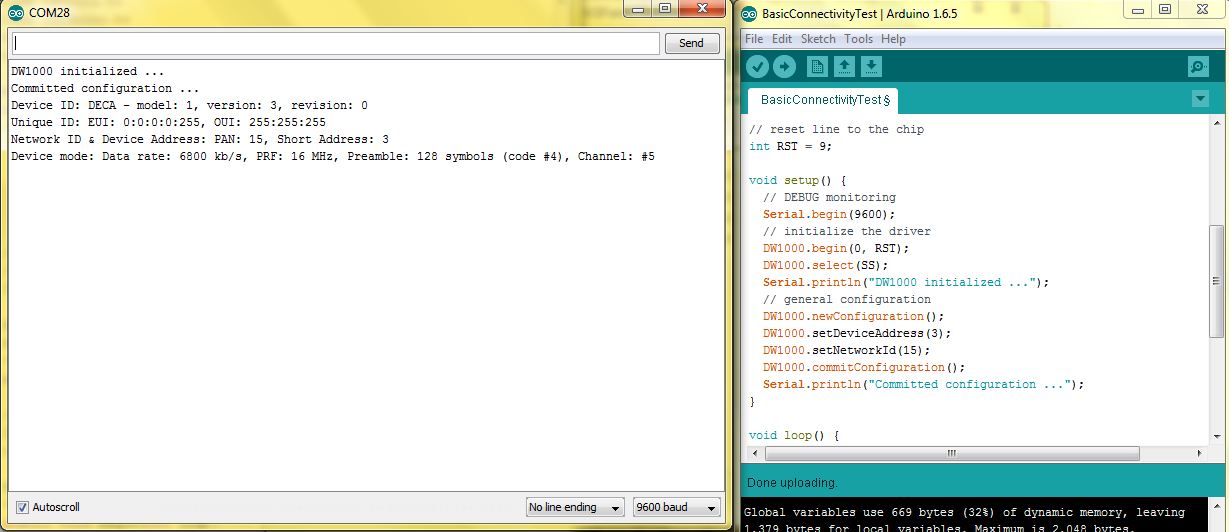
1. Parameter

* Pengujian dilakukan di laboratorium.
* Pengujian dilakukan untuk meguji konektifitas modul DWM1000 dengan mikrokontroler Arduino Pro Mini.

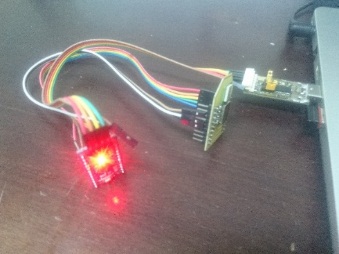
1. Prosedur Percobaan

* Membuat program untuk mensettng PAN ID dan Network ID modul DWM1000.
* Manghubungkan mikrokontroler dengan PC lalu meng-*upload* program yang telah dibuat dan di-*compile*.
* Mengamati hasil dari komunikasi mikrokontroler Aduino dengan modul DWM1000 dengan menggunakan *serial* *monitor* pada Arduino IDE.

1. Hasil dan Analisis



**Gambar 4.10.** Komunikasi Arduino dengan DWM1000.



**Gambar 4.11.** Pengujian konektifitas Arduino dengan DWM1000.

Dari hasil pengamatan seperti tampak pada gambar 4.10, pada pengujian konektifitas antara Arduino Pro Mini dengan modul DWM1000 didapat bahwa Arduino dapat melakukan inisialisasi terhadap modul DWM1000. Dari pengujian didapat bahwa Arduino berhasil men-*set* PAN ID dan Network ID untuk modul DWM1000 dan berhasil membaca kedua parameter tersebut dan mengirimkan hasilnya ke PC melalui komunikasi UART. Gambar 4.11 merupakan ilustrasi pengujian konektifitas arduino dengan modul DWM1000.

## **Pengujian *Ranging* Modul DWM1000**

1. Tujuan

Melakukan pengujian pengukuran jarak menggunakan modul DWM1000.

1. Peralatan

* Modul mikrokontroler Arduino Pro Mini
* Arduino IDE
* 2 Modul DWM1000, 1 sebagai *anchor* dan 1 sebagai *tag*
* *Converter* USB to TTL

1. Parameter

* Pengujian dilakukan di lorong gedung D4 PENS.
* Pengujian dilakukan untuk menguji ketepatan jarak yang didapatkan oleh modul DWM1000 seagai *anchor* dan *tag* dengan menggunakan mikrokontroler Arduino Pro Mini.

1. Prosedur Percobaan

* Membuat program untuk mensettng melakukan pengukuran jarak antara *tag* *node* dengan *anchor* *node.*
* Manghubungkan mikrokontroler dengan PC lalu meng-*upload* program yang telah dibuat di-*compile*.
* Mengamati hasil dari komunikasi mikrokontroler Aduino dengan modul DWM1000 dengan menggunakan *serial* *monitor* pada Arduino IDE.

1. Hasil dan Analisis

Tabel pengujian 4.10 menunjukkan bahwa *error* yang didapat dari perbandingan dengan jarak yang terukur adalah sebesar 5.02%. Dapat dilihat dari tabel 4.10 dan gambar 4.12 bahwa untuk penggunaan aplikasi menggunakan modul DWM1000 pada jarak 1 sampai 20 meter modul ini dapat diandalkan, namun untuk penggunaan aplikasi dengan jarak 1 sampai 5 meter mungkin akan menghasilkan *error* pengukuran yang besar sehingga tidak cocok untuk diaplikasikan pada jarak pendek. Dari pengamatan ketika melakukan pengujian didapatkan bahwa ketika banyak halangan yang menghalangi *anch*or dan *tag* pembacaan jarak megalami *error* yang besar. Dalam proses pengujian kali ini tidak ada halangan benda apapun yang menghalangi *anchor* dan *tag.* Gambar 4.13 merupakan ilustrasi proses pengujian *ranging* modul DWM1000.

**Tabel 4.10.** Pengujian modul DWM1000.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| jarak terukur (m) | jarak oleh DWM1000 (m) | *Error*(%) | jarak terukur (m) | jarak oleh DWM1000 (m) | *Error*(%) |
| 1 | 1.14 | 14.00 | 11 | 11.43 | 3.91 |
| 2 | 2.21 | 10.50 | 12 | 12.42 | 3.50 |
| 3 | 3.26 | 8.67 | 13 | 13.36 | 2.77 |
| 4 | 4.42 | 10.50 | 14 | 14.46 | 3.29 |
| 5 | 5.25 | 5.00 | 15 | 15.52 | 3.47 |
| 6 | 6.32 | 5.33 | 16 | 16.61 | 3.81 |
| 7 | 7.23 | 3.29 | 17 | 17.64 | 3.76 |
| 8 | 8.27 | 3.37 | 18 | 18.47 | 2.61 |
| 9 | 9.32 | 3.56 | 19 | 19.46 | 2.42 |
| 10 | 10.35 | 3.50 | 20 | 20.61 | 3.05 |
| Rata - rata |  |  |  |  | 5.02 |

**Gambar 4.12.** *Plotting* pengujian jarak modul DWM1000.



**Gambar 4.13.** Proses pengujian *ranging* modul DWM1000.

## **Pembacaan Jarak Dan Penentuan Posisi Modul DWM1000**

1. Tujuan

Melakukan pengujian fluktasi pengukuran jarak menggunakan modul DWM1000.

1. Peralatan

* Modul mikrokontroler Arduino Pro Mini
* Arduino IDE
* 2 Modul DWM1000, 1 sebagai *anchor* dan 1 sebagai *tag*
* *Converter* USB to TTL

1. Parameter

* Pengujian dilakukan di lorong gedung D4 PENS.
* Pengujian dilakukan untuk menguji ketepatan jarak yang didapatkan oleh modul DWM1000 seagai *anchor* dan *tag* dengan menggunakan mikrokontroler Arduino Pro Mini.

1. Prosedur Percobaan

* Membuat program untuk mensettng melakukan pengukuran jarak antara *tag* *node* dengan *anchor* *node.*
* Manghubungkan mikrokontroler dengan PC lalu meng-*upload* program yang telah dibuat di-*compile*.
* Mengamati hasil dari komunikasi mikrokontroler Aduino dengan modul DWM1000 dengan menggunakan *serial* *monitor* pada Arduino IDE.

1. Hasil dan Analisis

**Tabel 4.11.** Pegujianpengukuran jarak dengan DWM1000.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | Pengukuran jarak pada titik- | | | |
| 1meter(m) | 2meter(m) | 3meter(m) | 4meter(m) |
| Percobaanke- | 1 | 0.98 | 1.99 | 2.95 | 3.94 |
| 2 | 1.00 | 1.95 | 2.88 | 3.99 |
| 3 | 0.90 | 1.98 | 2.91 | 4.00 |
| 4 | 0.81 | 1.99 | 2.94 | 3.99 |
| 5 | 1.04 | 2.00 | 3.00 | 3.98 |
| 6 | 0.91 | 2.02 | 2.96 | 3.96 |
| 7 | 0.96 | 2.00 | 2.98 | 4.00 |
| 8 | 0.85 | 1.96 | 2.99 | 4.02 |
| 9 | 0.96 | 2.00 | 2.94 | 4.00 |
| 10 | 0.97 | 1.98 | 3.03 | 4.02 |
| 11 | 0.94 | 2.01 | 2.99 | 4.00 |
| 12 | 0.97 | 2.01 | 3.02 | 4.02 |
| 13 | 0.90 | 1.99 | 3.03 | 4.00 |
| 14 | 1.00 | 2.00 | 3.00 | 4.01 |
| 15 | 0.97 | 1.97 | 2.99 | 3.99 |
| 16 | 0.98 | 2.01 | 3.02 | 4.01 |
| 17 | 0.98 | 1.99 | 3.04 | 3.98 |
| 18 | 1.03 | 1.99 | 3.04 | 4.03 |
| 19 | 0.97 | 2.00 | 3.00 | 4.02 |
| 20 | 1.02 | 1.95 | 2.96 | 4.00 |
| Rata – rata |  | 95.7 | 198.95 | 298.35 | 399.8 |
| Simpangan baku |  | 5.67 | 1.91 | 4.27 | 2.17 |

**Gambar 4.14.** Grafik pengukuran jarak DWM1000.

Tabel 4.11 merupakan hasil pengujian pembacaan jarak pengukuran oleh modul DWM1000. Pada tabel 4.11 disajikan data pengukuran pada jarak 1 meter, 2 meter, 3 meter, dan 4 meter, data yang disajikan adalah data pengukuran yang dihasilkan oleh modul DWM1000.Gambar 4.14 adalah grafik representasi dari data yang ada pada tabel 4.11.

**Tabel 4.12.** Latensi pengiriman data 3 jarak *anchor.*

|  |  |
| --- | --- |
| Jarak(meter) | Rata - rata latensi(ms) |
| 1 | 145.83 |
| 2 | 145.92 |
| 3 | 146.01 |
| 4 | 145.90 |
| 5 | 147.35 |
| 6 | 147.25 |
| 7 | 147.30 |
| 8 | 148.84 |
| 9 | 148.82 |
| 10 | 151.71 |

Tabel 4.12 menunjukkan latensi pengiriman data pengukuran jarak dari *tag node* ke mikrokontroler utama. Pengujian dilakukan pada jarak antara *tag node* dengan 3 *anchor node* sekaligus dengan jara yang sama untuk ketiga – tiganya. Pengujian dilakukan dengan melihat pewaktu yang dibuat di dalam program mikrokontroler utama untuk melihat lama jeda waktu pengiriman antara data satu dengan data berikutnya. Data untuk setiap jarak yang ditentukan diambil 20 data latensi lalu dirata – rata untuk mendapatkan data rata – rata latensi dalam satuan milidetik. Dari tabel 4.12 diketahui bahwa rata – rata latensi paling lama didapatkan pada jarak 10 meter dengan latensi rata – rata selama 151.71 milidetik.

Untuk mendapatkan jarak antara *tag node* dengan *anchor node* dibutuhkan beberapa lama waktu. Dari tabel 4.12 diketahui bahwa rata – rata latensi semaki lama pada jarak yang semakin jauh. Hal ini disebabkan proses pengukuran jarak antara *tag node* dengan anchor node menggunakan teknik *TDOA,* yaitu teknik pengukuran jarak yang menggunakan selisih waktu antara data yang terkirim dari *anchor node* dengan data yang diterima oleh *tag node,* sehingga semakin panjang jarak yang harus diukur maka semakin lama waktu pengukuranya.

**Tabel 4.13.**Perbandingan jarak DWM1000 dengan RSSI Bluetooth.

|  |  |  |  |
| --- | --- | --- | --- |
| Jarak sesungguhnya (m) | Jarak terukur oleh DWM (m) | Jarak terukur oleh Blueooth1 (m) | Jarak terukur oleh Bluetooth2 (m) |
| 1 | 1.14 | 1.51 | 0.81 |
| 2 | 2.21 | 2.06 | 1.43 |
| 3 | 3.26 | 1.93 | 1.74 |
| 4 | 4.42 | 4.03 | 4.76 |
| 5 | 5.25 | 4.00 | 3.94 |
| 6 | 6.32 | 5.02 | 3.83 |
| 7 | 7.23 | 6.03 | 5.93 |
| 8 | 8.27 | 11.01 | 11.15 |

**Gambar 4.15.** Perbandingan DWM1000 dengan RSSI Bluetooth.

Tabel 4.13 merupakan tabel perbandingan pengukuran jarak yang dilakukan menggunakan modul DWM1000 dengan pengukuran jarak dengan menggunakan perhitungan RSSI Bluetooth. Data pengkuran dengan mengunakan perhitungan RSSI Bluetooth didapatkan dari penelitian tentang *indoor positioning system* berbasis Bluetoooth *low energy.*[9]

**Tabel 4.14.** Pengujian posisi dengan modul DWM1000.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Xt | Yt | X (uji) | Y (uji) | Jarak *error* (m) | *Error* (%) |
| 1 | 1 | 0.87 | 0.88 | 0.18 | 12.51 |
| 2 | 2 | 1.9 | 1.6 | 0.41 | 14.58 |
| 3 | 3 | 3.07 | 3.14 | 0.16 | 3.69 |
| 4 | 4 | 4.03 | 4.08 | 0.09 | 1.51 |
| 5 | 5 | 5.08 | 4.96 | 0.09 | 1.26 |
| 6 | 6 | 6.19 | 5.91 | 0.21 | 2.48 |
| 7 | 7 | 7.14 | 6.96 | 0.15 | 1.47 |
| 8 | 8 | 8.32 | 8.05 | 0.32 | 2.86 |
| Rata – rata *error* | 5.05 | | | | |

**Gambar 4.16.** Grafik pengujian posisi dengan modul DWM1000.

Tabel 4.14 menunjukkan hasil pengujian posisi DWM1000 dengan menggunakan trilaterasi. Hasil dari tabel 4.14 menunjukkan bahwa rata – rata *error* penghitngan trilaterasi adalah 5.05%. Gambar 4.16 merupakan grafik pengujian posisi menggunakan trilaterasi. Gambar 4.15 menunjukkan perbandingan jarakyang terukur oleh modul DWM1000 dengan perhitungan jarak dengan RSSI Bluetooth.

## **Pengujian ketepatan *Path* *Planning* robot**

1. Tujuan

Melakukan pengujian ketepatan pergerakaan robot yang bergrak dari titik satuke titik lain.

1. Peralatan

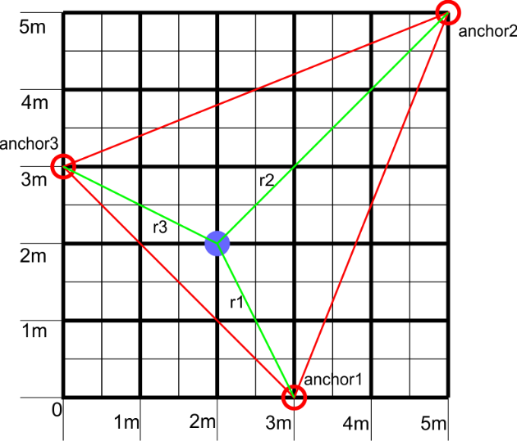
* Robot
* 4 Modul DWM1000, 3 sebagai *anchor* dan 1 sebagai *tag*
* HTerm
* Modul BluetoothHC-05

1. Parameter

* Pengujian dilakukan di lapagan gedung D4 PENS.
* Pengujian dilakukan untuk menguji ketepatan pergerakaan robot yang didapatkan.

1. Prosedur Percobaan

* Membuat program *path* *planning* pada robot.
* Menghubungkan robot dengan PC melalui Bluetooth.
* Mengubungkan robot dengan Hterm pada PC untuk melakukan *monitoring* data dan melakukan perintah untuk menentukan posisi yang dituju.
* Meletakkan robot dan *anchor* *node* pada bidang yang telah diukur dan ditandai seperti tampak pada gambar 4.17.
* Menjalankan robot dengan mengirimkan perintah melalui Hterm.
* Mengamati hasil pergerakan robot melalui *data streaming* pada Hterm dan pengukuran pada kenyataan.



**Gambar 4.17.** Peletakkan robot dan *anchor* *node.*

1. Hasil dan Analisis

**Tabel 4.15.**Posisi pergerakan robot dari ttik (0,0)m ke titik (2,2)m.

|  |  |  |
| --- | --- | --- |
| Data ke -n | Posisi robot sumbu x(m) | Posisi robot sumbu y(m) |
| 1 | 0.01 | 0.13 |
| 2 | 0.21 | 0.33 |
| 3 | 0.61 | 0.74 |
| 4 | 0.79 | 0.86 |
| 5 | 1.03 | 1.08 |
| 6 | 1.24 | 1.15 |
| 7 | 1.43 | 1.37 |
| 8 | 1.59 | 1.48 |
| 9 | 1.71 | 1.64 |
| 10 | 1.98 | 2.01 |

**Tabel 4.16.** Posisi pergerakan robot dari ttik (2,2)m ke titik (4,1)m.

|  |  |  |
| --- | --- | --- |
| Data ke -n | Posisi robot sumbu x(m) | Posisi robot sumbu y(m) |
| 1 | 2.09 | 2.08 |
| 2 | 2.04 | 1.93 |
| 3 | 2.29 | 1.79 |
| 4 | 2.96 | 1.6 |
| 5 | 3.29 | 1.41 |
| 6 | 3.41 | 1.35 |
| 7 | 3.64 | 1.27 |
| 8 | 3.73 | 1.22 |
| 9 | 3.88 | 1.08 |
| 10 | 3.93 | 0.97 |

|  |  |  |
| --- | --- | --- |
| Data ke -n | Posisi robot sumbu x(m) | Posisi robot sumbu y(m) |
| 1 | 3.97 | 0.97 |
| 2 | 3.99 | 1.17 |
| 3 | 4.03 | 1.27 |
| 4 | 3.94 | 1.64 |
| 5 | 3.98 | 1.96 |
| 6 | 3.97 | 2.19 |
| 7 | 4.01 | 2.39 |
| 8 | 3.98 | 2.75 |
| 9 | 4.04 | 3.05 |
| 10 | 3.99 | 3.7 |

**Tabel 4.17.** Posisi pergerakan robot dari ttik (4,1)m ke titik (4,4)m.

Pada tabel 4.15 disajikan tabel pergerakan robot dari titik awal (0,0)m ke titik tujuan (2,2)m. Data yang disajikan berupa 10 titik sampel posisi robot ketika bergerak dari titik awal menuju ke titik akhir. Pada gambar 4.18 grafik pergerakan yang berhubungan dengan tabel 4.15 disajikan dengan *series* bernama *path*1 dan pengujian1.

Pada tabel 4.16 disajikan tabel pergerakan robot dari titik awal (2,2)m ke titik tujuan (4,1)m. Data yang disajikan berupa 10 titik sampel posisi robot ketika bergerak dari titik awal menuju ke titik akhir. Pada gambar 4.18 grafik pergerakan yang berhubungan dengan tabel 4.16 disajikan dengan *series* bernama *path*2 dan pengujian2

Pada tabel 4.17 disajikan tabel pergerakan robot dari titik awal (4,1)m ke titik tujuan (4,4)m. Data yang disajikan berupa 10 titik sampel posisi robot ketika bergerak dari titik awal menuju ke titik akhir. Pada gambar 4.18 grafik pergerakan yang berhubungan dengan tabel 4.17 disajikan dengan *series* bernama *path*3 dan pengujian3.

**Gambar 4.18.** Grafik pergerakan robot terhadap garis acuan.

Pengujian selanjutnya adalah pengujian ketepatan robot untuk mencapai titik tujuan. Pengujian dilakukan untuk menguji ketepatan robot mencapai titik tujuan (2,2)m dan (4,4)m dari titik awal (0,0)m.

Untuk menghitung Jarak dari titik awal ke target digunakan permsaman 4.8 berikut:

……………...………...(4.8)

Dimana: r = jarak dari posisi awal ke posisi akhir

∆x = jarak posisi awal dengan posisi akhir pada sumbu x

∆y = jarak posisi awal dengan posisi akhir pada sumbu y

Untuk menghitung jarak *error* digunakan persamaan 4.9 berikut:

……...(4.9)

Dimana: *dis\_error* = Jarak robot ke posisi target

*act\_y* = posisi target sumbu y

*res\_y* = Posisi robot sumbux

= posisi target sumbu y

*res\_x* = Posisi robot sumbuy

Untuk menghitung nilai *error* ketepatan robot menuju titik tujuan dihitung dengan mensubtitusikan nilai nilai *r* dari persamaan 4.8 dan nilai *dis\_error* dari persamaan 4.9 ke persamaan 4.10 berikut :

.................................................(4.10)

**Tabel 4.18.** Akurasi pencapaian tujuan (2,2)m dari tiik awal (0,0)m.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Percobaan ke- | Posisi robot sumbux (m) | Posisi robot sumbuy (m) | Jarak robot ke posisi target (m) | Jarak dari titik awal ke target (m) | *Error* (%) |
| 1 | 2.15 | 2.11 | 0.19 | 2.83 | 6.60 |
| 2 | 2.12 | 2.12 | 0.17 | 2.83 | 6.02 |
| 3 | 1.92 | 1.97 | 0.09 | 2.83 | 3.03 |
| 4 | 1.9 | 1.89 | 0.15 | 2.83 | 5.27 |
| 5 | 1.98 | 2.01 | 0.02 | 2.83 | 0.79 |
| 6 | 2.03 | 2.04 | 0.05 | 2.83 | 1.77 |
| 7 | 2.06 | 2.08 | 0.10 | 2.83 | 3.55 |
| 8 | 2.08 | 2.14 | 0.16 | 2.83 | 5.72 |
| 9 | 1.87 | 1.83 | 0.21 | 2.83 | 7.59 |
| 10 | 1.92 | 1.97 | 0.09 | 2.83 | 3.03 |
| Rata – rata *error* |  | | | | 4.34 |

Pada tabel 4.18 disajikan data posisi akhir robot ketika bergerak menuju posisi tujuannya. Dalam tabel tersebut terdapat 10 kali data pengujian untuk melihat keteptan robot menju titik akhir (2,2). Pada tabel pengujian ditampilkan *error* pencapain robot menuju titik akhir. Dari tabel 4.18 dapat dilihat bahwa robot memiliki rata – rata *error* untuk mencapai titik tujuan (2,2)m sebesar 4,34 %.

Pada gambar 4.19 ditampilkan grafik persebaran titik yang dicapai robot dan perbandinganya dengan titik referensi (2,2)m.

**Gambar 4.19.** Akurasirobot mencapai titik akhir (2,2)m.

**Tabel 4.19.** Akurasi pencapaian tujuan (4,4)m dari tiik awal (0,0)m.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Percobaan ke- | Posisi robot sumbux (meter) | Posisi robot sumbuy (meter) | Jarak ke posisi target (m) | *Error* (%) |
| 1 | 3.97 | 4.12 | 0.12 | 2.19 |
| 2 | 4.19 | 3.91 | 0.21 | 3.71 |
| 3 | 3.93 | 4.17 | 0.18 | 3.25 |
| 4 | 4.19 | 3.91 | 0.21 | 3.71 |
| 5 | 3.99 | 3.7 | 0.30 | 5.30 |
| 6 | 3.95 | 4.14 | 0.15 | 2.63 |
| 7 | 4.14 | 3.89 | 0.18 | 3.15 |
| 8 | 3.95 | 4.11 | 0.12 | 2.13 |
| 9 | 4.14 | 3.87 | 0.19 | 3.38 |
| 10 | 4.12 | 3.94 | 0.13 | 2.37 |
| Rata- rata *error* (%) |  | | | 3.30 |

**Gambar 4.20.** Akurasirobot mencapai titik akhir (4,4)m.

Pada tabel 4.19 disajikan data posisi akhir robot ketika bergerak menuju posisi tujuannya. Dalam tabel tersebut terdapat 10 kali data pengujian untuk melihat keteptan robot menuju titik akhir (4,4). Pada tabel pengujian ditampilkan *error* pencapain robot menuju titik akhir. Dari tabel 4.19 dapat dilihat bahwa robot memiliki rata – rata *error* untuk mencapai titik tujuan (2,2)m sebesar 3,30 %.

Pada gambar 4.20 ditampilkan grafik persebaran titik yang dicapai robot dan perbandinganya dengan titik referensi (4,4)m.

# PENUTUP

Bab ini membahas tentang kesimpulan sementara dari hasil pengujian alat dan analisis dari proses dan pengujian kerja sistem *monitoring* seperti yang telah dijelaskan pada bab-bab sebelumnya. Selain itu, penulis juga menyampaikan saran-sarannya kepada para pembaca yang ingin mengembangkan tugas akhir ini untuk kedepannya sehingga nantinya akan dapat menghasilkan alat yang mampu berguna bagi lingkungan dan masyarakat luas.

## **Kesimpulan**

Dari hasil perancangan dan pembuatan sistem yang kemudian dilanjutkan dengan pengujian dan analisis maka dapat diambil kesimpulan sebagai berikut:

* Dalam dunia navigasi, posisi dan orientasi suatu suatu objek sangat berpengaruh teradap penentuan perencanaan jalur (*path* *planning*).
* *Wireless sensor network* yang mampu digunakan untuk mengukur jarak dapat dimanfaatkan untuk menentukan posisi suatu objek pada suatu bidang dengan menerapkan metode trilaterasi. Dengan menggunakan modul DWM1000 penentuan posisi objek (robot) dengan *error* rata – rata sebesar 5,05%.
* Sensor *magnetometer* HMC5883 dapat digunakan sebagai sensor orientasi yang mengacu pada arah magnetik bumi dengan *error* penunjukan *heading* rata – rata sebesar 3°.
* Dengan mengguanakan *positioning* dengan *wireless sensor network* sebagai sensor absolut dapat menghindari gangguan yang terjadi pada sensor relativ seperti selip pada *rotary encoder*. Pada pengujian odometri rata – rata *error* yang dihasilkan sebesar 29,2%, dibandingkan dengan pengujian dengan *indoor positioning system* denganrata – rata *error* sebesar 4.34%.
* Dengan menerapkan *indoor positioning system*  pada *path* *planning* *mobile robot* didapat ketepatan dengan *error* rata – rata sebesar 3,3% pada pergerakan dari titik (0,0)m ke titik (4,4)m.

## **Saran**

Berdasarkan hasil proyek akhir ini, masih terdapat beberapa kekurangan dan dimungkinkan untuk pengembangan lebih lanjut. Oleh karenanya penulis merasa perlu untuk memberi saran-saran sebagai berikut :

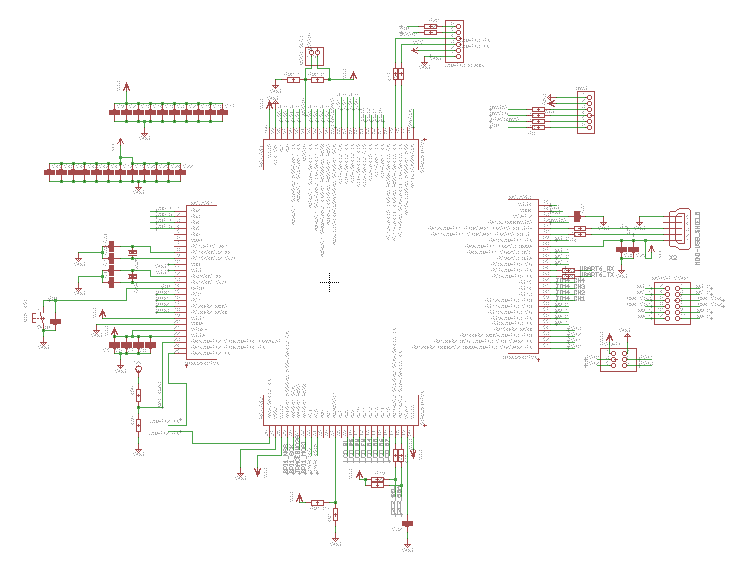
* Perlu dibuatkan GUI untuk dapat melakukan *monitoring* dan akuisisi data pergerakan sensor - sensor pada robot robot juga dapat mengirim *command* kepada robot.
* Perlu dilakukan riset lebih mendalam untuk mengakses fungsi – fungsi modul DWM100 yang sementara pada proyek akhir ini masih diakses menggunakan Arduino, agar modul dapat digunakan lebih opimal menggunakan mikrokontroler lain.
* Apabila menggunakan roda *bidirectional omniwheel* disarankan untuk tidak menggunakan roda dengan bantalan yang terbuat dari plastik karena lebih mudah mengalami selip.

# DAFTAR PUSTAKA

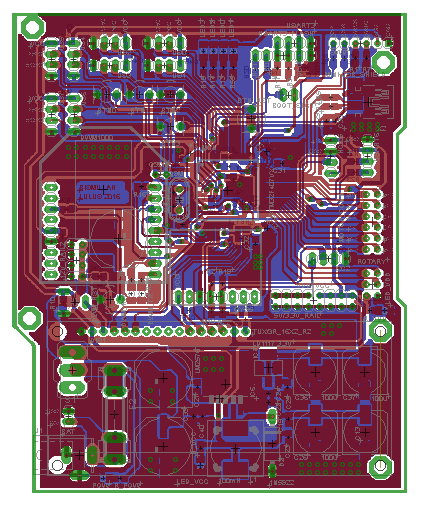
1. Jonathan P. How, Brett Bethke, Adrian Frank, Daniel Dale, And John Vian, *“*Real-*Time* Indoor Autonomous Vehicle Test Environment*”*, IEEE Kontrol Systems Magazine, APRIL 2008.
2. Torvald Ersson and Xiaoming Hu, “Path Planning And Navigation Of Mobile Robots In Unknown Environments”, Optimization and Systems TheoryCentre for Autonomous Systems, Royal Institute of Technology, SE 100 44 Stockholm Sweden.
3. Jong Bae Kim, and Hee Sung Jun, “Vision-Based Location Positioning Using Augmented Reality for Indoor Navigation”, IEEE Transactions on Consumer Electronics, Vol. 54, No. 3, AUGUST 2008.
4. Jusuf Dwi Kariyanto, Ali Husein Alasiry, Fernando Ardila, Nofria Hanafi, “Navigasi *Mobile Robot* Berbasis Trajektori dan Odometri Dengan Pemulihan Jalur Secara Otomatis”, Tugas akhir: Prodi Teknik Elektronika, Politeknik Elektronika Negeri Surabaya Kampus PENS-ITS Sukolilo, Surabaya.
5. Bayu Sandi Marta, Fernando Ardilla, A.R. Anom Besari. “*Path* *Tracking* Pada *Mobile Robot* dengan Umpan Balik Odometri”, Tugas akhir: Prodi Teknik Komputer, Politeknik Elektronika Negeri Surabaya Kampus PENS-ITS Sukolilo, Surabaya.
6. Anggarica Mudji Pratama, Taufiqurrahman, Prima Kristalina, “*Real* *Time Visualization System* untuk *Monitoring* Posisi *Node* Pada Jaringan Sensor Nirkabel Berbasis Komunikasi Zigbee”, Prodi Teknik Telekomunikasi, Politeknik Elektronika Negeri Surabaya Kampus PENS-ITS Sukolilo, Surabaya.
7. Lailul Rahmaniah, Yuliana Setiowati, Jauari Akhmad Nur Hasim, “Pemanfaatan Cubeacon Sebagai Media *Geotagging* Berbasis Android”, Prodi Teknik Informatika, Politeknik Elektronika Negeri Surabaya Kampus PENS-ITS Sukolilo, Surabaya.
8. Eduardo Navarro, Benjamin Peuker, Michael Quan, “Wi-Fi Localization Using RSSI Fingerprinting”, Computer Engineering, \*Child Development, California Polytechnic State University, San Luis Obispo, CA 93405; USA.
9. Agustí Corbacho Salas, Josep Paradells Aspas, “Indoor positioning system based on Bluetooth Low Energy”, Escola Tècnica d'Enginyeria de Telecomunicació de Barcelona Universitat Politècnica de Catalunya, Barcelona, June 2014.
10. *http://www.amazon.com/SainSmart-Modules-Accelerometer-Gyroscope-HMC5883L/dp/B00KQDIU3U (diakses pada 2 Januari 2016 pukul 10:00 WIB ).*
11. *Datasheet* HMC5883.
12. *Datasheet* DWM1000.
13. *Datasheet* STM32F407VG.
14. *http://paeae.com/components/lcd/lcd-16x2-black-on-green.html* (diakses pada 2 Desember 2014 pukul 10:00 WIB ).
15. *http://elektronika-dasar.web.id/teori-motor-dc-dan-jenis-jenis-motor-dc/* (diakses pada 5 Januari 2016 pukul 10:00 WIB ).
16. *https://github.com/thotro/arduino-dw1000* (diakses pada 5 Januari 2016 pukul 13:00 WIB )*.*
17. *http://maruzar.blogspot.co.id/2012/11/menghitung-sudut-heading-atau-bearing.html* (diakses pada 3 Juli 2016 pukul 23:00 WIB ).
18. *https://id.wikipedia.org/wiki/Kompas* (diakses pada 4 Juli 2016 pukul 01:00 WIB )*.*
19. *http://old.myhdl.org/* (diakses pada 6 Juli 2016 pukul 19:00 WIB ).
20. *https://purnomosejati.wordpress.com/2011/08/25/mengenal-komunikasi-i2cinter-integrated-circuit/* (diakses pada 6 Juli 2016 pukul 19:00 WIB )*.*

# LAMPIRAN

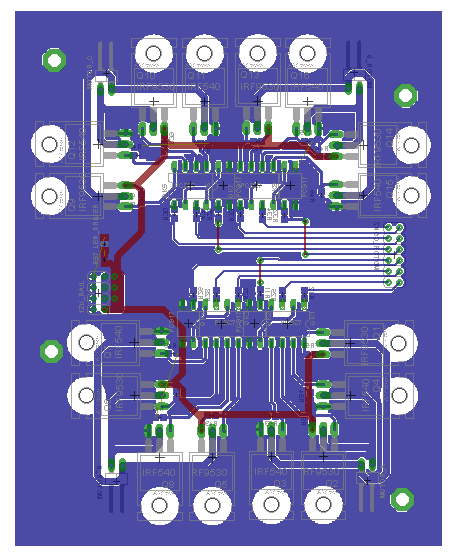
## A1. Gambar skematik Mikrokontroler Utama



## A2. Gambar *Board* Mikrokontroler Utama



## A3. Gambar *Board* *Driver* Motor



# BIODATA PENULIS



Nama Lengkap : Luqman Hakim

Tempat, Tanggal Lahir : Jombang, 2 Agustus 1993

Alamat : Desa Gambiran RT/RW 01/02

Kec. Mojoagung Kab. Jombang

No Telepon : 083831076129

Email : [luqman.hakim.eepis@gmail.com](mailto:luqman.hakim.eepis@gmail.com) /

[embohunited@gmail.com](mailto:embohunited@gmail.com)

Hobi : Futsal, Membaca, Bermain game

Moto : Tuhan, aku berguru kepadaMu

Riwayat Pendidikan :

* SDN Gambiran 01 Tahun 2000-2006
* SMPN 1 Mojoaung Tahun 2006-2009
* SMAN Mojoagung Tahun 2009-2012
* PENS Tahun 2012-2016

****

****