TUGAS AKHIR – EE 184801

DYNAMIC POSITIONING SYSTEM PADA PROTOTIPE *MARINE SEMI-SUBMERSIBLE PLATFORM* EMPAT PENDORONG DENGAN KENDALI JARAK JAUH MELALUI INTERNET.

Luthfi Halim
NRP 07111540000018

Dosen Pembimbing
Dr. Ir. Hendra Kusuma, M.Eng.Sc.
Ir. Tasripan, MT.

DEPATERMEN TEKNIK ELEKTRO
Fakultas Teknologi Elektro
Institut Teknologi Sepuluh Nopember
Surabaya 2019

TUGAS AKHIR – EE 184801

DYNAMIC POSITIONING SYSTEM PADA PROTOTIPE *MARINE SEMI-SUBMERSIBLE PLATFORM* EMPAT PENDORONG DENGAN KENDALI JARAK JAUH MELALUI INTERNET

Luthfi Halim
NRP 07111540000018

Dosen Pembimbing
Dr. Ir. Hendra Kusuma, M.Eng.Sc.
Ir. Tasripan, MT.

DEPARTEMEN TEKNIK ELEKTRO
Fakultas Teknologi Elektro
Institut Teknologi Sepuluh Nopember
Surabaya 2019

FINAL PROJECT – EE 184801

DYNAMIC POSITIONING SYSTEM ON MARINE SEMI-SUBMERSIBLE PROTOTYPE FOUR THRUSTERED WITH REMOTE CONTROL THROUGH THE INTERNET

Luthfi Halim
NRP 07111540000018

Supervisor
Dr. Ir. Hendra Kusuma, M.Eng.Sc.
Ir. Tasripan, MT.

ELECTRICAL ENGINEERING DEPARTMENT
Faculty of Electrical Technology
Sepuluh Nopember Institute of Technology
Surabaya 2019

PERNYATAAN KEASLIAN TUGAS AKHIR

Dengan ini saya menyatakan bahwa isi sebagian maupun keseluruhan Tugas Akhir saya dengan judul "*Dynamic Positioning System* Pada Prototipe *Marine Semi Submersible Platform* Empat Pendorong Dengan Kendali Jarak Jauh Melalui Internet" adalah benar-benar hasil karya intelektual mandiri, diselesaikan tanpa menggunakan bahan-bahan yang tidak diijinkan dan bukan merupakan karya pihak lain yang saya akui sebagai karya sendiri.

Semua referensi yang dikutip maupun dirujuk telah ditulis secara lengkap pada daftar pustaka. Apabila ternyata pernyataan ini tidak benar, saya bersedia menerima sanksi sesuai peraturan yang berlaku.

Surabaya, 11 Juli 2019

Luthfi Halim
NRP. 07111540000018

.....Halaman ini sengaja dikosongkan.....

DYNAMIC POSITIONING SYSTEM PADA PROTOTIPE *MARINE SEMI-SUBMERSIBLE PLATFORM* EMPAT PENDORONG DENGAN KENDALI JARAK JAUH MELALUI INTERNET

TUGAS AKHIR

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

Menyetujui :

Dosen Pembimbing I	Dosen Pembimbing II
Dr. Ir Hendra Kusuma, M.Eng.Sc.	Ir. Tasripan, MT.
NIP. 196409021989031003	NIP. 196204181990031004

SURABAYA
JULI, 2019

.....Halaman ini sengaja dikosongkan.....

DYNAMIC POSITIONING SYSTEM PADA PROTOTIPE *MARINE SEMI-SUBMERSIBLE PLATFORM* EMPAT PENDORONG DENGAN KENDALI JARAK JAUH MELALUI INTERNET

Nama : Luthfi Halim
Pembimbing : 1. Dr. Ir. Hendra Kusuma, M.Eng.Sc.
2. Ir. Tasripan, MT.

ABSTRAK

Marine Semi-Submersible platform merupakan bangunan lepas pantai diatas permukaan air yang secara umum digunakan untuk penambangan *offshore*, *basecamp* keamanan kapal, dan tempat produksi minyak di lepas pantai. Bangunan ini ditujukan untuk tetap stabil terhadap posisinya semula dengan cara dipancang kebawah laut dengan beberapa jangkar dan struktur penahan. Biaya dan struktur untuk mempertahankan posisi ini sangat besar. Digunakan sistem penggerak pada setiap sudut *platform* berjumlah empat buah sehingga *platform* mampu beradaptasi untuk kembali ke posisi semula dengan meminimalisir pengunaan jangkar dan dapat diposisikan dalam kedalaman laut berapapun.

Dynamic Positioning System (DPS), yaitu Sebuah sistem yang mampu untuk mempertahankan posisi akan digunakan pada prototipe *Marine Semi-submersible platform* yang dikembangkan pada tugas akhir ini. Dimana sistem DPS ini direalisasi menggunakan metoda *Differential* GPS, dimana perubahan posisi (Δx dan Δy) dan arah hadap ($\Delta\theta$) *platform* diketahui melalui GPS dan kompas. Dari perubahan nilai Δx, Δy dan $\Delta\theta$ platform, akan ditentukan reaksi tindakan untuk menyesuaikan kembali posisi. Karena adanya *error* pada nilai GPS yang cukup besar saat penerimaan sinyal informasi koordinat dari satelit, maka untuk mengoreksi *error* tersebut digunakan metoda D-GPS dimana posisi GPS *platform* mendapat koreksi dari pangkalan yang ada di darat atau kapal lain dengan mengirimkan sinyal berisi informasi tentang posisinya.

Sistem D-GPS yang dikembangakan pada penelitian ini, mampu mengurangi *error* GPS pada *platform* sebesar 31,22% dalam 11 percobaan dimana uji coba dilakukan di danau 8 ITS.

Kata kunci: *dynamic positioning system, electrical engineer, internet of things, semi-submersible.*

.....Halaman ini sengaja dikosongkan.....

DYNAMIC POSITIONING SYSTEM ON MARINE SUBMERSIBLE PROTOTYPE FOUR THRUSTERED WITH REMOTE CONTROL THROUGH THE INTERNET

Name : Luthfi Halim
Supervisor : 1. Dr. Ir. Hendra Kusuma, M.Eng.Sc.
2. Ir. Tasripan, MT.

ABSTRACT

Marine Semi-Submersible platform is an offshore building above the water surface that is commonly used for offshore mining, ship security basecamp, and oil production sites offshore. The building is intended to remain stable against its original position by being put under the sea with several anchors and retaining structures. The costs and structure for maintaining this position are very expensive. In solving the problem, a drive system is used in each corner of the building, amounting to four so that the platform is able to adapt to return to its original position so as to minimize the use of anchors and can be positioned in any depth of the sea.

Dynamic Positioning System (DPS), which is a system that is able to maintain the position will be used on the Marine Semi-submersible platform prototype developed in this final project. Where the DPS system is realized using the Differential GPS method, where changes in position (Δx and Δy) and direction (Δθ) of the platform are known through GPS and compass. From the changes in the values Δx, Δy and Δθ platform, the action reaction will be determined to readjust the position. Due to a large GPS value error when receiving a coordinate information signal from a satellite, the D-GPS method is used to correct the error where the GPS platform position is corrected from the base on land or other vessels by sending a signal containing information about its position.

The D-GPS system developed in this study was able to maintain the position of the prototype platform while reducing the error of GPS by 31.22 % based on 11 times of trials, where trials were carried out on lake 8 ITS.

Keyword: *dynamic positioning system, electrical engineer, internet of things, semi-submersible.*

.....Halaman ini sengaja dikosongkan.....

KATA PENGANTAR

Puji dan syukur kepada Allah SWT atas berkat rahmat dan karuniaNya, penulis mampu menyelesaikan laporan Tugas Akhir dengan judul "***DYNAMIC POSITIONING SYSTEM* PADA PROTOTIPE *MARINE SEMI-SUBMERSIBLE PLATFORM* EMPAT PENDORONG DENGAN KENDALI JARAK JAUH MELALUI INTERNET**", sebagai salah satu persyaratan dalam menyelesaikan pendidikan program Strata-Satu di Departemen Teknik Elektro, Fakultas Teknologi Elektro, Institut Teknologi Sepuluh Nopember.

Dalam penyusunan dan penyelesaian laporan Tugas Akhir ini penulis mengucapkan terima kasih yang sebesar – besarnya kepada :

1. Bapak Dr. Ir. Hendra Kusuma, M.Eng.Sc. dan Bapak Ir. Tasripan, MT. selaku dosen pembimbing Tugas Akhir saya atas bimbingan serta arahan kepada penulis.
2. Bapak Dr. Ir. Djoko Purwanto, M.Eng. , Bapak Dr. Ir. Ronny Mardiyanto, ST.,MT. dan Bapak Dr. Astria Nur Irfansyah, ST.,MT. selaku dosen evaluator.
3. Bapak Dr.Eng. Ardyono Priyadi, ST.,M.Eng. selaku Kepala Departemen Teknik Elektro ITS.
4. Yuniar Saptotri dan Amin Sholikah, kedua orangtua penulis, yang senantiasa memberikan semangat, do'a, motivasi serta dukungannya terhadap penulis.
5. Seluruh dosen dan teman-teman mahasiswa bidang studi Elektronika Departemen Teknik Elektro ITS.
6. Khalif Aji Puspito selaku rekan yang senantiasa selalu disisi penulis dalam menyelesaikan penelitian ini.
7. Tim Barunastra ITS, Christopher Reinaldo dan Wildan Ilhami yang rela sebagian alatnya dipinjam untuk menunjang penelitian ini.

Penulis menyadari masih banyak yang dapat dikembangkan pada tugas akhir ini. Penulis menerima setiap masukan dan kritik yang diberikan. Semoga penelitian ini dapat memberi banyak manfaat.

Surabaya, 11 Juli 2019

Luthfi Halim

.....Halaman ini sengaja dikosongkan.....

DAFTAR ISI

DAFTAR GAMBAR

.....Halaman ini sengaja dikosongkan.....

DAFTAR TABEL

.....Halaman ini sengaja dikosongkan.....

BAB I
PENDAHULUAN

Pada bab ini membahas mengenai hal – hal yang mendahului pelaksanaan penelitian tugas akhir ini. Hal tersebut meliputi latar belakang, perumusan masalah, tujuan penulisan, batasan masalah, metodologi penelitian, sistematika penulisan dan relevansi.

1.1. Latar Belakang

Dalam proses penambangan minyak dan gas di lepas pantai umumnya menggunakan *marine semi-submersible platform*. *Platform* tersebut harus tetap *stationer* atau terus berada pada posisi yang sama agar proses penambangan dapat terus berlangsung. Pergeseran sedikit saja pada pipa penyalur dapat mengakibatkan kebocoran dan kerusakan pada sistem penambangan. Sehingga dibutuhkan suatu sistem yang mampu mempertahankan posisi secara terus menerus. Pada kedalaman 300 m umum digunakan jangkar dan struktur penahan untuk tujuan mempertahankan posisi *platform.* Pada kedalaman lebih dari 1000 m atau lebih dikenal dengan istilah *"Deep water exploration"* jangkar dan struktur penahan membutuhkan biaya dan perawatan yang mahal. Untuk itu dibuat sistem yang mampu mempertahankan posisi. *Dynamic Position System* pada *Marine Semi-Submersible Platform* yang dibangun pada area laut mampu bertahan pada kedalaman laut lebih dari 1000 m. *Dynamic Position System* merupakan sistem yang mampu menyesuaikan diri untuk kembali ke posisi semula dengan mengatur kecepatan dari masing masing pendorong[2].

Dalam perkembangan teknologi informasi yang sangat cepat, kebutuhan akan control, kalibrasi dan pemantauan pada *platform* sangat dibutuhkan. Sehingga pemanfaatan Internet pada *Platform* akan mendukung pemantauan tersebut secara *real-time* kapan saja dan dimana saja.

Dr. Ir. Dwi Soetjipto mengatakan pada seminar kuliah tamunya yang bertemakan "Transformasi Hulu MIGAS dalam Rangka Pengembangan Kedaulatan Energi dan Energi Berkeadilan di Indonesia" bahwa "dalam beberapa tahun kedepan Indoenesia akan mengalami golden era pada dunia migas" (Dwi Soetjipto, April, 2019). Bergerak dari motivasi tersebut penulis berinisiatif untuk melakukan penelitian yang mengembangkan dunia maritim Indonesia.

1.2. Perumusan Masalah

Permasalahan yang dibahas dalam tugas akhir ini adalah:

1. Bagaimana sistem elektronik pada *Dynamic Positioning System* sehingga *platform* mampu mempertahankan posisinya secara dinamis,
2. Bagaimana model pergerakan *platform* dengan empat pendorong agar mampu mempertahankan kestabilan,
3. Bagaimana memanfaatkan Internet untuk mempermudah pemantauan dan kendali pada *platform.*

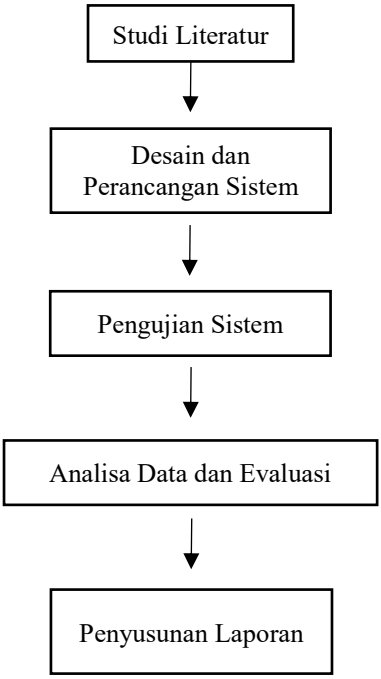
1.3. Tujuan Penelitian

1. Merancang dan membuat prototipe *Marine Semi-Submersible Platform* yang mampu mempertahankan posisinya,
2. Menerapkan *Differential* GPS pada *platform* agar mendapat nilai posisi *platform* secara akurat.

1.4. Batasan Masalah

1. *Dynamic Positioning System* dengan hanya terbatas pada tiga sudut kebebasan yaitu *Surge*, *Sway*, dan *Yaw,*
2. Jumlah penggerak berjumlah empat,
3. Internet sebagai jalan penghubung untuk pemantauan dan kendali digunakan untuk *Remote Desktop* melalui Team Viewer.

1.5. Metodologi Penelitian



Gambar 1.1 Diagram alir metodologi penelitian

1.5.1. Studi literatur

Studi Literatur merupakan tahap awal yang dilakukan untuk mengumpulkan dan mengkaji teori, data dan penelitian yang telah ada sebelumnya yang dianggap relevan dan mampu menunjang keberhasilan tugas akhir. Studi Literatur dapat bersumber dari paper, jurnal, buku, dan artikel tertentu baik skala nasional maupun internasional yang diterbitkan oleh Institusi Akademik terpercaya.

1.5.2. Desain dan Perancangan Sistem

Pada tahap ini semua kebutuhan elektronik berupa sensor, prosessor, kontroler, pendorong, *interface* dan seluruh jalur

kelistrikan dan jalur komunikasi didesain dan direncanakan dengan baik agar didapatkan sistem yang stabil.

Setelah desain sistem elektronik selesai dirancang maka semua peralatan elektronik akan dirangkai dan diuji apakah terdapat kesalahan pada jalur distribusi listrik dan jalur komunikasi pada *platform*.

1.5.4. Pengujian Sistem

Pada tahap ini akan dilakukan pengujian terhadap sistem yang telah selesai dirangkai. *Platform* akan diuji di darat apakah semua sistem telah berjalan sesuai prosedur yang telah dirancang atau tidak. Dan diuji apakah hasil rancangan telah memenuhi kebutuhan dari tujuan yang telah dimaksud di awal. Jika terdapat kesalahan maka proses akan kembali kepada perancangan sistem elektronik.

1.5.6. Analisa Data dan Evaluasi

Setiap Kali selesai pengujian *Platform* di atas air telah dilaksanakan, akan dilakukan analisa dan evaluasi terhadap hasil yang didapat. Terdapat optimalisasi disetiap percobaan yang kemudian *platform* akan kembali pada uji sistem dan uji air. Hasil analisa dan evaluasi akan dicatat untuk kemudian melakukan penyusunan laporan.

1.5.7. Penyusunan Laporan Tugas Akhir

Setelah berulang kali platform melakukan uji coba dan evaluasi akan didapat kesimpulan dan hasil akhir percobaan. Hasil akan dirangkum hasil percobaan dan analisanya untuk disusun dalam bentuk laporan dan dibukukan.

1.6. Sistematika Penulisan

Dalam buku tugas akhir ini, pembahasan mengenai sistem yang dibuat terbagi menjadi lima bab dengan sistematika sebagai berikut:

- **BAB I : Pendahuluan**
  Bab ini meliputi penjelasan latar belakang, rumusan masalah, batasan masalah, tujuan, metodologi, sistematika penulisan, dan relevansi.
- **BAB II : Tinjauan Pustaka**

Pada bab ini berisi mengenai teori yang mendasari penyusunan laporan tugas akhir secara umum khususnya yang berhubungan komponen yang akan digunakan.

- **BAB III : Perancangan Sistem**
  Bab ini menjelaskan tentang perencanaan sistem yang meliputi perangkat keras (*hardware*) dan perangkat lunak (*software*) untuk pembuatan alat ini.
- **BAB IV : Pengujian dan Analisis**
  Pada bab ini menguraikan tentang pengujian alat pada platform ujicoba dan analisa hasil pengujian.
- **BAB V : Penutup**
  Bab ini berisi tentang kesimpulan yang diperoleh dari pembuatan alat serta saran untuk pengembangan lebih lanjut.

1.7. Relevansi dan Manfaat

Pada penelitian ini diharapkan mampu menerapkan secara optimal *dynamic positioning system* pada *marine semi-submersible platform* sehingga mampu bertahan pada posisi tujuan.

.....Halaman ini sengaja dikosongkan.....

BAB II
TINJAUAN PUSTAKA

Pada bab ini diberikan teori dasar yang menjadi landasan untuk merumuskan dan menyelesaikan masalah yang akan dibahas pada penelitian ini. Pada bagian ini terdapat tinjauan pustaka tentang komponen yang akan digunakan untuk membuat alat pada penelitian ini.

2.1. *Marine Semi-Submersible Platform*

Gambar 2.1 *Marine Semi-Submersible Crane Vessel* **[2]**.

Marine semi-submersible platform adalah alat khusus yang digunakan dalam sejumlah peran lepas pantai tertentu seperti sebagai *rig* pengeboran, kapal penyelamat, *platform* produksi minyak serta sebagai derek untuk mengangkat muatan berat atau *platform* lainnya[1].

Marine semi-submersible platform memperoleh sebagian besar daya apungnya dari sistem *ballast* yang terdapat dalam *ponton*. *Ponton* kedap air tersebut terletak dibagian bawah *platform* yang bisa tenggelam dan juga bisa terapung. Itulah mengapa *platform* ini disebut *semi-submersible*. Dengan sistem *ponton* yang dapat tenggelam ini membuat *platform* memiliki stabilitas yang tinggi sehingga *deck* operasi pada *platform* bisa diletakkan lebih tinggi[1].

Dengan kondisi rancangan lambung *semi-submersible* yang terendam, membuat jenis *platform* ini lebih sedikit menerima dampak beban gelombang dibandingkan dengan jenis kapal lain. Namun dengan luasan permukaan basah yang cukup kecil membuat *platform semi-submersible* sensitif terhadap perubahanbeban yang terdapat di *platform*. Sehingga harus lebih berhati-hati dalam pemindahan beban yang terdapat pada *platform* untuk tetap menjaga kestabilan[1].

Marine semi-submersible platform dapat beroperasi di laut dalam ataupun dangkal dengan memanfaatkan sistem *ballasting*-nya yang dapat merubah sarat *platform* menjadi lebih dalam atau lebih rendah[1]. Adapun beberapa jenis *marine semi-submersible platform* sesuai fungsinya adalah sebagai berikut:

2.1.1. *Mobile Offshore Drilling Units* (MODU)

MODU merupakan *platform* yang dibuat dengan stabilitas tinggi untuk kepentingan pengeboran minyak dan gas di area lepas pantai. *Platform* ini dapat ditarik ke posisi pengeboran dengan menggunakan *tugboat* yang menariknya ataupun bisa juga dengan menggunakan *azimuth thruster* yang terpasang di *platform* itu sendiri[1].

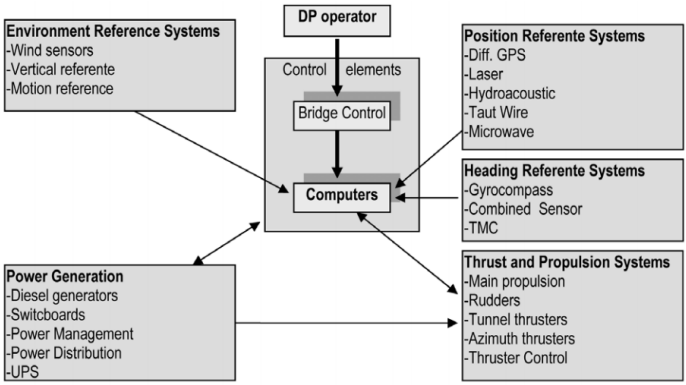
2.1.2. *Semi submersible Crane Vessel* (SSCV)

SSCV merupakan *semi-submersible platform* yang dilengkapi dengan dua *crane* besar yang digunakan sebagai *vessel* untuk pembangunan bangunan lepas pantai. Dengan sistem *submersible* dari *platform* inilah yang dimanfaatkan untuk tetap bisa menjaga stabilitas *platform* meskipun sedang mengangkat beban yang cukup berat[1].

2.1.3. *Offshore Support Vessel*

Merupakan *semi-submersible* yang digunakan sebagai *support vessel* dari bangunan lepas pantai lainnya karena memiliki stabilitas yang baik serta *deck* yang luas[1].

2.2. *Dynamic Positioning System (*DPS)

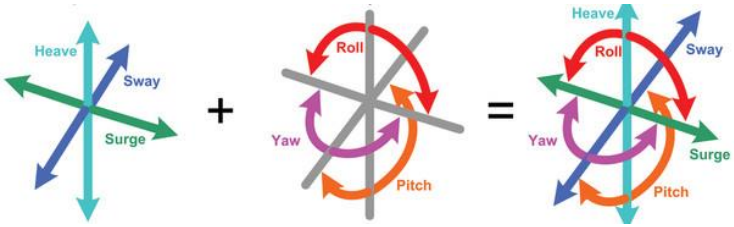


Gambar 2.2 Diagram Skematik *Dynamic Positioning System* secara umum **[3]**.

Pada Gambar 2.2 dijelaskan komponen yang dibutuhkan dalam menyusun *dynamic positioning system.* Keseluruhan sistem yang dibutuhkan antara lain, operator DP, sistem referensi lingkungan, sistem tenaga, sistem propulsi, arah hadap, referensi posisi dan elemen kontrol.

Dynamic Positioning System (DPS) pada kapal atau bangunan lepas pantai merupakan prosedur atau sistem yang dibuat agar *platform* mampu mempertahankan posisi dan arah hadapnya dengan menggunakan *Propeller* dan Rotornya sendiri. Dimana DPS memungkinkan untuk melakukan penambangan minyak dan segala operasi di laut dalam yang tidak mampu dilakukan dengan hanya menggunakan tambat atau jangkar. Definisi DPS juga mencakup bertahan pada suatu lokasi tetap, melakukan *maneuver* secara presisi, meraba perubahan lingkungan, dan kemampuan untuk mengatur posisi[3].

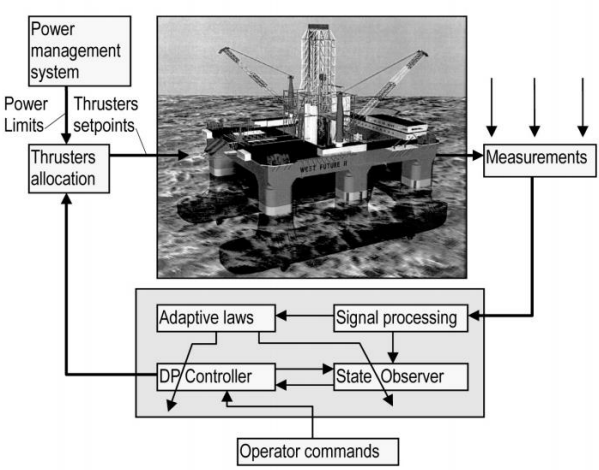
Semua *Vessels* (kapal dan bangunan lepas pantai) dapat bergerak dalam enam sudut kebebasan (*six degree of freedom*) yaitu *yaw*, *pitch, roll*, *surge*, *sway*, dan *heave*. Namun demikian, DPS hanya mengontrol secara otomatis perubahan dari *surge*, *sway*, dan *yaw*. Untuk dapat bekerja setidaknya DPS harus memiliki beberapa komponen berikut yaitu komputer, modul kontrol, referensi posisi, referensi arah hadap, referensi kondisi lingkungan, sistem tenaga, dan sistem propulsi[3].



Gambar 2.3 Enam Sudut Kebebasan (*six degree of freedom)* **[4]**.

Gambar 2.3 menunjukan enam sudut kebebasan yang mempengaruhi plaform dan dapat terjadi pada floating platform. Enam sudut kebebasan tersebut dibagi kedalam dua himpunan gerakan, yaitu tiga gerakan translasi (heave, sway, dan surge) dan tiga gerakan rotasi (pitch, roll dan yaw). Dari enam sudut kebebasan hanya tiga yang mampu dipertahankan oleh Dynamic Positioning System, yaitu Surge, Sway dan Yaw. Karena Surge dan Sway berkaitan dengan pergerakan posisi platform sedangkan Yaw berkaitan dengan perubahan arah hadap platform[4]. Penjelasan mengenai pengertian masing masing dari enam sudut kebebasan:

1. Surge : pergerakan maju atau mundur
2. Sway : pergerakan kanan atau kiri
3. Heave : pergerakan naik atau turun
4. Pitch : miring ke kiri atau ke kanan
5. Roll : miring ke depan atau ke belakang
6. Yaw : Berputar ke kiri atau ke kanan



Gambar 2.4 Susunan Secara Umum *Dynamic Positioning System* **[3]**

2.2.1. Fungsi *Platform* dengan DPS :

Penambangan, pengeboran eksplorasi (pengambilan sampel inti), pengeboran produksi, dukungan penyelam, pipelay (pipa kaku dan fleksibel), pemasangan kabel dan perbaikan, multi-peran, akomodasi atau layanan "flotel", survei hidrografi, survei sebelum atau sesudah operasi, survei kecelakaan, penyelamatan dan pemindahan, pengerukan, *rockdumping* (perlindungan pipa), instalasi bawah laut, pengangkatan (*topside* dan *subsea*), stimulasi dan *workover* yang baik, pasokan *platform*,*offtake tanker* ulang-alik, produksi terapung (dengan atau tanpa penyimpanan), kargo angkat berattransportasi, kapal pesiar penumpang, penanggulangan ranjau, penelitian oseanografi,penambangan dasar laut, penentuan posisi *platform* peluncuran roket, dukungan perbaikan / pemeliharaan untukkapal militer, transfer kapal ke kapal dan manuver kapal konvensional[3].

2.2.2. Keuntungan *Platform* dengan DPS :

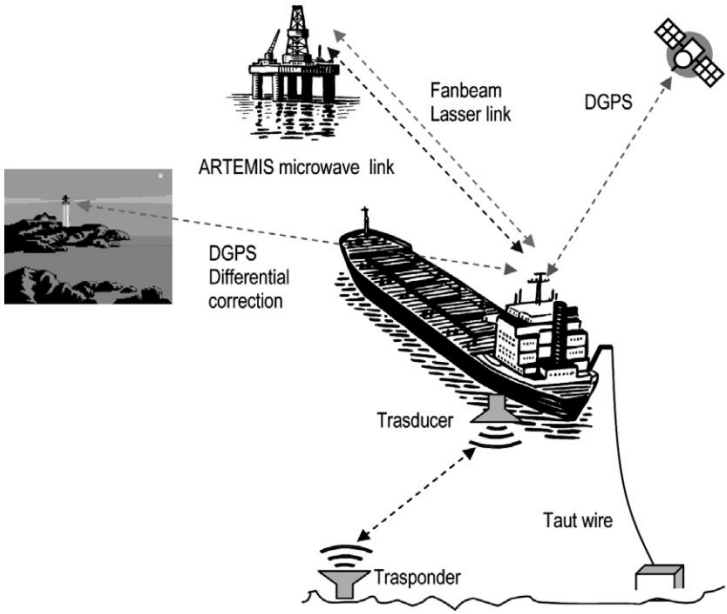
Platform sepenuhnya menggunakan pendorong sendiri, pengaturan di lokasi cepat dan mudah, kapal sangat bermanuver, respons cepat terhadap perubahan cuaca dimungkinkan, respons cepat terhadap perubahan dalam persyaratan operasi, fleksibilitas dalam sistem, kemampuan untuk bekerja di kedalaman air apa pun,

dapat menyelesaikan tugas-tugas pendek lebih cepat, sehingga lebih ekonomis, menghindari risiko kerusakan perangkat keras dasar laut dari jalur tambat danjangkar, menghindari tambatan lintas dengan kapal lain atau platform tetap dan bisapindah ke lokasi baru dengan cepat[3].

2.2.3. Kerugian Utama *Platform* dengan DPS :

Modal dan biaya operasi tinggi, terdapat kemungkinan gagal mempertahankan posisi karena kegagalan peralatan, lebih tinggi tarif hari dibandingkan sistem moor yang sebanding, konsumsi bahan bakar yang lebih tinggi, pendorong bahaya bagi penyelam, dapat kehilangan posisi dalam cuaca ekstrem atau di perairan dangkal dangelombang pasang yang kuat, kontrol posisi aktif dan bergantung pada operator manusia (dan juga peralatan) dan membutuhkan lebih banyak personel untuk mengoperasikan dan memelihara peralatan. Sistem kontrol DP pertama tidak beradaptasi dengan kondisi laut aktual dan kesalahan kapal dan pendorong, itulah alasan mengapa diperlukan perbaikan kontrol dalam akurasi pemeliharaan stasiun yang menyediakan prosedur algoritmik implementasi terkini untuk meningkatkan kinerja manuver, dan data digital yang cepat transmisi[3].

Dalam *Dynamic Posistioning System* sebuah referensi posisi dan referensi arah hadap dibutuhkan agar *platform* mampu menentukan tindakan yang akan dilakukan. Referensi arah hadap didapat dengan menambahkan *gyrocompass* pada sistem. Sementara referensi posisi pada *Dynamic Positioning System* didapat dengan lima metode yaitu *Hydrocaustic Position Reference*, *Taut Wire Posistion reference*, *Differential Global Positioning Sytem (DGPS), Laser-Base Position Reference*, dan Artemis seperti terdapat pada Gambar 2.5[3].



Gambar 2.5 Sistem Referensi Posisi **[3]**.

Pada tugas akhir ini metode referensi posisi yang digunakan adalah *Differential Global Positioning System Reference.*

2.3 *Differential Global Positioning System*

Global Positioning System (GPS) memiliki akurasi khas yang tersedia dari Layanan Pemosisian Standar GPS 20m, tetapi akurasi GPS tidak memadai untuk tujuan *Dynamic Positioning*. Untuk meningkatkan akurasi GPS, koreksi diferensial diterapkan pada data GPS. Hal ini dilakukan dengan membangun stasiun referensi di titik-titik yang diketahui pada spheroid WGS 84. Rentang semu yang diperoleh oleh penerima dibandingkan dengan yang dihitung dari lokasi satelit dan stasiun referensi yang diketahui, yang mendapatkan koreksi. Sebagian besar layanan DGPS menerima beberapa input diferensial yang diperoleh dari berbagai stasiun referensi yang dipisahkan secara luas[3].

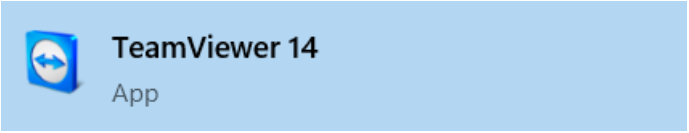
Jaringan Sistem DGPS memberikan stabilitas dan akurasi yang

lebih baik, dan menghilangkan lebih banyak kesalahan ionosfer daripada yang dapat diperoleh dari stasiun referensi tunggal. Keakuratan yang diperoleh dari sistem DGPS adalah di area 1-3m tergantung pada jarak ke stasiun referensi, kondisi ionosfer, dan konstelasi satelit yang tersedia. Beberapa operasi DP memerlukan pemosisian kapal relatif terhadap struktur yang bergerak[3].

2.4. *Remote Desktop Connection* dengan TeamViewer

Remote Desktop adalah teknologi untuk dapat memantau dan mengendalikan suatu perangkat computer yang sudah memiliki sistem operasi dari perangkat computer lain baik melaui local area network maupun dikendalikan jarak jauh melalui internet. Dalam penelitian ini penulis menggunakan aplikasi TeamViewer.

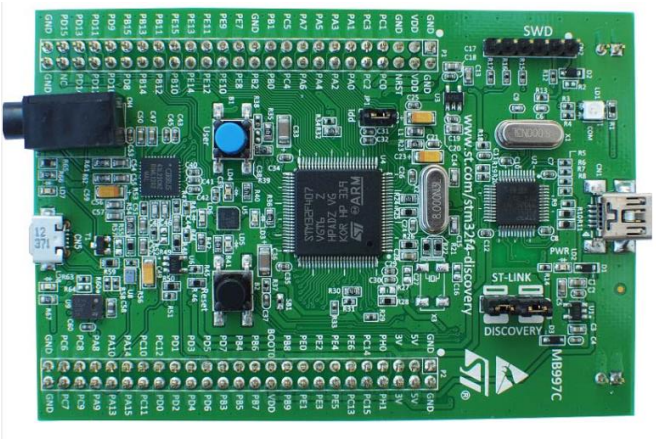
TeamViewer merupakan aplikasi dengan fungsi utama sabagai remote desktop. Penulis dimudahkan dalam troubleshooting platform dan proses kendali darurat pada platform karena lokasi penelitian berada pada daerah berair yaitu danau 8 ITS. Proses pemantauan kondisi platform secara real time juga dapa dilakukan tanpa harus bersentuhan langsung dengan platform yang berada di permukaan air.



Gambar 2.6 Logo Team Viewer.

2.5. Mikrokontroller STM32F4

STM32F4 *discovery* adalah modul dari ST Microelectronics berbasiskan mikrokontroller arsitektur ARM 32bit dengan prosesor STM32F407VGT6 dan termasuk sebuah ST-LINK/V2 sebagai alat *debug* mempunyai banyak f00itur seperti digital *accelerometer*, ST MEMS digital *microphone*, audio DAC yang terintegrasi pengendali speaker kelas D, Led, tombol dan sebuah konektor USB OTG Micro-AB. Untuk *chip* STM32F407VGT6 mempunyai banyak fitur yang dapat digunakan, seperti I/O, Timer, ADC, dan DMA. Kecepatan *clock* nya bisa mencapai 168Mhz, sehingga memungkinkan untuk mengerjakan perintah program yang cukup yang panjang dalam waktu yang cukup singkat[5].



Gambar 2.7 STM32F407VG *Discovery Board* **[5]**.

Alasan penggunaan STM32F4 pada tugas akhir ini dibanding mikrokontroller yang lain adalah karena STM32F4 memiliki timer yang sangat banyak yang memungkinkan mampu menerima input PWM dari sinyal remot sebanyak 8 channel serta mampu mengeluarkan output PWM untuk 4 servo dan 4 motor. STM32F4 memiliki 6 slot serial yang 4 diantaranya sudah digunakan untuk menerima input sensor.

2.6. Modul *Global Positioning System* (GPS) Ublox M8N

Module GPS merupakan module elektronik yang berfungsi sebagai penerima sinyal satelit untuk mengetahui posisi penerima. Seri NEO-M8 memberikan sensitivitas tinggi dan waktu akuisisi minimal dengan tetap mempertahankan daya sistem yang rendah. NEO-M8M dioptimalkan untuk aplikasi yang sensitif biaya, sementara NEO-M8N / M8Q memberikan kinerja terbaik dan integrasi RF yang lebih mudah. Faktor bentuk NEO memungkinkan migrasi mudah dari generasi NEO sebelumnya. Arsitektur RF yang canggih dan penekanan gangguan memastikan kinerja maksimum bahkan di lingkungan yang bermusuhan dengan GNSS. Seri NEO-M8 menggabungkan tingkat ketahanan dan integrasi tingkat tinggi dengan opsi konektivitas fleksibel. NEO-M8N yang tahan di masa depan mencakup Flash internal yang memungkinkan peningkatan firmware sederhana untuk mendukung sistem GNSS

tambahan. Ini membuat NEO-M8 sangat cocok untuk aplikasi industri dan otomotif[6].

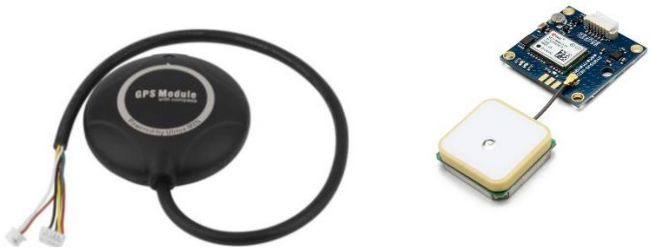
Document Information		
Title	**NEO-M8**	
Subtitle	u-blox M8 concurrent GNSS modules	
Document type	Data Sheet	
Document number	UBX-13003366	
Revision and Date	R10	21-Oct-2015
Document status	Production Information	

Document status explanation	
Objective Specification	Document contains target values. Revised and supplementary data will be published later.
Advance Information	Document contains data based on early testing. Revised and supplementary data will be published later.
Early Production Information	Document contains data from product verification. Revised and supplementary data may be published later.
Production Information	Document contains the final product specification.

This document applies to the following products:

Product name	Type number	ROM/FLASH version	PCN reference
NEO-M8N	NEO-M8N-0-01	ROM 2.01/Flash FW 2.01	N/A
NEO-M8M	NEO-M8M-0-01	ROM 2.01	UBX-15015253
NEO-M8Q	NEO-M8Q-0-01	ROM 2.01	UBX-15015253

Gambar 2.8 Spesifikassi GPS Ublox M8N **[6]**.

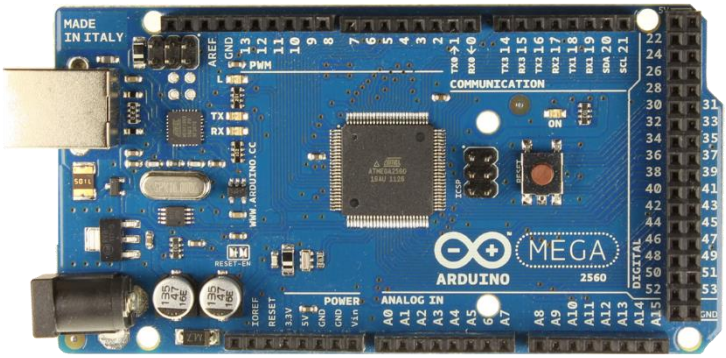


Gambar 2.9 Modul, *antenna,* dan *chip* Ublox M8N **[6]**.

2.7. Mikrokontroler Arduino Mega

Arduino Mega 2560 adalah papan mikrokontroler berdasarkan ATmega 2560. Ini memiliki 54 pin input / output digital (14 di antaranya dapat digunakan sebagai output PWM), 16 analog input, 4 UART (port serial perangkat keras), osilator kristal 16 MHz, koneksi USB, colokan listrik, header ICSP, dan tombol reset. Ini berisi semua yang diperlukan untuk mendukung mikrokontroler; cukup sambungkan ke komputer dengan kabel USB atau nyalakan

dengan adaptor AC-DC atau baterai untuk memulai. Mega kompatibel dengan sebagian besar perisai yang dirancang untuk Arduino Duemilanove atau Diecimila[7].



Gambar 2.10 Arduino Mega 2560 *Board* **[7]**.

2.7. *Graphical User Interface QT Designer*

Graphical User Interface atau yang biasa kita sebut sebagai GUI merupakan bentuk dari suatu alat yang memungkinkan pengguna untuk dapat berinteraksi dengan perangkat keras elektronik melalui sebuah desain grafis sehingga penggunaannya lebih mudah, bervariasi dan mudah dipahami. Dalam perkembangannya, GUI sudah digunakan pada seluruh *platform Internet of Things*. Pada tugas akhir ini penulis mencoba mengantarkan sebuah terobosan dalam bidang kontrol, navigasi dan kemaritiman yang dibumbui oleh teknologi GUI ini. Dengan hadirnya GUI akan mempermudah pengguna dalam fungsi *monitoring* dan *controlling platform* yang diciptakan.

Aplikasi yang mempersilahkan penulis untuk dapat membuat GUI adalah aplikasi Qt Designer. Qt dapat dioperasikan dalam berbagai Bahasa pemrograman salah satunya adalah C++. Qt merupakan *platform* professional yang sudah digunakan oleh industry besar dalam menciptakan karyanya salah satunya adalah LG. Qt sangat bagus digunakan pada sistem tertanam, *IoT platform*, alat medis bahkan otomotif dan otomasi[8].

Gambar 2.11 Logo Qt Designer **[8]**

2.8. F2838 *Motor DC Brushless* Tahan Air

F2838 merupakan jenis motor DC Brushless yang dikhususkan penggunaannya di dalam air. Motor DC jenis ini di gunakan karena platform tugas akhir ini merupakan bangunan lepas pantai yang seluruh dikegiatannya berada di air. Pada motor ini terdapat *seal* atau penutup pada *coil* motor sehingga mampu menghindari dari *short circuit* pada gulungan. Motor ini mendapat penguatan dari baterai yang besar penguatan pada gulungan di dalam motor diatur oleh module yang bernama *Electrical Speed Controller.*

Gambar 2.12 *Waterproof Motor DC Brushless* F2838.

2.9. *Electronic Speed Controller* (ESC)

Electronic Speed Controller (ESC) adalah module yang mampu mengatur kecepatan motor DC. ESC mengikuti sinyal referensi kecepatan (berasal dari tuas throttle, joystick, atau input manual lainnya) dan memvariasikan tingkat switching dari jaringan Field Efferct Transistor. Dengan menyesuaikan Duty Cycle atau frekuensi switching dari transistor, kecepatan motor diubah. Pergantian yang cepat dari transistor adalah apa yang menyebabkan motor itu sendiri mengeluarkan karakteristik rengekan bernada tinggi, terutama terlihat pada kecepatan yang lebih rendah.

Gambar 2.13 *Electrical Speed Cntroller.*

Berbagai jenis kontrol kecepatan diperlukan untuk motor DC brushed dan motor DC brushless. Motor Brushless DC memerlukan prinsip pengoperasian yang berbeda. Kecepatan motor bervariasi dengan menyesuaikan Duty Cycle arus yang dikirim ke beberapa belitan motor.

2.10. Motor *Servo*

Gambar 2.14 Motor *Servo*.

Motor servo adalah sebuah motor DC dengan sistem umpan balik tertutup di mana posisi rotor-nya akan diinformasikan kembali ke rangkaian kontrol yang ada di dalam motor servo. Motor ini terdiri dari sebuah motor DC, serangkaian gear, potensiometer, dan rangkaian kontrol. Potensiometer berfungsi untuk menentukan batas sudut dari putaran servo.

Sedangkan sudut dari sumbu motor servo diatur berdasarkan lebar pulsa yang dikirim melalui kaki sinyal dari kabel motor servo.

Secara putaran terdapat dua jenis motor servo, yaitu motor servo standar dan motor servo continuous. Motor servo standar sering dipakai pada sistem robotika, misalnya untuk membuat Robot Arm (Robot Lengan) sedangkan motor servo continuous sering dipakai untuk Mobile Robot. Pada bagian servo tertulis tipe servo yang bersangkutan. Contoh motor servo bisa dilihat pada Gambar 2.5.

Penggunaan motor servo di bidang robotika tentu ada alasannya. Pertama adalah motor servo memiliki putaran yang lambat dengan torsi yang kuat (berkat adanya sistem roda gigi). Hal ini cocok dengan bidang robotika.

2.11. *Remote Control* AT9

Gambar 2.15 *Remote Control* AT9s **[9]**.

AT9s merupakan alat remote control jarak jauh dengan frekuensi kerja 2.4 GHz. Merupakan pemancar dengan standar 9 saluran kanal yang berpasangan dengan penerima (receiver) R9DS. AT9s dapat diupgrade menjadi 12 kanal dengan cara mengupgrade firmware dan beralih menggunakan penerima R12DS. AT9s adalah perangkat remote control tercanggih yang dikeluarkan oleh RadioLink. Dapat digunakan untuk

mengendalikan berbagai perangkat seperti helicopter, fixed-wing, glider, dan multirotor. Pada AT9s terdapat teknologi pemancara sinyal yaitu Direct Sequence Spread Spectrum (DSSS) dan Frequency Hoping Dpread Spectrum (FHSS) yang mampu bekerja secara sinkron untuk pertama kali dan diproduksi oleh RadioLink. Teknologi FHSS menggunakan 16 saluran pseudo-random frequency hopping. AT9s juga menggunakan modulasi QPSK yang memastikan kinerja anti-interfensi yang sangat baik, bahkan dapat mengontrol stabilitas di pusat kota. AT9s mudah digunakan bagi pemula maupun professional[9].

2.12. Modul Kompas CMPS-11

Gambar 2.16 Modul Kompas CMPS-11 **[10]**.

CMPS11 adalah kompas generasi ke 3 dengan sistem kompensasi kemiringan. Sensor kompas CMPS11 ini dilengkapi dengan magnetometer 3-axis, gyro 3-axix, dan akselerometer 3-axix. Filter Kalman menggabungkan gyro dan accelerometer untuk menghilangkan kesalahan yang disebabkan oleh memiringkan PCB. CMPS11 menghasilkan data sudut 0-3599 yang mewakili 0-359.9 atau 0 hingga 255. Output dari ketiga sensor adalah pengukuran terhadap nilai x, y, dan z terhadap medan magnet yang sekaligus untuk bisa mendapatkan nilai pitcing dan roling. Modul CMPS11 membutuhkan catu daya pada 3,6 - 5v dan menarik arus nominal 25mA. Pilihan antarmuka serial atau I2C disediakan. Sensor kompas ini digunakan sebagai inputan sudut arah hadap dari sebuah platform yang nantinya akan tetap dipertahankan pada kondisi dynamic positioning system[10].

2.14. Modul Nirkabel HC-12

Modul HC-12 adalah modul generasi terbaru yang memiliki multichannel jalur transmisi data. HC-12 bekerja pada pita frekuensi 433,4 sampai 473,0 MHz, dan beberapa *channel* dapat diatur dengan loncatan 400kHz dari total 100 channel. Daya transmisi maksimum untuk modul ini adalah 100 mW (20 dBm), sensitivitas penerimaan adalah -117 dBm pada tingkat 5000 bps di udara. Modul ini mampu menempuh jarak 1000 meter (FU3 mode dengan kecepatan serial 4800 bps) di ruang terbuka, 1800 meter dengam FU4 mode[11].

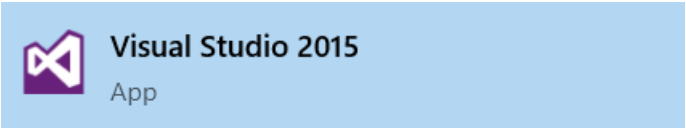
Gambar 2.17 Bentuk Fisik Modul HC-12 **[11]**.

Spesifikasi dari Modul HC-12 adalah sebagai berikut [11]:

1. Bekerja pada sumber tegangan 3,2 V – 5,5 V dengan kapasitas beban tidak kurang dari 200mA.
2. Transmisi jarak jauh (FU3: 1000 meter diruang terbuka, dengan kecepatan 5000 bps dan FU4 :1800 meter diruang terbuka dengan kecepatan 500 bps di udara).
3. Rentang frekuensi kerja antara 433,4-473,0 MHz dengan 100 channel komunikasi.
4. Maksimum daya pengiriman adalah 100 mW (20dBm) dengan 8 tingkat daya dapat diatur.
5. Built-in MCU untuk melakukan komunikasi dengan perangkat eksternal melalui port serial, tidak memerlukan program dan konfigurasi untuk penggunaan dasar.

2.15. Microsoft Visual Studio

Microsotf Visual Studio merupakan *Integrated Development Enviroment* (IDE) dari Microsoft. *Software* ini digunakan untuk mengembangkkan program kompputer, serta situs web, aplikasi web, layanan web, dan aplikasi seluler dalam bentuk aplikasi *console*, aplikasi windows, atau apilasi Web.

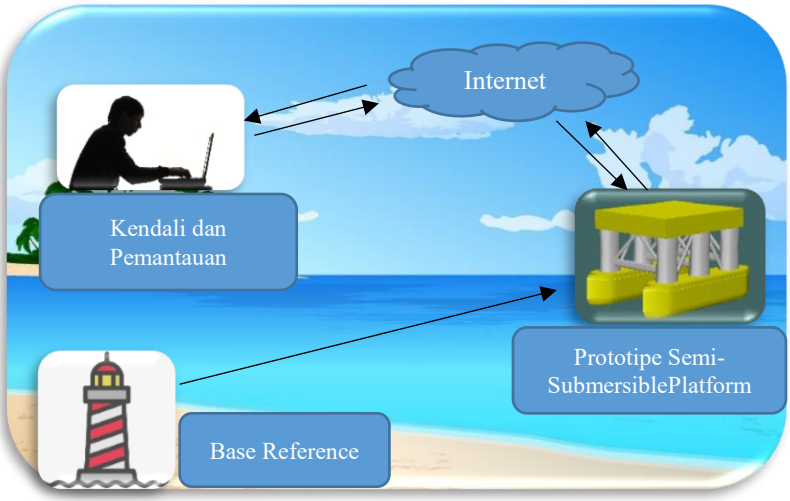


Gambar 2.18 Logo Visual Studio 2015.

Visual Studio merupakan editor kode yang mendukung *IntelliSense* (komponen penyelesaian kode) serta *refactoring.* Alat bawaan lainnya meliputi *code profiler,* perancangan bentuk untuk aplikasi GUI, *web designer, class designer,* dan skema perancangan *database*.

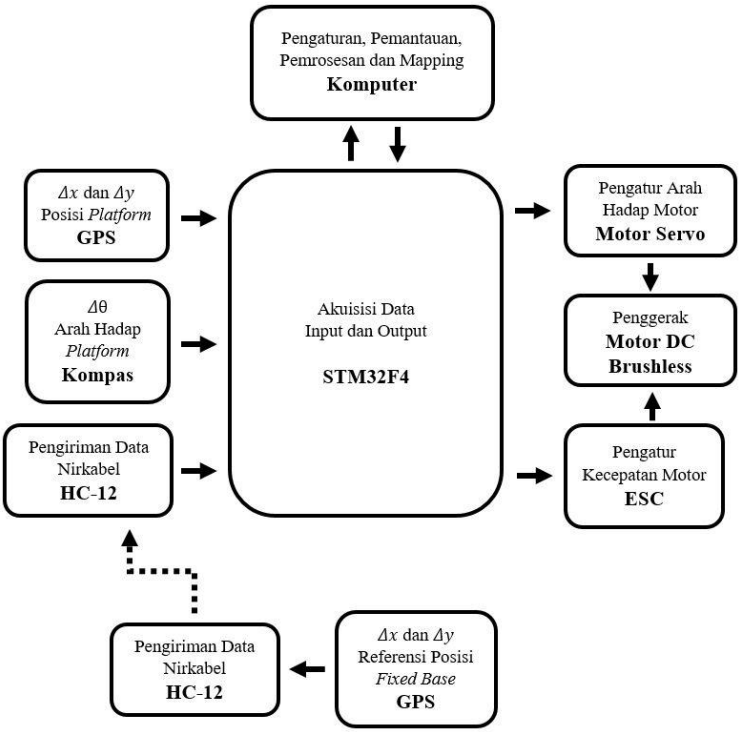
Visual Studio mendukung 36 bahasa pemrograman yang berbeda dan memungkinkan editor kode dan *debugger* untuk mendukung hampir semua bahasa pemrograman. Bahasa bawaan dari s*oftware* ini termasuk *C, C++, C++/CLI, Visual Basic .NET, C#, F#, JavaScript, TypeScript, XML, XSLT, HTML,* dan *CSS.* Bahasa pendukung lainnya seperti *Phyton, Ruby, Node.js,* dan *M* tersedia melalui *plug-in.*

BAB III
PERANCANGAN SISTEM



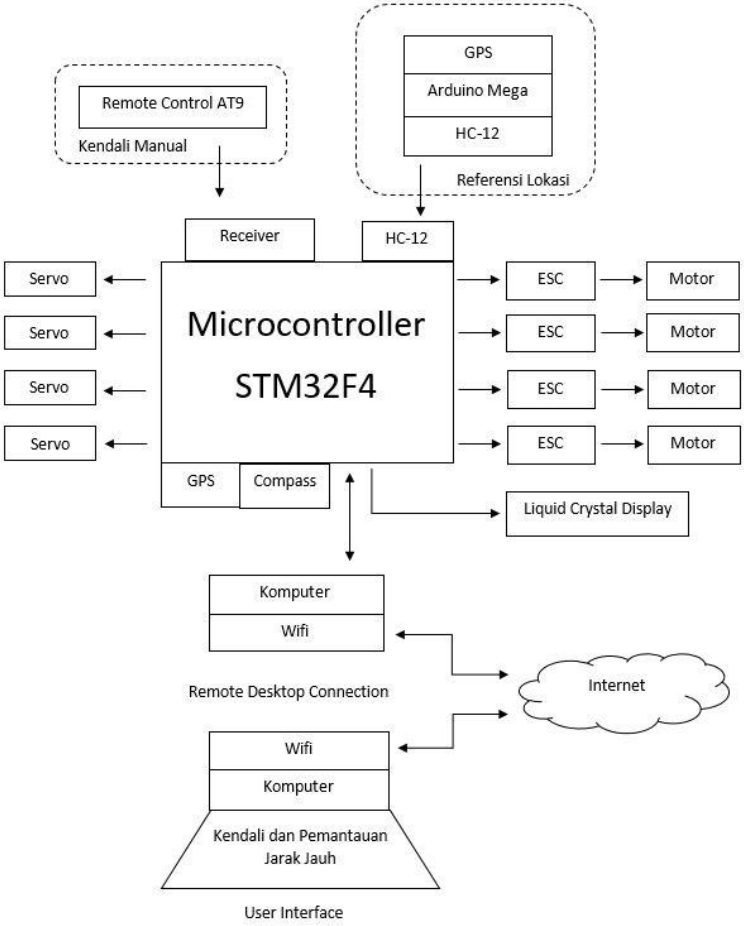
Gambar 3.1 Skema Sistem Keseluruhan.

Pada bab ini dijelaskan perancangan sistem secara keseluruhan. Pengaruh *input* dan proses pengambilan keputusan untuk *output* juga dijelaskan pada bab ini. Pada *platform* terdapat rancang bangun mulai dari sistem mekanik, sistem elektronik, sistem rancang gerak, sistem kontrol, dan sistem komunikasi. Pada keseluruhan sistem juga terdapat *fixed base* sabagai lokasi referensi yang akan memberikan informasi lokasinya kepada *platform* sehingga posisi dari *platform* akan lebih akurat. Keseluruhan sistem yang dibangun bertujuan agar *platform* dapat secara terus menerus mampu mempertahankan posisinya secara terus menerus.



Gambar 3.2 Diagram Blok Input, Output, dan Pemrosesan.

Pada perancangan sistem pada penelitian ini terdapat tiga obyek yang saling bertukar informasi secara nirkabel seperti terlihat pada Gambar 3.1. Ketiga obyek ini memiliki peran masing-masing yang saling mendukung satu sama lain sehingga membangun sistem yang mampu diandalkan. Ketiga obyek ini yaitu Pengguna selaku pemantau dan pengontrol, *fixed base* selaku pemberi referensi posisi kepada *platform* dan *platform* itu sendiri yang dalam tujuannya dimaksudkan untuk mempertahankan posisi. Keseluruhan perjalanan alur informasi antar ketiga obyek serta antar komponen elektronik utama pada platform dijelaskan pada Gambar 3.3.



Gambar 3.3 Rancangan Umum Alur Pengiriman Data dan Informasi Seluruh Sistem.

3.1. Perancangan Prototipe Marine Semi-Submersible Platform

3.1.1. Perancangan Sistem Mekanik Platform

Perancangan sistem mekanik dimulai dengan melakukan pembuataan desain prototipe *platform semi-submersible* dengan menggunakan aplikasi desain Maxsurf. Dimensi dari prototipe *platform* dibuat berdasarkan skala dari ukuran *platform* yang sesungguhnya dan umum digunakan. Tujuan dalam taham pendesainan prototipe selain untuk merancang bentuk prototipe, juga untuk mengetahui nilai-nilai koefisien serta parameter-parameter pada prototipe *platform*. Desain ini didapat dari kerjasama bersama rekan dari jurusan Sistem Perkapalan Institut Teknologi Sepuluh Nopember. Desain, hasil, dan ukuran rancangan *platform* adalah seperti pada Gambar 3.4 dan Tabel 3.1 berikut:

Gambar 3.4 Desain dan Pembuatan Prototipe *Body Platform.*

Dimensi Prototipe Platform Semi-submersible		
Parameter	Ukuran	Satuan
L	60	Cm
B	42	Cm
H	35	Cm
T	10	Cm
B ponton	12	Cm

H ponton	12	Cm
H pilar	17	Cm

Tabel 3.1 Dimensi Prototipe *Marine Semi-Submersible Platform.*

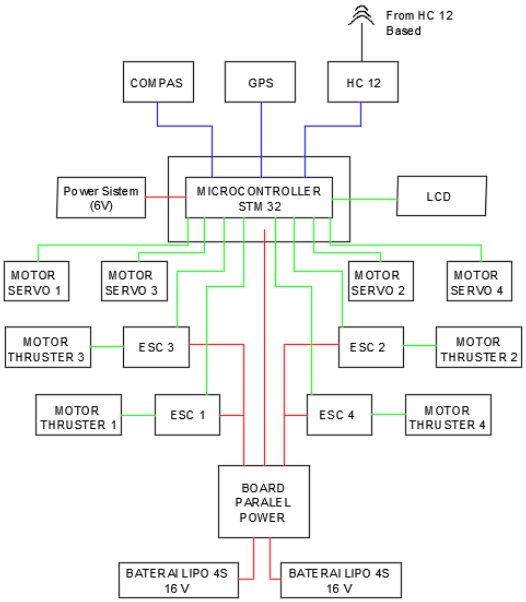
Untuk mendukung *semi-submersible platform* bergerak ke segala arah dengan kemampuan manuver yang baik, *semi-submersible platform* menggunakan *Azimuth Propulsion System*. Prototipe *marine semi-submersible platform* ini dilengkapi dengan empat motor *thruster* sehingga membuat *platform* dapat bergerak *surge*, *sway*, *yaw*, serta mampu bergerak secara serong.

3.1.2. Perancangan Sistem Elektronik Platform

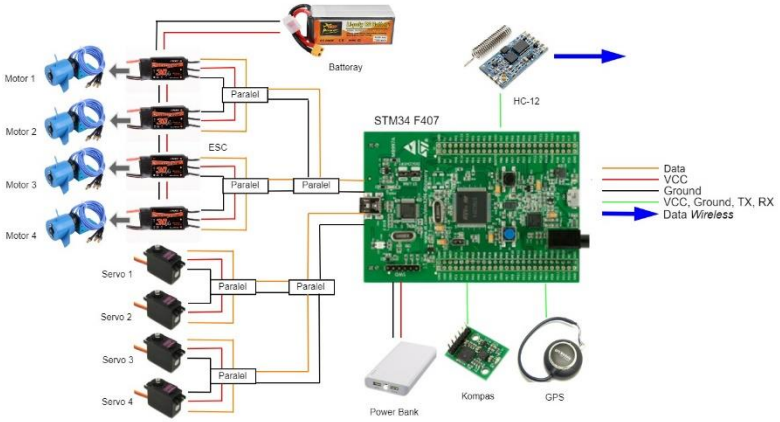
Agar *dynamic positioning system* pada *platform* dapat berjalan dan berkomunikasi secara optimal diperlukan beberapa komponen elektronik berikut :

1. GPS Ublox M8N, berfungsi sebagai *receiver* data *Global Positioning System* (GPS) dari satelit yang mengorbit di atas permukaan bumi, untuk mendapat titik koordinat *platform* maupun *fixed base* di darat dalam nilai *longitude* dan *latitude*.
2. Kompas CMPS-11, berfungsi sebagai penunjuk arah hadap *platform* saat ini dalam bentuk derajat sudut.
3. STM32F407 *Discovery*, berfungsi sebagai pusat kendali, dan pemroses *input* dan *output*.
4. Motor Servo, berfungsi sebagai aktuator yang mengarahkan arah buang resultan gaya dari motor DC *brushless*, sehingga *platform* mampu ber-*maneuver*.
5. Motor DC *Brushless*, berfungsi sebagai aktuator yang memberi gaya dorong pada *platform*, sehingga *platform* mampu bergerak.
6. Modul Komunikasi Nirkabel HC-12, berfungsi untuk komunikasi jarak jauh hingga 1,8 km yang mengirimkan data dari *fixed base* menuju *platform*.
7. *Universal Serial Bus Transistor Transistor Logic* (USB-TTL), berfungsi sebagai jalur komunikasi antara mikrokontroller pada *platform* dengan komputer pada *platform*.

8. Komputer, berfungsi sebagai media pengendali, pemetaan dan pemantauan *platform*.
9. *Remote Control* AT9, berfungsi sebagai alat kontrol manual dan sebagai *safety equipment* pada saat percobaan, sehingga saat *platform* lepas kendali, *platform* dapat diambil alih secara manual melaui *remote*.
10. *Liquid Crystal Display*, berfungsi untuk menampilkan data pada mikrokontroller.



Gambar 3.5 Skema Rangkaian Elektronik pada Prototipe *Marine Semi-Submersible Platform.*



Gambar 3.6 *Wiring Diagram* Rangkaian Elektronik pada Prototipe *Marine Semi-Submersible Platform.*

3.1.3. Desain Skematik Board PCB Platform

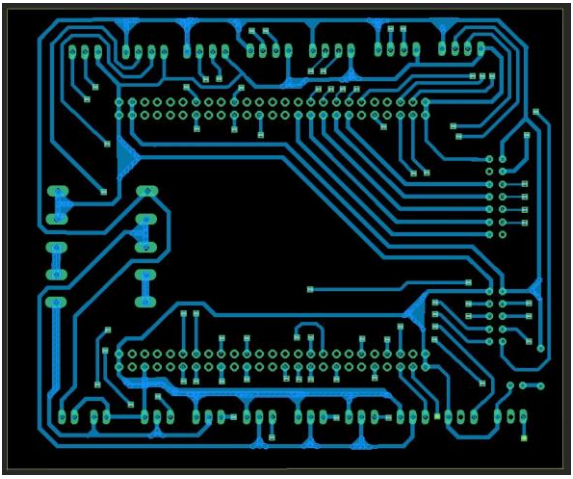
Untuk dapat membuat jalur komunikasi data antar tiap modul dengan STM32F4 yang rapih dan terstruktur dengan baik, sebelumnya dirancang dan didesain skematik *Printed Circuit Board* (PCB) menggunakan *software* Eagle Autodesk. Hasil desain skematik dapat dilihat pada Lampiran A Rancangan *layout* PCB tampak atas dapat dilihat pada Gambar 3.8 dan tampak bawah dapat dilihat pada Gambar 3.9.

Pada prototipe *marine semi-submersible platform ini* juga diperlukan seperangkat sistem tenaga yang mampu mendistribusikan tenaga untuk setiap komponen elektronik. Sistem tenaga ini dirancang supaya mampu menyesuaikan kebutuhan tiap komponen. Tegangan yang didistribusikan baik arus maupun tegangan harus dapat terus menyuplai kebutuhan selama jangka waktu yang dibutuhkan. Sedikit kesalahan pada sistem tenaga dapat membuat keseluruhan sistem bekerja tidak normal bahkan dapat bersifat destructive. Untuk kebutuhan itu semua maka perlu adanya rancangan sistem tenaga. Rancangan sistem tenga, desain skematik dan layout PCB tampak bawahnya dapat dilihat berturut turut pada Gambar 3.10, Gambar 3.11, dan Gambar 3.12.

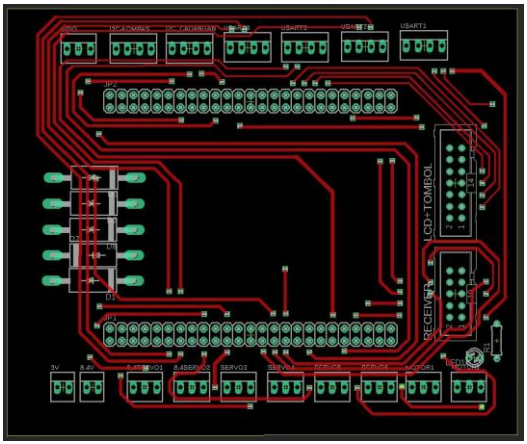
No	Komponen/Modul	Koneksi Pin		Keterangan
		Pin Modul	STM32F4	
1	GPS Ublox M8N	Tx	D9	U(S)ART3
		Vcc	5V	Power
		Gnd	Gnd	
2	Compass CMPS-11	Tx	A1	UART4
		Rx	A0	
		Vcc	5V	Power
		Gnd	Gnd	
3	Wireless Serial Communication Module HC-12	Tx	A3	U(S)ART2
		Rx	A2	
		Vcc	5V	Power
		Gnd	Gnd	
4	USB-TTL	Tx	A10	U(S)ART1
		Rx	B6	
		Vcc	5V	Power
		Gnd	Gnd	
5	LCD 16x4 display	Vcc	5V	Power
		Gnd	Gnd	
		Rs	PD7	GPIO
		Rw	PA15	
		E	PC10	
		D4	PC11	
		D5	PC12	
		D6	PD0	
		D7	PD1	
6	Receiver	Vcc	5V	Power
		Gnd	Gnd	
		Ch1	E9	Timer1
		Ch2	E11	
		Ch3	E13	
		Ch4	E14	
		Ch5	C6	Timer8
		Ch6	C7	
		Ch7	C8	

			Ch8	C9	
7	Motor Servo	Servo1	Data	PD14	Tim4/Ch3
		Servo2	Data	PD15	Tim4/Ch4
		Servo3	Data	PB0	Tim3/Ch3
		Servo4	Data	PB1	Tim3/Ch4
		Vcc		6V	Power
		Gnd		Gnd	
8	ESC Motor	ESC1	Data	PD12	Tim4/Ch1
		ESC2	Data	PD13	Tim4/Ch2
		ESC3	Data	PA6	Tim3/Ch1
		ESC4	Data	PA7	Tim3/Ch2
		Vcc		16 V	Power
		Gnd		Gnd	

Tabel 3.2 Konfigurasi Pin Antar Modul dengan STM32F4.

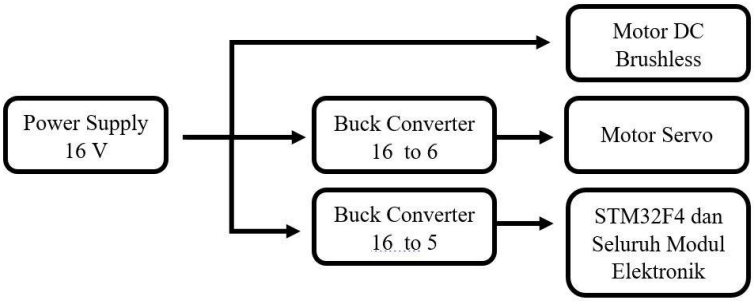


Gambar 3.7 *Layout* Tampak Bawah PCB.

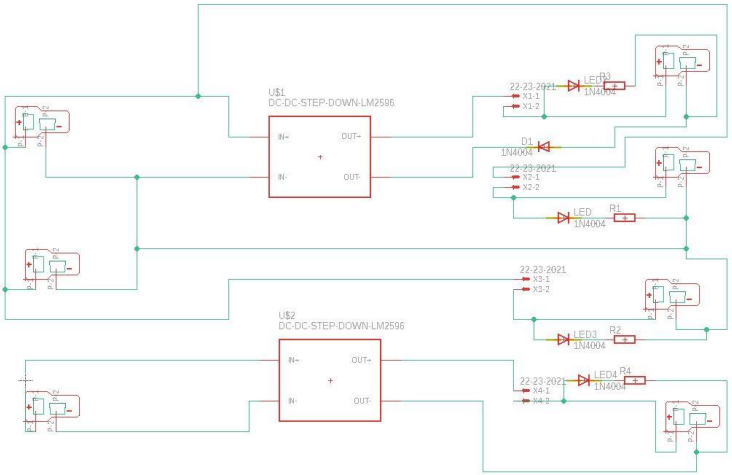


Gambar 3.8 *Layout* Tampak Atas PCB.

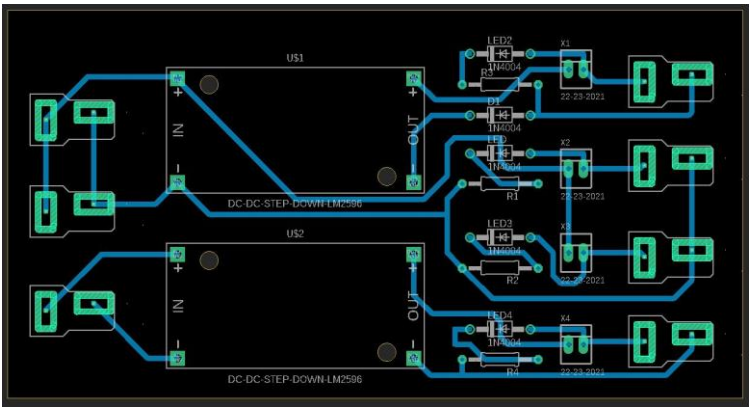
3.1.4. Rancang Bangun Sistem Tenaga Platform



Gambar 3.9 Diagram Sistem Tenaga.



Gambar 3.10 Desain Skemaitk Sistem Tenaga.

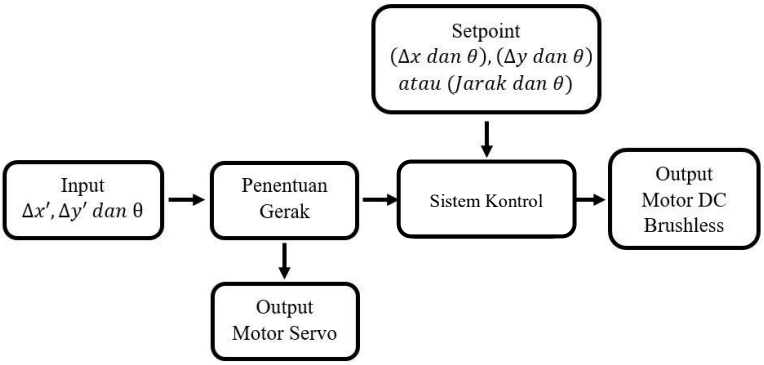


Gambar 3.11 *Layout* Tampak Bawah PCB Sistem Tenaga.

Gambar 3.12 Implementasi Keseluruhan Sistem Elektronik.

3.1.5. Perancangan Sistem Kontrol Platform

Sistem kontrol pada *platform* menggunakan algoritma *proportional integral derivative control*. Proses kontrol *platform* mngikuti proses berikut:

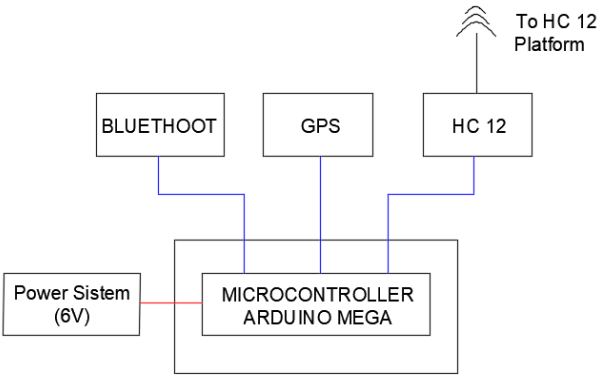


Gambar 3.13 Proses Kontrol *Platform.*

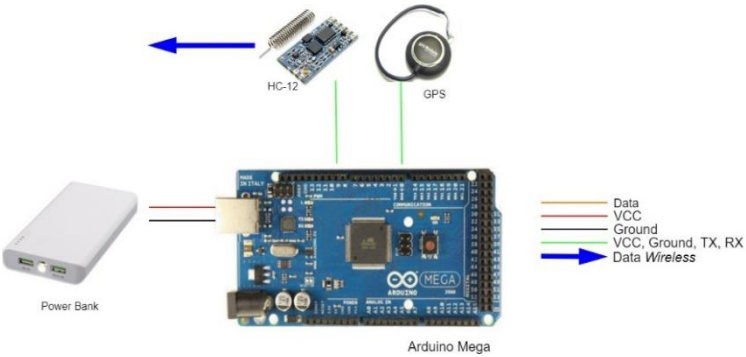
3.2. Perancangan Sistem *Base Reference*

3.2.1. Sistem Elektronik Base Reference

Metode *dynamic positioning system* pada tugas akhir ini menggunakan *differential global positioning system,* dimana terdapat *fixed base* yang diketahui lokasinya secara pasti untuk memberitahu posisi keberadaan *base* dan jaraknya terhadap *platform*. Dari keberadaan *fixed base* ini diharapkan mampu memperkecil *error* posisi yang diterima oleh modul GPS. Rancangan sistem elektronik pada *base* dapat dilihat pada Gmabar 3.14 dan Gambar 3.15.



Gambar 3.14 Skema Rangkaian Elektronik *Fixed Base.*



Gambar 3.15 *Wiring Diagram* Rangkaian Elektronik *Fixed Base.*

No	Komponen/Modul	Koneksi Pin		Keterangan
		Pin Modul	Arduino MEGA	
1	GPS Ublox M8N	Tx	Rx3	Serial
		Vcc	5V	Power
		Gnd	Gnd	
2	Wireless Serial Communication Module HC-12	Tx	Rx3	Serial
		Rx	Tx3	
		Vcc	5V	Power
		Gnd	Gnd	
		Rx	B6	
		Vcc	5V	Power

Tabel 3.3 Konfigurasi Pin Modul Elektronik dengan Arduino Mega pada *Fixed Base.*

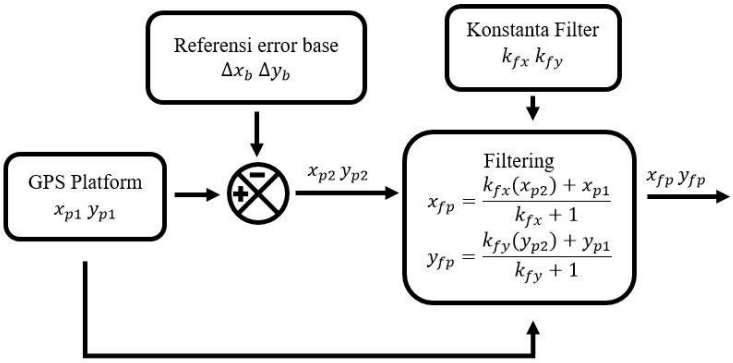
Berikut adalah komponen elektronik yang berada pada *fixed base* beserta fungsinya:

1. GPS Ublox M8N, berfungsi sebagai *receiver* data *Global Positioning System* (GPS) dari satelit yang mengorbit di atas permukaan bumi, untuk mendapat titik koordinat *platform* maupun *fixed base* di darat dalam nilai *longitude* dan *latitude*.
2. Mikrokontroller Arduino Mega, berfungsi sebagai pusat kendali, dan pemroses *input* dan *output.*
3. Modul Komunikasi Nirkabel HC-12, berfungsi untuk komunikasi jarak jauh hingga 1,8 km yang mengirimkan data posisi referensi dari *fixed base* menuju *platform*.
4. *Power Bank*, berfungsi sebagai suplai tenaga.

Gambar 3.16 Implementasi *Fixed Base.*

3.3. Perancangan Metoda DGPS untuk mengurangi *error* GPS

Penelitian ini memanfaatkan *Global Positioning System* untuk mengetahui posisi *platform*. Dalam mempertahankan posisi dibutuhkan akurasi yang cukup tinggi pada waktu pembacaan posisi *platform*. Namun dalam prakteknya teknologi GPS memiliki *error* yang cukup besar yaitu sebesar 1,5m untuk Modul GPS Ublox M8N di ruang terbuka di lingkungan darat[6]. Untuk memperkecil *error* tersebut digunakan metoda *differential global positioning system* dimana metoda ini menggunakan lokasi referensi dari tempat lain yang dalam penelitian disebut sebagai *fixed base*. *Fixed base* sudah diketahui posisinya secara pasti dan memiliki modul GPS dengan spesifikasi yang sama dengan *platform*. GPS pada *fixed base* akan memiliki *error* terhadap posisi pastinya yang *error* tersebut akan disandingkan dengan nilai GPS *platform* untuk menghilangkan *error* GPS pada *platform*. Perhitungan untuk menemukan nilai GPS *platform* yang baru :



Gambar 3.17 Rancangan Filter DGPS untuk Mengurangi *Error GPS*.

$\Delta x_b = x_{b1} - x_{b0}$ $\Delta y_b = y_{b1} - y_{b0}$

$x_{p2} = x_{p1} + \Delta x_b$ $y_{p2} = y_{p1} + \Delta y_b$

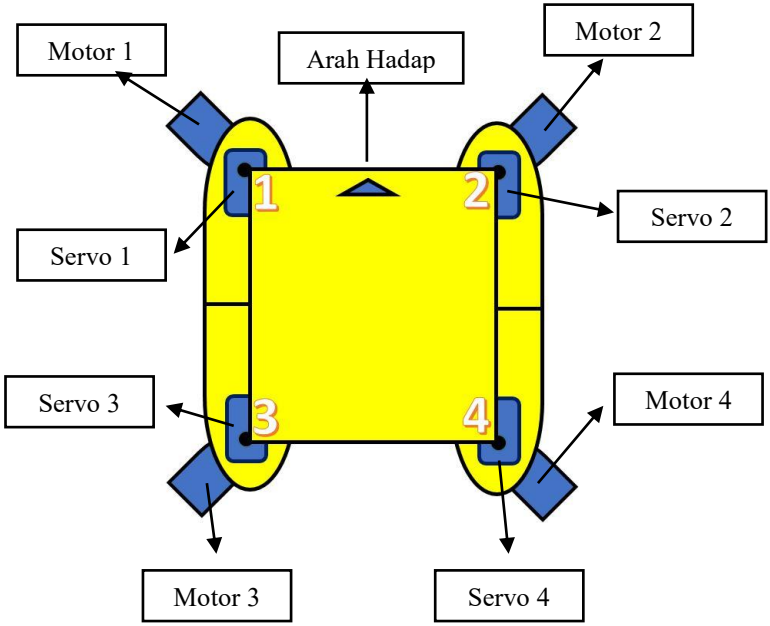
$x_{fp} = \frac{k_{fx}(x_{p2}) + x_{p1}}{k_{fx} + 1}$ $y_{fp} = \frac{k_{fy}(y_{p2}) + y_{p1}}{k_{fy} + 1}$

dimana,

x_{b0} = posisi *longitude fixed base* yang sudah diketahui sebelumnya
x_{b1} = posisi *longitude fixed base* berdasarkan *input* GPS pada *base*
y_{b0} = posisi *latitude fixed base* yang sudah diketahui sebelumnya
y_{b1} = posisi *latitude fixed base* berdasarkan *input* GPS pada *base*
x_{p1} = posisi *longitude platform* berdasarkan *input* GPS pada *platform*
x_{p2} = hasil penjumlahan *longitude platform* dengan *error base*
y_{p1} = posisi *latitude platform* berdasarkan *input* GPS pada *platform*
y_{p2} = hasil penjumlahan *latitude platform* dengan *error base*
k_{fx} = konstanta filter *longitude*
k_{fy} = konstanta filter *latitude*
x_{fp} = posisi *longitude platform* setelah difilter
y_{fp} = posisi *latitude platform* setelah difilter

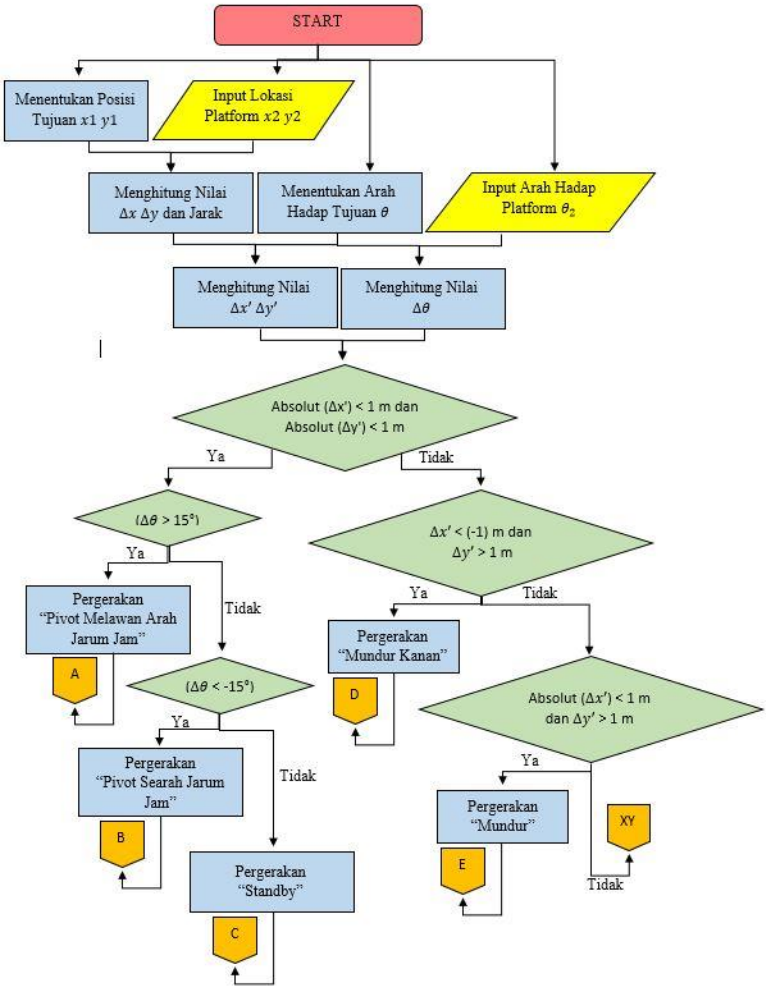
Hasil filter x_{fp} dan y_{fp} selanjutnya akan didefinisikan sebagai $x2$ dan $y2$ yaitu posisi *platform* saat ini setelah melalui proses filter. Nilai dari posisi *platform* yang telah difilter diharapkan memiliki nilai *error* yang lebih kecil. Untuk nilai konstanta filter k_{fx} dan k_{fy} akan dilakukan percobaan untuk mendapat nilai yang optimal dengan rentang konstanta filter 1-20.

3.4. Perancangan Skema Pergerakan Platform

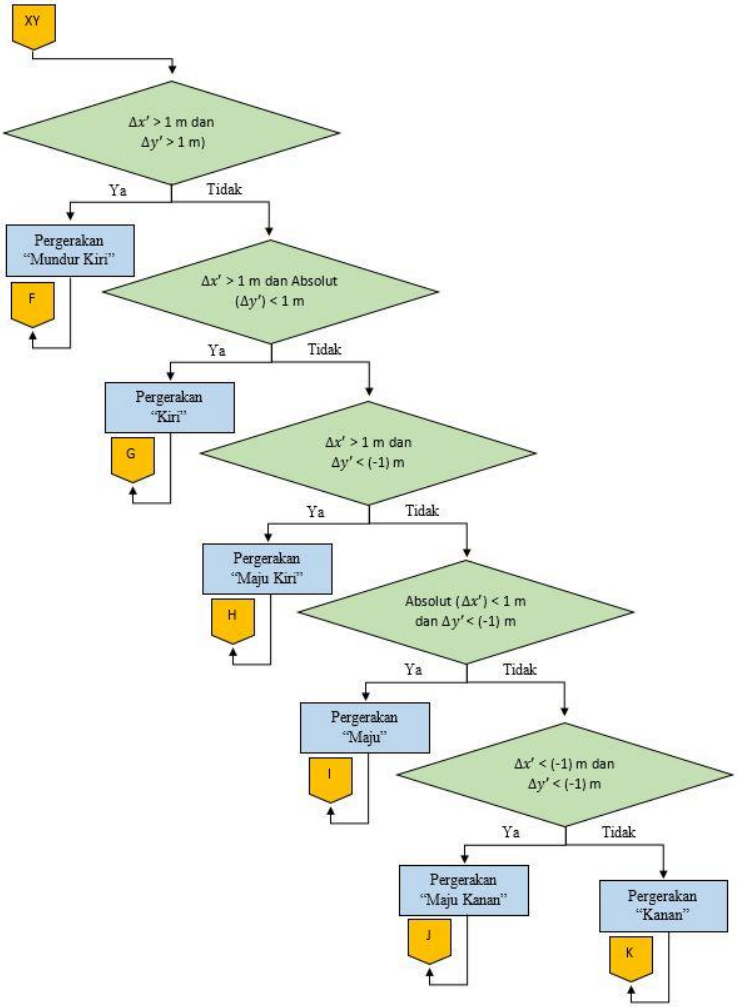


Gambar 3.18 Permodelan *Platform.*

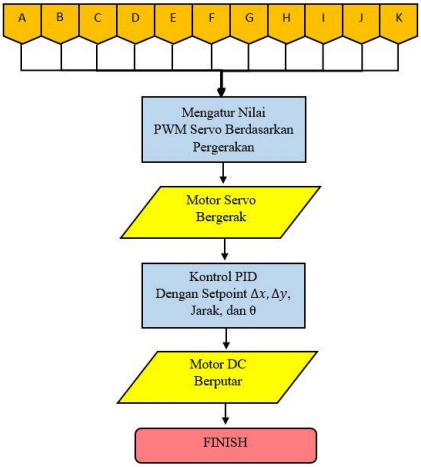
Pada Gambar 3.17 dijelaskan representasi model terhadap protipe. Bahwa prototipe yang dirancang pada tugas akhir ini memiliki 8 aktuator yaitu 4 *motor servo* dan 4 *motor dc brushless* sebagai *thruster*. Pada gambar juga ditunjukan bahwa segitiga biru menunjukan arah hadap *platform*.



Gambar 3.19 *Flowchart (1)* Logika Perencanaan Gerak *Platform.*

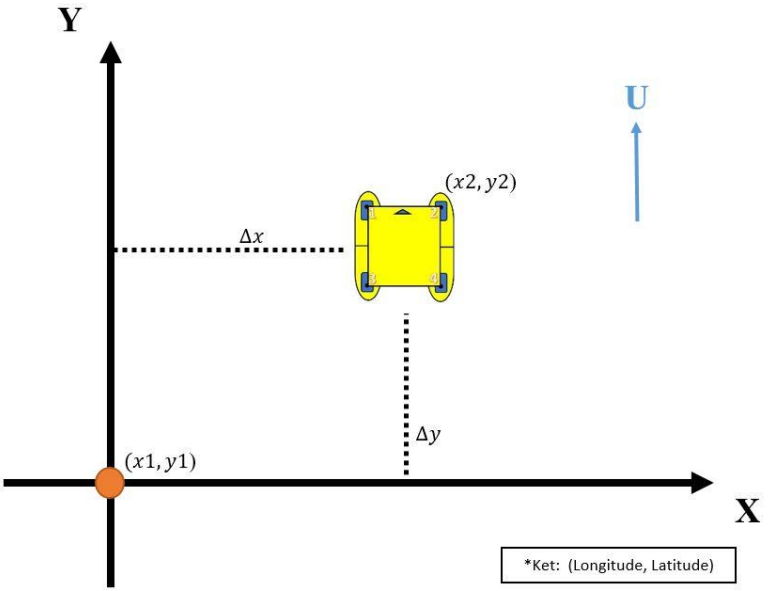


Gambar 3.20 *Flowchart (2)* Logika Perencanaan Gerak *Platform.*



Gambar 3.21 *Flowchart (3)* Logika Perencanaan Gerak *Platform.*

Pada *Flowchart* dijelaskan Logika yang digunakan penulis dalam mengambil keputusan perencanaan gerak yang dimulai dari penentuan *setpoint* dan mengambil data *input* dari sensor. Penulis menggunakan logika '*if*' untuk setiap kondisi dan menghasilkan 11 jenis pergerakan yaitu "pivot melawan arah jarum jam", "pivot searah jarum jam", "Standby", "Mundur Kanan", "Mundur", "Mundur Kiri", "Kiri", "Maju Kiri", "Maju", "Maju Kanan", dan "Kanan". Kemudian dilanjutkan untuk mengatur nilai PWM *servo* berdasarkan pergerakan dengan persyaratan kondisi yang terpenuhi. Kemudian sistem akan memberi *output* PWM kepada *servo*. Kemudian proses kontrol dilakukan berdasarkan *setpoint* yang sebelumnya sudah didapatkan yaitu $\Delta x, \Delta y,$ Jarak dan θ. Penulis menggunakan metoda Kontrol PID. Penentuan setiap nilai yang dibutuhkan dalam perencanaan gerak akan dijabarkan pada penjelasan berikut :



Gambar 3.22 Penggambaran Penentuan Nilai Δx dan Δy.

Dalam perencanaan gerak atau dengan tujuan untuk menentukan tindakan apa yang akan diambil oleh *platform* dibutuhkan 3 variabel yaitu Δx, Δy dan Δθ yang masing masing ditentukan melaui rumus berikut :

$$\Delta x = (x2 - x1) \times Ln$$

dimana,
Δx = jarak membujur antara *platform* dengan titik yang dipertahankan
$x2$ = *longitude platform* dalam desimal, hasil filter
$x1$ = *longitude* tujuan dalam desimal
Ln = konstanta *longitude* desimal ke jarak meter = 110574,6108

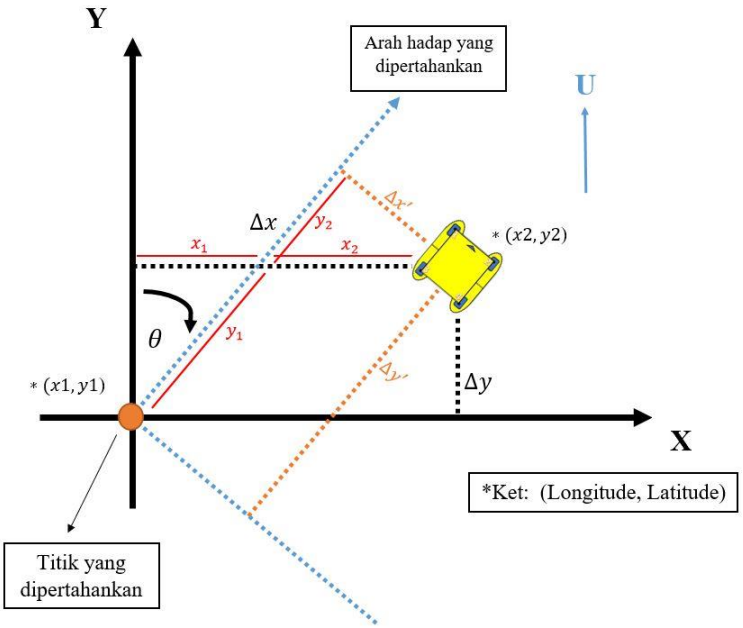
$$\Delta y = (y2 - y1) \times Lt$$

dimana,
Δy = jarak melintang antara *platform* dengan titik yang dipertahankan
$y2$ = *latitude platform* dalam desimal, hasil filter
$y1$ = *latitude* tujuan dalam desimal, dari GPS pada *fixed base*

Lt = konstanta *latitude* desimal ke jarak meter = 111320

$$r = \sqrt[2]{|\Delta y^2 - \Delta x^2|}$$

dimana,
r = jarak antara platform dengan titik tujuan



Gambar 3. 23 Penggambaran Penentuan $\Delta x'$ dan $\Delta y'$.

Penurunan rumus $\Delta x'$:

$\Delta x = x_1 + x_2$
$x_2 = \Delta x - x_1$
$x_1 = \Delta y \times \tan(\theta)$
$x_2 = \Delta x - \Delta y \times \tan(\theta)$

$\Delta x' = x_2 \times \cos(\theta)$
$\Delta x' = (\Delta x - \Delta y \times \tan(\theta)) \times \cos(\theta)$
$\Delta x' = (\Delta x \times \cos(\theta)) - (\Delta y \times \sin(\theta))$

Penurunan rumus $\Delta y'$:

$\Delta y' = y_1 + y_2$

$y_1 = \frac{\Delta y}{\cos(\theta)}$ $\Delta y' = y_1 + y_2$

$y_2 = \Delta x' \times \tan(\theta)$ $\Delta y' = \frac{\Delta y}{\cos(\theta)} + (\Delta x' \times \tan(\theta))$

dimana,
Δx = *error* jarak membujur
Δy = *error* jarak melintang
θ = sudut yang dipertahankan *platform*
$\Delta x'$ = *error* jarak membujur setelah dipengaruhi θ
$\Delta y'$ = *error* jarak melintang setelah dipengaruhi θ

Penurunan rumus $\Delta\theta$:

$$\Delta\theta = \theta_2 - \theta$$

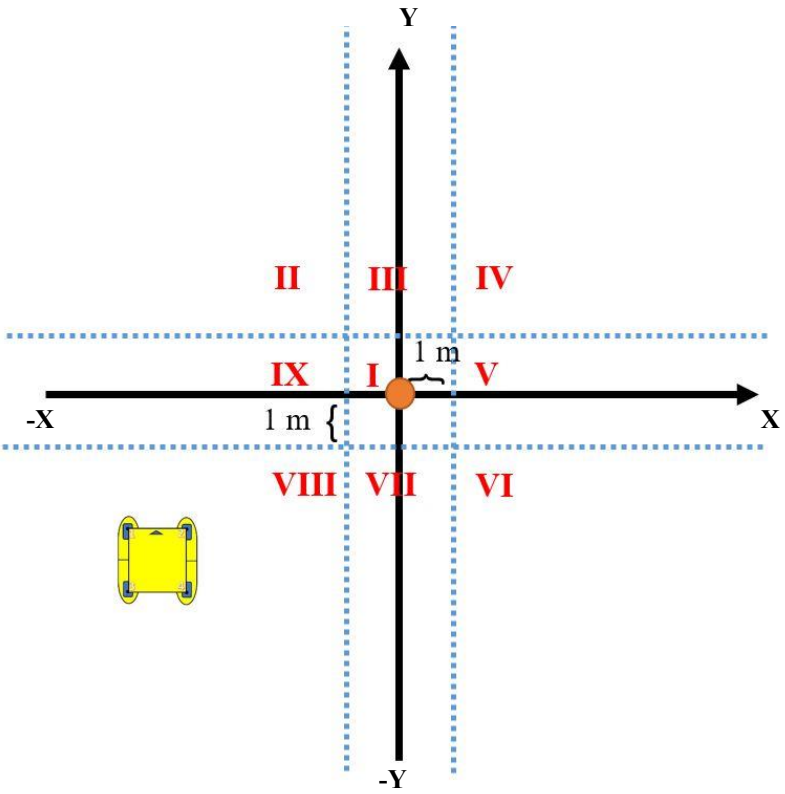
dimana,
$\Delta\theta$ = *error* sudut
θ_2 = posisi arah haadp kapal saat ini
θ = sudut yang dipertahankan *platform*

Setelah mendapat nilai $\Delta x'$, $\Delta y'$, dan $\Delta\theta$ dapat ditentukan pergerakan apa yang akan diambil oleh *platform* berdasarkan peraturan berikut :

1. **[Kuadran I]** Jika (absolut ($\Delta x'$) < 1 m dan absolut($\Delta y'$) < 1 m) maka :
    a. Jika ($\Delta\theta$ > 15°) maka {Pergerakan "pivot melawan arah jarum jam"}
    b. Jika ($\Delta\theta$ < -15°) maka {Pergerakan "pivot searah jarum jam"}
    c. Jika ($\Delta\theta$ < 15° dan $\Delta\theta$ > (-15°)) maka {Pergerakan "Standby"}
2. **[Kuadran II]** Jika ($\Delta x'$ < (-1) m dan $\Delta y'$ > 1 m) maka {Pergerakan "Mundur kanan"}
3. **[Kuadran III]** Jika (absolut($\Delta x'$) < 1 m dan $\Delta y'$ > 1 m) maka {Pergerakan "Mundur"}
4. **[Kuadran IV]** Jika ($\Delta x'$ > 1 m dan $\Delta y'$ > 1 m) maka {Pergerakan "Mundur kiri"}
5. **[Kuadran V]** Jika ($\Delta x'$ > 1 m dan absolut($\Delta y'$) < 1 m) maka {Pergerakan "Kiri"}
6. **[Kuadran VI]** Jika ($\Delta x'$ > 1 m dan $\Delta y'$ < (-1) m) maka

{Pergerakan “Maju kiri”}

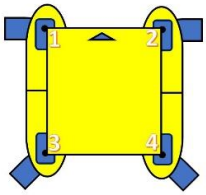
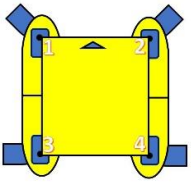
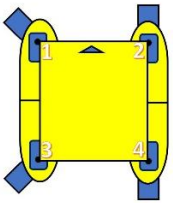
7. **[Kuadran VII]** Jika (absolut($\Delta x'$) < 1 m dan $\Delta y'$ < (-1) m) maka {Pergerakan “Maju”}
8. **[Kuadran VIII]** Jika ($\Delta x'$ < (-1) m dan $\Delta y'$ < (-1) m) maka {Pergerakan “Maju kanan
9. **[Kuadran IX]** Jika ($\Delta x'$ < (-1) m dan absolut($\Delta y'$) < 1 m) maka {Pergerakan “Kanan”}

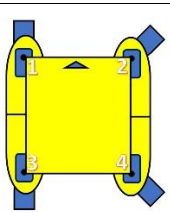
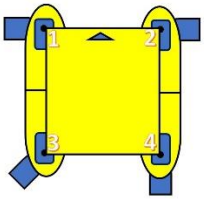
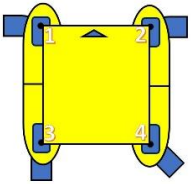
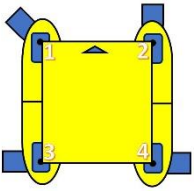


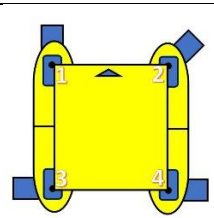
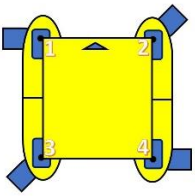
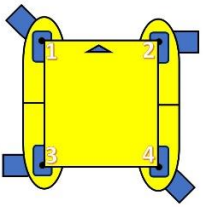
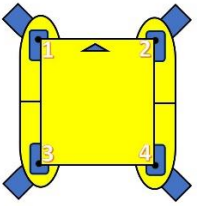
Gambar 3.24 Penggambarkan Kuadran Pergerakan *Platform.*

Setelah pergerakan ditentukan, selanjutnya adalah memposisikan setiap aktuator agar sesuai dengan tiap – tiap pergerakan. Kombinasi

posisi dari tiap – tiap motor *servo* akan berbeda. Tiap motor *servo* akan bergerak mengarahkan *thuster* agar gaya dorong yang dihasilkan mengakibatkan resultan gaya $\sum F$ yang sesuai dari pergerakan yang sebelumnya telah ditentukan. Pada Tabel 3.4 akan dijelaskan posisi aktuator pada masing – masing pergerakan.

No	Pergerakan	Skema Pergerakan	PWM Servo	Keterangan
1	Maju		Servo 1 800 Servo 2 2150 Servo 3 1810 Servo 4 1140	Motor 1 dan 2 ON dengan kecepatan berdasarkan feedback dari error arah hadap. Motor 3 dan 4 ON dengan feedback dari jarak platform terhadap tujuan
2	Mundur		Servo 1 1150 Servo 2 1810 Servo 3 2300 Servo 4 780	Motor 3 dan 4 ON dengan kecepatan berdasarkan feedback dari error arah hadap. Motor 1 dan 2 ON dengan feedback dari jarak platform terhadap tujuan
3	Kanan		Servo 1 1150 Servo 2 1500 Servo 3 1810 Servo 4 1500	Motor 2 dan 4 ON dengan kecepatan berdasarkan feedback dari error arah hadap. Motor 1 dan 3 ON dengan feedback dari jarak platform terhadap tujuan
4	Kiri		Servo 1 1500 Servo 2 1810 Servo 3 1500	Motor 1 dan 3 ON dengan kecepatan berdasarkan feedback dari error arah hadap. Motor 2 dan 4 ON

			Servo 4 1140	dengan feedback dari jarak platform terhadap tujuan
5	Maju Kanan		Servo 1 800 Servo 2 2150 Servo 3 1810 Servo 4 1500	Motor 1 dan 2 ON dengan kecepatan berdasarkan feedback dari error arah hadap. Motor 3 ON dengan feedback dari jarak platform terhadap tujuan. Motor 4 OFF
6	Maju Kiri		Servo 1 800 Servo 2 2150 Servo 3 1500 Servo 4 1140	Motor 1 dan 2 ON dengan kecepatan berdasarkan feedback dari error arah hadap. Motor 4 ON dengan feedback dari jarak platform terhadap tujuan. Motor 3 OFF
7	Mundur Kanan		Servo 1 1150 Servo 2 1500 Servo 3 2300 Servo 4 780	Motor 3 dan 4 ON dengan kecepatan berdasarkan feedback dari error arah hadap. Motor 1 ON dengan feedback dari jarak platform terhadap tujuan. Motor 2 OFF
8	Mundur Kiri		Servo 1 1500 Servo 2 1810 Servo 3	Motor 3 dan 4 ON dengan kecepatan berdasarkan feedback dari error arah hadap.

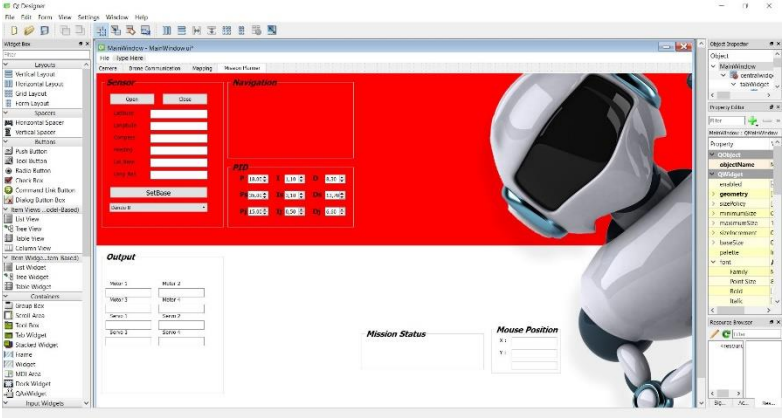
			2300 Servo 4 780	Motor 2 ON dengan feedback dari jarak platform terhadap tujuan. Motor 1 OFF
9	Pivot Searah Jarum Jam		Servo 1 800 Servo 2 1810 Servo 3 1810 Servo 4 780	Motor 1 dan 4 ON dengan kecepatan berdasarkan feedback dari error arah hadap. Motor 2 dan 3 OFF
10	Pivot Melawanan Jarum Jam		Servo 1 1150 Servo 2 2150 Servo 3 2300 Servo 4 1140	Motor 2 dan 3 ON dengan kecepatan berdasarkan feedback dari error arah hadap. Motor 1 dan 4 OFF
11	Stand-by		Servo 1 1150 Servo 2 1810 Servo 3 1810 Servo 4 1140	Motor 1, 2, 3, dan 4 OFF

Tabel 3.4 Tabel Rancangan Pergerakan *Platform.*

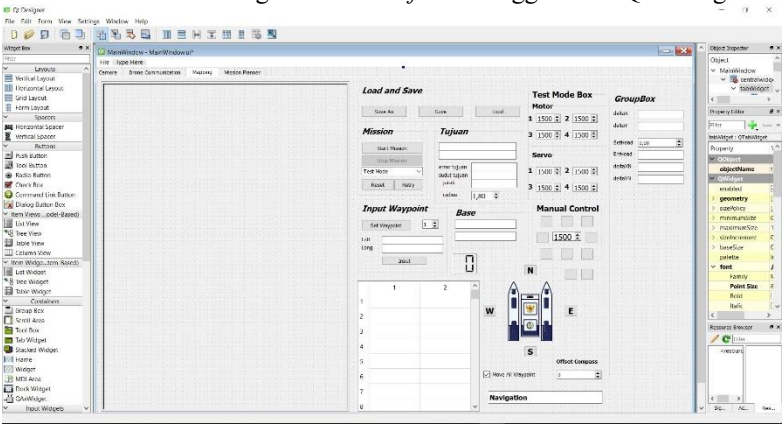
3.5. Perancangan Kendali dan Pemantauan Jarak Jauh

3.5.1. Perancangan Graphical User Interface

Pada tugas akhir ini juga dibahaas bagaimana cara mengontrol dan memonitor *platform* dari jarak jauh melalui internet. Dalam tujuan mengontrol dan memonitor *platform* dibutuhkan komputer pada *platform* dan juga sebuah user interface yang mempermudah penggunaan.



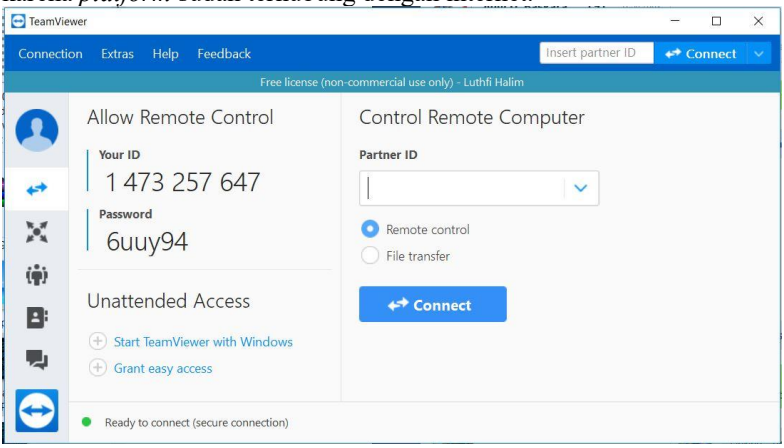
Gambar 3.25 Perancangan *User Interface* Menggunakan QT Designer.



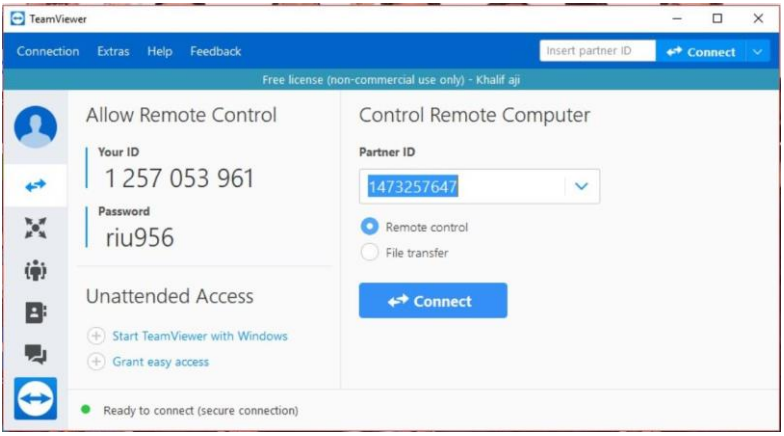
Gambar 3.26 Perancangan *User Interface* Menggunakan QT Designer.

3.5.2. Perancangan Sistem Komunikasi

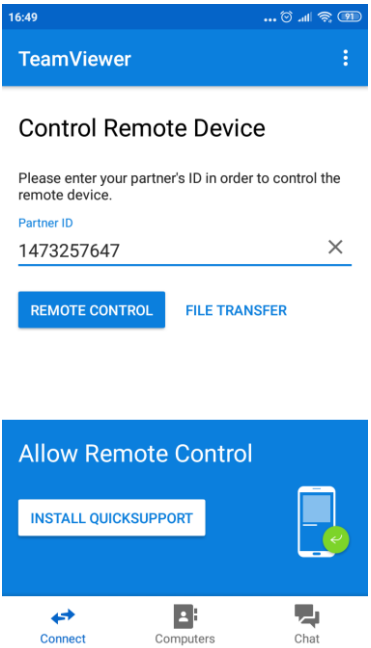
Dalam penelitian ini penulis menggunakan Aplikasi TeamViewer sebagai aplikasi penyedia layanan *remote desktop* untuk berbagai macam *platform*. TeamViewer sebagai aplikasi yang menjembatani *User* untuk mengambil alih komputer yang berada pada *platform* sehingga *platform* secara keseluruhan dapat dan dikendalikan dan dipantau dari jarak jauh karena *platform* sudah terhubung dengan internet.



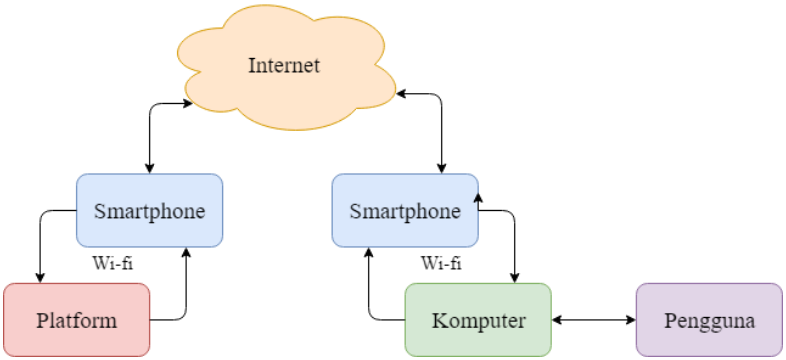
Gambar 3.27 Tampilan TeamViewer pada *User* Pengendali.



Gambar 3.28 Tampilan TeamViewer pada *Platform* yang Diambil Alih.



Gambar 3.29 Tampilan TeamViewer pada Pengendali Menggunakan Smartphone.



Gambar 3.30 Diagram Sistem Komunikasi.

Pada proses untuk mngambil alih kendali *platform,* baik *platform* maupun pengguna keduanya harus sama-sama terhubung ke internet. Pada penelitian ini penulis menggunakan *smartphone* agar keduanya dapat terhubung ke internet dengan baik. Untuk dapat terhubung pengguna harus mengetahui ID dan *password* TeamViewer *platform*. Agar terhubung keduanya juga harus telah menjalankan aplikasi TeamViewer. Pengguna hanya perlu memasukkan ID dan *password platform* untuk dapat sepenuhnya mengendalikan *platform* dari jarak jauh. Selain melalui komputer pengguna juga dapat menggunakan *smartphone* untuk mengendalikan *platform* atau hanya sekedar memantau *platform* dari jarak jauh.

.....Halaman ini sengaja dikosongkan.....

BAB IV
PENGUJIAN DAN ANALISIS

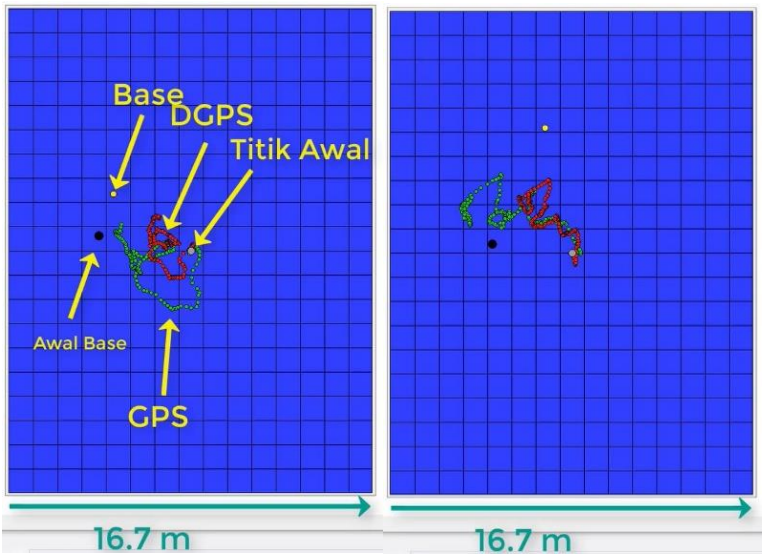
Pada bab ini akan dibahas mengenai pengujian dari sistem yang telah dirancang pada bab sebelumnya. Bab ini bertujuan untuk mendapatkan data yang kemudian dilakukan analisa pada masing-masing pengujian. Pengujian yang dilakukan meliputi pengujian GPS dan perbandingannya dengan metoda DGPS, pengujian model pergerakan, pengujian sistem komunikasi

4.1. Pengujian Sensor GPS dan metoda DGPS

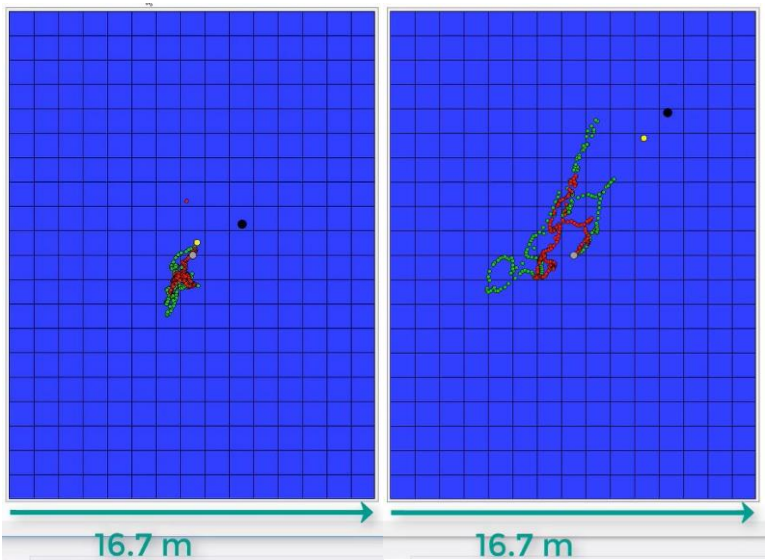
Pada pengujian sensor GPS ini dilakukan pada ruang terbuka yaitu danau 8 ITS. Metoda yang diterapkan adalah *Differential Global Positioning System* dimana GPS pada *platform* mendapat koreksi dari *fixed base* darat untuk mengurangi *error*. Dibawah ini merupakan hasil yang didapat melalui *User Interface* yang telah dikembangkan oleh penulis menggunakan Qt Designer.

Dalam pengujian ini konstanta filter pada metoda DGPS yang digunakan adalah $k_{fx} = 9/7$ dan $k_{fy} = 9/7$ yang menunjukan bahwa nilai GPS pada platform itu sendiri lebih dominan dibandingkan dengan pengaruh oleh referensi posisi yang diberikan oleh fixed base di darat dalam menentukan nilai posisi DGPS.

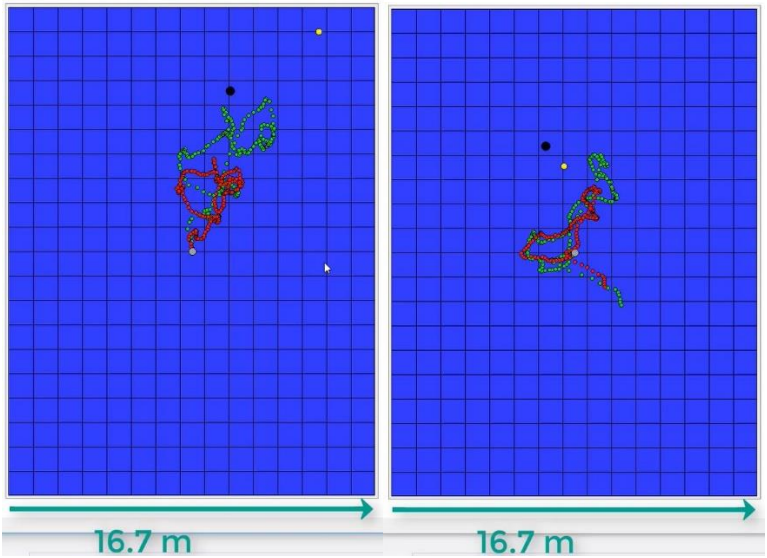
Pada Gambar 4.1, Gambar 4.2, Gambar 4.3, Gambar 4.4, Gmambar 4.5 dan Gambar 4.6 terlihat hasil perbandingan antara hasil plot GPS dan DGPS. Titik berwarna kuning merupakan posisi base sekarang berdasarkan GPS pada base. Titik berwarna hitam merupakan posisi awal base berdasarkan GPS yang nilainya tidak berubah-ubah. Titik-titik berwarna hijau merupakan plot posisi platform berdasarkan nilai GPS belum menggunakan metode DGPS. Titik-titik berwarna merah merupakan plot posisi platform setelah melalui proses koreksi dari metode DGPS. Dan titik berwarna abu-abu merupakan posisi awal platform berdasarkan GPS yang nilainya telah ditentukan diawal. Dan masing-masing kotak menggambarkan luas daerah persegi yang memiliki panjang dan lebar $1{,}1\ m\ x\ 1{,}1\ m$ pada kondisi sebenarnya.



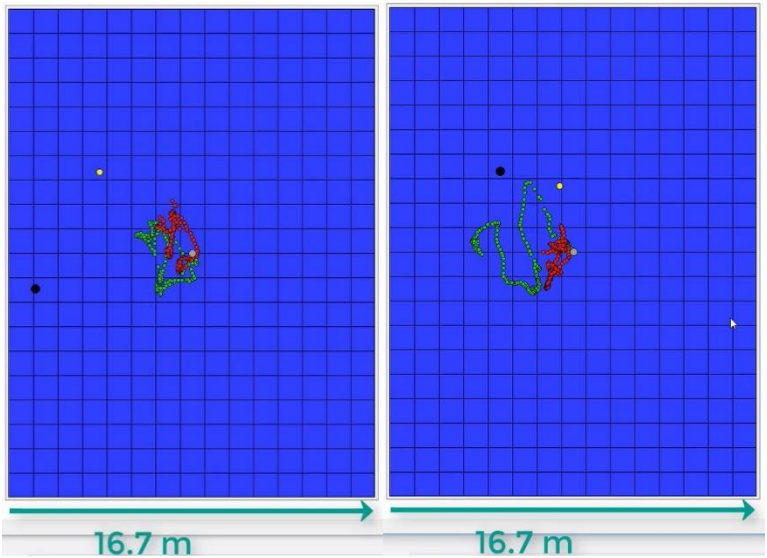
Gambar 4.1 Hasil 1 dan 2 plot posisi platform pada percobaan DGPS.



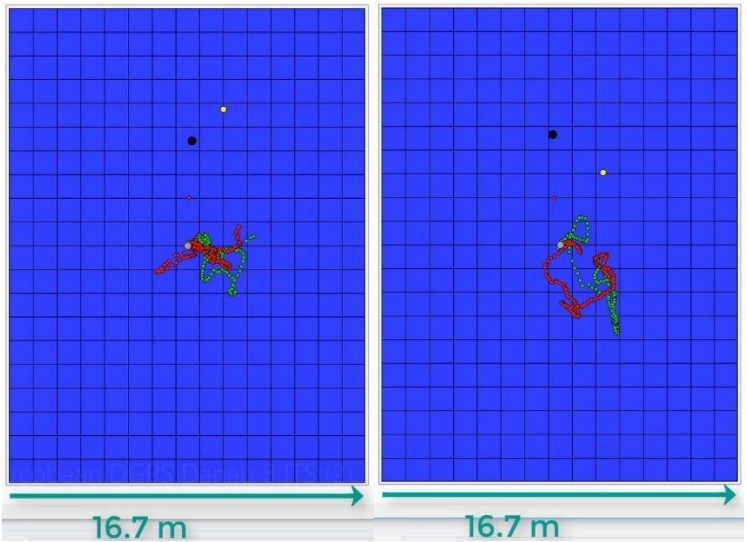
Gambar 4.2 Hasil 3 dan 4 plot posisi platform pada percobaan DGPS.



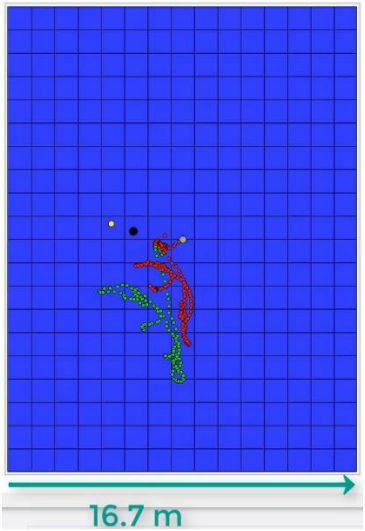
Gambar 4.3 Hasil 5 dan 6 plot posisi platform pada percobaan DGPS.



Gambar 4.4 Hasil 7 dan 8 plot posisi platform pada percobaan DGPS.



Gambar 4.5 Hasil 9 dan 10 plot posisi platform pada percobaan DGPS.



Gambar 4.6 Hasil 11 plot posisi platform pada percobaan DGPS.

Percobaan	Maks Error GPS	Maks Error DGPS	Pengurangan Error Metoda DGPS
1	3,68 m	2,83 m	23,09 %
2	6,81 m	5,00 m	26,57 %
3	5,02 m	3,42 m	31,87 %
4	3,18 m	2,37 m	25,47 %
5	4,67 m	2,18 m	53,32 %
6	2,72 m	2,52 m	7,35 %
7	4,53 m	3,32 m	26,71 %
8	7,73 m	4,39 m	43,20 %
9	6,29 m	3,70 m	41,17 %
10	3,08 m	2,13 m	30,84 %
11	5,50 m	3,64 m	33,82 %
Rata-rata	4,83 m	3,22 m	31,22 %

Tabel 4.1 Hasil Percobaan DGPS.

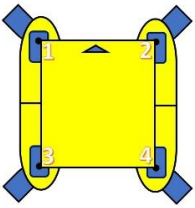
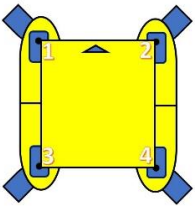
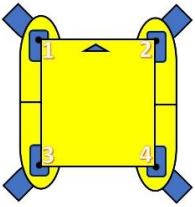
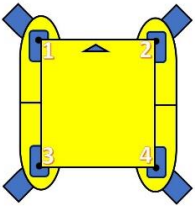
Pada Tabel 4.1 terdapat hasil rangkuman dari 11 percobaan perbandingan antara GPS dan DGPS. Hasil pengujian pada sensor GPS saja tidak terlalu akurat sehingga digunakan koreksi *error*. Penulis menggunakan metoda DGPS untuk mengurangi error yang terdapat pada sensor GPS Ublox M8N. Dari hasil percobaan sensor GPS dan DGPS didapat hasil rata rata error masing-masing sebesar 4,83m dan 3,22 m. Dari hasil tersebut disimpulkan bahwa pada percobaan ini mekanisme DGPS mampu mengurangi error GPS sebesar 31,22 %.

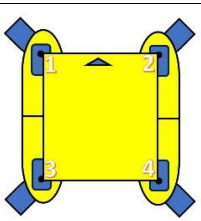
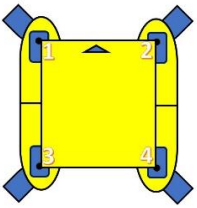
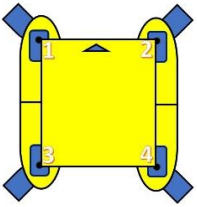
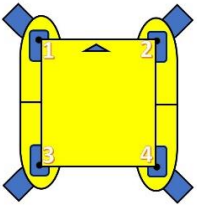
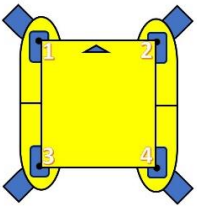
4.2. Pengujian Model Pergerakan

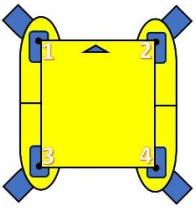
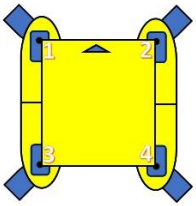
Pada bab 3.4 Perancangan Skema Pergerakan Platform sudah ditentukan bagaimana posisi tiap motor dc yang sudah diarahkan oleh tiap motor servo agar platform mampu bermaneuver omni-directional dengan baik. Omni-directinal yang dimaksud adalah pergerakan surge, sway dan yaw. Untuk mendapat hasil tersebut sebelumnya penulis sudah melakukan percobaan berbagai kondisi posisi motor untuk mendapat respon terbaik.

Pada proses uji coba penulis melakukan 3 jenis model pergerakan. Dalam proses uji coba penulis mengetahui bahwa perubahan arah motor sangat berpengaruh pada pergerakan dan keseimbangan platform. Sehingga model pergerakan yang akan digunakan adalah model pergerakan dengan meminimalisir perubahan arah motor.

4.2.1 Model Pergerakan 1 pada Uji Pergerakan

No	Pergerakan	Skema Pergerakan	Keterangan
1	Maju		Motor 1 OFF
			Motor 2 OFF
			Motor 3 ON
			Motor 4 ON
2	Mundur		Motor 1 ON
			Motor 2 ON
			Motor 3 OFF
			Motor 4 OFF
3	Kanan		Motor 1 ON
			Motor 2 OFF
			Motor 3 ON
			Motor 4 OFF
4	Kiri		Motor 1 OFF
			Motor 2 ON
			Motor 3 OFF
			Motor 4 ON
5	Maju Kanan		Motor 1 OFF
			Motor 2 OFF
			Motor 3 ON

			Motor 4 OFF
6	Maju Kiri		Motor 1 OFF Motor 2 OFF Motor 3 OFF Motor 4 ON
7	Mundur Kanan		Motor 1 ON Motor 2 OFF Motor 3 OFF Motor 4 OFF
8	Mundur Kiri		Motor 1 OFF Motor 2 ON Motor 3 OFF Motor 4 OFF
9	Pivot Searah Jarum Jam		Motor 1 ON Motor 2 OFF Motor 3 OFF Motor 4 ON
10	Pivot Melawanan		Motor 1 OFF

	Jarum Jam		Motor 2 ON
			Motor 3 ON
			Motor 4 OFF
11	Stand-by		Motor 1 OFF
			Motor 2 OFF
			Motor 3 OFF
			Motor 4 OFF

Tabel 4.2 Tabel Model 1 Pergerakan *Platform.*

Kelebihan :

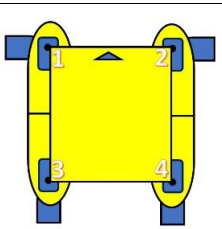
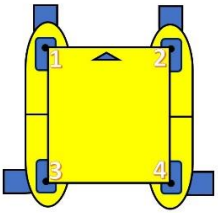
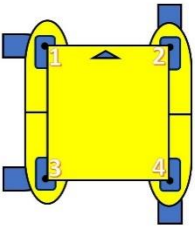
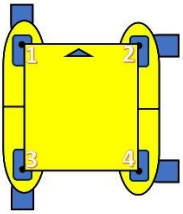
- Tidak terdapat perubahan arah motor sehingga efisien secara energy dan tidak terdapat gangguan akibat gaya yang diakibatkan perubahan arah motor.

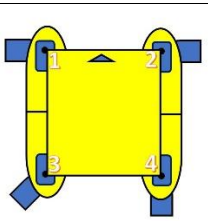
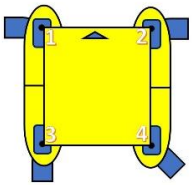
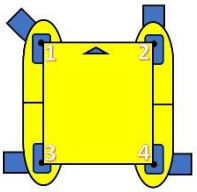
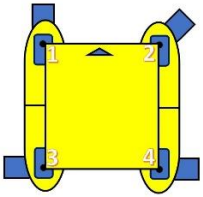
Kekurangan :

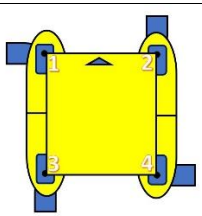
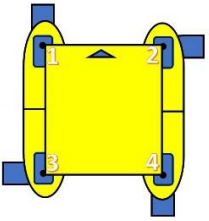
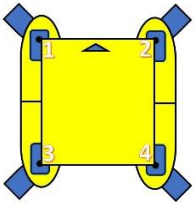
- Untuk mendapat hasil pergerakan yang baik harus memposisikan arah tiap motor secara sempurna dari segi mekanik. Resultan gaya saat semua motor menyala pada kecepatan yang sama platform harus diam ditempat.
- Sulit memasukan feedback error arah hadap terhadap kecepatan motor untuk memperbaiki arah hadap.

4.2.2 Model Pergerakan 2 pada Uji Pergerakan

No	Pergerakan	Skema Pergerakan	PWM Servo	Keterangan
1	Maju		Servo 1 800	Motor 1 dan 2 ON dengan kecepatan berdasarkan feedback dari error arah hadap. Motor 3 dan 4 ON
			Servo 2 2150	
			Servo 3 1500	

			Servo 4 1500	dengan feedback dari jarak platform terhadap tujuan
2	Mundur		Servo 1 1500 Servo 2 1500 Servo 3 2300 Servo 4 780	Motor 3 dan 4 ON dengan kecepatan berdasarkan feedback dari error arah hadap. Motor 1 dan 2 ON dengan feedback dari jarak platform terhadap tujuan
3	Kanan		Servo 1 800 Servo 2 1500 Servo 3 2300 Servo 4 1500	Motor 2 dan 4 ON dengan kecepatan berdasarkan feedback dari error arah hadap. Motor 1 dan 3 ON dengan feedback dari jarak platform terhadap tujuan
4	Kiri		Servo 1 1500 Servo 2 2150 Servo 3 1500 Servo 4 780	Motor 1 dan 3 ON dengan kecepatan berdasarkan feedback dari error arah hadap. Motor 2 dan 4 ON dengan feedback dari jarak platform terhadap tujuan
5	Maju Kanan		Servo 1 800 Servo 2 2150 Servo 3 1810	Motor 1 dan 2 ON dengan kecepatan berdasarkan feedback dari error arah hadap. Motor 3 ON dengan feedback

			Servo 4 1500	dari jarak platform terhadap tujuan. Motor 4 OFF
6	Maju Kiri		Servo 1 800 Servo 2 2150 Servo 3 1500 Servo 4 1140	Motor 1 dan 2 ON dengan kecepatan berdasarkan feedback dari error arah hadap. Motor 4 ON dengan feedback dari jarak platform terhadap tujuan. Motor 3 OFF
7	Mundur Kanan		Servo 1 1150 Servo 2 1500 Servo 3 2300 Servo 4 780	Motor 3 dan 4 ON dengan kecepatan berdasarkan feedback dari error arah hadap. Motor 1 ON dengan feedback dari jarak platform terhadap tujuan. Motor 2 OFF
8	Mundur Kiri		Servo 1 1500 Servo 2 1810 Servo 3 2300 Servo 4 780	Motor 3 dan 4 ON dengan kecepatan berdasarkan feedback dari error arah hadap. Motor 2 ON dengan feedback dari jarak platform terhadap tujuan. Motor 1 OFF
9	Pivot Searah Jarum Jam		Servo 1 800 Servo 2 1500 Servo 3	Motor 1 dan 4 ON dengan kecepatan berdasarkan feedback dari error arah hadap.

			1500 Servo 4 780	Motor 2 dan 3 OFF
10	Pivot Melawanan Jarum Jam		Servo 1 1500 Servo 2 2150 Servo 3 2300 Servo 4 1500	Motor 2 dan 3 ON dengan kecepatan berdasarkan feedback dari error arah hadap. Motor 1 dan 4 OFF
11	Stand-by		Servo 1 1150 Servo 2 1810 Servo 3 1810 Servo 4 1140	Motor 1, 2, 3, dan 4 OFF

Tabel 4.3 Tabel Model 2 Pergerakan *Platform.*

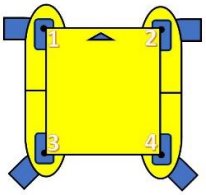
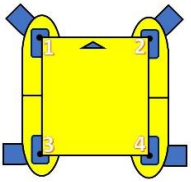
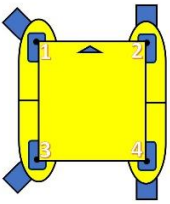
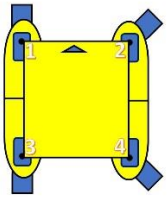
Kelebihan :

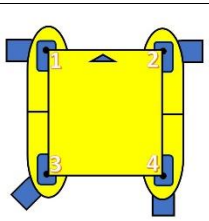
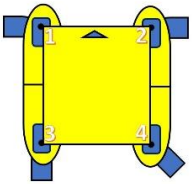
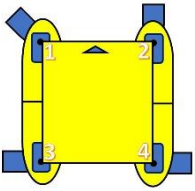
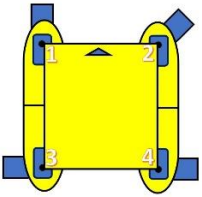
- Penambahan feedback error arah hadap terhadap kecepatan motor mudah diterapkan sehingga platform mampu mempertahankan arah hadap di tiap keadaan.
- Penentuan pergerakan mudah dilakukan

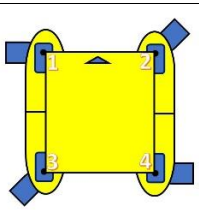
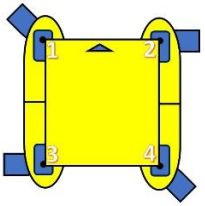
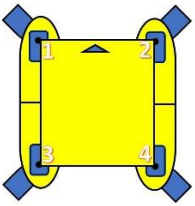
Kekurangan :

- Terlalu banyak perubahan arah motor yang menimbulkan gaya, sehingga memiliki dampak yang cukup mengganggu pergerakan dan keseimbangan platform.

4.2.3 Model Pergerakan 3 pada Uji Pergerakan

No	Pergerakan	Skema Pergerakan	PWM Servo	Keterangan
1	Maju		Servo 1 800 Servo 2 2150 Servo 3 1810 Servo 4 1140	Motor 1 dan 2 ON dengan kecepatan berdasarkan feedback dari error arah hadap. Motor 3 dan 4 ON dengan feedback dari jarak platform terhadap tujuan
2	Mundur		Servo 1 1150 Servo 2 1810 Servo 3 2300 Servo 4 780	Motor 3 dan 4 ON dengan kecepatan berdasarkan feedback dari error arah hadap. Motor 1 dan 2 ON dengan feedback dari jarak platform terhadap tujuan
3	Kanan		Servo 1 1150 Servo 2 1500 Servo 3 1810 Servo 4 1500	Motor 2 dan 4 ON dengan kecepatan berdasarkan feedback dari error arah hadap. Motor 1 dan 3 ON dengan feedback dari jarak platform terhadap tujuan
4	Kiri		Servo 1 1500 Servo 2 1810 Servo 3 1500 Servo 4 1140	Motor 1 dan 3 ON dengan kecepatan berdasarkan feedback dari error arah hadap. Motor 2 dan 4 ON dengan feedback dari jarak platform terhadap tujuan
5	Maju Kanan		Servo 1 800	Motor 1 dan 2 ON dengan kecepatan

			Servo 2 2150 Servo 3 1810 Servo 4 1500	berdasarkan feedback dari error arah hadap. Motor 3 ON dengan feedback dari jarak platform terhadap tujuan. Motor 4 OFF
6	Maju Kiri		Servo 1 800 Servo 2 2150 Servo 3 1500 Servo 4 1140	Motor 1 dan 2 ON dengan kecepatan berdasarkan feedback dari error arah hadap. Motor 4 ON dengan feedback dari jarak platform terhadap tujuan. Motor 3 OFF
7	Mundur Kanan		Servo 1 1150 Servo 2 1500 Servo 3 2300 Servo 4 780	Motor 3 dan 4 ON dengan kecepatan berdasarkan feedback dari error arah hadap. Motor 1 ON dengan feedback dari jarak platform terhadap tujuan. Motor 2 OFF
8	Mundur Kiri		Servo 1 1500 Servo 2 1810 Servo 3 2300 Servo 4 780	Motor 3 dan 4 ON dengan kecepatan berdasarkan feedback dari error arah hadap. Motor 2 ON dengan feedback dari jarak platform terhadap tujuan. Motor 1 OFF
9	Pivot Searah Jarum Jam		Servo 1 800 Servo 2 1810 Servo 3	Motor 1 dan 4 ON dengan kecepatan berdasarkan feedback dari error arah hadap.

			1810 Servo 4 780	Motor 2 dan 3 OFF
10	Pivot Melawanan Jarum Jam		Servo 1 1150 Servo 2 2150 Servo 3 2300 Servo 4 1140	Motor 2 dan 3 ON dengan kecepatan berdasarkan feedback dari error arah hadap. Motor 1 dan 4 OFF
11	Stand-by		Servo 1 1150 Servo 2 1810 Servo 3 1810 Servo 4 1140	Motor 1, 2, 3, dan 4 OFF

Tabel 4.4 Tabel Model 3 Pergerakan *Platform.*

Kelebihan :

- Penambahan feedback error arah hadap terhadap kecepatan motor mudah diterapkan sehingga platform mampu mempertahankan arah hadap di tiap keadaan.
- Penentuan pergerakan mudah dilakukan

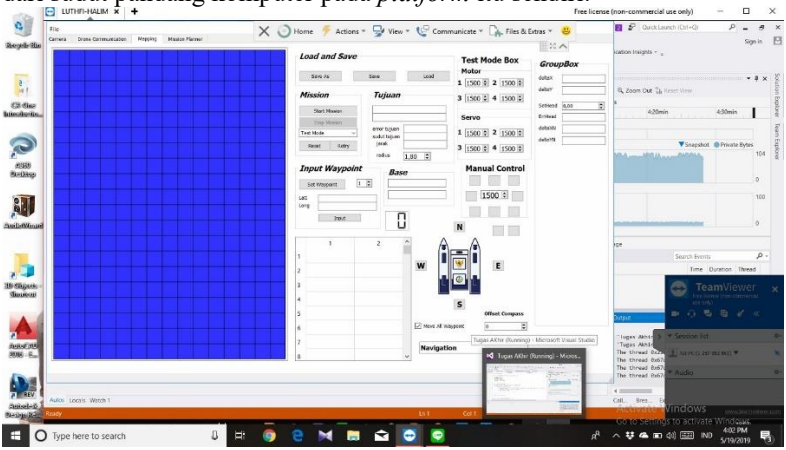
Kekurangan :

- Terdapat perubahan arah motor yang menimbulkan gaya, sehingga memiliki dampak yang cukup mengganggu pergerakan dan keseimbangan platform. Namun sudah diminimalisir sehingga lebih baik dari model pergerakan 2.

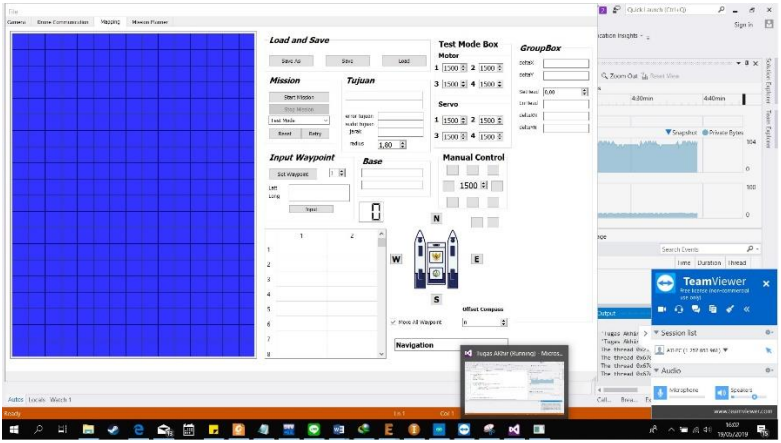
Setelah melakukan uji coba pada ketiga jenis pergerakan, penulis memilih jenis pergerakan 3 untuk diterapkan pada platform.

4.3. Pengujian Sistem Komunikasi

Pada pengujian ini penulis sekaligus sebagai pengguna mencoba untuk mengambil alih komputer yang berada pada *platform*. Aplikasi TeamViewer sebagai perantara digunakan dalam prosesnya. Pada pengujian ini didapatkan hasil bahwa TeamViewer dapat digunakan untuk mengambil alih komputer pada *platform* dimana pengguna maupun platform sudah terlebih dahulu tersambung ke internet. *Platform* dapat dikendalikan layaknya komputer normal dan dapat melakukan task apa saja seperti menggunakan komputer pada platform secara langsung. Pada Gambar 4.7 dan Gambar 4.8 terlihat kondisi saat komputer pada *platform* sudah terambil alih secara utuh. Gambar 4.7 merupakan gambaran saat *platform* terambil alih dari sudut pandang pengguna, sedangkan Gambar 4.8 merupakan gambaran saat platform sudah terambil alih yang dilihat dari sudut pandang komputer pada *platform* itu sendiri.



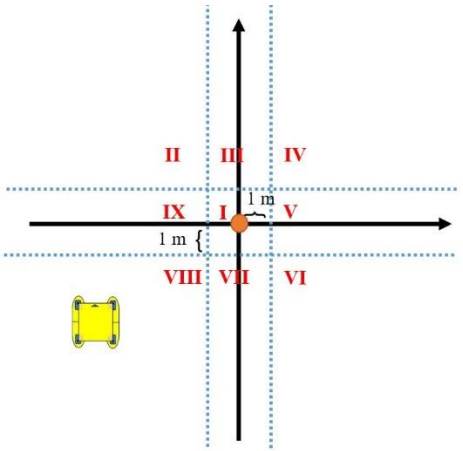
Gambar 4.7 *Interface Platform* dari sudut pandang pengguna saat dikendalikan melalui TeamViewer.



Gambar 4.8 *Interface Platform* saat dikendalikan melalui TeamViewer.

Pada pengujian ini juga diketahui bahwa pemantauan dan kendali dapat dilakukan dengan baik namun terdapat delay yang bergantung pada kecepatan internet pada saat pengujian.

4.4. Pengujian Dynamic Positioning System

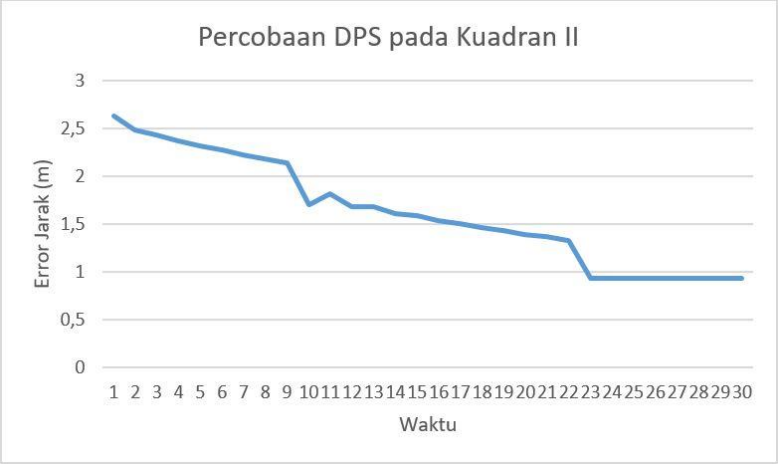


Gambar 4.9 Kuadran Pergerakan *Platform.*

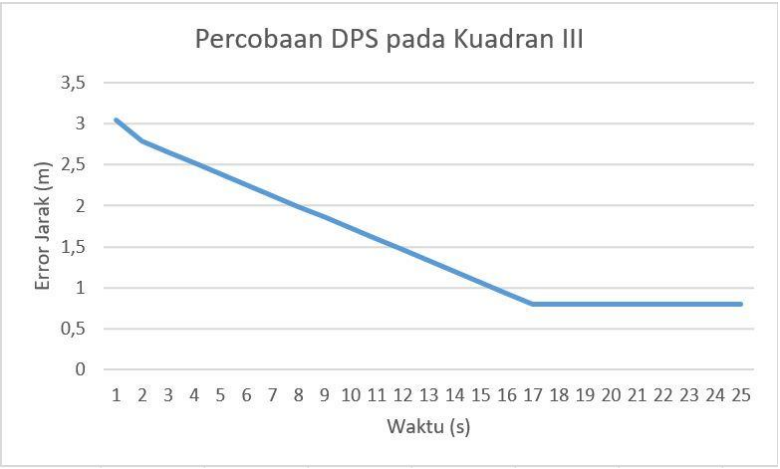
Kuadran	Respon Pergerakan
I	*Standby,* Hold Heading
II	Mundur Kanan
III	Mundur
IV	Mundur Kiri
V	Kiri
VI	Maju Kiri
VII	Maju
VIII	Maju Kanan
IX	Kanan

Tabel 4.5 Respon Pergerakan Platform pada tiap Kuadran

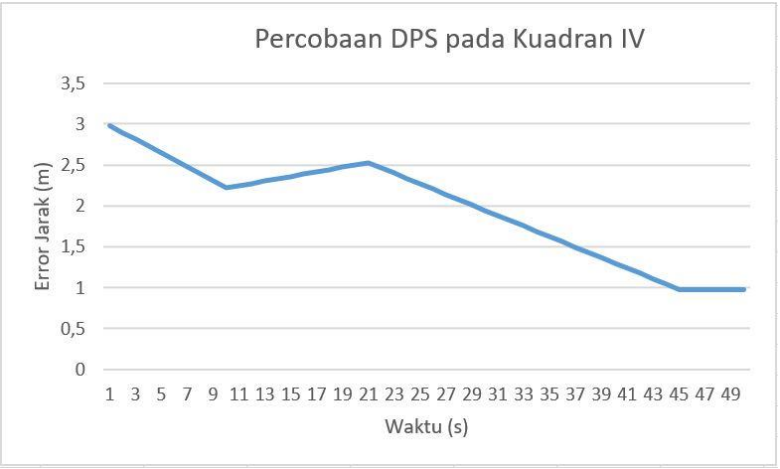
Pada pengujian ini *platform* diuji kemampuannya dalam menentukan respon yang diambil berdasarkan kuadran dimana posisi *platform* berada. Kesesuaian respon yang diambil oleh *platform* sangat menentukan dalam proses platform untuk kembali ke posisi semula yang sudah ditentukan. Penggambaran kuadran posisi *platform* dapat dilihat pada Gambar 4.9. Dari Tabel 4.5 dapat dilihat respon yang diambil oleh *platform* dan diketahui bahwa *platform* sudah mampu mengambil respon yang tepat untuk kembali ke posisi semula.



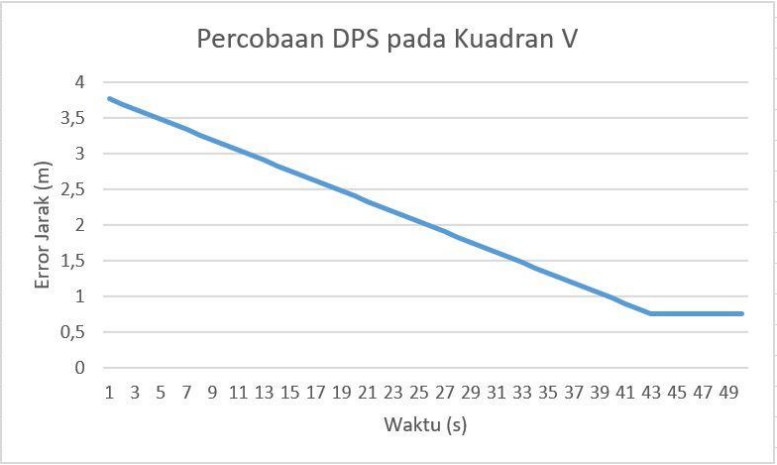
Gambar 4. 10 Respon percobaan DPS pada Kuadran II.



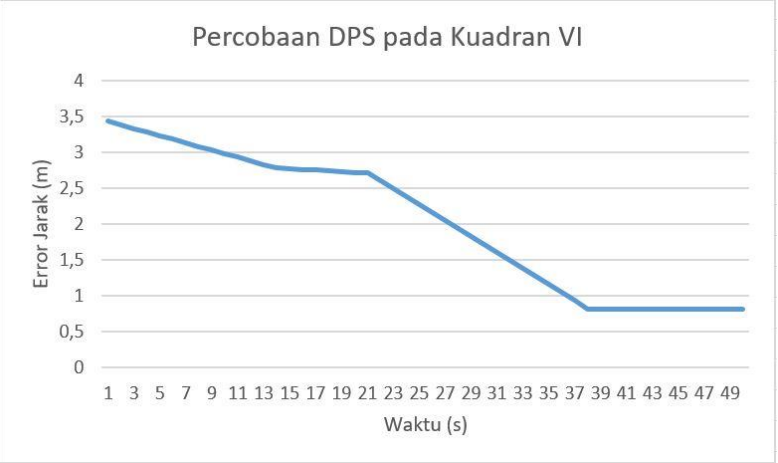
Gambar 4. 11 Respon percobaan DPS pada Kuadran III



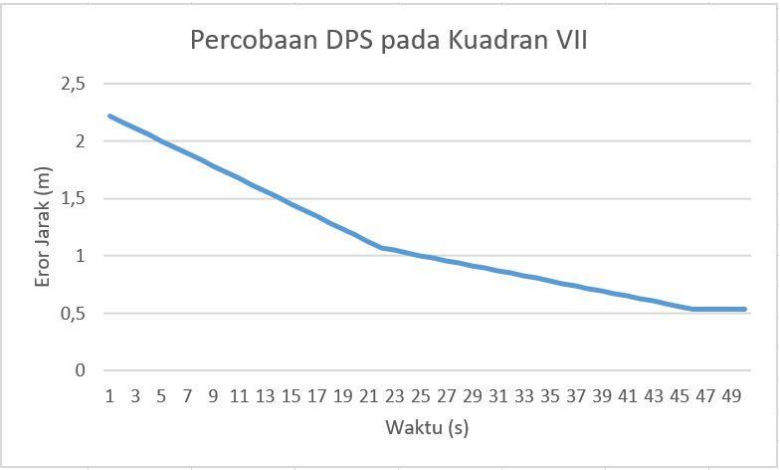
Gambar 4. 12 Respon percobaan DPS pada Kuadran IV



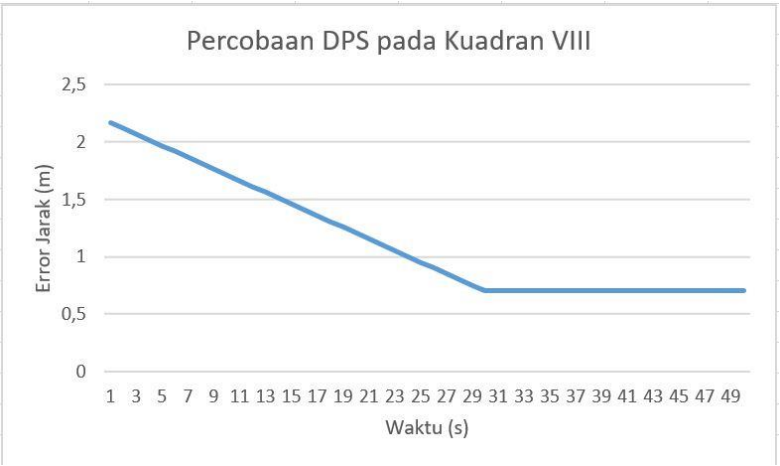
Gambar 4. 13 Respon percobaan DPS pada Kuadran V



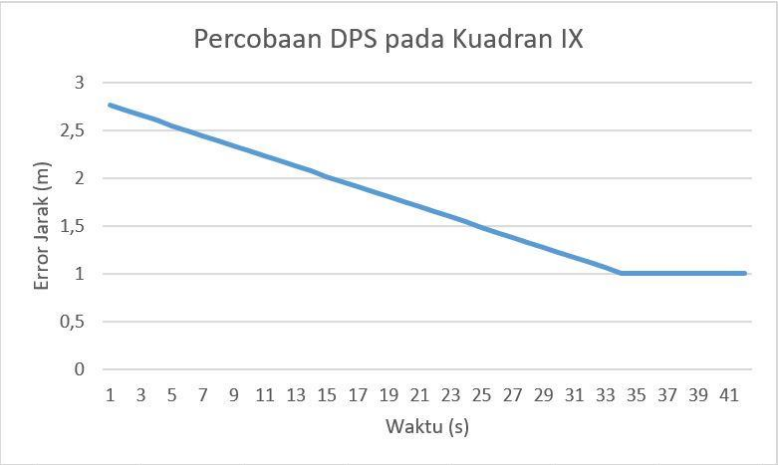
Gambar 4. 14 Respon percobaan DPS pada Kuadran VI



Gambar 4. 15 Respon percobaan DPS pada Kuadran VII



Gambar 4. 16 Respon percobaan DPS pada Kuadran VIII



Gambar 4. 17 Respon percobaan DPS pada Kuadran IX

Gambar 4. 18 Respon percobaan DPS pada Kuadran I, mempertahankan sudut arah hadap *Platform.*

4.5. Pengujian *Platform* di Danau 8 ITS

Pengujian platform pada Danau 8 ITS dilakukan untuk menguji kehandalan *platform* dalam mempertahankan posisi pada lokasi yang telah ditentukan yang terdapat arus. Pengujian ini dilakukan untuk mempertahankan posisi *platform* pada lokasi persegi dengan panjang dan lebar sebesar $2\ m\ x\ 2\ m$ dan luas 4 m^2 karena *error* GPS sampai dengan 3,22 m. Pada Gambar 4.10, Gambar 4.11 dan Gambar 4.12 terlihat saat platform bermaneuver untuk kembali ke posisi semula.

Gambar 4.19 Pengujian *Platform* di danau 8 ITS (1).

Hasil pengujian mendapat hasil yang sangat baik karena saat *platform* berada di luar area persegi, *platform* mampu kembali ke dalam area bersamaan dengan mempertahankan arah hadapnya.

Gambar 4.20 Pengujian *Platform* di danau 8 ITS (2).

Gambar 4.21 Pengujian *Platform* di danau 8 ITS (3).

.....Halaman ini sengaja dikosongkan.....

BAB V
PENUTUP

5.1. Kesimpulan

Berdasarkan hasil pengujian yang dilakukan penulis pada tugas akhir ini dapat ditarik kesimpulan bahwa prototipe *marine semi-submersible platform* dengan empat pendorong yang dilengkapi dengan *Dynamic Positioning System* mampu mempertahankan posisi pada area 4 m^2. Metoda *Differential Global Positioning System* digunakan pada keseluruhan sistem mampu mengurangi *error* GPS sebesar 31,22 % Penggunaan *Remote Desktop* dengan TeamViewer dapat digunakan untuk memantau *platform* dari jauh dengan memanfaatkan internet.

5.2. Saran

Pada proses penelitian yang dilakukan penulis masih terdapat banyak kekurangan. Proses pengembangan masih perlu dilanjutkan untuk mendapat hasil yang optimal pada platform sehingga pengaplikasiannya dapat benar benar terealisasi. Beberapa saran dari penulis untuk pengembagan platform ini adalah memanfaatan Kalman Filter untuk mengurangi error GPS dan Menggunakan Inertial Navigation System untuk membantu mengurangi error GPS.

.....Halaman ini sengaja dikosongkan.....

DAFTAR PUSTAKA

[1] "Semi-submersible platform," *Wikipedia*. 05-Mar-2019.

[2] J. Lopez-Cortijo, I. Fene, A. S. Duggal, dan F. S. F. Systems, "DP FPSO – A Fully Dynamically Positioned FPSO for Ultra Deep Waters," hlm. 9.

[3] C. S. Chas dan R. Ferreiro, "Introduction To Ship Dynamic Positioning Systems," hlm. 19.

[4] "Intro to VR: Degrees of Freedom." [Daring]. Tersedia pada: http://www.leadingones.com/articles/intro-to-vr-4.html. [Diakses: 24-Mei-2019].

[5] User Manual UM1472, "Discovery kit for STM32F407/417 lines," *STMicroelectronics*, hlm. 42, 2014.

[6] NEO-M8 - Datasheet, "NEO-M8 u-blox M8 concurrent GNSS modules." Ublox, 2015.

[7] A. Robotshop, "Arduino Mega 2560 Datasheet." Arduino, 2014.

[8] T. Q. Company, "Qt | Cross-platform software development for embedded & desktop." [Daring]. Tersedia pada: https://www.qt.io. [Diakses: 25-Mei-2019].

[9] E. L. RadioLink, "AT9S-user-manual.pdf." RadioLink, 2018.

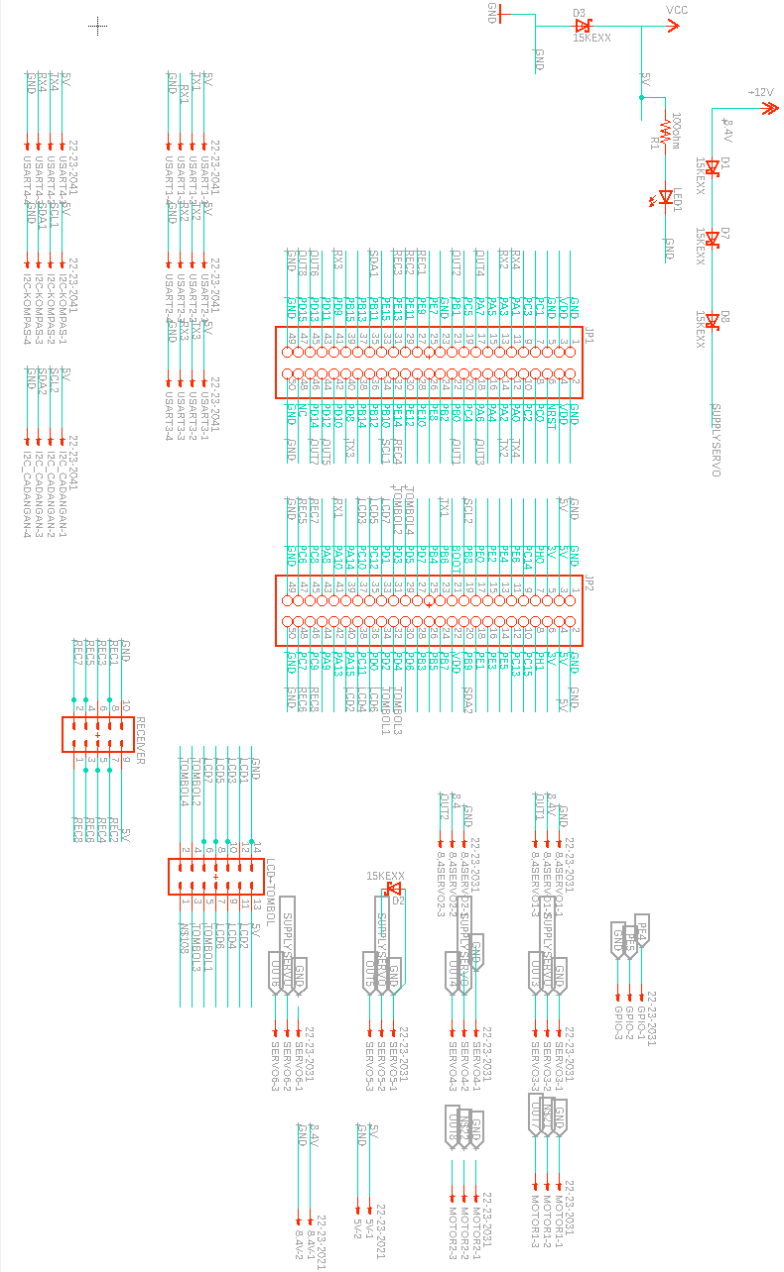
[10] User Manual CMPS-11, "CMPS-11 Tilt Compansated Compass Module." pishrobot.com, 2015.

[11] "HC-12 Wireless Serial Port Communication Module," hlm. 10.

.....Halaman ini sengaja dikosongkan.....

LAMPIRAN A

Desain Skematik Sistem Elektronik Prototipe Marine Semi-Submersible Platform dengan Mikrokontroller STM32F4



LAMPIRAN B

Program Arduino untuk mengambil nilai GPS, menerima input Bluetooth, dan mengirim data dari Base menuju *Platform*

```
#include <string.h>

float lattitude;    //float tdk di dalam loop ??
float longitude;

int count = 0;
int penandaGPS = 2;
int penandaGPS2 = 0;
char buffer_gps[100];
int t = 0;

int bluetooth = 3;
int blue;
char kirim[40];

void setup() {
   // put your setup code here, to run once:
   Serial1.begin(9600); //bluetooth
   Serial.begin(9600);
   Serial2.begin(9600); // ke STM
   Serial3.begin(9600); //GPS
}
char   aaa[20];
char   bbb[20];
char   ccc[20];
char   ddd[20];
char   eee[20];
char   fff[20];
char   ggg[20];
char   hhh[20];
char   iii2[20];
char   jjj7[20];
char   kkk[20];
```

```
int la = 0;
int lo = 0;
  char su;

void loop() {

    if(Serial1.available()>0)
    {
       bluetooth = (int)Serial1.read();
       bluetooth = bluetooth - 48;
    }
    if(Serial3.available()>0)
    {
       su = Serial3.read();

    if(penandaGPS == 2)
    {
       if(su == 'R')
       {
          penandaGPS = 3;
          //Serial.print("[R]");
       }
       else{
          penandaGPS = 2;
       }
    }
    else if(penandaGPS == 3)
    {
       if(su == 'M')
       {
          penandaGPS = 4;
          //Serial.print("[M]");
       }
       else{
          penandaGPS = 2;
       }
    }
    else if(penandaGPS == 4)
    {
```

```
    if(su == 'C')
    {
       penandaGPS = 5;
       //Serial.print("[C]");
    }
    else{
       penandaGPS = 2;
    }
}
else if(penandaGPS == 5)
{
    if(su != '$')
          {
             if(penandaGPS2 == 0){
                if( su != ',')
                {

                }
                else{
                   //Serial.print("{}");
                   penandaGPS2 = 1;
                }
             }
             else if(penandaGPS2 == 1){
                if( su != ',')
                {

                }
                else{
                   //Serial.print("{}");
                   penandaGPS2 = 2;
                }
             }
             else if(penandaGPS2 == 2){
                if( su != ',')
                {

                }
                else{
```

```
      //Serial.print("{}");
        penandaGPS2 = 3;
    }
}
else if(penandaGPS2 == 3){
    if( su != ',')
    {
        //Serial.print(la);
        ccc[la++] = su;
        //Serial.print("|");

    }
    else{
        //Serial.print("{}");
        penandaGPS2 = 4;
    }
}
else if(penandaGPS2 == 4){
    if( su != ',')
    {

    }
    else{
        //Serial.print("{}");
        penandaGPS2 = 5;
    }
}
else if(penandaGPS2 == 5){
    if( su != ',')
    {
        //Serial.print(lo);
        eee[lo++] = su;
        //Serial.print("|");

    }
    else{
        //Serial.print("{}");
        penandaGPS2 = 6;
    }
```

```
            }
            else if(penandaGPS2 == 6){
              if( su != ',')
              {

              }
              else{
                //Serial.print("{}");
                penandaGPS2 = 7;
              }
            }
          }
          else
          {
            penandaGPS = 2;
            penandaGPS2 = 0;
            t=0;
            la = 0;
            lo = 0;

            sprintf(kirim,"AB%d,",(int)bluetooth);
            Serial2.print(kirim);
            Serial.print(kirim);
            Serial2.print(ccc);
            Serial.print(ccc);
            sprintf(kirim,",");
            Serial2.print(kirim);
            Serial.print(kirim);
            Serial2.print(eee);
            Serial.print(eee);
            sprintf(kirim,"C");
            Serial2.print(kirim);
            Serial.println(kirim);
          }
    }
    }
}
```

LAMPIRAN C

Program STM32F4 sebagai kontroler utama dalam memproses semua input dan memberi seluruh output

```
#include "stm32f4xx.h"
#include "stm32f4xx_conf.h"
#include "stdio.h"
#include "delay.h"
#include "stdlib.h"
#include "lcd_biasa.h"
#include "math.h"
#include "filter.h"
#include "string.h"
#include "inisialisasi_timer.h"
#include "inisialisasi_io.h"
#include "inisialisasi_USART.h"
#include "tm_stm32f4_i2c.h"

// Definisi variabel fungsi
void monitoringData();
void ambildata();
void ambildata2();
void ambil_data_pc();
void ambil_data_arduino();
void konvertStringGps();

int Motor1 = 1500;
int Motor2 = 1500;
int Motor3 = 1500;
int Motor4 = 1500;
int Servo1 = 1500;
int Servo2 = 1500;
int Servo3 = 1500;
int Servo4 = 1500;

char tampil[32];
char kirimString[100];
int counter_drone=0;

#define out1 TIM3->CCR1
```

```
#define out2 TIM3->CCR2
#define out3 TIM3->CCR3
#define out4 TIM3->CCR4
#define out5 TIM4->CCR1
#define out6 TIM4->CCR2
#define out7 TIM4->CCR3
#define out8 TIM4->CCR4

signed long int data1[2];
signed long int data2[2];
signed long int data3[2];
signed long int data4[2];
signed long int data5[2];
signed long int data6[2];
signed long int data7[2];
signed long int data8[2];

signed long int lebar_pulsa1=3000;
signed long int lebar_pulsa2=3000;
signed long int lebar_pulsa3=3000;
signed long int lebar_pulsa4=3000;
signed long int lebar_pulsa5=3000;
signed long int lebar_pulsa6=3000;
signed long int lebar_pulsa7=3000;
signed long int lebar_pulsa8=3000;

#define ADDRESS_KOMPAS 0xC0;
#define transmit 0x00;
#define receive 0x01;
#define tombol_lcd
GPIO_ReadInputDataBit(GPIOD,GPIO_PIN_3)
int counter_lcd = 0;


long int pewaktu;
long int pewaktu_i2c;
long int pewaktu1ms;
long int pewaktu100ms;
long int pewaktu_kontrol;
int state_remot=0;
```

```
char data_siap;

//Data ke Remote
#define falling 1
#define rising   0

#define kondisi_polaritas1_rising ((TIM1-
>CCER)>>1 & 1)==0
#define kondisi_polaritas1_falling ((TIM1-
>CCER)>>1 & 1)==1
#define kondisi_polaritas2_rising ((TIM1-
>CCER)>>5 & 1)==0
#define kondisi_polaritas2_falling ((TIM1-
>CCER)>>5 & 1)==1
#define kondisi_polaritas3_rising ((TIM1-
>CCER)>>9 & 1)==0
#define kondisi_polaritas3_falling ((TIM1-
>CCER)>>9 & 1)==1
#define kondisi_polaritas4_rising ((TIM1-
>CCER)>>13 & 1)==0
#define kondisi_polaritas4_falling ((TIM1-
>CCER)>>13 & 1)==1
#define kondisi_polaritas5_rising ((TIM8-
>CCER)>>1 & 1)==0
#define kondisi_polaritas5_falling ((TIM8-
>CCER)>>1 & 1)==1
#define kondisi_polaritas6_rising ((TIM8-
>CCER)>>5 & 1)==0
#define kondisi_polaritas6_falling ((TIM8-
>CCER)>>5 & 1)==1
#define kondisi_polaritas7_rising ((TIM8-
>CCER)>>9 & 1)==0
#define kondisi_polaritas7_falling ((TIM8-
>CCER)>>9 & 1)==1
#define kondisi_polaritas8_rising ((TIM8-
>CCER)>>13 & 1)==0
#define kondisi_polaritas8_falling ((TIM8-
>CCER)>>13 & 1)==1

#define polaritas1_falling (TIM1->CCER|=(1<<1))
```

```
#define polaritas1_rising (TIM1->CCER &=
~(1<<1))
#define polaritas2_falling (TIM1->CCER|=(1<<5))
#define polaritas2_rising (TIM1->CCER &=
~(1<<5))
#define polaritas3_falling (TIM1->CCER|=(1<<9))
#define polaritas3_rising (TIM1->CCER &=
~(1<<9))
#define polaritas4_falling (TIM1->CCER|=(1<<13))
#define polaritas4_rising (TIM1->CCER &=
~(1<<13))
#define polaritas5_falling (TIM8->CCER|=(1<<1))
#define polaritas5_rising (TIM8->CCER &=
~(1<<1))
#define polaritas6_falling (TIM8->CCER|=(1<<5))
#define polaritas6_rising (TIM8->CCER &=
~(1<<5))
#define polaritas7_falling (TIM8->CCER|=(1<<9))
#define polaritas7_rising (TIM8->CCER &=
~(1<<9))
#define polaritas8_falling (TIM8->CCER|=(1<<13))
#define polaritas8_rising (TIM8->CCER &=
~(1<<13))

#define periode_tim1 40000-1
#define periode_tim8 40000-1

#define SERVOKIRI out3
#define SERVOKANAN out4
#define SERVO_DRONE  out5=out6
#define MOTORKIRI out7
#define MOTORKANAN   out8

float sudut16;
float heading = 0;
float error_heading = 0;
float error_heading_sebelum = 0;
float P_head = 18.0;
float P_heads = 35.0;
float I_head = 1.1;
```

```
float I_heads = 3.1;
float D_head = 8.3;
float D_heads = 13.7;

float pid_motor;
float pid_motor_serong;

int gerak = 0;
int move = 0;

void lcd(char kolom,char baris,char*string)
{
    lcd_gotoxy(kolom,baris);
    lcd_puts(string);
}

signed long int scale_motor;
signed long int scale_motor_mundur;
int batas_bawah_servo=800;
int batas_atas_servo=2200;

void olah_pwm_remot();
void test_servo();
void test_motor();

void test_servo()
{
    error_heading = sudut16 - heading;
        if(error_heading>180.0)
        {
            error_heading = error_heading - 360.0;
        }
        else if(error_heading<(-180.0))
        {
            error_heading = 360.0 - error_heading;
        }
        pid_motor = P_head*error_heading -
D_head*error_heading_sebelum +
I_head*error_heading_sebelum;
```

```
        pid_motor_serong = P_heads*error_heading
- D_heads*error_heading_sebelum +
I_heads*error_heading_sebelum;
        error_heading_sebelum = error_heading;

        if(lebar_pulsa4<3070 && lebar_pulsa4>2950
&& lebar_pulsa1 > 2900 && lebar_pulsa1 < 3100 &&
lebar_pulsa2 > 2900 && lebar_pulsa2 < 3100)
            {
                gerak = 0;
                move = 1;
            }
            else {
                gerak = 1;
                if(move == 1){
                    heading = sudut16;
                    move = 0;
                }
            }
    if(lebar_pulsa4<3070 && lebar_pulsa4>2950){
        if(lebar_pulsa2 < 2900) //
        {
            if(lebar_pulsa1 < 2900) //maju kiri
            {
                out3 = 1810;
                out4 = 1140;
                out7 = 800;
                out8 = 2150;
                if(error_heading < 0)
                {
                    out2 = 1670 + abs(pid_motor);
                    out5 = 1490;
                    out6 = 1490;
                }
                else{
                    out6 = 1670 +
abs(pid_motor_serong);
                    out2 = 1490;
                    out5 = 1490;
                }
```

```
                if(abs(lebar_pulsa2-3000) >
abs(lebar_pulsa1-3000))
                {
                    out1 = 4520 - lebar_pulsa2;
                }
                else{
                    out1 = 4520 - lebar_pulsa1;
                }
            }
            else if(lebar_pulsa1 > 3100) //maju
kanan
            {
                out3 = 1810;
                out4 = 1140;
                out7 = 800;
                out8 = 2150;
                if(error_heading < 0)
                {
                    out5 = 1670 +
abs(pid_motor_serong);
                    out1 = 1490;
                    out6 = 1490;
                }
                else{
                    out1 = 1670 + abs(pid_motor);
                    out6 = 1490;
                    out5 = 1490;
                }

                if(abs(lebar_pulsa2-3000) >
abs(lebar_pulsa1-3000))
                {
                    out2 = 4520 - lebar_pulsa2;
                }
                else{
                    out2 = lebar_pulsa1 - 1480;
                }
            }
            else{ //maju
```

```
                out3 = 1810;
                out4 = 1140;
                out7 = 800;
                out8 = 2150;

                out1 = 4520 - lebar_pulsa2;
                out2 = 4520 - lebar_pulsa2;

                if(error_heading < 0)
                {
                    out5 = 1590 + abs(pid_motor);
                    out6 = 1490;
                }
                else{
                    out6 = 1590 + abs(pid_motor);
                    out5 = 1490;
                }
            }
        }
        else if(lebar_pulsa2 > 3100) //mundur
        {
            //out5 = out6 = lebar_pulsa2 - 1480;
            if(lebar_pulsa1 < 2900) //mundur kiri
            {
                out3 = 2300;
                out4 = 780;
                out7 = 1150;
                out8 = 1810;
                if(error_heading < 0)
                {
                    out1 = 1670 +
abs(pid_motor_serong);
                    out3 = 1490;
                    out5 = 1490;
                }
                else{
                    out5 = 1670 + abs(pid_motor);
                    out3 = 1490;
                    out1 = 1490;
                }
```

```
                if(abs(lebar_pulsa2-3000) >
abs(lebar_pulsa1-3000))
                {
                    out6 = lebar_pulsa2 - 1480;
                }
                else{
                    out6 = 4520 - lebar_pulsa1;
                }
            }
            else if(lebar_pulsa1 > 3100) //mundur
kanan
            {
                if(error_heading < 0)
                {
                    out2 = 1490;
                    out6 = 1670 + abs(pid_motor);
                    out1 = 1490;
                }
                else{
                    out2 = 1670 +
abs(pid_motor_serong);
                    out1 = 1490;
                    out6 = 1490;
                }

                if(abs(lebar_pulsa2-3000) >
abs(lebar_pulsa1-3000))
                {
                    out5 = 4520 - lebar_pulsa2;
                }
                else{
                    out5 = lebar_pulsa1 - 1480;
                }
            }
            else{ //mundur
                out7 = 1150;
                out8 = 1810;

                out3 = 2300;
```

```
        out4 = 750;

        out5 = lebar_pulsa2 - 1480;
        out6 = lebar_pulsa2 - 1480;

        if(error_heading < 0)
        {
            out1 = 1590 + abs(pid_motor);
            out2 = 1490;
        }
        else{
            out2 = 1590 + abs(pid_motor);
            out1 = 1490;
        }
    }
}
else{
    if(lebar_pulsa1 < 2900) // kiri
    {
        out4 = 1140;
        out8 = 1810;

        out3 = 1500;
        out7 = 1500;

        out1 = 4520 - lebar_pulsa1;
        out6 = 4520 - lebar_pulsa1;

        if(error_heading < 0)
        {
            out2 = 1590 + abs(pid_motor);
            out5 = 1490;
        }
        else{
            out5 = 1590 + abs(pid_motor);
            out2 = 1490;
        }
    }
    else if(lebar_pulsa1 > 3100) //kanan
    {
```

```
            out4 = 1500;
            out8 = 1500;

            out3 = 1810;
            out7 = 1150;

            out2 = lebar_pulsa1 - 1480;
            out5 = lebar_pulsa1 - 1480;

            if(error_heading < 0)
            {
                out6 = 1590 + abs(pid_motor);
                out1 = 1490;
            }
            else{
                out1 = 1590 + abs(pid_motor);
                out6 = 1490;
            }
        }
        else{  // diam
            out3 = 1810;
            out4 = 1140;
            out7 = 1150;
            out8 = 1810;
            out5 = out6 = out1 = out2 = 1490;
        }
    }
}

else if(lebar_pulsa4<=2950) //rotate kiri
{
    out3 = 2300;
    out4 = 1140;
    out7 = 1150;
    out8 = 2150;
    out2 = out6 = 4520 - lebar_pulsa4;
    out1 = out5 = 1490;
}
else if(lebar_pulsa4>=3070) // rotate kanan
{
```

```
        out3 = 1810;
        out4 = 780;
        out7 = 800;
        out8 = 1810;
        out1 = out5 = lebar_pulsa4 - 1480;
        out2 = out6 = 1490;
    }


}
void test_motor()
{
    out3 = out4 = out7 = out8 = 1500;
    out1 = out2 = out5 = out6 = 4520 -
lebar_pulsa2;
}

void olah_pwm_remot()
{

    error_heading = sudut16 - heading;
    if(error_heading>180.0)
    {
        error_heading = error_heading - 360.0;
    }
    else if(error_heading<(-180.0))
    {
        error_heading = 360.0 - error_heading;
    }
    pid_motor = P_head*error_heading -
D_head*error_heading_sebelum +
I_head*error_heading_sebelum;
    pid_motor_serong = P_heads*error_heading -
D_heads*error_heading_sebelum +
I_heads*error_heading_sebelum;
    error_heading_sebelum = error_heading;
    //out1 = 1500;
    //out2 = 1500;
    //out7 = 1500;
    //out8 = 1500;
```

```
    //out3 = out4 = out7 = out8 = 4500 -
lebar_pulsa3;
    //out3 = out4 = out7 = out8 = 1500;

    //SERVO BELAKANG KIRI
    //out3 = 1500;  //1810//1950
    //SERVO BELAKANG KANAN
    //out4 = 1500;  //1000
    //SERVO DEPAN KIRI
    //out7 = 1500;  //1100
    //SERVO DEPAN KANAN
    //out8 = 1500;  //1950//1850
    //out1 = out2 = out5 = out6 = 4520 -
lebar_pulsa2;

    //MAJU LEBAR PULSA3 2100
    //out5 = out6 = 4520 - lebar_pulsa2;

    //out1 motor belakang kanan
    //out2 motor belakang kiri
    //out3 servo belakang kiri
    //out4 servo belakang kanan

    //out5 motor depan kiri
    //out6 motor depan kanan
    //out7 servo depan kiri
    //out8 servo depan kanan

    if(lebar_pulsa4<3070 && lebar_pulsa4>2950 &&
lebar_pulsa1 > 2900 && lebar_pulsa1 < 3100 &&
lebar_pulsa2 > 2900 && lebar_pulsa2 < 3100)
    {
        gerak = 0;
        move = 1;
    }
    else {
        gerak = 1;
        if(move == 1){
            heading = sudut16;
            move = 0;
```

```
        }
    }



 if(lebar_pulsa4<3070 && lebar_pulsa4>2950){
    if(lebar_pulsa2 < 2900) //
    {
        //out1 = out2 = 4520 - lebar_pulsa2;


        if(lebar_pulsa1 < 2900) //maju kiri
        {
            if(error_heading < 0)
            {
                out2 = 1670 + abs(pid_motor);
                //out5 = 1600 +
abs(pid_motor_serong);
                out5 = 1490;
                out6 = 1490;
            }
            else{
                out6 = 1670 +
abs(pid_motor_serong);
                out2 = 1490;
                out5 = 1490;
            }

            if(abs(lebar_pulsa2-3000) >
abs(lebar_pulsa1-3000))
            {
                out1 = 4520 - lebar_pulsa2;
            }
            else{
                out1 = 4520 - lebar_pulsa1;
            }
            //out5 = 1490;

            out7 = 740;
```

```
            out8 = 2314;
            out3 = 1500;

            out4 = 1130 ;/*+ (abs(lebar_pulsa2-
3000)-abs(lebar_pulsa1-3000));
            if(out4 > 1300)
            {
                out4 = 1300;
            }
            else if(out4 < 1000)
            {
                out4 = 1000;
            }*/

        }
        else if(lebar_pulsa1 > 3100) //maju kanan
        {
            if(error_heading < 0)
            {
                out5 = 1670 +
abs(pid_motor_serong);
                out1 = 1490;
                out6 = 1490;
            }
            else{
                out1 = 1670 + abs(pid_motor);
                out6 = 1490;
                out5 = 1490;
            }

            if(abs(lebar_pulsa2-3000) >
abs(lebar_pulsa1-3000))
            {
                out2 = 4520 - lebar_pulsa2;
            }
            else{
                out2 = lebar_pulsa1 - 1480;
            }
            //out6 = 1490;
```

```
            out7 = 740;
            out8 = 2314;
            out4 = 1500;

            out3 = 1810 ;/*- (abs(lebar_pulsa2-
3000)-abs(lebar_pulsa1-3000));
            if(out3 > 2060)
            {
                out3 = 2060;
            }
            else if(out3 < 1650)
            {
                out3 = 1650;
            }*/
        }
        else{ //maju
            out3 = 1500;
            out4 = 1500;

            out7 = 740;
            out8 = 2314;

            out1 = 4520 - lebar_pulsa2;
            out2 = 4520 - lebar_pulsa2;

            if(error_heading < 0)
            {
                out5 = 1590 + abs(pid_motor);
                out6 = 1490;
            }
            else{
                out6 = 1590 + abs(pid_motor);
                out5 = 1490;
            }


            //out5 = out6 = 1480;
        }
    }
    else if(lebar_pulsa2 > 3100) //mundur
```

```
    {
        //out5 = out6 = lebar_pulsa2 - 1480;
        if(lebar_pulsa1 < 2900) //mundur kiri
        {
            if(error_heading < 0)
            {
                out1 = 1670 + abs(pid_motor);
                out5 = 1490;
            }
            else{
                out5 = 1670 + abs(pid_motor);
                out3 = 1600 +
abs(pid_motor_serong);
                out1 = 1490;
            }

            if(abs(lebar_pulsa2-3000) >
abs(lebar_pulsa1-3000))
            {
                out6 = lebar_pulsa2 - 1480;
            }
            else{
                out6 = 4520 - lebar_pulsa1;
            }
            //out2 = 1490;

            out4 = 750;
            out7 = 1500;
            out3 = 2300;

            out8 = 1890 ;/*- (abs(lebar_pulsa2-
3000)-abs(lebar_pulsa1-3000));
            if(out8 > 2150)
            {
                out8 = 2150;
            }
            else if(out8 < 1700)
            {
                out8 = 1700;
            }*/
```

```
        }
        else if(lebar_pulsa1 > 3100) //mundur
kanan
        {
            if(error_heading < 0)
            {
                out5 = 1670 + abs(pid_motor);
                out6 = 1600 +
abs(pid_motor_serong);
                out1 = 1490;
            }
            else{
                out1 = 1670 + abs(pid_motor);
                out5 = 1490;
            }

            if(abs(lebar_pulsa2-3000) >
abs(lebar_pulsa1-3000))
            {
                out2 = 4520 - lebar_pulsa2;
            }
            else{
                out2 = lebar_pulsa1 - 1480;
            }
            //out6 = 1490;

            out3 = 2300;
            out8 = 1500;
            out4 = 750;

            out7 = 1100 ;/*+ (abs(lebar_pulsa2-
3000)-abs(lebar_pulsa1-3000));
            if(out7 > 1250)
            {
                out7 = 1250;
            }
            else if(out7 < 950)
            {
                out7 = 950;
            }*/
```

```
        }
        else{ //mundur
            out7 = 1500;
            out8 = 1500;

            out3 = 2300;
            out4 = 750;

            out5 = lebar_pulsa2 - 1480;
            out6 = lebar_pulsa2 - 1480;

            if(error_heading < 0)
            {
                out1 = 1590 + abs(pid_motor);
                out2 = 1490;
            }
            else{
                out2 = 1590 + abs(pid_motor);
                out1 = 1490;
            }
        }
    }
    else{
        if(lebar_pulsa1 < 2900) // kiri
        {
//          out6 = out1 = 4520 - lebar_pulsa1;
//          out5 = out2 = 1480;

            out4 = 750;
            out8 = 2314;

            out3 = 1500;
            out7 = 1500;

            out1 = 4520 - lebar_pulsa1;
            out6 = 4520 - lebar_pulsa1;

            if(error_heading < 0)
            {
                out2 = 1590 + abs(pid_motor);
```

```
                out5 = 1490;
            }
            else{
                out5 = 1590 + abs(pid_motor);
                out2 = 1490;
            }
        }
        else if(lebar_pulsa1 > 3100) //kanan
        {
//          out2 = out5 = lebar_pulsa1 - 1480;
//          out1 = 1480;

            out4 = 1500;
            out8 = 1500;

            out3 = 2300;
            out7 = 750;

            out2 = lebar_pulsa1 - 1480;
            out5 = lebar_pulsa1 - 1480;

            if(error_heading < 0)
            {
                out6 = 1590 + abs(pid_motor);
                out1 = 1490;
            }
            else{
                out1 = 1590 + abs(pid_motor);
                out6 = 1490;
            }
        }
        else{  // diam
            out5 = out6 = out1 = out2 = 1480;

            out3 = out4 = out7 = out8 = 1500;
        }
    }
}

else if(lebar_pulsa4<=2950)
```

```
 {
     out8 = 2314;
     out3 = 2300;
     out2 = out6 = 4520 - lebar_pulsa4;
 }
 else if(lebar_pulsa4>=3070)
 {
     out7 = 750;
     out4 = 750;
     out1 = out5 = lebar_pulsa4 - 1480;
 }


}

void TIM1_CC_IRQHandler()
{
    if(TIM_GetFlagStatus(TIM1,TIM_FLAG_CC1)!=0)
    {
        if(kondisi_polaritas1_rising)
        {
            data1[0]=TIM_GetCapture1(TIM1);
            polaritas1_falling;
            data_siap=0;
        }

        else if(kondisi_polaritas1_falling)
        {
            data1[1]=TIM_GetCapture1(TIM1);
            polaritas1_rising;


    if(data1[0]<data1[1])lebar_pulsa1=data1[1]-
data1[0];
            else
lebar_pulsa1=data1[1]+periode_tim1-data1[0];
            data_siap=1;
        }
        TIM_ClearFlag(TIM1,TIM_FLAG_CC1);
    }
```

```
    if(TIM_GetFlagStatus(TIM1,TIM_FLAG_CC2)!=0)
    {
        if(kondisi_polaritas2_rising)
        {
            data2[0]=TIM_GetCapture2(TIM1);
            polaritas2_falling;
            data_siap=0;
        }

        else if(kondisi_polaritas2_falling)
        {
            data2[1]=TIM_GetCapture2(TIM1);
            polaritas2_rising;


    if(data2[0]<data2[1])lebar_pulsa2=data2[1]-
data2[0];
            else
lebar_pulsa2=data2[1]+periode_tim1-data2[0];
            data_siap=1;
        }
        TIM_ClearFlag(TIM1,TIM_FLAG_CC2);
    }

    if(TIM_GetFlagStatus(TIM1,TIM_FLAG_CC3)!=0)
    {
        if(kondisi_polaritas3_rising)
        {
            data3[0]=TIM_GetCapture3(TIM1);
            polaritas3_falling;
            data_siap=0;
        }

        else if(kondisi_polaritas3_falling)
        {
            data3[1]=TIM_GetCapture3(TIM1);
            polaritas3_rising;
```

```
    if(data3[0]<data3[1])lebar_pulsa3=data3[1]-
data3[0];
            else
lebar_pulsa3=data3[1]+periode_tim1-data3[0];
            data_siap=1;
        }
        TIM_ClearFlag(TIM1,TIM_FLAG_CC3);
    }

    if(TIM_GetFlagStatus(TIM1,TIM_FLAG_CC4)!=0)
    {
        if(kondisi_polaritas4_rising)
        {
            data4[0]=TIM_GetCapture4(TIM1);
            polaritas4_falling;
            data_siap=0;
        }

        else if(kondisi_polaritas4_falling)
        {
            data4[1]=TIM_GetCapture4(TIM1);
            polaritas4_rising;


    if(data4[0]<data4[1])lebar_pulsa4=data4[1]-
data4[0];
            else
lebar_pulsa4=data4[1]+periode_tim1-data4[0];
            data_siap=1;
        }
        TIM_ClearFlag(TIM1,TIM_FLAG_CC4);
    }

}

void TIM8_CC_IRQHandler()
{
    if(TIM_GetFlagStatus(TIM8,TIM_FLAG_CC1)!=0)
    {
```

```
        if(kondisi_polaritas5_rising)
        {
            data5[0]=TIM_GetCapture1(TIM8);
            polaritas5_falling;
            data_siap=0;
        }
        else if(kondisi_polaritas5_falling)
        {
            data5[1]=TIM_GetCapture1(TIM8);
            polaritas5_rising;


    if(data5[0]<data5[1])lebar_pulsa5=data5[1]-
data5[0];
            else
lebar_pulsa5=data5[1]+periode_tim8-data5[0];
            data_siap=1;
        }
        TIM_ClearFlag(TIM8,TIM_FLAG_CC1);
    }
    if(TIM_GetFlagStatus(TIM8,TIM_FLAG_CC2)!=0)
    {
        if(kondisi_polaritas6_rising)
        {
            data6[0]=TIM_GetCapture2(TIM8);
            polaritas6_falling;
            data_siap=0;
        }
        else if(kondisi_polaritas6_falling)
        {
            data6[1]=TIM_GetCapture2(TIM8);
            polaritas6_rising;


    if(data6[0]<data6[1])lebar_pulsa6=data6[1]-
data6[0];
            else
lebar_pulsa6=data6[1]+periode_tim8-data6[0];
            data_siap=1;
        }
```

```
        TIM_ClearFlag(TIM8,TIM_FLAG_CC2);
    }

    if(TIM_GetFlagStatus(TIM8,TIM_FLAG_CC3)!=0)
    {
        if(kondisi_polaritas7_rising)
        {
            data7[0]=TIM_GetCapture3(TIM8);
            polaritas7_falling;
            data_siap=0;
        }
        else if(kondisi_polaritas7_falling)
        {
            data7[1]=TIM_GetCapture3(TIM8);
            polaritas7_rising;


    if(data7[0]<data7[1])lebar_pulsa7=data7[1]-
data7[0];
            else
lebar_pulsa7=data7[1]+periode_tim8-data7[0];
            data_siap=1;
        }
        TIM_ClearFlag(TIM8,TIM_FLAG_CC3);
    }
    if(TIM_GetFlagStatus(TIM8,TIM_FLAG_CC4)!=0)
    {
        if(kondisi_polaritas8_rising)
        {
            data8[0]=TIM_GetCapture4(TIM8);
            polaritas8_falling;
            data_siap=0;
        }
        else if(kondisi_polaritas8_falling)
        {
            data8[1]=TIM_GetCapture4(TIM8);
            polaritas8_rising;
```

```
    if(data8[0]<data8[1])lebar_pulsa8=data8[1]-
data8[0];
            else
lebar_pulsa8=data8[1]+periode_tim8-data8[0];
            data_siap=1;
        }
        TIM_ClearFlag(TIM8,TIM_FLAG_CC4);
    }
}

void baca_kondisi_remot()
{
    if(lebar_pulsa8>3200)
    {
        state_remot=3;  //Mode Otomatis
    }
    else
    {
        if(lebar_pulsa7 > 3200){
            state_remot = 0; //Mode Test Motor
Servo 1500
        }
        else if(lebar_pulsa7 < 2800)
        {
            state_remot = 2; //Mode Manual
        }
        else{
            state_remot = 1; //Mode Test Servo
        }
    }
}

void TIM6_DAC_IRQHandler()
{
    TIM_ClearFlag(TIM6,TIM_FLAG_Update);
    pewaktu1ms++;
    pewaktu_i2c++;

    baca_kondisi_remot();
```

```
    if(state_remot==0)olah_pwm_remot();

    if(pewaktu1ms%100==0)
        {
        pewaktu_kontrol++;
        pewaktu++;
        }
}

double lat_combo;
double long_combo;

char data_uart4;
int penanda_kompas = 0;
unsigned char data_kompas_serial[3];

void UART4_IRQHandler()
{
    if(USART_GetITStatus(UART4, USART_IT_RXNE))
        {
            if(penanda_kompas==0)
            {

    data_kompas_serial[0]=USART_ReceiveData(UART4
);
                penanda_kompas=1;
            }
            else if(penanda_kompas==1)
            {

    data_kompas_serial[1]=USART_ReceiveData(UART4
);
                penanda_kompas=0;
            }
        }
        USART_ClearITPendingBit(UART4,
USART_IT_RXNE);
}
void minta_data_kompas(char data)
{
```

```
    USART_SendData(UART4,data);
    while(USART_GetFlagStatus(UART4,USART_FLAG_TX
E) == RESET);
}

char data_usart2;
char penanda_gps=0;
char buffer_gps[100];

int t2=0;
int t1=0;
char ambil_usart2[50];

void USART2_IRQHandler(void)
{
    if(USART_GetITStatus(USART2, USART_IT_RXNE))
    {
        data_usart2=USART_ReceiveData(USART2);
        ambil_data_arduino();
    }
    USART_ClearITPendingBit(USART2,
USART_IT_RXNE);
}

int penanda_usart2 = 0;
double lattitude;
double latt_base;
double latt_base_sebelum;
double latt_base_murni_sebelum;
double error_latt_base;
double longitude;
double long_base;
double long_base_sebelum;
double long_base_murni_sebelum;
double error_long_base;
int bluetooth;

void ambil_data_arduino()
{
    if(penanda_usart2==0)
```

```
        {

    if(data_usart2=='A'){penanda_usart2=1;}
            else {penanda_usart2 = 0;}
        }
        else if(penanda_usart2==1)
            {

    if(data_usart2=='B'){penanda_usart2=2;}
                else {penanda_usart2 = 0;}
            }
        else if(penanda_usart2==2)
        {
            if(data_usart2!='C')
            {
                ambil_usart2[t2++]= data_usart2;
            }
            else
            {

    sscanf(ambil_usart2,"%d,%lf,%lf",&bluetooth,&
lattitude,&longitude);
                //konvertStringGps();
                penanda_usart2=0;
                if(bluetooth>11)
                {
                    bluetooth = 9;
                }
                t2=0;
                //memset(ambil_usart2,'0',50);
            }
        }
}

char  a[20];
char  b[20];
char  c[20];
char  d[20];
char  e[20];
char  f[20];
```

```
char  g[20];
char  h[20];
char  i2[20];
char  j7[20];
char  k[20];

void ambildata()
{
    int i=1;
    int j=0;

    while(buffer_gps[i]!=',')
    {
       a[j++]=buffer_gps[i++];
    }

    i++;j=0;
    while(buffer_gps[i]!=',')
    {
       b[j++]=buffer_gps[i++];
    }

    i++;j=0;
    while(buffer_gps[i]!=',')
    {
       c[j++]=buffer_gps[i++];
    }

    i++;j=0;
    while(buffer_gps[i]!=',')
    {
       d[j++]=buffer_gps[i++];
    }

    i++;j=0;
    while(buffer_gps[i]!=',')
    {
       e[j++]=buffer_gps[i++];
    }
```

```
    i++;j=0;
    while(buffer_gps[i]!=',')
    {
        f[j++]=buffer_gps[i++];
    }

    i++;j=0;
    while(buffer_gps[i]!=',')
    {
        g[j++]=buffer_gps[i++];
    }

    i++;j=0;
    while(buffer_gps[i]!=',')
    {
        h[j++]=buffer_gps[i++];
    }

    i++;j=0;
    while(buffer_gps[i]!=',')
    {
        i2[j++]=buffer_gps[i++];
    }

    i++;j=0;
    while(buffer_gps[i]!=',')
    {
        j7[j++]=buffer_gps[i++];
    }

    i++;j=0;
    while(buffer_gps[i]!=',' )
    {
        k[j++]=buffer_gps[i++];
        if(buffer_gps[i]==0)break;
    }

    i++;j=0;

}
```

```
double lattDouble;
double longDouble;
double lattDecimal=-7.277865;
double longDecimal=112.765421;
int ratusan,puluhan,satuan;
int ratusan2,puluhan2,satuan2;
double menit_ke_sudut,menit_ke_sudut2;
void konvertStringGps()
{
    latt_base_sebelum = latt_base;
    long_base_sebelum = long_base;

    //konversi dms ke desimal lattitude
    ratusan=lattitude/100;
    menit_ke_sudut=(lattitude-(ratusan*100))/60;
    latt_base=(ratusan+menit_ke_sudut)*(-1);

    if(latt_base != latt_base_sebelum)
    {
        latt_base_murni_sebelum =
latt_base_sebelum;
        error_latt_base = latt_base -
latt_base_sebelum;
    }

    //  //konversi dms ke desimal longitude
    ratusan2=longitude/100;
    menit_ke_sudut2=(longitude-
(ratusan2*100))/60;
    long_base=(ratusan2+menit_ke_sudut2);

    if(long_base != long_base_sebelum)
    {
        long_base_murni_sebelum =
long_base_sebelum;
        error_long_base = long_base -
long_base_sebelum;
    }
```

```
}

unsigned int data_kontrol[10];

char ambil_usart1[50];
char data_usart1;
int penanda_usart1=0;
int hitung_usart1 = 0;

void USART1_IRQHandler(void)
{

    if(USART_GetITStatus(USART1, USART_IT_RXNE))
    {
        data_usart1=USART_ReceiveData(USART1);
        //hitung_usart1++;
        ambil_data_pc();
    }
    USART_ClearITPendingBit(USART1,
USART_IT_RXNE);

}

void ambil_data_pc()
{
    if(penanda_usart1==0)
    {
        if(data_usart1=='A'){penanda_usart1=1;}
        else {penanda_usart1 = 0;}
    }
    else if(penanda_usart1==1)
        {

    if(data_usart1=='B'){penanda_usart1=2;}
            else {penanda_usart1 = 0;}
        }
    else if(penanda_usart1==2)
    {
        if(data_usart1!='C')
        {
```

```
            ambil_usart1[t1++]= data_usart1;
        }
        else
        {

    sscanf(ambil_usart1,"%4d,%4d,%4d,%4d,%4d,%4d,
%4d,%4d,C",&Motor1,&Motor2,&Motor3,&Motor4,&Serv
o1,&Servo2,&Servo3,&Servo4);
            penanda_usart1=0;
            t1=0;
        }
    }
}


void buffer_data_pc()
{
    int i=1;
        int j=0;

        while(buffer_gps[i]!=',')
        {
            a[j++]=buffer_gps[i++];
        }

        i++;j=0;
        while(buffer_gps[i]!=',')
        {
            b[j++]=buffer_gps[i++];
        }

        i++;j=0;
        while(buffer_gps[i]!=',')
        {
            c[j++]=buffer_gps[i++];
        }

        i++;j=0;
        while(buffer_gps[i]!=',')
        {
```

```
            d[j++]=buffer_gps[i++];
        }
}

int data_usart3;
unsigned char data_kompas[2];
char buffer_gps2[100];
int t3 = 0;

int penanda_gps2 = 0;

void USART3_IRQHandler(void)
{
    if(USART_GetITStatus(USART3, USART_IT_RXNE))
        {
            data_usart3=USART_ReceiveData(USART3);
            if(penanda_gps2==0)
            {

    if(data_usart3=='G')penanda_gps2=1;
                else penanda_gps2 = 0;
            }
            else if(penanda_gps2==1)
            {

    if(data_usart3=='P'||data_usart3=='N')penanda
_gps2=2;
                else penanda_gps2 = 0;
            }
            else if(penanda_gps2==2)
            {

    if(data_usart3=='R')penanda_gps2=3;
                else penanda_gps2 = 0;
            }
            else if(penanda_gps2==3)
            {

    if(data_usart3=='M')penanda_gps2=4;
                else penanda_gps2 = 0;
```

```
            }
            else if(penanda_gps2==4)
            {

    if(data_usart3=='C')penanda_gps2=5;
                else penanda_gps2 = 0;
            }
            else if(penanda_gps2==5)
            {
                if(data_usart3!='$')
                {
                    buffer_gps2[t3++]= data_usart3;
                }
                else
                {   penanda_gps2=0;
                    t3=0;
                    ambildata2();
                }
            }

        }
        USART_ClearITPendingBit(USART2,
USART_IT_RXNE);
}
char  aaa[20];
char  bbb[20];
char  ccc[20];
char  ddd[20];
char  eee[20];
char  fff[20];
char  ggg[20];
char  hhh[20];
char  iii2[20];
char  jjj7[20];
char  kkk[20];

void ambildata2()
{
    int iii=1;
    int jjj=0;
```

```
while(buffer_gps2[iii]!=',')
{
    aaa[jjj++]=buffer_gps2[iii++];
}

iii++;jjj=0;
while(buffer_gps2[iii]!=',')
{
    bbb[jjj++]=buffer_gps2[iii++];
}

iii++;jjj=0;
while(buffer_gps2[iii]!=',')
{
    ccc[jjj++]=buffer_gps2[iii++];
}

iii++;jjj=0;
while(buffer_gps2[iii]!=',')
{
    ddd[jjj++]=buffer_gps2[iii++];
}

iii++;jjj=0;
while(buffer_gps2[iii]!=',')
{
    eee[jjj++]=buffer_gps2[iii++];
}

iii++;jjj=0;
while(buffer_gps2[iii]!=',')
{
    fff[jjj++]=buffer_gps2[iii++];
}

iii++;jjj=0;
while(buffer_gps2[iii]!=',')
{
    ggg[jjj++]=buffer_gps2[iii++];
```

```
    }

    iii++;jjj=0;
    while(buffer_gps2[iii]!=',')
    {
        hhh[jjj++]=buffer_gps2[iii++];
    }

    iii++;jjj=0;
    while(buffer_gps2[iii]!=',')
    {
        iii2[jjj++]=buffer_gps2[iii++];
    }

    iii++;jjj=0;
    while(buffer_gps2[iii]!=',')
    {
        jjj7[jjj++]=buffer_gps2[iii++];
    }

    iii++;jjj=0;
    while(buffer_gps2[iii]!=',' )
    {
        kkk[jjj++]=buffer_gps2[iii++];
        if(buffer_gps2[iii]==0)break;
    }

    iii++;jjj=0;

}

double lattDouble2;
double longDouble2;
double lattDecimal2=-7.277865;
double lattDecimal2Sebelum;
double lattDecimal2MurniSebelum;
double errorLatt1;
double longDecimal2=112.765421;
double longDecimal2Sebelum;
double longDecimal2MurniSebelum;
```

```
double errorLong1;
int ratusan11,puluhan11,satuan11;
int ratusan22,puluhan22,satuan22;
double menit_ke_sudut11,menit_ke_sudut22;

void konvertStringGps2()
{
    lattDecimal2Sebelum = lattDecimal2;
    longDecimal2Sebelum = longDecimal2;
    lattDouble2=atof(ccc);
    longDouble2=atof(eee);

    //konversi dms ke desimal lattitude
    ratusan11=lattDouble2/100;
    puluhan11=(lattDouble2-ratusan11*100)/10;
    satuan11=(lattDouble2-ratusan11*100-
puluhan11*10);
    menit_ke_sudut11=(lattDouble2-
(ratusan11*100))/60;

    lattDecimal2=(ratusan11+menit_ke_sudut11)*(-
1);
    if(lattDecimal2 != lattDecimal2Sebelum)
    {
        lattDecimal2MurniSebelum =
lattDecimal2Sebelum;
       errorLatt1 = lattDecimal2 -
lattDecimal2Sebelum;
    }

    //konversi dms ke desimal longitude
    ratusan22=longDouble2/100;
    puluhan22=(longDouble2-ratusan22*100)/10;
    satuan22=(longDouble2-ratusan22*100-
puluhan22*10);
    menit_ke_sudut22=(longDouble2-
(ratusan22*100))/60;

    longDecimal2=(ratusan22+menit_ke_sudut22);
    if(longDecimal2 != longDecimal2Sebelum)
```

```
    {
        longDecimal2MurniSebelum =
longDecimal2Sebelum;
        errorLong1 = longDecimal2 -
longDecimal2Sebelum;
    }

}

unsigned int angle16;

int kirimsudut;

void kirim_data_ke_pc(char data)
{
    USART_SendData(USART1,data);
    while(USART_GetFlagStatus(USART1,USART_FLAG_T
XE) == RESET);
}

int counter=0;

void kirim_ke_pc()
{
    int a;
    sprintf(kirimString,"A%2.7lfB%3.7lfC%dD%2.7fE
%3.7fG",fabs(lattDecimal2),longDecimal2,angle16,
fabs(latt_base),long_base);     //Dihilangkan
Degrees
    for(a=0;a<=strlen(kirimString);a++)
    {
        USART_SendData(USART1,kirimString[a]);

    while(USART_GetFlagStatus(USART1,USART_FLAG_T
XE) == RESET);
    }
}

uint8_t compass[4];
```

```
void monitoringData(int menu)
{
    switch(menu)
    {
        case 0:
                sprintf(tampil,"1. Tugas Akhir
2019");
                lcd(0,0,tampil);
                sprintf(tampil,"   Luthfi Halim
E55");
                lcd(0,1,tampil);
                sprintf(tampil,"   Khalif Aji
P55");
                lcd(0,2,tampil);
                sprintf(tampil,"   Wisuda 120
Amin");
                lcd(0,3,tampil);
                break;
        case 1:
                sprintf(tampil,"2. Motor");
                lcd(0,0,tampil);
                sprintf(tampil,"   M1:%4d
M2:%4d",(int)Motor1, (int)Motor2);
                lcd(0,1,tampil);
                sprintf(tampil,"   M3:%4d
M4:%4d",(int)Motor3, (int)Motor4);
                lcd(0,2,tampil);
                break;
        case 2:
                sprintf(tampil,"3. Servo");
                lcd(0,0,tampil);
                sprintf(tampil,"   S1:%4d
S2:%4d",(int)Servo1, (int)Servo2);
                lcd(0,1,tampil);
                sprintf(tampil,"   S3:%4d
S4:%4d",(int)Servo3, (int)Servo4);
                lcd(0,2,tampil);
                break;
        case 3:
```

```
                sprintf(tampil,"4. DATA GPS
Compass");
                lcd(0,0,tampil);
                sprintf(tampil,"
%2.7lf",lattDecimal2);
                lcd(0,1,tampil);
                sprintf(tampil,"
%3.7lf",longDecimal2);
                lcd(0,2,tampil);
                sprintf(tampil,"  %5.1f
",sudut16);
                lcd(0,3,tampil);
                break;
        case 4:
                sprintf(tampil,"5.Output");
                lcd(0,0,tampil);
                sprintf(tampil,"%4d %4d
%4d",(int)out1, (int)out2, (int)out3);
                lcd(0,1,tampil);
                sprintf(tampil,"%4d %4d
%4d",(int)out4, (int)out5, (int)out6);
                lcd(0,2,tampil);
                sprintf(tampil,"%4d
%4d",(int)out7, (int)out8);
                lcd(0,3,tampil);
                break;
        case 5:
                sprintf(tampil,"6.Data Remote");
                lcd(0,0,tampil);
                sprintf(tampil,"%4d %4d
%4d",(int)lebar_pulsa1,(int)lebar_pulsa2,(int)le
bar_pulsa3);
                lcd(0,1,tampil);
                sprintf(tampil,"%4d %4d
%4d",(int)lebar_pulsa4,(int)lebar_pulsa5,(int)le
bar_pulsa6);
                lcd(0,2,tampil);
                sprintf(tampil,"%4d
%4d",(int)lebar_pulsa7,(int)lebar_pulsa8);
                lcd(0,3,tampil);
```

```
                break;
        case 6:
                sprintf(tampil,"7.Test Data PC");
                lcd(0,0,tampil);
                sprintf(tampil,"Usart 1 :
%d",(int)data_usart1);
                lcd(0,1,tampil);
                sprintf(tampil,"Count : %d",
(int)hitung_usart1);
                lcd(0,2,tampil);
                sprintf(tampil,"Moly");
                lcd(0,3,tampil);
                break;
        case 7:

    sprintf(tampil,"8.GPS1%2.7lf",lattDecimal2Mur
niSebelum);
                lcd(0,0,tampil);
                sprintf(tampil,"
%2.7lf",lattDecimal2);
                lcd(0,1,tampil);
                sprintf(tampil,"
GPS2%2.7lf",latt_base_murni_sebelum);
                lcd(0,2,tampil);
                sprintf(tampil,"
%2.7lf",latt_base);
                lcd(0,3,tampil);
                break;
        case 8:

    sprintf(tampil,"9.GPS1%3.7lf",longDecimal2Mur
niSebelum);
                lcd(0,0,tampil);
                sprintf(tampil,"
%3.7lf",longDecimal2);
                lcd(0,1,tampil);
                sprintf(tampil,"
GPS2%3.7lf",long_base_murni_sebelum);
                lcd(0,2,tampil);
```

```
                sprintf(tampil,"
%3.7lf",long_base);
                lcd(0,3,tampil);
                break;
        case 9:
                sprintf(tampil,"10. Data
Arduino");
                lcd(0,0,tampil);
                sprintf(tampil,"lat :
%4.5f",lattitude);
                lcd(0,1,tampil);
                sprintf(tampil,"long:
%5.5f",longitude);
                lcd(0,2,tampil);
                sprintf(tampil,"bluetooth : %4d",
(int)bluetooth);
                lcd(0,3,tampil);
                break;
        case 10:
                sprintf(tampil,"11. Test Sensor");
                lcd(0,0,tampil);
                sprintf(tampil,"Sudut   :
%4.1f",sudut16);
                lcd(0,1,tampil);
                sprintf(tampil,"Heading :
%4.1f",heading);
                lcd(0,2,tampil);

    sprintf(tampil,"Err:%4.1fMot:%4.1f",error_hea
ding,pid_motor);
                lcd(0,3,tampil);
                break;
        case 11 :
                sprintf(tampil,"Latt :
%3.7f",errorLatt1);
                lcd(0,0,tampil);
                sprintf(tampil,"LatB :
%3.7f",error_latt_base);
                lcd(0,1,tampil);
```

```
                sprintf(tampil,"Long :
%3.7f",errorLong1);
                lcd(0,2,tampil);
                sprintf(tampil,"LonB :
%3.7f",error_long_base);
                lcd(0,3,tampil);
                break;

    }

}

void servo_motor()
{
    out7 = Servo1;
    out8 = Servo2;
    out5 = Motor1;
    out6 = Motor2;
    out2 = Motor3;
    out1 = Motor4;
    out3 = Servo3;
    out4 = Servo4;
}

void reset_data()
{
    Motor1 = 1500;
    Motor2 = 1500;
    Motor3 = 1500;
    Motor4 = 1500;
    Servo1 = 1500;
    Servo2 = 1500;
    Servo3 = 1500;
    Servo4 = 1500;
}

int main(void)
{
    //Semua Inisiasi tulis disini
```

```
    SystemInit();

    RCC_APB2PeriphClockCmd(RCC_APB2Periph_SYSCFG,
ENABLE);   //enable interrupt
    RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOA,
ENABLE);
    RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOB,
ENABLE);
    RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOC,
ENABLE);
    RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOD,
ENABLE);
    RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOE,
ENABLE);
    RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOH,
ENABLE);
    RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOG,
ENABLE);

    inisialisasi_timer6();
    inisialisasi_io();

    lcd_init(20);

    reset_data();

    pwm_motor();
    pwm_remot();

    inisialisasi_USART2();
    inisialisasi_USART1();
    inisialisasi_USART3();
    inisialisasi_UART4();

    out1=1500;
    out2=1500;
    out3=1500;
    out4=1500;
    out5=1500;
    out6=1500;
```

```
    out7=1500;
    out8=1500;

    TIM8->CNT=0;
    TIM1->CNT=0;

    while(1)
    {

        konvertStringGps(); //olah data GPS 1
        konvertStringGps2(); //olah data GPS 2




        //gabungin 2 data GPS

        minta_data_kompas(19);
        angle16=((data_kompas_serial[0]<<8) +
data_kompas_serial[1]);
        sudut16=angle16/10.0;

        baca_kondisi_remot();

        if(tombol_lcd == 1  || lebar_pulsa6 >
3200)//ganti tampilan LCD
        {
            lcd_clear();
            counter_lcd++;
            if(counter_lcd>11)
            {
                counter_lcd = 0;
            }
            delay_ms(600);
        }

        switch(state_remot)
        {
            case 0:               //manual
                test_motor();
```

```
                GPIO_SetBits(GPIOE, GPIO_Pin_5);
                GPIO_ResetBits(GPIOE, GPIO_Pin_4);
                break;
            case 1:
                test_servo();
                GPIO_SetBits(GPIOE, GPIO_Pin_5);
                GPIO_ResetBits(GPIOE, GPIO_Pin_4);
                break;
            case 2:
                olah_pwm_remot();
                //reset_data();
                GPIO_SetBits(GPIOE, GPIO_Pin_5);
                GPIO_ResetBits(GPIOE, GPIO_Pin_4);
                break;
            case 3:                 //otomatis
                servo_motor();
                GPIO_SetBits(GPIOE, GPIO_Pin_4);
                GPIO_ResetBits(GPIOE, GPIO_Pin_5);
                break;
        }

        monitoringData(counter_lcd);
//menampilkan data ke LCD
        kirim_ke_pc();
    }
}
```

LAMPIRAN D

Program *Graphical User Interface* pada QT Designer untuk pengujian GPS dan DGPS

```
#include "mainwindow.h"

using namespace std;
SerialConnection *serialConnection;
KontrolKapal *kontrol;

unsigned int titik_awal_lat = 72862585;
unsigned int titik_akhir_lat = 72872555;
unsigned int titik_awal_long = 1127957235;
unsigned int titik_akhir_long = 1127964815;

void MainWindow::initConnection() {

	// kontrol manual
	connect(ui->pushButtonMajuKiri, SIGNAL(pressed()), kontrol,
SLOT(kontrolManualMajuKiri()), Qt::DirectConnection);
	connect(ui->pushButtonMaju, SIGNAL(pressed()), kontrol,
SLOT(kontrolManualMaju()), Qt::DirectConnection);
	connect(ui->pushButtonMajuKanan, SIGNAL(pressed()), kontrol,
SLOT(kontrolManualMajuKanan()), Qt::DirectConnection);
	connect(ui->pushButtonKanan, SIGNAL(pressed()), kontrol,
SLOT(kontrolManualKanan()), Qt::DirectConnection);
	connect(ui->pushButtonMundurKanan, SIGNAL(pressed()),
kontrol, SLOT(kontrolManualMundurKanan()), Qt::DirectConnection);
	connect(ui->pushButtonMundur, SIGNAL(pressed()), kontrol,
SLOT(kontrolManualMundur()), Qt::DirectConnection);
	connect(ui->pushButtonMundurKiri, SIGNAL(pressed()), kontrol,
SLOT(kontrolManualMundurKiri()), Qt::DirectConnection);
	connect(ui->pushButtonKiri, SIGNAL(pressed()), kontrol,
SLOT(kontrolManualKiri()), Qt::DirectConnection);
	connect(ui->pushButtonRotateLeft, SIGNAL(pressed()), kontrol,
SLOT(kontrolManualRotateLeft()), Qt::DirectConnection);
	connect(ui->pushButtonRotateRight, SIGNAL(pressed()), kontrol,
SLOT(kontrolManualRotateRight()), Qt::DirectConnection);
```

```
        connect(ui->pushButtonMajuKiri, SIGNAL(released()), kontrol,
SLOT(kontrolManualMajuKiriRelease()), Qt::DirectConnection);
        connect(ui->pushButtonMaju, SIGNAL(released()), kontrol,
SLOT(kontrolManualMajuRelease()), Qt::DirectConnection);
        connect(ui->pushButtonMajuKanan, SIGNAL(released()), kontrol,
SLOT(kontrolManualMajuKananRelease()), Qt::DirectConnection);
        connect(ui->pushButtonKanan, SIGNAL(released()), kontrol,
SLOT(kontrolManualKananRelease()), Qt::DirectConnection);
        connect(ui->pushButtonMundurKanan, SIGNAL(released()),
kontrol, SLOT(kontrolManualMundurKananRelease()),
Qt::DirectConnection);
        connect(ui->pushButtonMundur, SIGNAL(released()), kontrol,
SLOT(kontrolManualMundurRelease()), Qt::DirectConnection);
        connect(ui->pushButtonMundurKiri, SIGNAL(released()), kontrol,
SLOT(kontrolManualMundurKiriRelease()), Qt::DirectConnection);
        connect(ui->pushButtonKiri, SIGNAL(released()), kontrol,
SLOT(kontrolManualKiriRelease()), Qt::DirectConnection);
        connect(ui->pushButtonRotateLeft, SIGNAL(released()), kontrol,
SLOT(kontrolManualRotateLeftRelease()), Qt::DirectConnection);
        connect(ui->pushButtonRotateRight, SIGNAL(released()), kontrol,
SLOT(kontrolManualRotateRightRelease()), Qt::DirectConnection);

        connect(ui->spinBoxMotorManual, SIGNAL(valueChanged(int)),
kontrol, SLOT(ubahMotorManual(int)), Qt::DirectConnection);

        // mode test
        connect(ui->spinBoxTestMotor1, SIGNAL(valueChanged(int)),
kontrol, SLOT(ubahMotor1TestMode(int)), Qt::DirectConnection);
        connect(ui->spinBoxTestMotor2, SIGNAL(valueChanged(int)),
kontrol, SLOT(ubahMotor2TestMode(int)), Qt::DirectConnection);
        connect(ui->spinBoxTestMotor3, SIGNAL(valueChanged(int)),
kontrol, SLOT(ubahMotor3TestMode(int)), Qt::DirectConnection);
        connect(ui->spinBoxTestMotor4, SIGNAL(valueChanged(int)),
kontrol, SLOT(ubahMotor4TestMode(int)), Qt::DirectConnection);
        connect(ui->spinBoxTestServo1, SIGNAL(valueChanged(int)),
kontrol, SLOT(ubahServo1TestMode(int)), Qt::DirectConnection);
        connect(ui->spinBoxTestServo2, SIGNAL(valueChanged(int)),
kontrol, SLOT(ubahServo2TestMode(int)), Qt::DirectConnection);
```

```
    connect(ui->spinBoxTestServo3, SIGNAL(valueChanged(int)),
kontrol, SLOT(ubahServo3TestMode(int)), Qt::DirectConnection);
    connect(ui->spinBoxTestServo4, SIGNAL(valueChanged(int)),
kontrol, SLOT(ubahServo4TestMode(int)), Qt::DirectConnection);

    //PID Kontrol
    connect(ui->doubleSpinBoxPhead,
SIGNAL(valueChanged(double)), kontrol, SLOT(setPhead(double)),
Qt::DirectConnection);
    connect(ui->doubleSpinBoxIhead,
SIGNAL(valueChanged(double)), kontrol, SLOT(setIhead(double)),
Qt::DirectConnection);
    connect(ui->doubleSpinBoxDhead,
SIGNAL(valueChanged(double)), kontrol, SLOT(setDhead(double)),
Qt::DirectConnection);
    connect(ui->doubleSpinBoxPheads,
SIGNAL(valueChanged(double)), kontrol, SLOT(setPheads(double)),
Qt::DirectConnection);
    connect(ui->doubleSpinBoxIheads,
SIGNAL(valueChanged(double)), kontrol, SLOT(setIheads(double)),
Qt::DirectConnection);
    connect(ui->doubleSpinBoxDheads,
SIGNAL(valueChanged(double)), kontrol, SLOT(setDheads(double)),
Qt::DirectConnection);
    connect(ui->doubleSpinBoxPjarak,
SIGNAL(valueChanged(double)), kontrol, SLOT(setPjarak(double)),
Qt::DirectConnection);
    connect(ui->doubleSpinBoxIjarak,
SIGNAL(valueChanged(double)), kontrol, SLOT(setIjarak(double)),
Qt::DirectConnection);
    connect(ui->doubleSpinBoxDjarak,
SIGNAL(valueChanged(double)), kontrol, SLOT(setDjarak(double)),
Qt::DirectConnection);

    // Ubah Mode
    connect(ui->comboBoxMode,
SIGNAL(currentIndexChanged(int)), kontrol, SLOT(ubahMode(int)),
Qt::DirectConnection);
    //Set
```

```
        connect(ui->pushButtonSetBase, SIGNAL(clicked(bool)), this,
SLOT(setBase()));
        connect(ui->doubleSpinBoxSetHeadAuto,
SIGNAL(valueChanged(double)), kontrol,
SLOT(setHeadAuto(double)));

        // save and load hsv connection
        connect(ui->actionSave, SIGNAL(triggered(bool)), this,
SLOT(on_save_new()));
        connect(ui->actionSave_As, SIGNAL(triggered(bool)), this,
SLOT(on_save_as_new()));
        connect(ui->actionOpen, SIGNAL(triggered(bool)), this,
SLOT(on_load_new()));
        connect(this, SIGNAL(updateHSV()), this, SLOT(loadHSV()));

        // camera thread connection
        connect(camThread, SIGNAL(started()), cam,
SLOT(openCamera()));
        connect(ui->startButton, SIGNAL(clicked(bool)), this,
SLOT(startCam()));
        connect(ui->stopButton, SIGNAL(clicked(bool)), this,
SLOT(stopCam()));
        connect(cam, SIGNAL(sendImage()), this, SLOT(updateLabel()),
Qt::DirectConnection);
        connect(this, SIGNAL(sendHSV(int, int, int, int, int, int, int, int,
int)), cam, SLOT(receiveHSV(int, int, int, int, int, int, int, int, int)),
Qt::DirectConnection);
        connect(this, SIGNAL(currentColorId(int)), cam,
SLOT(reciveColorId(int)), Qt::DirectConnection);
        connect(ui->labelOri, SIGNAL(clickedPos(QPoint)), cam,
SLOT(reciveInitialPos(QPoint)), Qt::DirectConnection);
        connect(ui->labelOri, SIGNAL(currentPos(QPoint)), cam,
SLOT(reciveMousePos(QPoint)), Qt::DirectConnection);
        connect(ui->labelOri, SIGNAL(endPos(QPoint)), cam,
SLOT(thresholding(QPoint)), Qt::DirectConnection);

        // serial connection
        connect(ui->pushButtonOpenPort, SIGNAL(clicked(bool)), this,
SLOT(startSerial()));
```

```
        connect(ui->pushButtonClosePort, SIGNAL(clicked(bool)), this,
SLOT(stopSerial()));
        connect(serialThread, SIGNAL(started()), serialConnection,
SLOT(initSerPortGPS())); // mulai jalanin thread koneksi serial

        connect(serialConnection,
SIGNAL(serial_kirim_data_GPS(double, double, float, double,
double)), this, SLOT(showDataGPS(double, double, float, double,
double)));
        connect(ui->pushButtonClosePort, SIGNAL(clicked(bool)),
serialConnection, SLOT(close()));

        connect(kontrol, SIGNAL(kirim_ke_serial(int, int, int, int, int, int,
int, int)), serialConnection, SLOT(terima_kontrol(int, int, int, int, int,
int, int, int)), Qt::DirectConnection);

        //klikmapping
        connect(ui->labelMapping,
SIGNAL(sendMousePosition(QPoint&)), this,
SLOT(showMousePosition(QPoint&)));
        connect(ui->labelMapping, SIGNAL(klik_kiri()), this,
SLOT(left_clicked()));
        connect(ui->labelMapping, SIGNAL(klik_kanan()), this,
SLOT(right_clicked()));

        // Waypoint
        connect(ui->spinBoxWaypoint, SIGNAL(valueChanged(int)), this,
SLOT(setIndexWaypoint(int)));
        connect(ui->pushButtonSetWaypoint, SIGNAL(clicked(bool)),
this, SLOT(setWaypoint()));
        connect(ui->pushButtonStartMission, SIGNAL(clicked(bool)),
this, SLOT(startplot()));
        connect(ui->pushButtonStartMission, SIGNAL(clicked(bool)),
kontrol, SLOT(mulai_misi()));
        connect(ui->pushButtonStopMission, SIGNAL(clicked(bool)),
this, SLOT(stopplot()));
        connect(ui->pushButtonStopMission, SIGNAL(clicked(bool)),
kontrol, SLOT(stop_misi()));
        connect(ui->pushButtonReset, SIGNAL(clicked(bool)), this,
```

```
SLOT(reset()));
    connect(ui->pushButtonSaveAsWaypoint, SIGNAL(clicked(bool)),
SLOT(on_save_waypoint()));
    connect(ui->pushButtonLoadWaypoint, SIGNAL(clicked(bool)),
SLOT(on_load_waypoint()));
    connect(ui->pushButtonRetry, SIGNAL(clicked(bool)), this,
SLOT(retry()), Qt::DirectConnection);
//    connect(ui->pushButtonRetry2, SIGNAL(clicked(bool)), this,
SLOT(retrytomisi2()), Qt::DirectConnection);
    connect(this, SIGNAL(reset_kontrol()), kontrol,
SLOT(retry_kontrol()), Qt::DirectConnection);
    connect(this, SIGNAL(reset_serialConnection()), serialConnection,
SLOT(retry_serialConnection()), Qt::DirectConnection);
    connect(ui->pushButtonInputWaypoint, SIGNAL(clicked(bool)),
this, SLOT(input_waypoint()));
    connect(ui->pushButtonNorth, SIGNAL(clicked(bool)), this,
SLOT(all_waypoint_geser_atas()));
    connect(ui->pushButtonWest, SIGNAL(clicked(bool)), this,
SLOT(all_waypoint_geser_kiri()));
    connect(ui->pushButtonEast, SIGNAL(clicked(bool)), this,
SLOT(all_waypoint_geser_kanan()));
    connect(ui->pushButtonSouth, SIGNAL(clicked(bool)), this,
SLOT(all_waypoint_geser_bawah()));

    connect(ui->spinBoxOffset, SIGNAL(valueChanged(int)), this,
SLOT(ubah_offset())); //check
    connect(this, SIGNAL(ubah_offset_serial(int)), serialConnection,
SLOT(terima_offset_serial(int)));
    connect(ui->doubleSpinBoxRadiusNavigasi,
SIGNAL(valueChanged(double)), this, SLOT(ubah_radius_navigasi()));
//    connect(ui->doubleSpinBoxRadiusMisi4,
SIGNAL(valueChanged(double)), this, SLOT(ubah_radius_misi4()));

    //connect(ui->pushButtonSetFrame, SIGNAL(clicked(bool)), this,
SLOT(set_frame()));
    connect(ui->comboBoxSetFrame,
SIGNAL(currentIndexChanged(int)), this, SLOT(set_frame_2()));
```

```
        // change pid
        connect(ui->spinBoxP1, SIGNAL(valueChanged(int)), this,
SLOT(pidChanged()));
        connect(ui->spinBoxI1, SIGNAL(valueChanged(int)), this,
SLOT(pidChanged()));
        connect(ui->spinBoxD1, SIGNAL(valueChanged(int)), this,
SLOT(pidChanged()));
        connect(this, SIGNAL(changePID(float, float, float)), cam,
SLOT(recivePID(float, float, float)), Qt::DirectConnection);
        connect(ui->spinBoxP2, SIGNAL(valueChanged(int)), this,
SLOT(pidChanged_2()));
        connect(ui->spinBoxI2, SIGNAL(valueChanged(int)), this,
SLOT(pidChanged_2()));
        connect(ui->spinBoxD2, SIGNAL(valueChanged(int)), this,
SLOT(pidChanged_2()));
        connect(this, SIGNAL(changePID_2(float, float, float)), cam,
SLOT(recivePID_2(float, float, float)), Qt::DirectConnection);
        connect(ui->spinBoxP3, SIGNAL(valueChanged(int)), this,
SLOT(pidChanged_3()));
        connect(ui->spinBoxI3, SIGNAL(valueChanged(int)), this,
SLOT(pidChanged_3()));
        connect(ui->spinBoxD3, SIGNAL(valueChanged(int)), this,
SLOT(pidChanged_3()));
        connect(this, SIGNAL(changePID_3(float, float, float)), kontrol,
SLOT(ubah_PID_misi3(float, float, float)), Qt::DirectConnection);

        connect(ui->spinBoxBatasY, SIGNAL(valueChanged(int)), cam,
SLOT(ubah_batasY(int)), Qt::DirectConnection);

        connect(droneConnection, SIGNAL(imageRecieved(QImage)),
this, SLOT(imageDrone(QImage)), Qt::DirectConnection);
        connect(droneConnection, SIGNAL(imageRecieved(QImage)),
cam, SLOT(predictImageDrone(QImage)), Qt::DirectConnection);
        connect(serialConnection, SIGNAL(drone_status_changed(int)),
this, SLOT(takePictOnWaypoint(int)));
        connect(this, SIGNAL(imageDroneRecieved()), droneConnection,
SLOT(doDisconnect()));
        connect(ui->pushButtonTakePict, SIGNAL(clicked(bool)), this,
SLOT(connectDrone()));
```

```
    connect(droneConnection, SIGNAL(imageRecieved(QImage)),
net, SLOT(sendImage(QImage)), Qt::DirectConnection); // koneksi buat
testing dan ambil data

    //kontrol
    connect(this, SIGNAL(kontrol_waypoint(double, double, int)),
kontrol, SLOT(tujuanWaypoint(double, double, int)));
    connect(serialConnection,
SIGNAL(serial_kirim_data_GPS(double, double, float, double,
double)), kontrol, SLOT(ambilDataKontrolGPS(double, double, float,
double, double)));
    connect(this, SIGNAL(ubah_kontrol_PID(double, double,
double)), kontrol, SLOT(ubah_PID(double, double, double)));
    connect(this, SIGNAL(send_index_plot_waypoint(int)), kontrol,
SLOT(terima_index(int)),Qt::DirectConnection);
    connect(kontrol, SIGNAL(kirim_data_kontrol(int, int, int, int, int,
int, int, int, float, float, double, float, double, double, float, double,
double)), this, SLOT(ambil_olah_kontrol(int, int, int, int, int, int, int, int,
float, float, double, float, double, double, float, double, double)),
Qt::DirectConnection);
    connect(serialConnection,
SIGNAL(serial_kirim_data_GPS(double, double, float, double,
double)), kontrol, SLOT(kontrol_waypoint()));

    connect(kontrol, SIGNAL(change_servo_camera(int, int)), this,
SLOT(ubah_posisi_camera_misi(int, int)));

    //timer mapping
    QTimer *timer_mapping = new QTimer(this);
    connect(timer_mapping, SIGNAL(timeout()), this,
SLOT(paintEvent()));
    timer_mapping->start(100);
    QTimer *timer_gps_plot = new QTimer(this);
    connect(timer_gps_plot, SIGNAL(timeout()), this,
SLOT(setWaypoint()));
    timer_gps_plot->start(2000);

    // server comunication
    connect(ui->btnStartConnServer, SIGNAL(clicked(bool)), this,
```

```
SLOT(startNet()));
	connect(ui->btnStopConnServer, SIGNAL(clicked(bool)), net,
SLOT(stop()));
	connect(ui->btnStopConnServer, SIGNAL(clicked(bool)), this,
SLOT(stopNet()));
	connect(netThread, SIGNAL(started()), net, SLOT(start()));
	connect(kontrol, SIGNAL(mission_changed(int)), net,
SLOT(setCurrentMission(int)), Qt::DirectConnection);
	connect(serialConnection,
SIGNAL(serial_kirim_data_GPS(double, double, float,double, double)),
net, SLOT(updateGPS(double, double, float, double, double)),
Qt::DirectConnection);
	connect(droneConnection, SIGNAL(imageRecieved(QImage)),
net, SLOT(setImage(QImage)), Qt::DirectConnection);
}

void MainWindow::init()
{
	//skala area
	skala_latitude = (float)800 / (titik_akhir_lat - titik_awal_lat); //
ngga pake minus 1 kalo di indonesia
	skala_longitude = (float)600 / (titik_akhir_long - titik_awal_long);
// ngga pake minus 1 kalo di indonesia

	waypen_now.setWidth(0);
	//waypen_now.setWidthF(0.5);
	waypen_now.setColor(Qt::black);
	waypen_now.setJoinStyle(Qt::RoundJoin);

	waypen_garis.setWidth(2);
	waypen_garis.setColor(Qt::yellow);
	waypen_garis.setJoinStyle(Qt::RoundJoin);



	// init all variable and all connection
	for (int i = 0; i < 7; i++) {
		hMin[i] = 0;
		sMin[i] = 0;
```

```
            vMin[i] = 0;
            hMax[i] = 255;
            vMax[i] = 255;
            sMax[i] = 255;
            erode[i] = 0;
            dilate[i] = 0;
        }

        kontrol = new KontrolKapal();
        // init thread serial connection
        serialThread = new QThread();
        serialConnection = new SerialConnection();
        serialConnection->moveToThread(serialThread);

        //init thread mapping

        // init thread drone connection
        // sementara belum bentuk tread
        droneConnection = new DroneConnection();


        // init thread communication
        net = new Networking();
        netThread = new QThread();
        net->moveToThread(netThread);

        // init all qt connection
        initConnection();
        // set view in stacked widget
        ui->stackedWidget->setCurrentIndex(0);
        emit currentColorId(0);
        loadStatus = false;
}

MainWindow::MainWindow(QWidget *parent)
        : QMainWindow(parent), ui(new Ui::MainWindow)
{
        ui->setupUi(this);
        init();
```

```
}

MainWindow::~MainWindow()
{
	stopCam();
	stopSerial();
	delete camThread;
	delete cam;
	delete ui;
	delete serialThread;
	delete serialConnection;
}


void MainWindow::startSerial() {
	// handling is running condition
	if (!serialThread->isRunning()) serialThread->start();
	//if (!kontrolThread->isRunning()) kontrolThread->start();
	ui->pushButtonOpenPort->setEnabled(false);
	ui->pushButtonClosePort->setEnabled(true);
}

void MainWindow::stopSerial() {
	serialThread->wait(100);
	if (serialThread->isRunning()) serialThread->quit();
	//if (kontrolThread->isRunning()) kontrolThread->quit();
	ui->pushButtonOpenPort->setEnabled(true);
	ui->pushButtonClosePort->setEnabled(false);
}

float radius_navigasi = 1.8;

void MainWindow::ubah_radius_navigasi()
{
	radius_navigasi = ui->doubleSpinBoxRadiusNavigasi->value();
}

void MainWindow::ubah_offset()
{
```

```
	offset = ui->spinBoxOffset->value();
	emit ubah_offset_serial(offset);
}
float kfx = 0.0;
float kfy = 0.0;

void MainWindow::showDataGPS(double latt, double lon, float comp,
double la_base, double lo_base)
{
	if (status_waypoint)
	{
		if (jarak < radius_navigasi && index_plot_waypoint !=
max_waypoint)
		{
			index_plot_waypoint++;
			emit
send_index_plot_waypoint(index_plot_waypoint);
		}
	}
	latitude_sekarang = latt;
	longitude_sekarang = lon;
	latitude_base_sekarang = la_base;
	longitude_base_sekarang = lo_base;


	latitude_combo = (double)((9 * latitude_sekarang + 7 *
(latitude_base_sekarang - BaseLatt + referenceLatt)) / 16) -
(latitude_base_sekarang - BaseLatt) / 5;
	longitude_combo = (double)((9 * longitude_sekarang + 7 *
(longitude_base_sekarang - BaseLong + referenceLong)) / 16) -
(longitude_base_sekarang - BaseLong) / 5;

	compass_sekarang = comp;

	ui->lineEditLattitude->setText(QString::number(latt, 'g', 10));
	ui->lineEditLongitude->setText(QString::number(lon, 'g', 10));
	ui->lineEditLatBase->setText(QString::number(la_base, 'g', 10));
	ui->lineEditLongBase->setText(QString::number(lo_base, 'g', 10));
	ui->lineEditCompass->setText(QString::number(comp));
```

```
    ui->lineEditTujuanLat->setText(QString::number(lattitudeWaypoin
t[index_plot_waypoint], 'g', 10));
    ui->lineEditTujuanLong->setText(QString::number(longitudeWayp
oint[index_plot_waypoint], 'g', 10));

    ui->lcdNumberWaypoint->display(index_plot_waypoint);
}


void MainWindow::left_clicked()
{
    //posisi_kapal_sekarang.setY((abs(lattitudeWaypoint[index_plot_w
aypoint]) * 10000000 - titik_awal_lat) * skala_latitude);
    int s[2];
    double ss[2];
    s[0] = ui->lineEditYposition->text().toInt();
    ss[0] = ((s[0] / skala_latitude) + titik_awal_lat) / (-10000000);
    s[1] = ui->lineEditXposition->text().toInt();
    ss[1] = ((s[1] / skala_longitude) + titik_awal_long) / (10000000);
    latitude[index_waypoint] = QString::number(ss[0]);
    longitude[index_waypoint] = QString::number(ss[1]);

    QTableWidgetItem* latt = new
QTableWidgetItem(latitude[index_waypoint]);
    QTableWidgetItem* longi = new
QTableWidgetItem(longitude[index_waypoint]);

    ui->tableWidgetWaypoint->setItem(index_waypoint - 1, 0, latt);
    ui->tableWidgetWaypoint->setItem(index_waypoint - 1, 1, longi);

    lattitudeWaypoint[index_waypoint] = ss[0];
    longitudeWaypoint[index_waypoint] = ss[1];

    emit kontrol_waypoint(lattitudeWaypoint[index_waypoint],
longitudeWaypoint[index_waypoint], index_waypoint);
}

void MainWindow::right_clicked()
```

```
{
}


void MainWindow::setIndexWaypoint(int index)
{
	index_waypoint = index;
	if (index_waypoint > max_waypoint)
	{
		max_waypoint = index_waypoint;
	}
}


void MainWindow::setWaypoint()
{
	latitude[index_waypoint] = QString::number(latitude_sekarang, 'g',
7);
	longitude[index_waypoint] =
QString::number(longitude_sekarang, 'g', 7);

	QTableWidgetItem* latt = new
QTableWidgetItem(latitude[index_waypoint]);
	QTableWidgetItem* longi = new
QTableWidgetItem(longitude[index_waypoint]);

	ui->tableWidgetWaypoint->setItem(index_waypoint - 1, 0, latt);
	ui->tableWidgetWaypoint->setItem(index_waypoint - 1, 1, longi);

	lattitudeWaypoint[index_waypoint] =
ui->lineEditLattitude->text().toDouble();
	longitudeWaypoint[index_waypoint] =
ui->lineEditLongitude->text().toDouble();
	lattitudeWaypoint1[index_waypoint] = latitude_combo;
	longitudeWaypoint1[index_waypoint] = longitude_combo;

	emit kontrol_waypoint(lattitudeWaypoint[index_waypoint],
longitudeWaypoint[index_waypoint], index_waypoint);
```

```
    ui->spinBoxWaypoint->setValue(index_waypoint + 1);
}

void MainWindow::input_waypoint()
{
    lattitudeWaypoint[index_waypoint] =
ui->lineEditInputLatt->text().toDouble();
    longitudeWaypoint[index_waypoint] =
ui->lineEditInputLong->text().toDouble();

    latitude[index_waypoint] =
QString::number(lattitudeWaypoint[index_waypoint], 'g', 7);
    longitude[index_waypoint] =
QString::number(longitudeWaypoint[index_waypoint], 'g', 7);

    QTableWidgetItem* latt = new
QTableWidgetItem(latitude[index_waypoint]);
    QTableWidgetItem* longi = new
QTableWidgetItem(longitude[index_waypoint]);

    ui->tableWidgetWaypoint->setItem(index_waypoint - 1, 0, latt);
    ui->tableWidgetWaypoint->setItem(index_waypoint - 1, 1, longi);

    emit kontrol_waypoint(lattitudeWaypoint[index_waypoint],
longitudeWaypoint[index_waypoint], index_waypoint);

}

QPoint posisi_kapal_sekarang;
QPoint posisi_tujuan_kapal;
QPoint posisi_waypoint;

float latitudePlot;
float longitudePlot;

void MainWindow::startplot()
{
    status_waypoint = true;
    ui->pushButtonStopMission->setEnabled(true);
```

```
	ui->pushButtonStartMission->setEnabled(false);
}

void MainWindow::stopplot()
{
	status_waypoint = false;
	ui->pushButtonStopMission->setEnabled(false);
	ui->pushButtonStartMission->setEnabled(true);
	index_plot_waypoint = 1;
	emit send_index_plot_waypoint(index_plot_waypoint);
}

void MainWindow::reset()
{
	memset(lattitudeWaypoint, NULLPOINT,
sizeof(lattitudeWaypoint));
	memset(longitudeWaypoint, NULLPOINT,
sizeof(longitudeWaypoint));
	ui->tableWidgetWaypoint->clear();
	ui->spinBoxWaypoint->setValue(0);
	index_plot_waypoint = 1;
	index_waypoint = 1;
	max_waypoint = 1;

	emit send_index_plot_waypoint(index_plot_waypoint);
}

void MainWindow::retry() {
	index_plot_waypoint = 1;
	emit send_index_plot_waypoint(index_plot_waypoint);

	emit reset_kontrol();
	emit reset_serialConnection();
}


QApplication app();
QPainter p;
```

```
void MainWindow::ambil_olah_kontrol(int mot1, int mot2, int mot3, int
mot4, int serv1, int serv2, int serv3, int serv4, float sudut, float eror,
double distance, float head, double delX, double delY, float errhead,
double delxn, double delyn) {
//      ui->lineEditServoKiri->setText(QString::number(serv1));
//      ui->lineEditServoKanan->setText(QString::number(serv2));
        ui->lineEditSudutTujuan->setText(QString::number(sudut));
        ui->lineEditErrorSudut->setText(QString::number(eror));
        ui->lineEditJarak->setText(QString::number(distance));
        ui->lineEditHeading->setText(QString::number(head));
        ui->lineEditOutMotor1->setText(QString::number(mot1));
        ui->lineEditOutMotor2->setText(QString::number(mot2));
        ui->lineEditOutMotor3->setText(QString::number(mot3));
        ui->lineEditOutMotor4->setText(QString::number(mot4));
        ui->lineEditOutServo1->setText(QString::number(serv1));
        ui->lineEditOutServo2->setText(QString::number(serv2));
        ui->lineEditOutServo3->setText(QString::number(serv3));
        ui->lineEditOutServo4->setText(QString::number(serv4));
        ui->lineEditDeltaX->setText(QString::number(delX));
        ui->lineEditDeltaY->setText(QString::number(delY));
        ui->lineEditDeltaXN->setText(QString::number(delxn));
        ui->lineEditDeltaYN->setText(QString::number(delyn));
        ui->lineEditErrorHeading->setText(QString::number(errhead));
//      ui->lcdNumberPenandaMisi3->display(kontrol->penanda_misi3);
//      ui->lineEditDock->setText(QString::number(kontrol->first_dock));

        jarak = distance;
        deltaX = delX;
        deltaY = delY;
}

void MainWindow::setBase()
{
        qDebug() << "Base";
        BaseLatt = latitude_base_sekarang;
        BaseLong = longitude_base_sekarang;

        referenceLatt = latitude_sekarang;
```

```
        referenceLong = longitude_sekarang;

        base = true;

        titik_awal_lat = abs(referenceLatt) * 10000000 - 1000;
        titik_akhir_lat = abs(referenceLatt) * 10000000 + 1000;
        titik_awal_long = abs(referenceLong) * 10000000 - 750;
        titik_akhir_long = abs(referenceLong) * 10000000 + 750;

        skala_latitude = (float)800 / (titik_akhir_lat - titik_awal_lat); //
ngga pake minus 1 kalo di indonesia
        skala_longitude = (float)600 / (titik_akhir_long - titik_awal_long);
// ngga pake minus 1 kalo di indonesia

        ui->lineEditBaseLatt->setText(QString::number(BaseLatt, 'g', 10));
        ui->lineEditBaseLong->setText(QString::number(BaseLong, 'g',
10));
}


void MainWindow::set_frame()
{
        //titik_awal_lat = ui->lineEditLatAwal->text().toInt();
        //titik_akhir_lat = ui->lineEditLatAkhir->text().toInt();
        //titik_awal_long = ui->lineEditLongAwal->text().toInt();
        //titik_akhir_long = ui->lineEditLongAkhir->text().toInt();

        skala_latitude = (float)(-1) * 800 / (titik_awal_lat - titik_akhir_lat);
// ngga pake minus 1 kalo di indonesia
        skala_longitude = (float)(-1) * 600 / (titik_awal_long -
titik_akhir_long); // ngga pake minus 1 kalo di indonesia

}

void MainWindow::set_frame_2()
{
        qDebug() << "test frame";
        switch (ui->comboBoxSetFrame->currentIndex())
        {
```

```
        case 0 : // Danau 8
                titik_awal_lat =          72862585;
                titik_akhir_lat =         72872555;
                titik_awal_long =     1127957235;
                titik_akhir_long =    1127964815;

                skala_latitude = (float)800 / (titik_akhir_lat - titik_awal_lat);
// ngga pake minus 1 kalo di indonesia
                skala_longitude = (float)600 / (titik_akhir_long -
titik_awal_long); // ngga pake minus 1 kalo di indonesia
                break;
        case 1: // Danau Trial Daytona
                titik_awal_lat =          72821000;
                titik_akhir_lat =         72821800;
                titik_awal_long =     1127968200;
                titik_akhir_long =    1127968800;

                skala_latitude = (float)800 / (titik_akhir_lat - titik_awal_lat);
// ngga pake minus 1 kalo di indonesia
                skala_longitude = (float)600 / (titik_akhir_long -
titik_awal_long); // ngga pake minus 1 kalo di indonesia
                break;
        case 2: // Robotika
                titik_awal_lat =        72817025;
                titik_akhir_lat =       72827025;
                titik_awal_long =     1127964385;
                titik_akhir_long =    1127971885;

                skala_latitude = (float)800 / (titik_akhir_lat - titik_awal_lat);
// ngga pake minus 1 kalo di indonesia
                skala_longitude = (float)600 / (titik_akhir_long -
titik_awal_long); // ngga pake minus 1 kalo di indonesia
                break;
        case 3: // Reed Canal Park
                titik_awal_lat = 291520905;
                titik_akhir_lat = 291502665;
                titik_awal_long = 810173445;
                titik_akhir_long = 810157455;
```

```
            skala_latitude = (float)(-1) * 800 / (titik_awal_lat -
titik_akhir_lat); // ngga pake minus 1 kalo di indonesia
            skala_longitude = (float)(-1) * 600 / (titik_awal_long -
titik_akhir_long); // ngga pake minus 1 kalo di indonesia
            break;
        case 4: // Course A

            titik_awal_lat =        291518596;
            titik_akhir_lat =       291508596;
            titik_awal_long =       810179959;
            titik_akhir_long =      810169959;

            /*titik_awal_lat = ui->lineEditLatAwal->text().toInt();
            titik_akhir_lat = ui->lineEditLatAkhir->text().toInt();
            titik_awal_long = ui->lineEditLongAwal->text().toInt();
            titik_akhir_long = ui->lineEditLongAkhir->text().toInt();*/

            skala_latitude = (float)(-1) * 800 / (titik_awal_lat -
titik_akhir_lat); // ngga pake minus 1 kalo di indonesia
            skala_longitude = (float)(-1) * 600 / (titik_awal_long -
titik_akhir_long); // ngga pake minus 1 kalo di indonesia
            break;
        case 5: // Course B
            titik_awal_lat = 291524282;
            titik_akhir_lat = 291514282;
            titik_awal_long = 810168216;
            titik_akhir_long = 810158216;

            skala_latitude = (float)(-1) * 800 / (titik_awal_lat -
titik_akhir_lat); // ngga pake minus 1 kalo di indonesia
            skala_longitude = (float)(-1) * 600 / (titik_awal_long -
titik_akhir_long); // ngga pake minus 1 kalo di indonesia
            break;

        }
}

void MainWindow::paintEvent()
{
```

```
        // Original image
        QPixmap
rocket("C:/Users/lhlut/Desktop/AUVSI2018/KapalIllustration-01.png");

        // Original rotated once to 30 degrees
        QPixmap gambar(rocket.size());
        gambar.fill(Qt::transparent);
        p.begin(&gambar);
        p.translate(rocket.width() / 2, rocket.height() / 2);
        p.rotate(compass_sekarang);
        p.translate(-rocket.width() / 2, -rocket.height() / 2);
        p.drawPixmap(0, 0, rocket);
        p.end();

        ui->labelArahKapal->setPixmap(gambar);

        if (lattitudeWaypoint[index_plot_waypoint] != NULLPOINT &&
longitudeWaypoint[index_plot_waypoint] != NULLPOINT) {
                posisi_kapal_sekarang.setY((abs(latitude_combo) *
10000000 - titik_awal_lat) * skala_latitude);
                posisi_kapal_sekarang.setX((abs(longitude_combo) *
10000000 - titik_awal_long) * skala_longitude);


        posisi_tujuan_kapal.setY((abs(lattitudeWaypoint[index_plot_wayp
oint]) * 10000000 - titik_awal_lat) * skala_latitude);

        posisi_tujuan_kapal.setX((abs(longitudeWaypoint[index_plot_way
point]) * 10000000 - titik_awal_long) * skala_longitude);

                QPixmap pixmapWaypoint(600, 800);
                QPainter painterWaypoint(&pixmapWaypoint);
                pixmapWaypoint.fill(QColor(51, 51, 255, 255));
                painterWaypoint.setPen(Qt::black);
                // gambar grid
                for (int loops = 40; loops < 800; loops = loops + 40)
                {
                        painterWaypoint.drawLine(0, loops, 600, loops);
                }
```

```
            for (int loops = 40; loops < 600; loops = loops + 40)
            {
                painterWaypoint.drawLine(loops, 0, loops, 800);
            }
            QFont font = painterWaypoint.font();
            font.setPixelSize(16);
            painterWaypoint.setFont(font);

            painterWaypoint.setPen(waypen_garis);
            //painterWaypoint.drawLine(posisi_kapal_sekarang,
posisi_tujuan_kapal);

            painterWaypoint.setPen(waypen_now);

            painterWaypoint.setBrush((Qt::red));
            //painterWaypoint.drawEllipse(posisi_kapal_sekarang, 3, 3);

            //painterWaypoint.setPen(waypen_now);
            painterWaypoint.setBrush((Qt::red));
            painterWaypoint.drawEllipse(posisi_tujuan_kapal, 2, 2);
            painterWaypoint.drawText(posisi_tujuan_kapal,
QString::number(index_plot_waypoint));
            painterWaypoint.setBrush((Qt::green));
            for (int loop = 1; loop <= max_waypoint; loop++)
            {
                posisi_waypoint.setY((abs(lattitudeWaypoint[loop]) *
10000000 - titik_awal_lat) * skala_latitude); // dirubah
                posisi_waypoint.setX((abs(longitudeWaypoint[loop]) *
10000000 - titik_awal_long) * skala_longitude); //dirubah

                painterWaypoint.drawEllipse(posisi_waypoint, 3, 3);
                //painterWaypoint.drawText(posisi_waypoint,
QString::number(loop));
            }
            painterWaypoint.setBrush((Qt::red));
            for (int loop = 1; loop <= max_waypoint; loop++)
            {
                posisi_waypoint.setY((abs(lattitudeWaypoint1[loop]) *
10000000 - titik_awal_lat) * skala_latitude); // dirubah
```

```
                posisi_waypoint.setX((abs(longitudeWaypoint1[loop]) *
10000000 - titik_awal_long) * skala_longitude); //dirubah

                painterWaypoint.drawEllipse(posisi_waypoint, 3, 3);
                //painterWaypoint.drawText(posisi_waypoint,
QString::number(loop));
            }

            if (base)
            {
                QPoint posisi_base;
                painterWaypoint.setBrush((Qt::black));
                posisi_base.setY((abs(BaseLatt) * 10000000 -
titik_awal_lat) * skala_latitude);
                posisi_base.setX((abs(BaseLong) * 10000000 -
titik_awal_long) * skala_longitude);
                painterWaypoint.drawEllipse(posisi_base, 7, 7);

                painterWaypoint.setBrush((Qt::gray));
                posisi_base.setY((abs(referenceLatt) * 10000000 -
titik_awal_lat) * skala_latitude);
                posisi_base.setX((abs(referenceLong) * 10000000 -
titik_awal_long) * skala_longitude);
                painterWaypoint.drawEllipse(posisi_base, 6, 6);

                painterWaypoint.setBrush((Qt::yellow));
                posisi_base.setY((abs(latitude_base_sekarang) *
10000000 - titik_awal_lat) * skala_latitude);
                posisi_base.setX((abs(longitude_base_sekarang) *
10000000 - titik_awal_long) * skala_longitude);
                painterWaypoint.drawEllipse(posisi_base, 5, 5);
            }
            ui->labelMapping->setPixmap(pixmapWaypoint);
        }
        else {
            posisi_kapal_sekarang.setY((abs(latitude_sekarang) *
10000000 - titik_awal_lat) * skala_latitude);
            posisi_kapal_sekarang.setX((abs(longitude_sekarang) *
10000000 - titik_awal_long) * skala_longitude);
```

```
            QPixmap pixmapWaypoint(600, 800);
            QPainter painterWaypoint(&pixmapWaypoint);
            pixmapWaypoint.fill(QColor(51, 51, 255, 255));
            painterWaypoint.setPen(waypen_now);
            painterWaypoint.setBrush((Qt::red));
            painterWaypoint.drawEllipse(posisi_kapal_sekarang, 6, 6);

            ui->labelMapping->setPixmap(pixmapWaypoint);
        }
}
```

BIODATA PENULIS

Luthfi Halim lahir di Blitar pada tanggal 8 November 1998 yang merupakan anak kedua dari empat bersaudara. Penulis menyelesaikan pendidikan dasar di SDN Pajeleran 1, melanjutkan pendidikannya di SMPN 2 Cibinong, dan melanjutakan sekolah menengah atas di SMAN 3 Bogor. Penulis kemudian melanjutkan studi sarjana di Departemen Teknik Elektro, Fakultas Teknologi Elektro, Institut Teknologi Sepuluh Nopember Surabaya. Selama masa perrkuliahan penulis aktif di UKM Robotika ITS dan Tim Roboboat Barunastra ITS. Penulis berkesempatan menjadi Kepala Bagian Ristek dan Teknologi di UKM Robotika. Bersama Tim Barunastra ITS penulis mendapat berbagai juara antara lain, Juara 1 Kontes Kapal Cepat Tak Berawak Nasional 2016, Juara 4 International Roboboat Competition di Amerika pada 2017, Juara 1 sekaligus Best Design pada Deconbotion 2017 dan pada 2018 meraih *Champion* atau Juara 1 pada International Roboboat Competition di Amerika yang di gelar oleh Robonation. Penulis juga aktif sebagai asisten laboratorium Mikroelektronik dan Sistem Tertanam.

Email : lhluthfihalim@gmail.com
HP/WA : 081227381328