SISTEM NAVIGASI HELIKOPTER BERDASARKAN RINTANGAN

SKRIPSI

Oleh

DIMAS ARYO WICAKSONO

06 06 04 245 6

DEPARTEMEN TEKNIK ELEKTRO
FAKULTAS TEKNIK UNVERSITAS INDONESIA
GENAP 2007/2008

SISTEM NAVIGASI HELIKOPTER BERDASARKAN RINTANGAN

SKRIPSI

Oleh

DIMAS ARYO WICAKSONO
06 06 04 245 6

SKRIPSI INI DIAJUKAN UNTUK MELENGKAPI SEBAGIAN
PERSYARATAN MENJADI SARJANA TEKNIK

DEPARTEMEN TEKNIK ELEKTRO
FAKULTAS TEKNIK UNVERSITAS INDONESIA
GENAP 2007/2008

PERNYATAAN KEASLIAN SKRIPSI

Saya menyatakan dengan sesungguhnya bahwa Skripsi dengan judul:

SISTEM NAVIGASI HELIKOPTER BERDASARKAN RINTANGAN

yang dibuat untuk melengkapi sebagian persyaratan menjadi Sarjana Teknik pada Program Pendidikan Sarjana Teknik Ekstensi Departemen Teknik Elektro Fakultas Teknik Universitas Indonesia, sejauh yang saya ketahui bukan merupakan tiruan atau duplikasi dari skripsi yang sudah dipublikasikan dan atau pernah dipakai untuk mendapatkan gelar kesarjanaan di lingkungan Universitas Indonesia maupun di Perguruan Tinggi atau instansi manapun, kecuali bagian yang sumber informasinya dicantumkan sebagaimana mestinya.

Depok, 18 Juli 2008

Dimas Aryo Wicaksono

NPM. 06 06 04 245 6

PENGESAHAN

Skripsi dengan judul:

SISTEM NAVIGASI HELIKOPTER BERDASARKAN RINTANGAN

dibuat untuk melengkapi sebagian persyaratan menjadi Sarjana Teknik pada Program Studi Teknik Elektro Departemen Teknik Elektro Fakultas Teknik Universitas Indonesia. Skripsi ini telah diujikan pada sidang ujian skripsi pada tanggal 10 Juli 2008 dan dinyatakan memenuhi syarat/sah sebagai skripsi pada Departemen Teknik Elektro Fakultas Teknik Universitas Indonesia.

Depok, 18 Juli 2008

Dosen Pembimbing

Dr. Abdul Muis,ST,MEng

NIP. 132 233 210

UCAPAN TERIMA KASIH

Puji dan syukur kehadirat ALLAH SWT atas berkat dan rahmat-Nya sehingga skripsi ini dapat diselesaikan dengan baik. Tak lupa penulis juga mengucapkan terima kasih kepada:

Dr. Abdul Muis, ST, MEng

selaku dosen pembimbing yang telah meluangkan waktu untuk memberi pengarahan, diskusi dan bimbingan, serta persetujuan sehingga skripsi ini dapat selesai dengan baik. Terima kasih pula kepada kedua orang tua dan seluruh anggota keluarga atas dukungan yang telah diberikan. Tidak lupa terima kasih kepada semua rekan-rekan yang tidak dapat disebutkan satu per satu.

ABSTRAK

DIMAS ARYO WICAKSONO
NPM 06 06 04 245 6
Departemen Teknik Elektro

Dosen Pembimbing
Dr. Abdul Muis ST, M.Eng

SISTEM NAVIGASI HELIKOPTER BERDASARKAN RINTANGAN

ABSTRAK

Pada saat ini, dunia *air modelling* menggunakan helikopter mini sudah banyak digemari orang. Agar dapat bernavigasi secara *autonomous* sebuah helikopter yang cerdas tentunya harus mampu mengenali keadaan lintasan yang akan ditempuhnya. Untuk itu diperlukan sebuah sistem navigasi yang mampu mendeteksi dan menghindari objek – objek rintangan.

Skripsi ini mengimplementasikan suatu aplikasi dari sensor sonar untuk mendeteksi objek-objek rintangan dan kompas digital sebagai sistem navigasi otomatis pada penerbangan helikopter dengan tujuan agar helikopter dapat menghindari rintangan yang ada di depannya. Untuk itu, helikopter yang dirancang harus memiliki kemampuan mendeteksi objek-objek penghalang yang bersifat statis maupun dinamis. Untuk tujuan tersebut, maka sistem ini dilengkapi dengan sebuah motor servo dc yang digunakan untuk men-*scanning* lingkungan lintasannya secara *real time*.

Sensor sonar dan kompas digital yang digunakan berupa modul yang terintegrasi dengan mikrokontroler. Data yang diperoleh dikirimkan secara *telemetry* ke komputer untuk selanjutnya diolah dan dimonitor. Dari program, penerbangan helikopter akan dipandu agar dapat menghindari rintangan.

Skripsi ini berhasil mensimulasikan sistem navigasi helikopter untuk menghindari rintangan pada cakupan 10 meter di depannya.

Kata kunci : SENSOR SONAR, MOTOR SERVO, KOMPAS, RINTANGAN, HELIKOPTER, *TELEMETRY*

ABSTRACT

DIMAS ARYO WICAKSONO
NPM 06 06 04 245 6
Electrical Engineering Department

Counsellors
Dr. Abdul Muis ST, M.Eng

HELICOPTER NAVIGATION SYSTEM BASED ON OBSTACLE

ABSTRACT

Nowdays, air modelling world using mini helycopter has benn popular among the people. To navigate autonomously, a smart helycopter must can identify its path condition. For that reason, the helycopter needs a navigation system that can detects and avoids obstacle objects.

This final project applys an application of sonar sensor to detect obstacle objects and a digital compass as an automatic helicopter navigation system to avoid obstacle on the face. So that, the designed helicopter must have ability to detect static and dynamic obstacle objects. Because of that, the system should be completed with a servo dc motor which is used for scanning its path environment in real time.

The sonar and digital compass used in this project is a modul which is integrated to microcontroller. Data from sonar and digital compass is then sent via telemetry system to computer for later processed and monitored. From navigation program, the helycopter will be guided in order to avoid the obstacle.

Finally, this final project is succeed in simulating helycopter navigation system to avoid obstacle in the range 10 metres on the face.

Keywords : SONAR SENSOR, SERVO MOTOR, COMPASS, OBSTACLE, HELICOPTER

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

Halaman

BAB I

PENDAHULUAN

1.1 LATAR BELAKANG MASALAH

Pada saat ini, dunia *air modelling* menggunakan helikopter mini sudah banyak digemari orang. Agar dapat bernavigasi secara autonomous sebuah helikopter yang cerdas tentunya harus mampu mengenali keadaan lintasan yang akan ditempuhnya. Untuk itu diperlukan sebuah sistem navigasi yang mampu mendeteksi dan menghindari objek – objek rintangan.

Dalam banyak aplikasi, sensor sonar adalah sensor yang umum digunakan untuk menentukan jarak sebuah objek yang di deteksinya. Salah satu aplikasi penting pemanfaatan sensor sonar adalah pada bidang penerbangan seperti helikopter.

Untuk tujuan tersebut maka sebuah helikopter harus dilengkapi dengan sensor yang dapat men-*scanning* lingkungan lintasannya secara *real time*. Salah satu sensor yang banyak digunakan untuk hal diatas adalah sensor sonar atau ultrasonik. Hal ini terkait dengan kemampuan jangkauan deteksinya yang relative jauh, tingkat radiasi yang aman serta harganya relative murah.

Selain itu, dibutuhkan juga penggerak sensor untuk dapat melakukan *scanning* lingkungan pada jalur lintasannya, untuk itu digunakan motor servo dc. Sedangkan sebagai pengatur arah helikopter digunakan kompas digital sebagai panduan arah untuk mencapai lokasi yang diinginkan.

1.2 TUJUAN PENULISAN SKRIPSI

Tujuan penulisan skripsi ini adalah untuk mengimplementasikan teknologi dari sensor sonar dan kompas digital yang dibuat menjadi satu sistem dalam navigasi helikopter untuk menghindari rintangan yang ada di depannya.

1.3 RUANG LINGKUP DAN PEMBATASAN MASALAH

1.3.1 Ruang Lingkup

Ruang lingkup skripsi ini adalah sebagai berikut:

1. Perancangan sistem navigasi secara telemetri.
2. Pemanfaatan informasi dari sensor sonar untuk menghindari rintangan.
3. Pemanfaatan informasi dari kompas digital untuk mengetahui arah tujuan.

1.3.2 Pembatasan Masalah

Pembatasan masalah pada skripsi ini adalah:

1. Deteksi rintangan dilakukan hanya dengan sebuah sensor sonar.
2. *Scanning* rintangan dilakukan dengan menggunakan motor servo sebesar 15^0 dan 30^0 di sebelah kanan dan kiri dari rintangan di depannya.
3. Sistem dikendalikan dengan menggunakan mikrokontroler *MinSys* DT-51/52 AT89S52.
4. Informasi rintangan yang dideteksi dikirimkan secara telemetri.
5. Petunjuk untuk menghindari rintangan dilakukan dengan memberikan informasi arah dan kecepatan kepada helikopter.
6. Bentuk informasi arah hindar rintangan yang digunakan adalah kanan, kiri dan naik.
7. Untuk arah naik, diasumsikan tidak ada sesuatu diatas helikopter. Batasan jarak naiknya helikopter adalah sejauh jangkauan dari sistem telemetri yang digunakan.
8. Bentuk informasi kecepatan yang digunakan adalah berupa variabel bahasa yaitu ’fast’, ’ normal’, ’slow’, dan ’stop’. Nilai kecepatan masing – masing variabel tidak dibahas pada skripsi ini.
9. Adapun dari skripsi ini, implementasi pada helikopter secara langsung belum dilakukan.

1.4 METODE PERANCANGAN

Perancangan dimulai dengan mempelajari tentang penggunaan sensor sonar sebagai alat untuk mendeteksi objek. Setelah itu, dilakukan pengujian sonar dalam mendeteksi jenis – jenis objek benda serta bentuknya. Selain itu, dilakukan

juga pengujian keakuratan jarak dari sonar serta lebar sudut pancaran deteksi sonar.

Kemudian perancangan dilanjutkan dengan mengintegrasikan sensor sonar dan motor servo pada satu mikrokontroler. Dengan mikrokontroler ini, maka data yang diperoleh akan diproses untuk menggerakan motor servo dalam melakukan *scanning*. Setelah itu, data yang diperoleh digunakan untuk menggerakan helikopter dengan panduan arah dari kompas digital sehingga dapat menghindari rintangan.

1.5 SISTEMATIKA PENULISAN

Sistematika penulisan skripsi ini dibagi menjadi 5 bab. Bab I berisi pendahuluan, yang membahas latar belakang masalah, tujuan penelitian, ruang lingkup dan pembatasan masalah, metode penelitian, dan sistematika penulisan skripsi ini. Selanjutnya bab II membahas landasan teori yang diperlukan untuk merancang sistem dengan sensor sonar, motor servo, kompas digital dan mikrokontroler. Berikutnya bab III membahas perancangan sistem dan penjelasan dari sistem yang akan dibuat serta modul – modul yang digunakan. Lalu bab IV membahas pengujian sistem dan analisis pada setiap pengujian. Dan yang terakhir bab V membahas kesimpulan hasil penelitian terhadap rancangan yang dihasilkan.

BAB II

LANDASAN TEORI

2.1 SENSOR SONAR

Pada bab ini akan dijelaskan ilmu fisika dan hal - hal mengenai sensor sonar yang digunakan untuk mendeteksi benda, mengukur jarak benda tersebut serta klasifikasinya dalam aplikasi robot. Sumber dari struktur sonar akan dijelaskan bagaimana hal tersebut bisa dilakukan.

Sensor sonar atau ultrasonik menggunakan rambatan energi akustik pada frekuensi yang lebih tinggi dibanding pendengaran normal manusia untuk memperoleh informasi dari lingkungan.

Sonar adalah salah satu sensor yang populer di dalam robotika yang menghasilkan pulsa akustik dan sinyal echo (gema) untuk mengukur jarak dari suatu benda yang di deteksi[1]. Dengan kecepatan bunyi telah diketahui maka jarak dari benda adalah sebanding dengan waktu tempuh dari sinyal echo. Pada frekuensi ultrasonik energi sonar dipusatkan dalam suatu berkas cahaya serta menyediakan informasi secara langsung sebagai tambahan terhadap jarak.

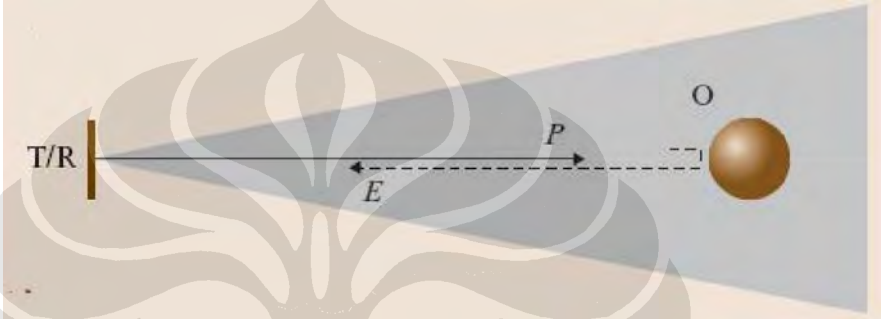
Kelebihan dari sensor sonar adalah biaya yang murah,ringan,konsumsi daya yang rendah, dan perhitungan yang mudah, bila dibandingkan dengan sensor jarak yang lainnya. Dalam beberapa aplikasi, seperti di dalam air dan lingkungan dengan jarak penglihatan rendah, sonar sering digunakan.

Di dalam robotika, sensor sonar mempunyai tiga tujuan yang berbeda, tetapi berhubungan,yaitu [2] :

1. Penghindaran rintangan (*Obstacle avoidance*): Gema (sinyal echo) pertama yang dideteksi pertama diasumsikan untuk mengukur jarak dari benda terdekat. Robot-robot menggunakan informasi ini untuk merencanakan lintasan di sekitar rintangan dan mencegah benturan/tabrakan.
2. Pemetaan sonar (*Sonar Mapping*): Beberapa sinyal echo(gema) yang diperoleh dari pencarian secara putaran atau dari beberapa sensor sonar

digunakan untuk membangun peta lingkungan. Seperti halnya tampilan pada radar,satu titik ditempatkan pada cakupan yang dideteksi.

3. Pengenalan objek (*Object recognition*): Satu urutan dari sinyal echo atau pemetaan sonar diproses untuk menggolongkan sinyal echo, lalu digunakan untuk menggambarkan struktur-struktur dar objek yang dideteksi. Bila berhasil, informasi ini akan bermanfaat untuk navigasi robot.



Gambar 2.1 Sistem dari sensor sonar

Gambar 2.2 Waktu transmisi sonar

Gambar 2.3 Jarak deteksi objek ke sensor sonar

Gambar 2.1 – 2.3 menunjukan sebuah sistem sonar yang sederhana. Sebuah transducer sonar,T/R,berlaku sebagai pemancar (T) untuk menghasilkan pulsa akustik (P) dan sebagai penerima(R) dari sinyal echo(E). Sebuah objek (O) terletak pada cakupan dari pancaran sonar, lalu memantulkan pulsa P. Sinyal yang dipantulkan oleh objek (O) ke tranduser akan dideteksi sebagai sinyal echo(gema). Waktu tempuh echo t_0, biasa disebut time-of-flight (TOF) yang diukur dari waktu pada saat pulsa ditransmisikan (Gambar 2.2). Jarak objek r_0 (Gambar 2.3) dapat dihitung dengan rumus :

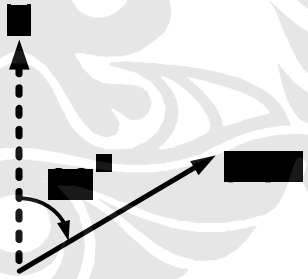
$$r_0 = \frac{ct_0}{2} \quad \text{.....(2.1)}$$

Dimana c merupakan kecepatan bunyi (besarnya adalah 344 m/s pada suhu dan tekanan standar). Angka 2 merupakan perjalanan sinyal dari P dan E untuk sekali pengukuran.

2.2 KOMPAS

Sejak dulu kala, kompas digunakan untuk mengetahui arah mata angin. Kompas ini bekerja berdasarkan medan magnet yang dihasilkan oleh bumi. Seiring dengan kemajuan jaman, maka telah ditemukan sebuah rangkaian dan sensor medan magnet yang digunakan untuk mengukur medan magnet bumi dan menghasilkan sebuah kompas digital. Data yang diperoleh dari kompas digital ini merupakan sudut yang dibentuk terhadap mata angin arah Utara (0^o).

Sebagai contoh, bila pembacaan menunjukkan 60^0 U, berarti sudut kompas membentuk sudut 60^0 terhadap mata angin utara. Penggambarannya adalah sebagai berikut :



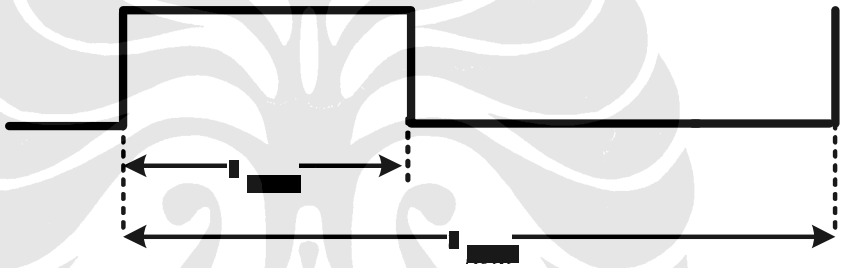
Gambar 2.4 Contoh pembacaan arah kompas digital

2.3 MOTOR SERVO

Motor servo adalah sebuah motor dengan sistem *closed feedback* di mana posisi dari motor akan diinformasikan kembali ke rangkaian kontrol yang ada di dalam motor servo[3]. Motor ini terdiri dari sebuah motor, 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.

Gambar 2.5 Contoh Motor Servo



Gambar 2.6 Teknik PWM untuk mengatur sudut motor servo

Teknik PWM atau *Pulse Width Modulation* digunakan mengatur sudut pergerakan motor servo. Seperti tampak pada gambar 2.6, terdapat t_{arah} dan t_{hold} pada pulsa yang dikirimkan ke motor servo. Pulsa 1 untuk t_{arah} berfungsi untuk mengatur arah putaran servo yang diinginkan. Motor servo dapat bergerak searah jarum jam dan berlawanan arah jarum jam. Besarnya t_{arah} ini berbeda – beda untuk setiap produk motor servo. Biasanya, besar t_{arah} lebih besar dari t_{hold}. Pulsa 0 untuk t_{hold} berfungsi untuk menahan pergerakan motor servo pada sudut yang diinginkan.

Motor servo akan bergerak sebesar x derajat sesuai dengan spesifikasi produk servo yang digunakan. Pemberian pulsa untuk x derajat ini adalah pulsa seperti pada gambar 2.6. Untuk mendapatkan sejumlah perubahan sudut yang diinginkan, maka pulsa PWM harus diberikan secara kontinyu. Pada umumnya, total pergerakan sudut yang bisa dilakukan oleh motor servo adalah 180^0.

2.4 MIKROKONTROLER

Mikrokontroler yang digunakan didalam skripsi ini adalah AT89S52 dari ATMEL yang mana merupakan keluarga MCS 51 yang banyak dijual di pasaran. Mikrokontroler ini merupakan mikrokontroler 8 bit yang merupakan single chip mikrokontroler, dimana semua rangkaian termasuk memori dan I/O (*input/output)* tergabung dalam satu pak IC. Di sini akan di bahas sedikit keterangan mengenai mikrokontrol, kontroler ini yang terdiri dari:

1. Konfigurasi I/O
2. Interupsi
3. Timer, counter

2.4.1 Konfigurasi I/O

Mikrokontroler 89C52 mempunyai 32 bit jalur I/O yang terbagi dalam 4 port yaitu port 0 sampai dengan port 3.

- **Port 0**

Port 0 merupakan 8 bit biderectional I/O dengan rangkaian open drain sebagai driver di dalamnya, sehingga saat diisi logika satu maka port ini akan bersifat mengambang (high impedance). Sehingga untuk penggunaan rangkaian I/O diperlukan resistor pull-up eksternal. Port 0 dapat dijadikan sebagai *multiplexed low order address bus* saat dikonfigurasikan dengan rangkaian memori eksternal. Saat flash programing port ini menerima byte kode program dan mengeluarkan byte kode saat proses verifikasi.

- **Port 1**

Port 1 terdiri dari 8 bit *bidirectiona*l I/O dengan internal resistor pull-up

- **Port 2**

Port 2 Merupakan 8 bit *bidirectiona*l I/O dengan internal resistor pullup. Saat dikonfigurasikan dengan rangkaian memori eksternal, port ini juga akan berfungsi sebagai addres bus byte yang tinggi (A8-A15). Port 2 berfungsi sebagai *multiplexed high order address bus* saat flash programing dan sebagai bit kontrol saat proses verifikasi.

- **Port 3**

Port 3 Merupakan 8 bit *bidirectional* I/O dengan internal resistor pullup. Port 3 dapat juga difungsikan untuk beberapa fungsi khusus seperti yang ditunjukkan pada tabel berikut:

Tabel 2.1 Konfigurasi alternatif dari port 3 89S52

Pin Port 3	Fungsi
P3.0	RXD (Input port serial)
P3.1	TXD (Output port serial)
P3.2	$\overline{INT0}$ (Interupsi eksternal 0) → aktif *low*
P3.3	$\overline{INT1}$ (Interupsi eksternal 1) → aktif *low*
P3.4	T0 (Input *timer* ekternal 0)
P3.5	T1 (Input *timer* eksternal 1)
P3.6	$\overline{WR}$ (Penulisan *strobe* pada memori data) eksternal) → aktif *low*
P3.7	$\overline{RD}$ (Pembacaan *strobe* pada memori data eksternal) → aktif *low*

2.4.2 Interupsi

Mikrokontroler 89S52 terdapat beberapa saluran interupsi. Interupsi pada 89C52 dibedakan 2 jenis,yaitu:

1. Interupsi yang tak dapat dihalangi oleh perangkat lunak (*non maskable interrupt*), misalnya reset.
2. Interupsi yang dapat dihalangi perangakat lunak (maskable interrupt) misalnya interupsi int0 dan int1 (eksternal) serta timer/counter 0 dan timer/counter 1 dan interupsi dari port serial (internal).

Mikrokontroller 89S52 mempunyai beberapa buah vektor interupsi yaitu dua buah interupsi timer (timer 0 dan timer 1), dua buah interupsi eksternal (int 0 dan int 1), dan sebuah interupsi serial . Pengaktifan interrupt pada mikrokontroler ini terdapat pada register *Interrupt Enable* (IE), yang terdapat pada Special Function Register.

2.4.3 Timer dan Counter

Seperti telah dikemukakan diatas bahwa mikrokontroler AT89S52 mempunyai 2 buah interupsi timer yaitu timer 0 dan timer 1. Perangkat Timer/Counter tersebut merupakan perangkat keras yang menjadi satu dalam chip

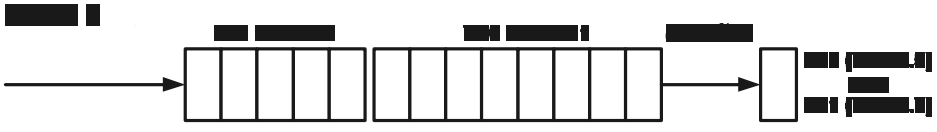
mikrokontroler AT89S52. Pencacah biner untuk Timer 0 dibentuk dengan register TL0 (Timer 0 *Low Byte*) dan register TH0 (Timer 0 *High Byte*). Pencacah biner untuk Timer 1 dibentuk dengan register TL1 (Timer 1 *Low Byte*) dan register TH1 (Timer 1 *High Byte*).

Timer dan Counter merupakan sarana input yang kurang mendapat perhatian pemakai mikrokontroler, dengan sarana input ini mikrokontroler dengan mudah bisa dipakai untuk mengukur lebar pulsa, membangkitkan pulsa dengan lebar yang pasti, dipakai dalam pengendalian tegangan secara PWM (*Pulse Width Modulation*) yang tentunya sangat diperlukan untuk aplikasi sonar dan motor servo.

Pencacah biner pembentuk Timer/Counter MCS51 merupakan pencacah biner menaik (*count up binary counter*) yang mencacah dari $0000 sampai $FFFF, saat kedudukan pencacah berubah dari $FFFF kembali ke $0000 akan timbul sinyal *overflow*.

TL0, TH0, TL1 dan TH1 merupakan SFR (*Special Function Register*) yang dipakai untuk membentuk pencacah biner perangkat Timer 0 dan Timer 1. Kapasitas keempat register tersebut masing-masing 8 bit, bisa disusun menjadi 4 macam Mode pencacah biner seperti terlihat dalam Gambar 2.6 sampai Gambar 2.7.

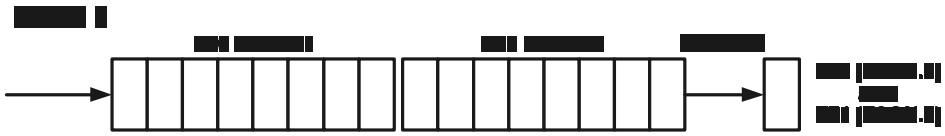
Pada Mode 0, Mode 1 dan Mode 2, Timer 0 dan Timer 1 masing-masing bekerja sendiri, artinya bisa dibuat Timer 0 bekerja pada Mode 1 dan Timer 1 bekerja pada Mode 2, atau kombinasi mode lainnya sesuai dengan keperluan. Pada Mode 3 TL0, TH0, TL1 dan TH1 digunakan bersama-sama untuk menyusun sistem timer yang tidak bisa di-kombinasi lain. Susunan TL0, TH0, TL1 dan TH1 pada masing-masing mode adalah sebagai berikut:



Gambar 2.6 Mode 0 - Pencacah Biner 13 Bit

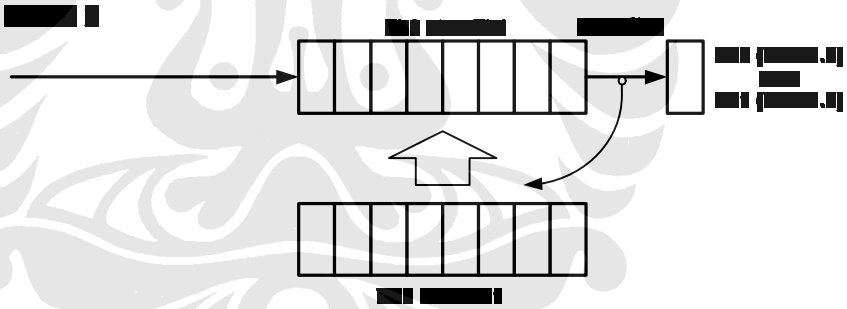
Pencacah biner dibentuk dengan TLx (maksudnya bisa TL0 atau TL1) sebagai pencacah biner 5 bit (meskipun kapasitas sesungguhnya 8 bit), *overflow* dari pencacah biner 5 bit ini dihubungkan ke THx (maksudnya bisa TH0 atau

TH1) membentuk sebuah untaian pencacah biner 13 bit, *overflow* dari pencacah 13 bit ini ditampung di flip-flop TFx (maksudnya bisa TF0 atau TF1) yang berada di dalam register TCON.



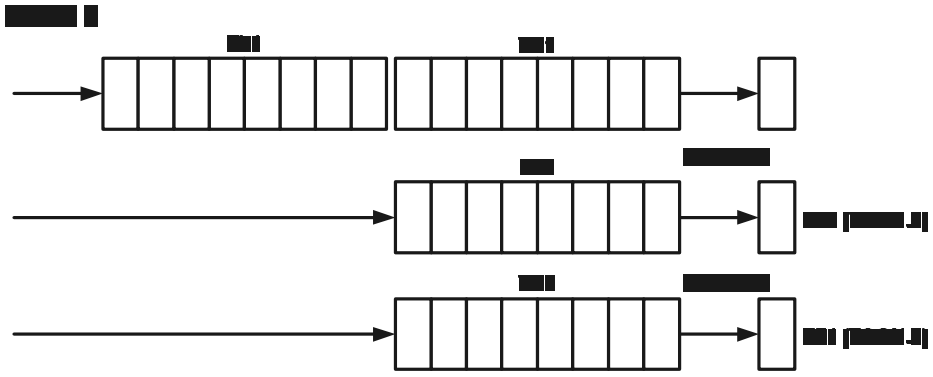
Gambar 2.7 Mode 1 - Pencacah Biner 16 Bit

Mode ini sama dengan *Mode 0*, hanya saja register TLx dipakai sepenuhnya sebagai pencacah biner 8 bit, sehingga kapasitas pencacah biner yang terbentuk adalah 16 bit. Seiring dengan masuknya sinyal persegi, kedudukan pencacah biner 16 bit ini akan bergerak dari $0000 (biner 0000 0000 0000 0000), $0001, $0002 … sampai $FFFF (biner 1111 1111 1111 1111), kemudian kembali menjadi $0000.



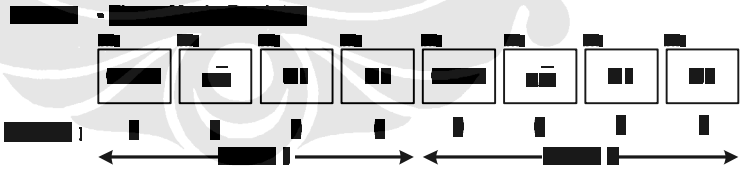
Gambar 2.8 Mode 2 - Pencacah Biner 8 Bit dengan Isi Ulang

TLx dipakai sebagai pencacah biner 8 bit, sedangkan THx dipakai untuk menyimpan nilai yang diisikan ulang ke TLx, setiap kali kedudukan TLx melimpah (berubah dari $FF menjadi $00). Dengan cara ini bisa didapatkan sinyal limpahan yang frekuensinya ditentukan oleh nilai yang disimpan dalam TH0.



Gambar 2.9 Mode 3 – Gabungan Pencacah Biner 16 Bit dan 8 Bit

Pada Mode 3 TL0, TH0, TL1 dan TH1 dipakai untuk membentuk 3 untaian pencacah, yang pertama adalah untaian pencacah biner 16 bit tanpa fasilitas pemantau sinyal limpahan yang dibentuk dengan TL1 dan TH1. Yang kedua adalah TL0 yang dipakai sebagai pencacah biner 8 bit dengan TF0 sebagai sarana pemantau limpahan. Pencacah biner ketiga adalah TH0 yang dipakai sebagai pencacah biner 8 bit dengan TF1 sebagai sarana pemantau limpahan. Untuk mengatur timer digunakan register TMOD dan register TCON. Register TMOD dan register TCON merupakan register pembantu untuk mengatur kerja Timer 0 dan Timer 1, kedua register ini dipakai bersama oleh Timer 0 dan Timer 1.



Gambar 2.10 Denah susunan bit dalam register TMOD

Register TMOD dibagi menjadi 2 bagian secara simetris, bit 0 sampai 3 register TMOD (TMOD bit 0 .TMOD bit 3) dipakai untuk mengatur Timer 0, bit 4 sampai 7 register TMODE (TMOD bit 4 .. TMOD bit 7) dipakai untuk mengatur Timer 1, pemakaiannya sebagai berikut :

• Bit M0/M1 dipakai untuk menentukan Mode Timer seperti yang terlihat dalam Tabel di Gambar 2.3.

• Bit $C/\overline{T}$ dipakai untuk mengatur sumber sinyal persegi yang diumpankan ke pencacah biner. Jika $C/\overline{T}$ =0 sinyal persegi diperoleh dari osilator kristal yang frekuensinya sudah dibagi 12, sedangkan jika $C/\overline{T}$ =1 maka sinyal persegi diperoleh atau dihasilkan di kaki T0 (untuk Timer 0) atau kaki T1 (untuk Timer 1).

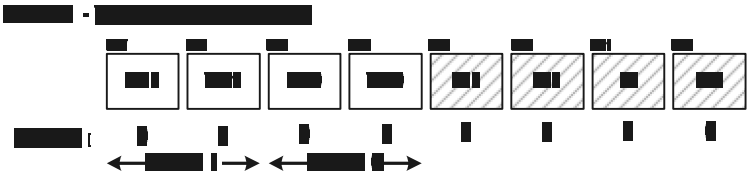
• Bit GATE merupakan bit pengatur saluran sinyal persegi. Bila bit GATE=0 saluran sinyal persegi hanya diatur oleh bit TRx (maksudnya adalah TR0 atau TR1 pada register TCON). Adapun bit GATE=1 kaki INT0 (untuk Timer 0) atau kaki INT1 (untuk Timer 1) dipakai juga untuk mengatur saluran sinyal persegi.

Tabel 2.2. Fungsi-Fungsi Bit TMOD

Simbol	**Alamat Bit**	**Fungsi**
GATE	TMOD.7 dan TMOD.3	Mengatur saluran sinyal denyut. Bila 0, saluran sinyal persegi hanya diatur bit TRx. Bila 1, kaki INT0 atau INT1 dipakai juga untuk mengatur saluran sinyal persegi.
$C/\overline{T}$	TMOD.6 dan TMOD.2	Mengatur sumber sinyal persegi yang diumpankan ke pencacah biner. Bila 0, sinyal denyut diperoleh dari osilator Kristal yang frekuensinya sudah dibagi 12. Bila 1, maka persegi diperoleh atau dihasilkan di kaki T0 (untuk Timer 0) atau kaki T1 (untuk Timer 1)sinyal denyut diperoleh dari kaki T0 atau kaki T1.
M1	TMOD.5 dan TMOD.1	Mode bit 1
M0	TMOD.4 dan TMOD.0	Mode bit 0

Tabel 2.3. Konfigurasi Mode *Timer/Counter*

M1	**M0**	**Mode**
0	0	0
0	1	1
1	0	2
1	1	3



Gambar 2.11 Denah susunan bit dalam register TCON

Register TCON dibagi menjadi 2 bagian, 4 bit pertama (bit 0 .. bit 3), bagian yang diarsir dalam Gambar 2.11, dipakai untuk keperluan mengatur kaki INT0 dan INT1. Sisa 4 bit dari register TCON (bit 4..bit 7) dibagi menjadi 2 bagian secara simetris yang dipakai untuk mengatur Timer0/Timer 1, sebagai berikut:

• Bit TFx (maksudnya adalah TF0 atau TF1) merupakan bit penampung *overflow.* TFx akan menjadi '1' setiap kali pencacah biner yang terhubung padanya melimpah (kedudukan pencacah berubah dari $FFFF kembali menjadi $0000). Bit TFx di-nol-kan dengan instruksi CLR TF0 atau CLR TF1. Jika sarana interupsi dari Timer 0/Timer 1 dipakai, TRx di-nol-kan saat MCS51 menjalankan rutin layanan interupsi (ISR – *Interupt Service Routine*).

• Bit TRx (maksudnya adalah TR0 atau TR1) merupakan bit pengatur saluran sinyal persegi, bila bit ini =0 sinyal persegi tidak disalurkan ke pencacah biner sehingga pencacah berhenti mencacah. Bila bit GATE pada register TMOD =1, maka saluran sinyal persegi ini diatur bersama oleh TRx dan sinyal pada kaki INT0/INT1.

Tabel 2.4. Fungsi-Fungsi Bit TCON

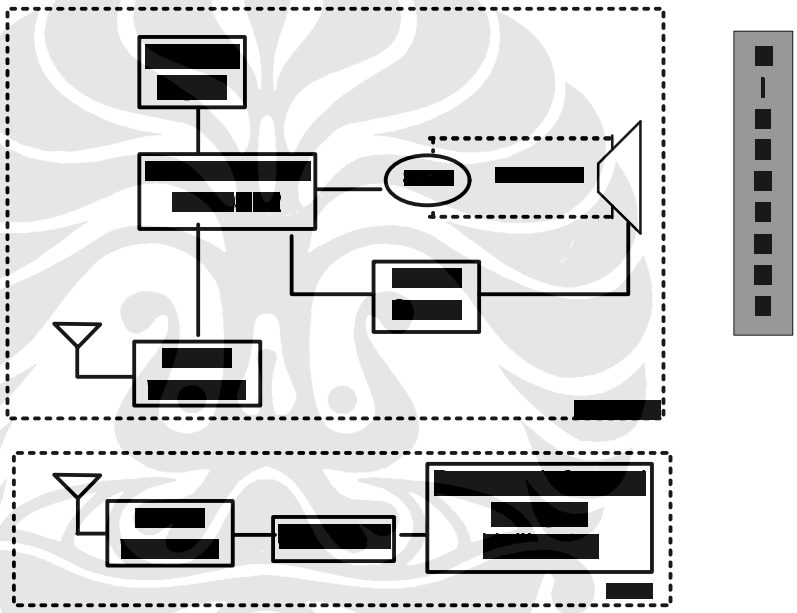
Simbol	Posisi Bit	Fungsi
TF1	TCON.7	*Timer* 1 *overflow flag*
TR1	TCON.6	*Timer* 1 *run control bit*
TF0	TCON.5	*Timer* 0 *overflow flag*
TR0	TCON.4	*Timer* 0 *run control bit*
IE1	TCON.3	*Interrupt* 1 *edge flag*
IT1	TCON.2	*Interrupt* 1 *type control bit*
IE0	TCON.1	*Interrupt* 0 *edge flag*
IT0	TCON.0	*Interrupt* 0 *type control bit*

BAB III

PERANCANGAN SISTEM

3.1 PRINSIP KERJA SISTEM

Fungsi dari sistem navigasi dengan menggunakan sensor sonar ini adalah bagaimana membuat helikopter yang bergerak bebas pada suatu area dapat melewati/menghindari rintangan yang ada di depannya.

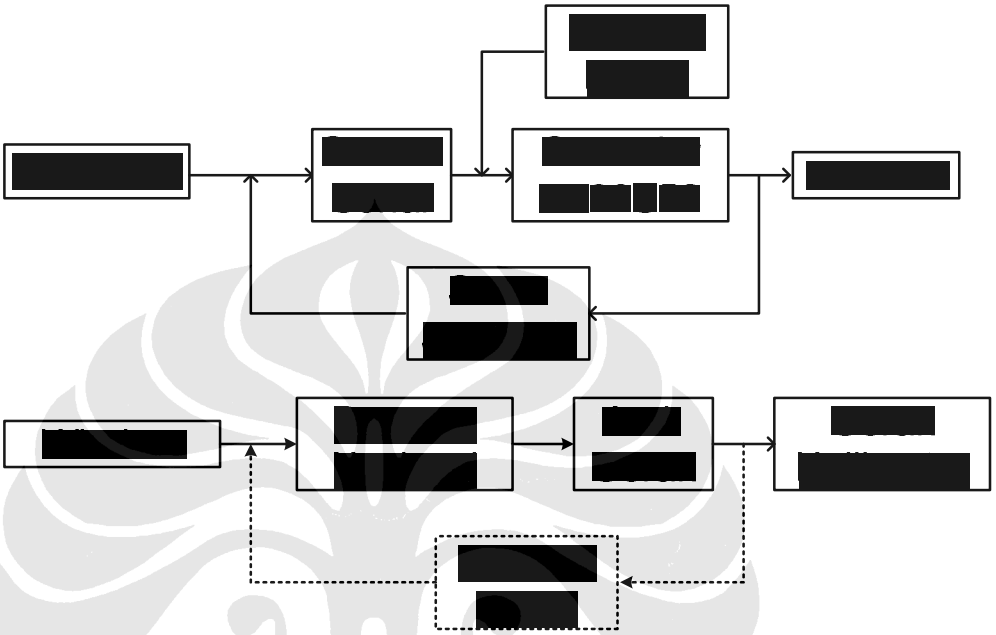


Gambar 3.1 Blok Sistem Navigasi Helikopter berdasarkan Rintangan

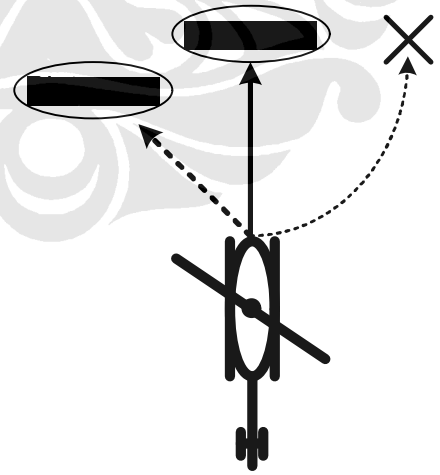
Seperti terlihat pada gambar diatas, sistem ini mengunakan sebuah motor servo. Motor servo ini berfungsi untuk menggerakan transduser dari sonar untuk mencari posisi dimana tidak terdapat rintangan. Pada awal sistem bekerja servo tidak bergerak. Namun pada saat tranduser sonar mendeteksi objek, maka servo akan bergerak sekian derajat ke kanan dan ke kiri untuk mendeteksi apakah masih terdapat halangan atau tidak. Informasi dari sonar ini akan dikirimkan ke komputer melalui modul *wireless* yang terpasang. Dari informasi yang di dapat,

helikopter akan dipandu untuk menghindari rintangan. Arah pergerakan helikopter ini akan dibimbing oleh kompas digital.

Sistem navigasi penghindar rintangan untuk helikopter ini digambarkan pada bagan dibawah ini.



Gambar 3.2 Sistem navigasi penghindar rintangan helikopter



Gambar 3.3 Sistem penghindar rintangan helikopter

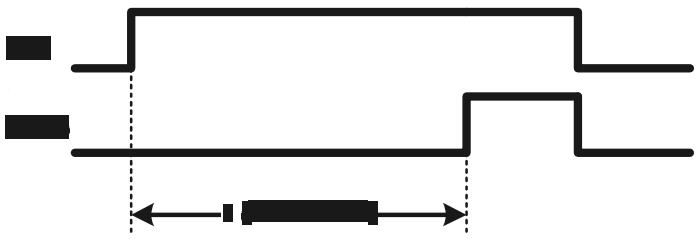
Gambar 3.3 menunjukkan bagaimana helikopter menghindari rintangan yang ada di depan dan samping kirinya. Helikopter akan menuju ke arah X setelah melakukan *scanning* lingkungannya.

3.2 RANCANGAN TATAP MUKA PERANGKAT KERAS

Perangkat keras secara keseluruhan dibuat dengan menggunakan komponen-komponen sesuai dengan fungsi dari blok-blok sistem yang digambarkan pada Gambar 3.1. Perangkat keras yang digunakan dalam sistem ini adalah sebagai berikut :

1. Modul *MinSys* DT 51/52 yang menggunakan mikrokontroler AT89S52
2. Modul Sonar Ranging Senscomp 6500
3. Tranduser Senscomp 600 Series
4. Kompas Digital CMPS03
5. Modul *Wireless* YS1020UA RF
6. Motor Servo SANWA SX-101z

Sensor sonar yang digunakan adalah modul sonar ranging senscomp 6500 yang menggunakan tranduser senscomp 600 series. Modul sonar ranging berfungsi sebagai pengatur pemancar dan penerima gelombang sonar dan pemberi informasi objek yang dideteksi kepada mikrokontroler. Masukan dan keluaran dari modul sonar ini adalah pulsa digital 5 volt DC. Transduser berfungsi sebagai pemancar dan penerima gelombang sonar. Frekuensi yang digunakan adalah 50 kHz. Pada modul sonar, pin yang digunakan adalah init dan echo. Pin Init merupakan masukkan untuk modul. Untuk mengaktifkan sonar maka pada init diberikan masukkan logika 1. Sedangkan untuk logika 0 digunakan untuk menonaktifkan sonar. Pin Echo merupakan keluaran keluaran dari modul. Pada saat sonar mendeteksi objek, maka echo akan berubah dari logika 0 ke 1.



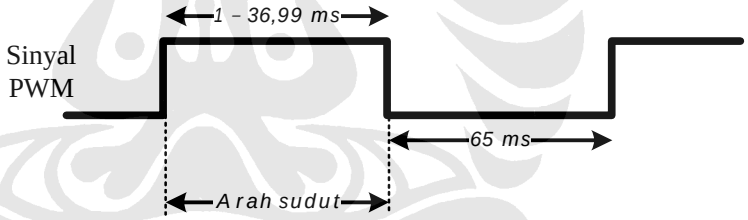
Gambar 3.4 Pulsa waktu modul sonar

Gambar 3.4 memperlihatkan pulsa waktu sonar. Untuk menghitung jarak objek yang terdeteksi (r_o), digunakan rumus berikut :

$$r_0 = \frac{ct_0}{2} \quad \text{.....(3.1)}$$

Nilai c merupakan kecepatan rambat bunyi di udara yaitu 344 m/s. Nilai t adalah nilai yang diukur pada saat init diberi pulsa 1 sampai nilai echo naik dari pulsa 0 ke pulsa 1.

Kompas digital berfungsi sebagai pemberi informasi arah sudut terhadap mata angin utara. Kompas digital menghasilkan sinyal PWM sebagai keluarannya. Sinyal PWM adalah sebuah sinyal yang telah dimodulasi lebar pulsanya. Pada CMPS03, lebar pulsa positif merepresentasikan sudut arah. Lebar pulsa bervariasi antara 1 mS (0^0) sampai 36.99 mS (359.9^0). Artinya, lebar pulsa berubah sebesar 0,1 mS setiap derajatnya. Sinyal akan low selama 65 mS diantara pulsa, sehingga total periodenya adalah 65 mS + lebar pulsa positif (antara 66 mS sampai 102 mS). Pulsa tersebut dihasilkan oleh timer 16 bit di dalam prosesornya, yang memberikan resolusi 1 µS. Pulsa PWM kompas digital ini dapat dilihat pada gambar berikut :

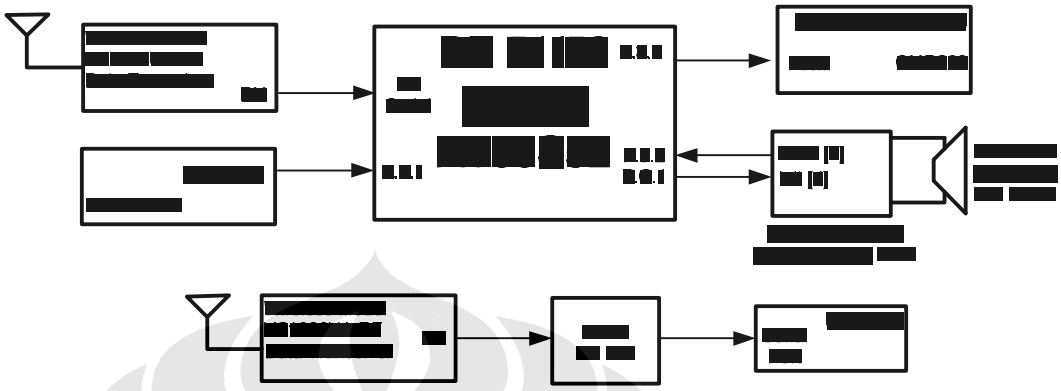


Gambar 3.5 Pulsa PWM kompas digital

Modul *MinSys* DT 51/52 digunakan untuk mengontrol modul sensor, menerima informasi arah sudut dari kompas digital, mengatur pergerakan motor servo dan mengirimkan data ke modul *wireless.*

Modul *Wireless* digunakan untuk mengirimkan data dari mikrokontroler ke komputer. Data yang dikirimkan dari mikrokontroler ke modul ini adalah berupa kode ASCII dari karakter yang digunakan.

Diagram blok lengkap untuk tatap muka perangkat keras dapat dilihat pada gambar berikut :



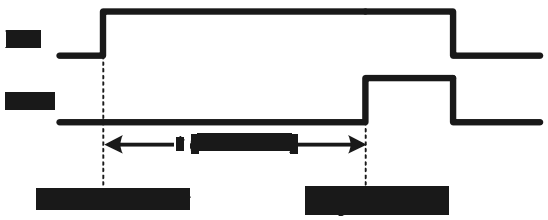
Gambar 3.6 Diagram Blok Tatap muka Perangkat Keras

3.3 PERANCANGAN PROGRAM PADA MIKROKONTROLER

Bahasa pemrograman yang digunakan adalah BASIC untuk MCS-51® dengan *compiler* BASCOM-8051©. Bahasa pemrogram ini cukup mudah dalam penggunaannya bila dibandingkan dengan menggunakan bahasa assembler biasa. Pembuatan program dibagi menjadi beberapa bagian yaitu, program untuk mengontrol kerja modul sensor sonar, program untuk mengambil data dari kompas digital, program pengendalian motor servo, serta program keseluruhan dari sistem yang dibuat pada mikrokontroler.

3.3.1 Program Sensor Sonar

Sesuai penjelasan sebelumnya, untuk bisa menggunakan sensor sonar, maka pin init pada modul sonar diberi pulsa 1. Pada saat itu, timer pada mikrokontroler dimulai. Ketika pin echo menghasilkan pulsa 1, maka menandakan bahwa sonar mendeteksi objek. Pada saat itu, maka timer pada mikrokontroler dihentikan.



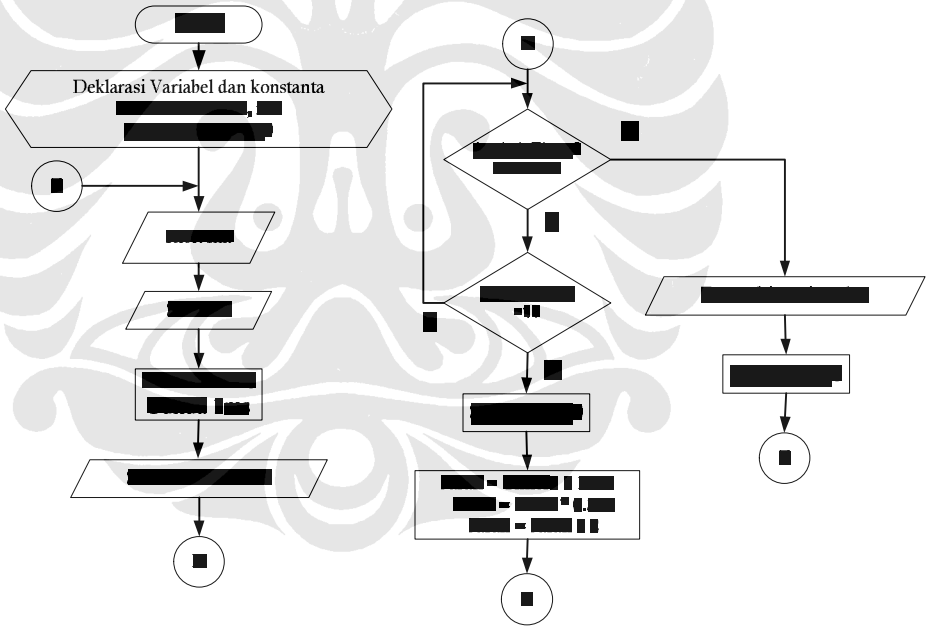
Gambar 3.7 Pulsa waktu sonar

Gambar 3.7 menunjukkan penggunaan timer untuk menghitung waktu perjalanan pulsa sonar saat dipancarkan dan saat sonar menerima pulsa balik setelah mengenai objek. Untuk mendapatkan nilai jarak objek yang dideteksi, maka dapat dihitung dengan rumus :

$$jarak = \frac{\left(\left(\frac{timer}{1000}\right) \times 0{,}344\right)}{2}(ms) \quad \text{.....(3.2)}$$

Nilai timer pada rumus diatas dibagi dengan 1000 karena timer yang didapat dari mikrokontroler masih dalam µs. Sedangkan nilai kecepatan rambat bunyi yang seharusnya sebesar 344 m/s menjadi 0,344 m/ms. Bila tidak ada objek yang terdeteksi maka timer mikrokontroler akan mengalami *overflow.*

Diagram alir untuk programnya adalah sebagai berikut :

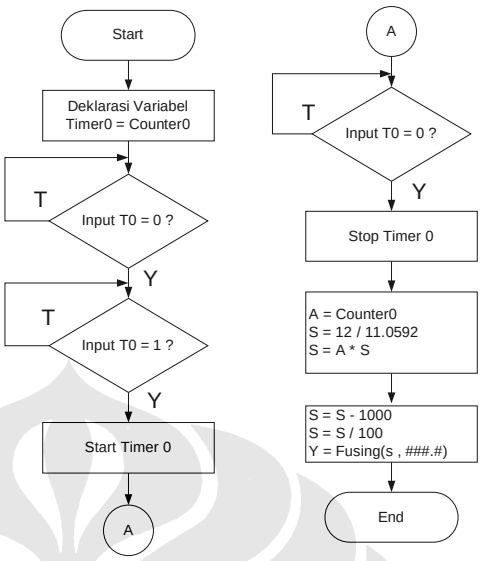


Gambar 3.8 Diagram Alir Program Sonar

3.3.2 Program kompas digital

Alur program kompas digital yang dirancang membaca pulsa sinyal PWM yang dikirimkan dari modul pada port mikrokontroler lalu mengkonversinya dengan sudut.

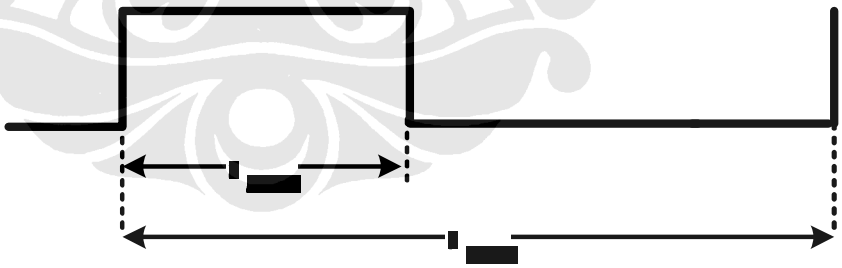
Diagram alir untuk programnya adalah sebagai berikut :



Gambar 3.9 Diagram Alir Kompas Digital

3.3.3 Program Pengendalian Motor Servo

Untuk mengendalikan motor servo, maka diperlukan sinyal PWM dari mikrokontroler. Sinyal PWM dapat dibentuk dengan memberikan pulsa 1(menset) atau me-reset dengan memberikan pulsa 0. Panjangnya pulsa cukup diatur dengan memberikan waktu tunda pada program.

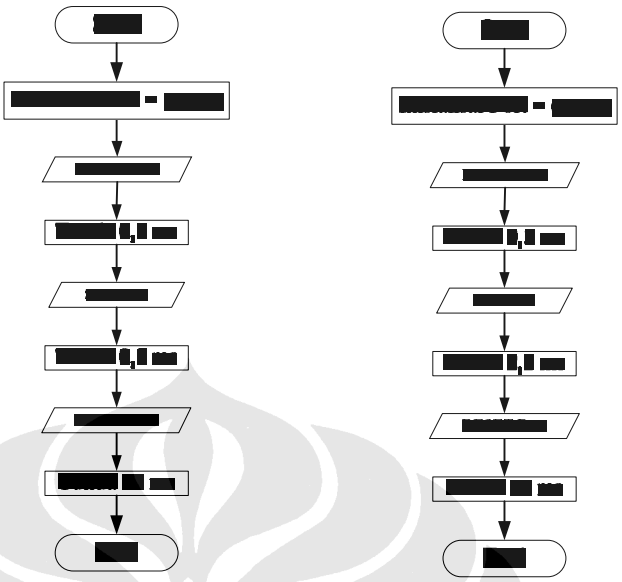


Gambar 3.10 Teknik PWM untuk mengatur sudut motor servo

Motor servo yang digunakan memiliki t_{arah} sebesar 0,7 ms – 2,2 ms dan t_{hold} sebesar 12 ms – 20 ms. Besar sudut pergerakan motor servo ini adalah 15^o.

Untuk pergerakan servo ke kiri sebesar 15^o, maka pulsa 1 diberikan sepanjang 0,7ms. Sedangkan untuk pergerakan servo ke kanan sebesar 15^o, maka pulsa 1 diberikan sepanjang 2,2 ms.

Diagram alir selengkapnya dapat dilihat pada gambar di bawah ini :



Gambar 3.11 Diagram Alir Motor Servo untuk gerak kiri dan kanan sebesar 15^{o}

3.3.4 Program Sistem Keselurahan Menggunakan Mikrokontroler

Diagram alir pemrosesan data secara keseluruhan menggunakan mikrokontroler dapat digambarkan pada Gambar 3.12 dan 3.13.

Pada awal diagram alir, mikrokontroler akan melakukan inisialisasi awal. Inisialisasi awal meliputi konfigurasi port serial yang digunakan yaitu berupa kecepatan baud, mode serial yang digunakan, port yang digunakan untuk sensor sonar, motor servo, dan kompas digital lalu mode timer yang dipakai. Setelah itu, mikrokontroler melakukan deklarasi counter untuk putaran servo. Counter ini digunakan untuk pengecekan putaran servo yang telah dilakukan.

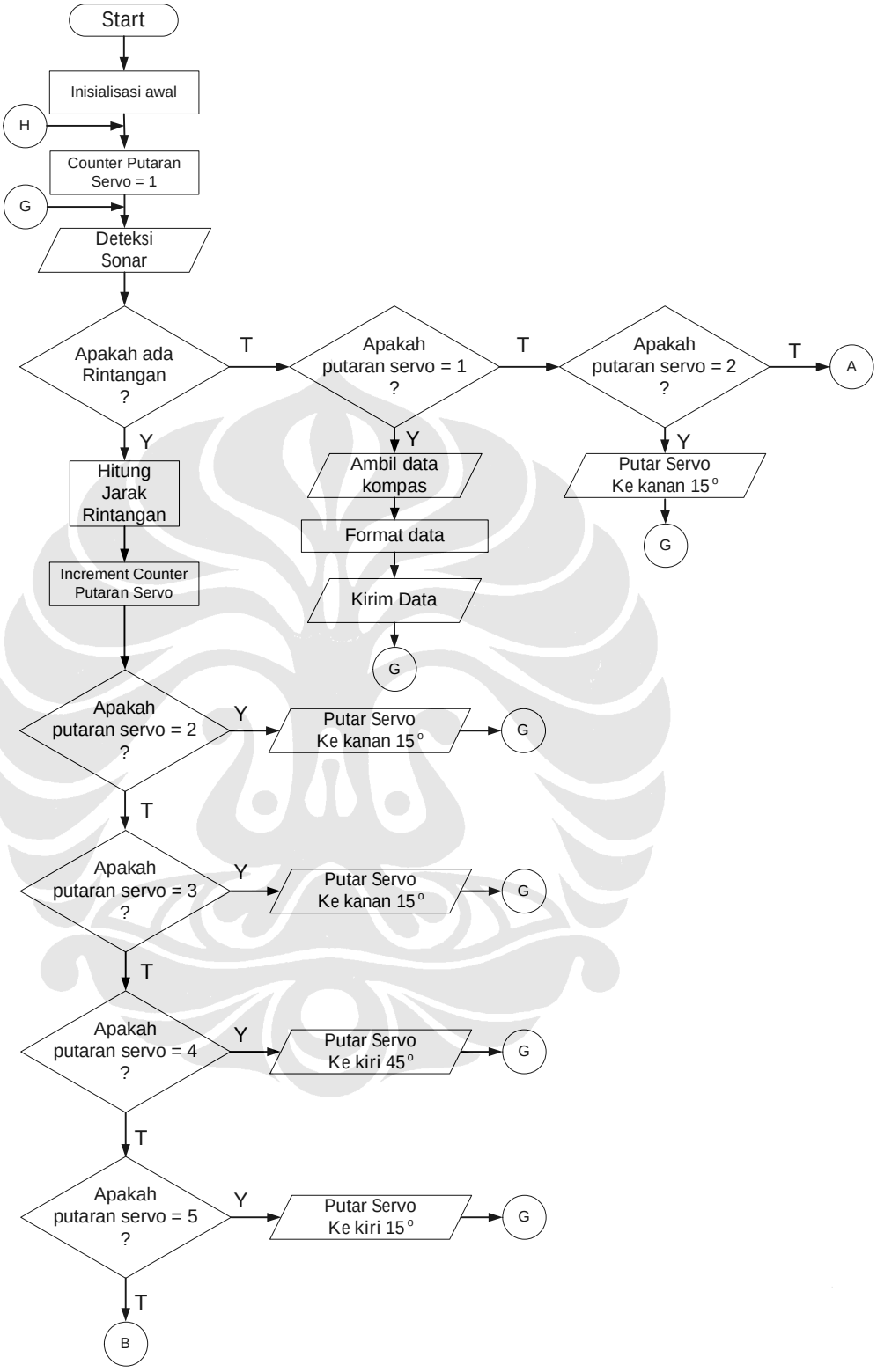
Setelah itu mikrokontroler akan melakukan deteksi sonar. Bila tidak ada rintangan maka mikrokontroler akan mengambil data dari kompas digital. Setelah itu mikrokontroler akan mengirimkan data ke komputer. Bila sensor sonar mendeteksi ada rintangan maka mikrokontroler akan memutar motor servo. Pada saat servo diputar mikrokontroler akan melakukan deteksi kembali. Proses putaran servo akan dilakukan sebanyak 5 kali. Setelah selesai maka mikrokontroler akan mengambil data dari kompas digital. Setelah itu, mikrokontroler akan mengirimkan data ke komputer. Sebelum melakukan pengiriman, data yang didapat dari hasil perhitungan jarak dan kompas digital dijadikan dalam 1 format.

Berikut format data yang dikirimkan :

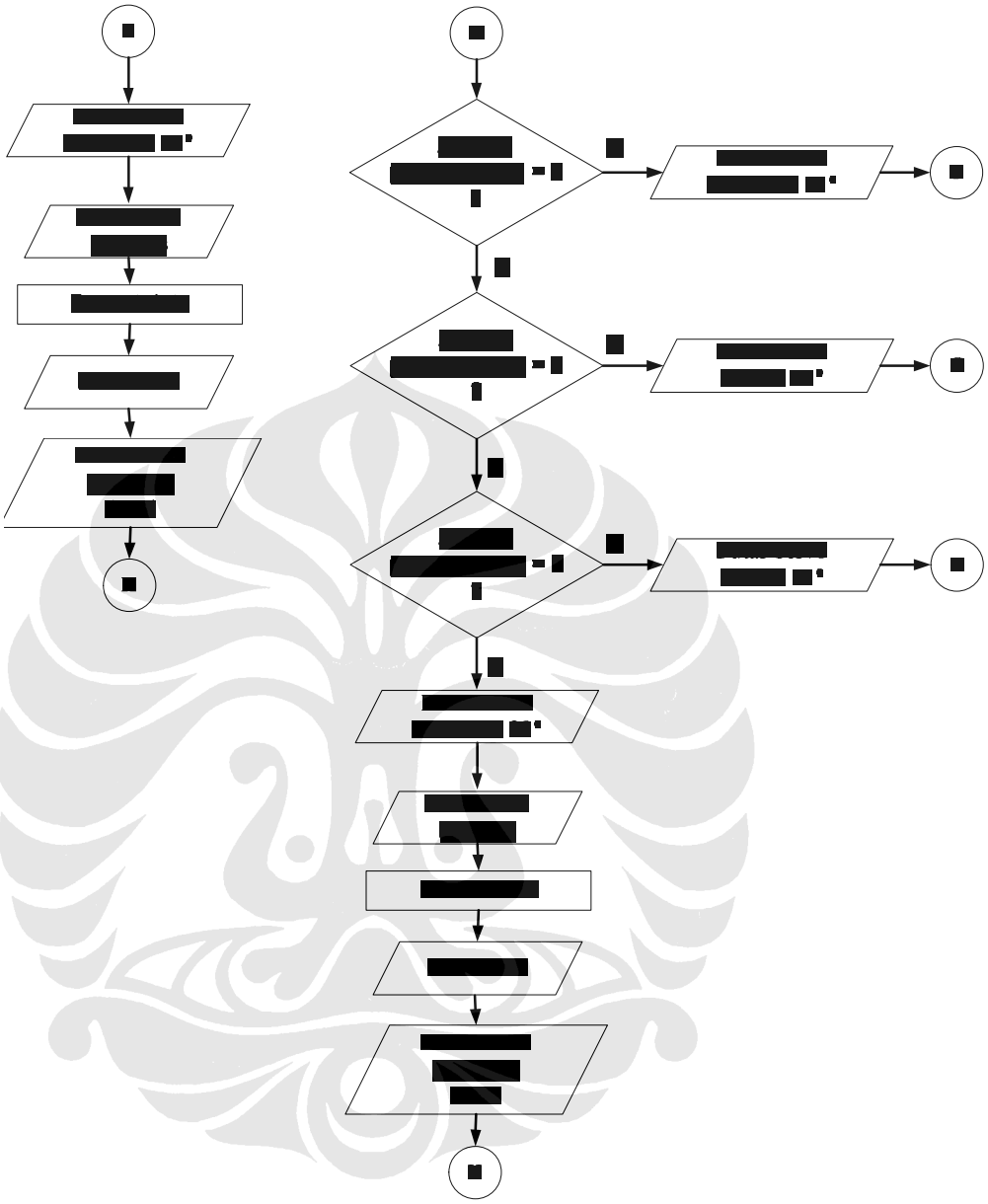
"C" ; "," ; Rintangan(1) ; "," ; "R10" ; "," ; Rintangan (2) ; "," ; "R20" ; "," ; Rintangan (3) ; "," ; "L10" ; "," ; Rintangan (4) ; "," ; "L20" ; "," ; Rintangan (5) ; "," ; "Compass" ; "," ; Y ; "," ; "U"

Ket :

- Rintangan merupakan hasil perhitungan jarak sensor ke objek yang dideteksi.
- ”C” adalah *header* data rintangan pada posisi tengah.
- ”R10” adalah *header* data pada posisi kanan dengan sudut 15^{o} dari tengah.
- ”R20” adalah *header* data pada posisi kanan dengan sudut 30^{o} dari tengah.
- ”L10” adalah *header* data pada posisi kiri dengan sudut 15^{o} dari tengah.
- ”L20” adalah *header* data pada posisi kiri dengan sudut 30^{o} dari tengah.
- Y adalah nilai data dari kompas digital
- ”U” adalah penunjuk Utara kompas



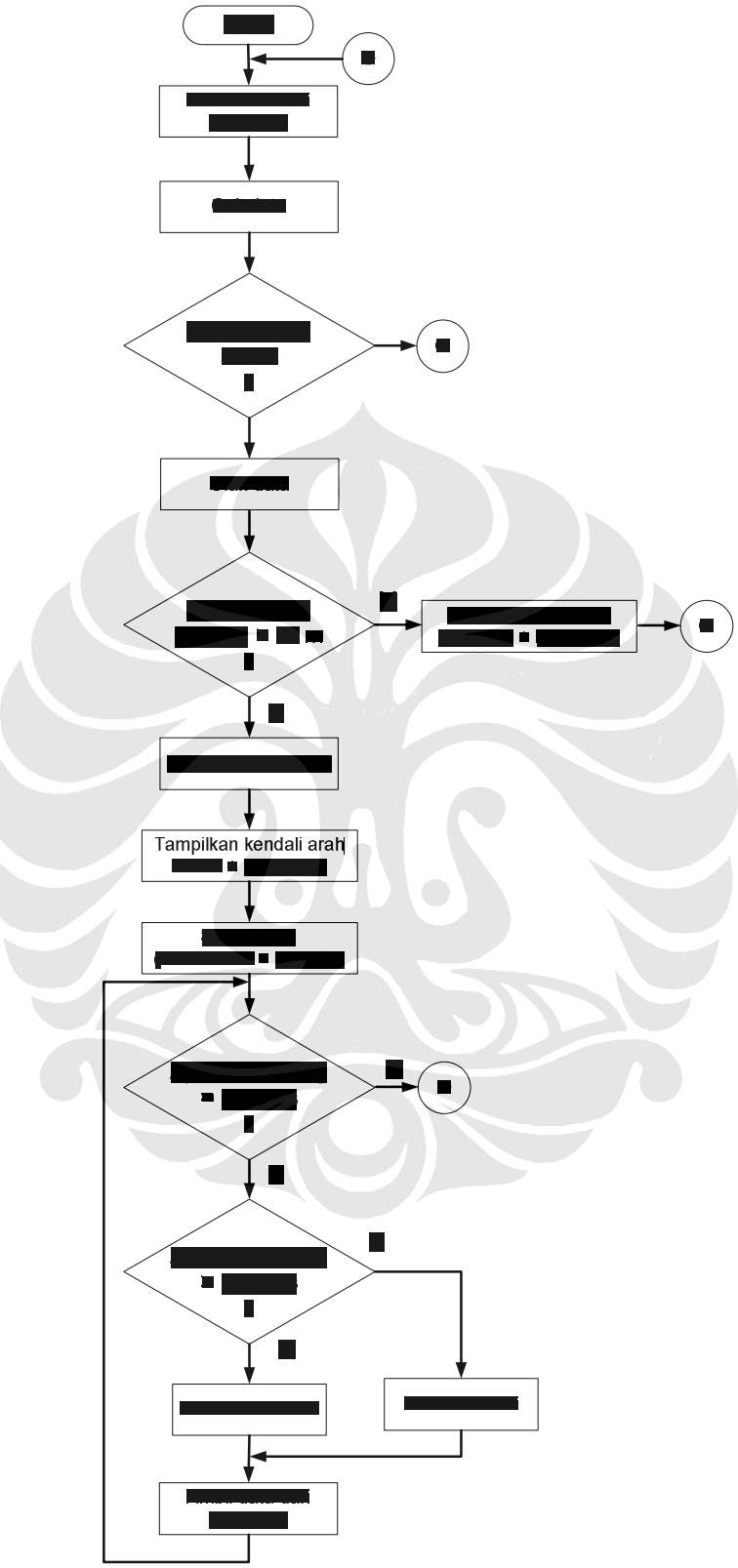
Gambar 3.12 Diagram Alir keselurahan 1



Gambar 3.13 Diagram Alir keselurahan 2

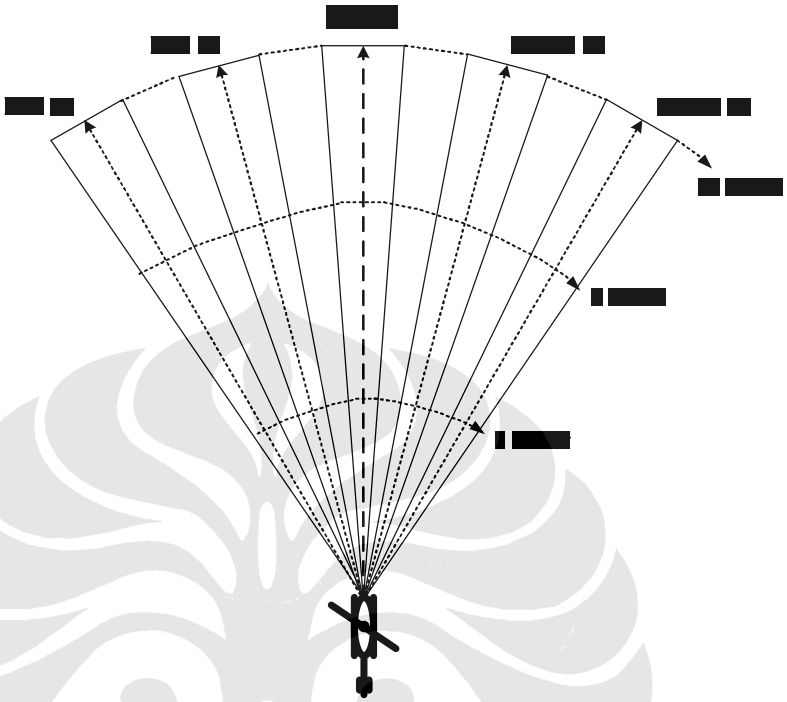
3.4 PROGRAM SISTEM NAVIGASI HELIKOPTER

Diagram alir keseluruhan sistem navigasi helikopter dapat dilihat pada gambar 3.14



Gambar 3.14 Diagram Alir program navigasi

Untuk proses pendeteksian rintangan akan dirancang seperti gambar dibawah ini:



Gambar 3.15 Daerah Kerja Sonar

Daerah kerja sonar dibagi menjadi 5 yaitu tengah sebagai daerah acuan utama sensor sonar dan 4 yang lainnya sebagai daerah *scanning* sonar. Sedangkan pada bagian jarak deteksi akan dibagi menjadi 4 bagian yaitu jarak $\geq$ 10 meter, 7 $\leq$ jarak < 10 meter, 3 < jarak < 7 meter dan jarak $\leq$ 3 meter. Dari beberapa parameter ini, maka dapat dibuat tabel kondisi bersyarat seperti berikut :

Tabel 3.1 Logika kondisi daerah tengah

Kondisi		**Input**	**Ouput**
		Tengah	**Kecepatan**
c		s $\geq$ 10 m	Fast
nc	jauh	7 $\leq$ s < 10	Normal
	sedang	3 < s < 7	Slow
	dekat	s $\leq$ 3	Stop

Dari tabel diatas dapat dilhat, sensor daerah tengah dijadikan sebagai *input*. Bila menggunakan bahasa verbal maka **c** (*clear*) digunakan bila tidak ada rintangan atau jarak $\geq$ 10 meter dan **nc** (*not clear*) digunakan bila ada rintangan dengan jarak < 10 meter. Pada **nc** ini, terdapat 3 pembagian yaitu jauh, sedang dan dekat.

Untuk parameter *output* adalah kecepatan. Besarnya kecepatan ini disesuaikan jarak yang dideteksi oleh sensor sonar. Karena daerah tengah digunakan sebagai acuan utama, maka disaat tidak ada rintangan maka besaranya kecepatan adalah *fast*. Bila dibuat menjadi algoritmanya maka menjadi:

If Tengah is c Then Kecepatan is Fast

If Tengah is jauh Then Kecepatan is Normal

If Tengah is sedang Then Kecepatan is Slow

If Tengah is dekat Then Kecepatan is Stop

Namun, bila pada daerah tengah ini terdapat rintangan maka sensor akan melakukan *scanning* ke empat daerah yaitu kanan sebesar 15^o dan 30^o serta kiri sebesar 15^o dan 30^o. Tabel kondisi untuk 4 daerah *scanning* dapat dilihat pada tabel berikut :

Tabel 3.2 Logika kondisi untuk daerah *scanning*

No	Input				Ouput	
	Kiri 30	Kiri 15	Kanan 15	Kanan 30	Control	Kecepatan
1	c	c	c	c	Kanan +15	Normal
2	nc	c	c	c	Kanan +15	Normal
3	c	nc	c	c	Kanan +15	Normal
4	nc	nc	c	c	Kanan +15	Normal
5	c	c	c	nc	Kiri +15	Normal
6	c	c	nc	c	Kiri +15	Normal
7	c	c	nc	nc	Kiri +15	Normal
8	nc	c	c	nc	???	---
9	c	nc	c	nc	???	---
10	nc	c	nc	c	???	---
11	c	nc	nc	c	???	---
12	nc	nc	c	nc	???	---
13	nc	nc	nc	c	???	---
14	nc	c	nc	nc	???	---
15	c	nc	nc	nc	???	---
16	nc	nc	nc	nc	Naik	Stop

Dari tabel 3.2 dapat dilihat terdapat 16 kemungkinan yang ada. Karena proses *scanning* yang dilakukan ke arah kanan terlebih dahulu, maka daerah kanan digunakan sebagai acuan utama. Jika pada daerah kanan 15 dan 30 (no.1-4) tidak ada rintangan maka helikopter akan bergerak ke kanan 15 dengan kecepatan normal. Bila pada daerah kanan terdapat rintangan(no.5-7) maka acuan dilihat ke daerah kiri. Jika pada daerah kiri 15 dan 30 tidak ada rintangan maka helikopter

akan bergerak ke kiri 15 dengan kecepatan normal. Bila pada semua daerah terdapat rintangan (no.16) maka helikopter harus berhenti dan bergerak naik ke atas. Dari beberapa pernyataan ini, maka dapat dibuat algoritmanya dengan ***If... Then...*** yaitu :

If Kanan 15 AND Kanan 30 is c Then Control is Kanan+15 AND Kecepatan is Normal

If Kanan 15 OR Kanan 30 is nc AND Kiri 15 AND Kiri 30 is c Then Control is Kiri+15 AND Kecepatan is Normal.

If Kanan 15 AND Kanan 30 AND Kiri 15 AND Kiri 30 is nc Then Control is Naik AND Kecepatan is Stop.

Bila pada daerah kiri dan kanan terdapat 1 rintangan(no.8 – 11 tabel 3.2) maka tabel kondisi perlu dibagi lagi seperti berikut :

Tabel 3.3. Kondisi Kanan 15 dan Kiri 15 tidak ada rintangan

Input				**Ouput**	
Kiri 30	**Kiri 15**	**Kanan 15**	**Kanan 30**	**Control**	**Kecepatan**
nc	c	c	nc	???	---
dekat			dekat	Naik	Stop
sedang			dekat	Kiri +15	Slow
jauh			dekat	Kiri +15	Normal
dekat			sedang	Kanan +15	Slow
sedang			sedang	Kanan +15	Slow
jauh			sedang	Kiri +15	Normal
x			jauh	Kanan +15	Normal

Algoritma untuk tabel 3.3 adalah sebagai berikut :

If Kanan 15 AND Kiri 15 is c AND Kiri 30 is dekat AND Kanan 30 is dekat Then Control is Naik AND Kecepatan is Stop.

If Kanan 15 AND Kiri 15 is c AND Kiri 30 is sedang AND Kanan 30 is dekat Then Control is Kiri+15 AND Kecepatan is Slow.

If Kanan 15 AND Kiri 15 is c AND Kiri 30 is jauh AND Kanan 30 is dekat Then Control is Kiri+15 AND Kecepatan is Normal.

If Kanan 15 AND Kiri 15 is c AND Kiri 30 is dekat AND Kanan 30 is sedang Then Control is Kanan+15 AND Kecepatan is Slow.

If Kanan 15 AND Kiri 15 is c AND Kiri 30 is sedang AND Kanan 30 is sedang Then Control is Kanan+15 AND Kecepatan is Slow.

If Kanan 15 AND Kiri 15 is c AND Kiri 30 is jauh AND Kanan 30 is sedang Then Control is Kiri+15 AND Kecepatan is Normal.

If Kanan 15 AND Kiri 15 is c Kanan 30 is jauh Then Control is Kanan+15 AND Kecepatan is Normal.

Tabel 3.4 Kondisi untuk Kanan 30 dan Kiri 15 tidak ada rintangan

Input				Ouput	
Kiri 30	**Kiri 15**	**Kanan 15**	**Kanan 30**	**Control**	**Kecepatan**
nc	c	nc	c	???	---
dekat		dekat		Naik	Stop
dekat		sedang		Kanan +30	Slow
dekat		jauh		Kanan +30	Normal
sedang		dekat		Kiri +15	Slow
sedang		sedang		Kanan +30	Slow
sedang		jauh		Kanan +30	Normal
jauh		x		Kiri +15	Normal

Algoritma untuk tabel 3.4 adalah sebagai berikut :

If Kanan 30 AND Kiri 15 is c AND Kiri 30 is dekat AND Kanan 15 is dekat Then Control is Naik AND Kecepatan is Stop.

If Kanan 30 AND Kiri 15 is c AND Kiri 30 is dekat AND Kanan 15 is sedangThen Control is Kanan+30 AND Kecepatan is Slow.

If Kanan 30 AND Kiri 15 is c AND Kiri 30 is dekat AND Kanan 15 is jauh Then Control is Kanan+30 AND Kecepatan is Normal.

If Kanan 30 AND Kiri 15 is c AND Kiri 30 is sedang AND Kanan 15 is dekat Then Control is Kiri+15 AND Kecepatan is Slow.

If Kanan 30 AND Kiri 15 is c AND Kiri 30 is sedang AND Kanan 15 is sedang Then Control is Kanan+30 AND Kecepatan is Slow.

If Kanan 30 AND Kiri 15 is c AND Kiri 30 is sedang AND Kanan 15 is jauh Then Control is Kanan+30 AND Kecepatan is Normal.

If Kanan 30 AND Kiri 15 is c AND Kiri 30 is jauh Then Control is Kiri+15 AND Kecepatan is Slow.

Tabel 3.5 Kondisi untuk Kanan 15 dan Kiri 30 tidak ada rintangan

Input				Ouput	
Kiri 30	**Kiri 15**	**Kanan 15**	**Kanan 30**	**Control**	**Kecepatan**
c	nc	c	nc	???	---
	dekat		dekat	Naik	Stop
	sedang		dekat	Kiri +30	Slow
	jauh		dekat	Kiri +30	Normal
	dekat		sedang	Kanan +15	Slow
	sedang		sedang	Kiri +30	Slow
	jauh		sedang	Kiri +30	Normal
	x		jauh	Kanan +15	Normal

Algoritma untuk tabel 3.5 adalah sebagai berikut :

If Kanan 15 AND Kiri 30 is c AND Kiri 15 is dekat AND Kanan 30 is dekat Then Control is Naik AND Kecepatan is Stop.

If Kanan 15 AND Kiri 30 is c AND Kiri 15 is sedang AND Kanan 30 is dekat Then Control is Kiri+30 AND Kecepatan is Slow.

If Kanan 15 AND Kiri 30 is c AND Kiri 15 is jauh AND Kanan 30 is dekat Then Control is Kiri+30 AND Kecepatan is Normal.

If Kanan 15 AND Kiri 30 is c AND Kiri 15 is dekat AND Kanan 30 is sedang Then Control is Kanan+15 AND Kecepatan is Slow.

If Kanan 15 AND Kiri 30 is c AND Kiri 15 is sedang AND Kanan 30 is sedang Then Control is Kiri+30 AND Kecepatan is Slow.

If Kanan 15 AND Kiri 30 is c AND Kiri 15 is jauh AND Kanan 30 is sedang Then Control is Kiri+30 AND Kecepatan is Normal.

If Kanan 15 AND Kiri 30 is c AND Kanan 30 is jauh Then Control is Kanan+15 AND Kecepatan is Normal.

Tabel 3.6 Kondisi untuk Kanan 30dan Kiri 30 tidak ada rintangan

Input				Ouput	
Kiri 30	**Kiri 15**	**Kanan 15**	**Kanan 30**	**Control**	**Kecepatan**
c	nc	nc	c	???	---
	dekat	dekat		Naik	Stop
	sedang	dekat		Kiri +30	Slow
	jauh	dekat		Kiri +30	Normal
	dekat	sedang		Kanan +30	Slow
	sedang	sedang		Kanan +30	Slow
	jauh	sedang		Kiri +30	Normal
	x	jauh		Kanan +30	Normal

Algoritma untuk tabel 3.6 adalah sebagai berikut :

If Kanan 30 AND Kiri 30 is c AND Kiri 15 is dekat AND Kanan 15 is dekat Then Control is Naik AND Kecepatan is Stop.

If Kanan 30 AND Kiri 30 is c AND Kiri 15 is sedang AND Kanan 15 is dekat Then Control is Kiri+30 AND Kecepatan is Slow.

If Kanan 30 AND Kiri 30 is c AND Kiri 15 is jauh AND Kanan 15 is dekat Then Control is Kiri+30 AND Kecepatan is Normal.

If Kanan 30 AND Kiri 30 is c AND Kiri 15 is dekat AND Kanan 15 is sedang Then Control is Kanan+30 AND Kecepatan is Slow.

If Kanan 30 AND Kiri 30 is c AND Kiri 15 is sedang AND Kanan 15 is sedang Then Control is Kanan+30 AND Kecepatan is Slow.

If Kanan 30 AND Kiri 30 is c AND Kiri 15 is jauh AND Kanan 15 is sedang Then Control is Kiri+30 AND Kecepatan is Normal.

If Kanan 30 AND Kiri 30 is c AND KANAN 15 is jauh Then Control is Kanan+30 AND Kecepatan is Normal.

Bila dari 4 daerah *scanning* terdapat 3 rintangan atau hanya 1 daerah yang tidak ada rintangan, maka tabel kondisi perlu dibagi lagi seperti berikut :

Tabel 3.7 Logika kondisi untuk Kanan 15 tidak ada rintangan

Input				**Ouput**	
Kiri 30	**Kiri 15**	**Kanan 15**	**Kanan 30**	**Control**	**Kecepatan**
nc	nc	c	nc	???	---
X	X		dekat	Naik	Stop
			sedang	Kanan +15	Slow
			jauh	Kanan +15	Normal

Algoritma untuk tabel 3.7 adalah sebagai berikut :

If Kanan 15 is c AND Kanan 30 is dekat AND Kiri 15 AND Kiri 30 is nc Then Control is Naik AND Kecepatan is Stop.

If Kanan 15 is c AND Kanan 30 is sedang AND Kiri 15 AND Kiri 30 is nc Then Control is Kanan+15 AND Kecepatan is Slow.

If Kanan 15 is c AND Kanan 30 is jauh AND Kiri 15 AND Kiri 30 is nc Then Control is Kanan+15 AND Kecepatan is Normal.

Tabel 3.8 Kondisi untuk Kanan 30 tidak ada rintangan

Input				Ouput	
Kiri 30	**Kiri 15**	**Kanan 15**	**Kanan 30**	**Control**	**Kecepatan**
nc	nc	nc	c	???	---
X	X	dekat		Naik	Stop
		sedang		Kanan +30	Slow
		jauh		Kanan +30	Normal

Algoritma untuk tabel 3.8 adalah sebagai berikut :

If Kanan 30 is c AND Kanan 15 is dekat AND Kiri 15 AND Kiri 30 is nc Then Control is Naik AND Kecepatan is Stop .

If Kanan 30 is c AND Kanan 15 is sedang AND Kiri 15 AND Kiri 30 is nc Then Control is Kanan+30 AND Kecepatan is Slow.

If Kanan 30 is c AND Kanan 15 is jauh AND Kiri 15 AND Kiri 30 is nc Then Control is Kanan+30 AND Kecepatan is Normal.

Tabel 3.9 Kondisi untuk Kiri 15 tidak ada rintangan

Input				Ouput	
Kiri 30	**Kiri 15**	**Kanan 15**	**Kanan 30**	**Control**	**Kecepatan**
nc	c	nc	nc	???	---
dekat		X	X	Naik	Stop
sedang				Kiri +15	Slow
jauh				Kiri +15	Normal

Algoritma untuk tabel 3.9 adalah sebagai berikut :

If Kiri 15 is c AND Kiri 30 is dekat AND Kanan 15 AND Kanan 30 is nc Then Control is Naik AND Kecepatan is Stop.

If Kiri 15 is c AND Kiri 30 is sedang AND Kanan 15 AND Kanan 30 is nc Then Control is Kiri+15 AND Kecepatan is Slow.

If Kiri 15 is c AND Kiri 30 is jauh AND Kanan 15 AND Kanan 30 is nc Then Control is Kiri+15 AND Kecepatan is Normal.

Tabel 3.10 Kondisi untuk Kiri 30 tidak ada rintangan

Input				Ouput	
Kiri 30	**Kiri 15**	**Kanan 15**	**Kanan 30**	**Control**	**Kecepatan**
c	nc	nc	nc	???	---
	dekat	X	X	Naik	Stop
	sedang			Kiri +30	Slow
	jauh			Kiri +30	Normal

Algoritma untuk tabel 3.10 adalah sebagai berikut :

If Kiri 30 is c AND Kiri 15 is dekat AND Kanan 15 AND Kanan 30 is nc Then Control is Naik AND Kecepatan is Stop.

If Kiri 30 is c AND Kiri 15 is sedang AND Kanan 15 AND Kanan 30 is nc Then Control is Kiri+30 AND Kecepatan is Slow.

If Kiri 30 is c AND Kiri 15 is jauh AND Kanan 15 AND Kanan 30 is nc Then Control is Kiri+30 AND Kecepatan is Normal.

BAB IV

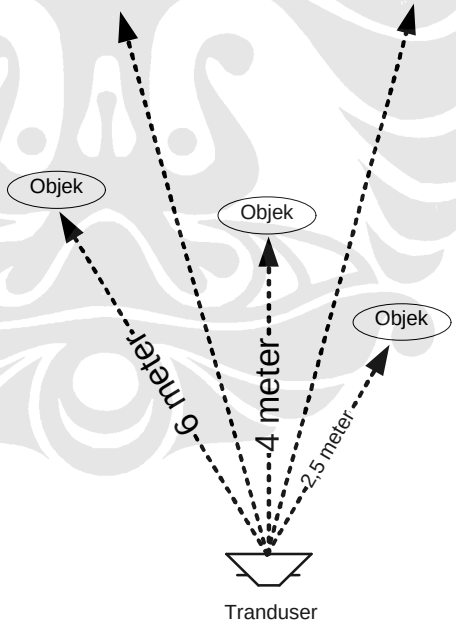
PENGUJIAN DAN ANALISIS SISTEM

4.1 PENGUJIAN SISTEM

Pengujian sistem dilakukan untuk menguji program yang telah dibuat apakah sudah sesuai dengan rancangan atau tidak. Pengujian yang akan dilakukan yaitu menguji penerimaan data yang dikirimkan dari mikrokontroler ke komputer secara *wireless*. Setelah itu menguji program sistem navigasi yang telah dibuat.

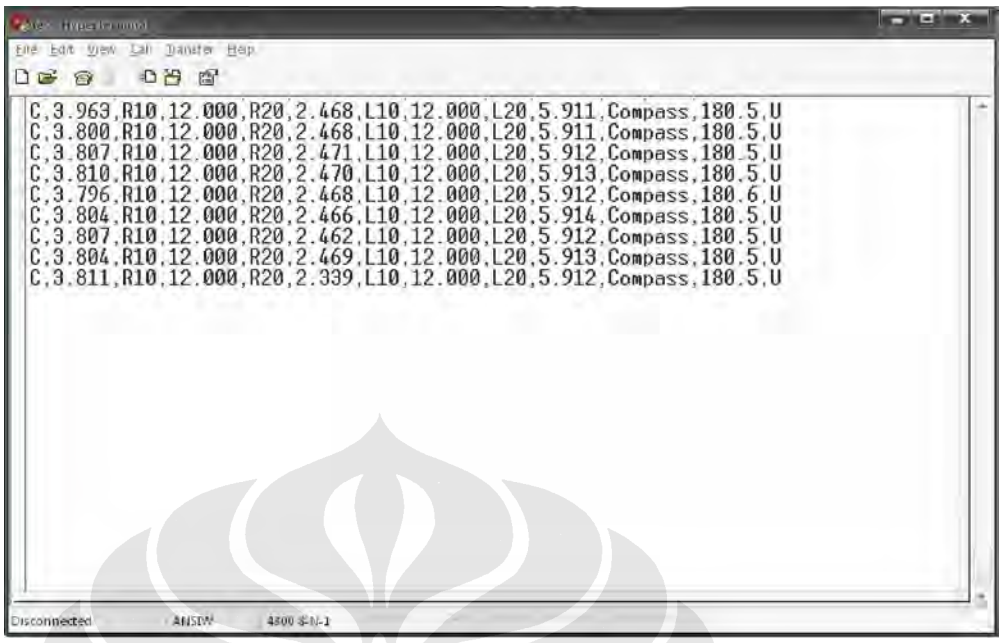
4.1.1 Hasil Pengujian Penerimaan Data yang Dikirimkan

Pengujian yang pertama adalah penerimaan data menggunakan program *hyper terminal* pada Windows. Skema pengujian terhadap objek deteksi adalah sebagai berikut :



Gambar 4.1 Skema pengujian 1

Tranduser pada skema pengujian 1 ini berada dalam kondisi diam atau tidak bergerak. Berikut ini adalah hasil yang diterima oleh komputer :

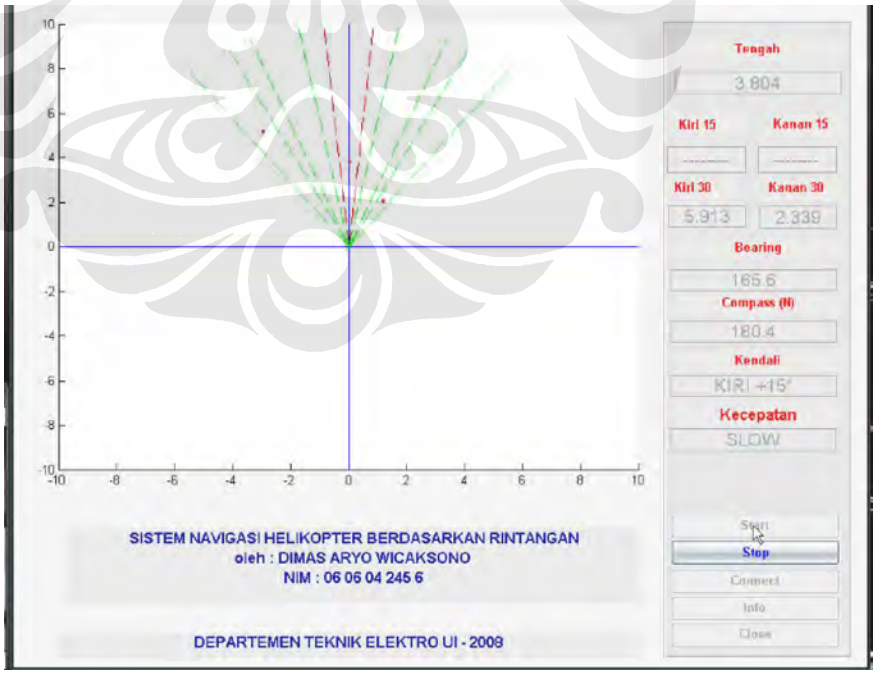


Gambar 4.2 Hasil data yang diterima oleh komputer

4.1.2 Hasil Pengujian Program Navigasi

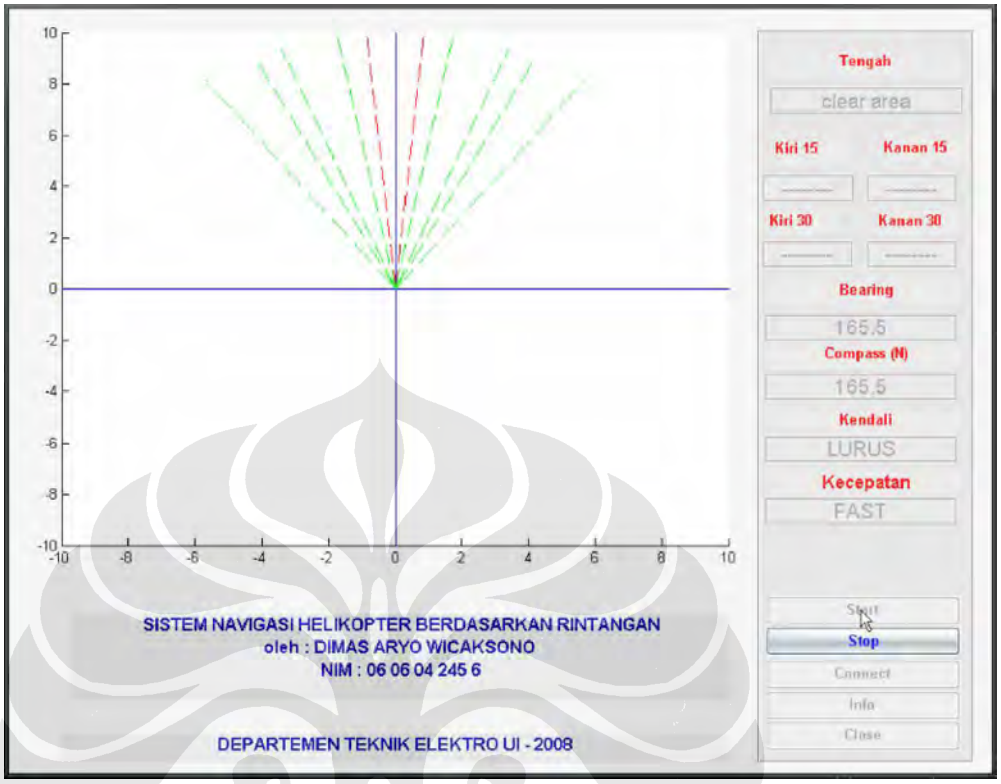
Pengujian program navigasi dilakukan dengan menjalankan program yang telah dibuat. Berikut ini adalah hasil – hasil pengujiannya :

- Hasil pengujian program sesuai dengan skema pengujian 1



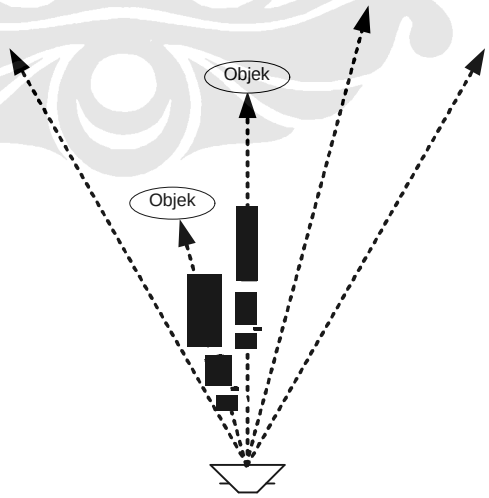
Gambar 4.3 Hasil pengujian sistem navigasi 1

- Hasil pengujian ketika setelah melewati rintangan



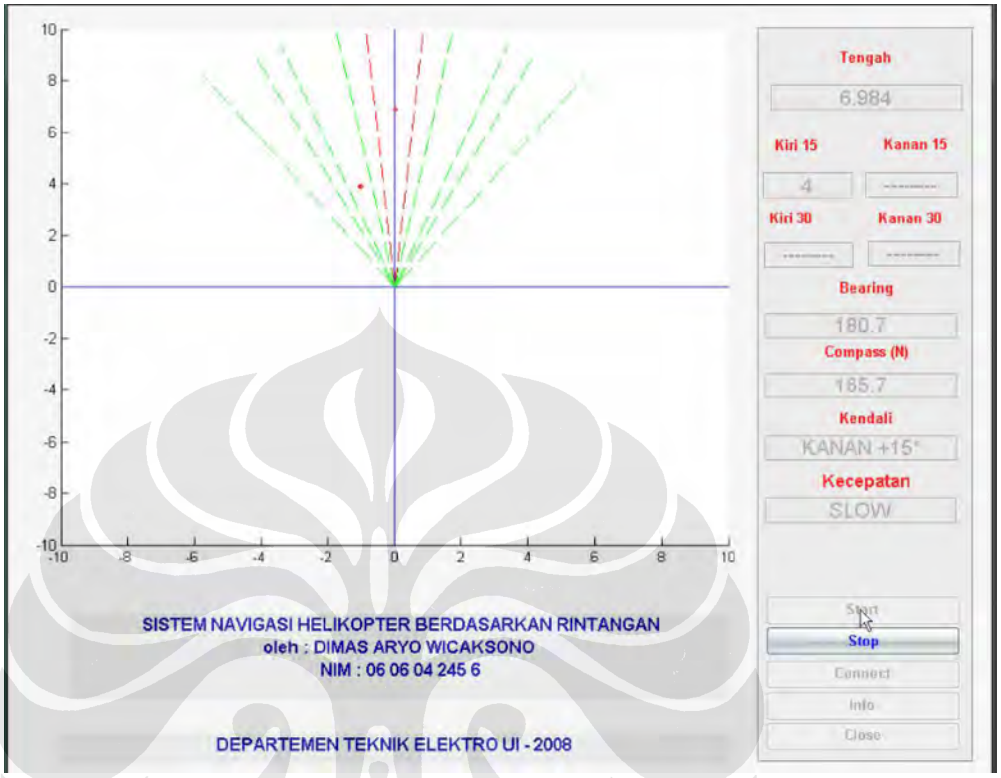
Gambar 4.4 Hasil pengujian setelah melewati rintangan

Selain skema pengujian 1, ada beberapa skema lagi yaitu ketika tidak ada rintangan di daerah kanan, dan ketika hanya daerah kanan 30^0 saja yang tidak ada rintangan.

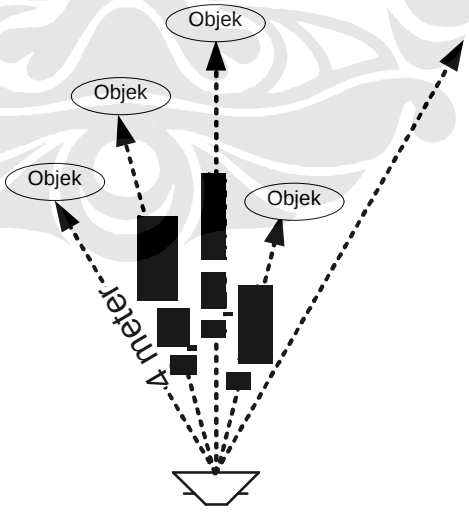


Gambar 4.5 Skema pengujian 2

- Hasil pengujian dengan skema 2.

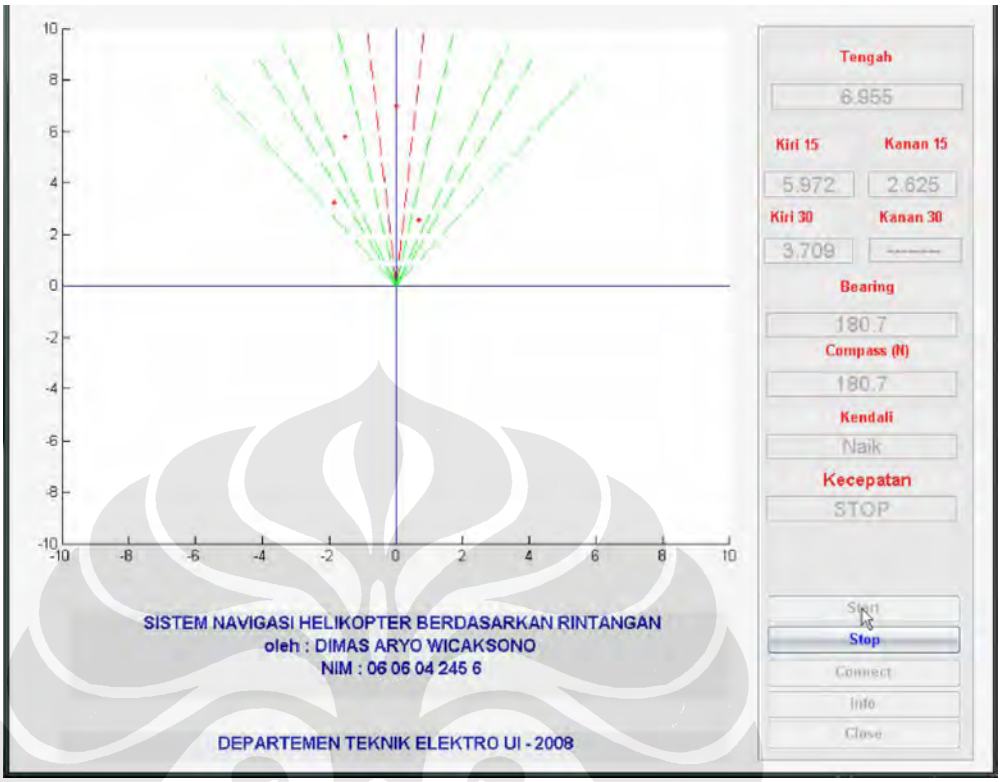


Gambar 4.6 Hasil pengujian dengan skema 2



Gambar 4.7 Skema pengujian 3

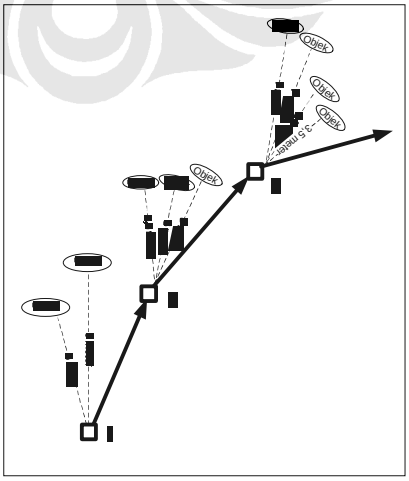
- Hasil pengujian dengan skema 3.



Gambar 4.8 Hasil pengujian dengan skema 3

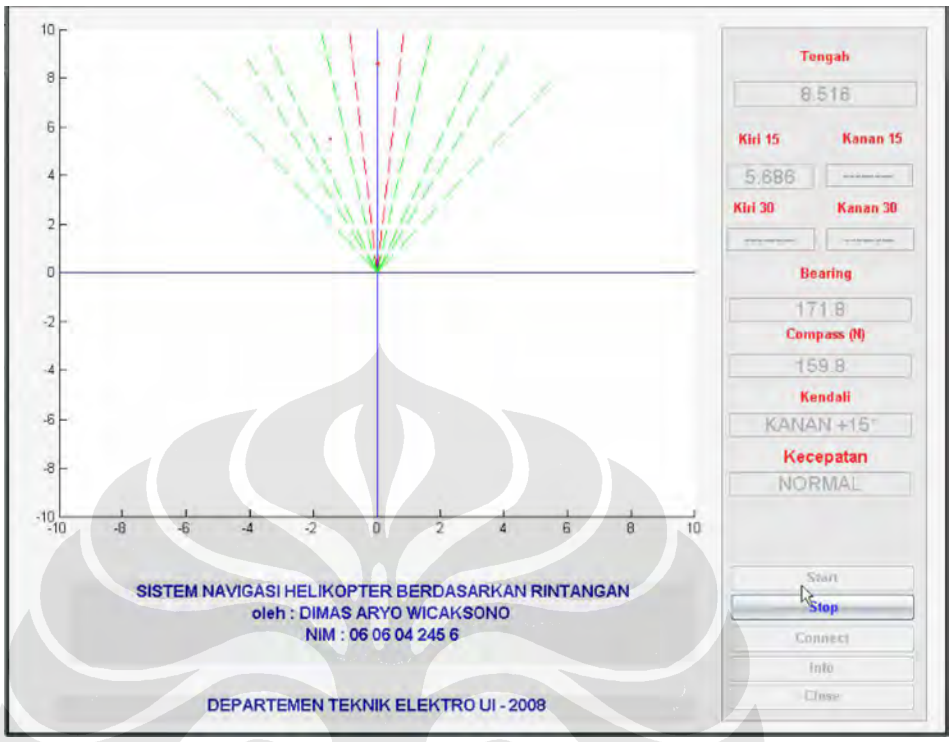
4.1.3 Hasil Pengujian Program Navigasi Dalam Keadaan Bergerak

Berbeda dengan sebelumya, pengujian program navigasi berikutnya dilakukan dalam keadaan bergerak dengan beberapa rintangan. Percobaan pertama dilakukan dengan 2 rintangan, berikutnya dengan 3 rintangan dan yang terakhir dengan 4 rintangan. Skema pengujian dapat dilihat pada gambar berikut :

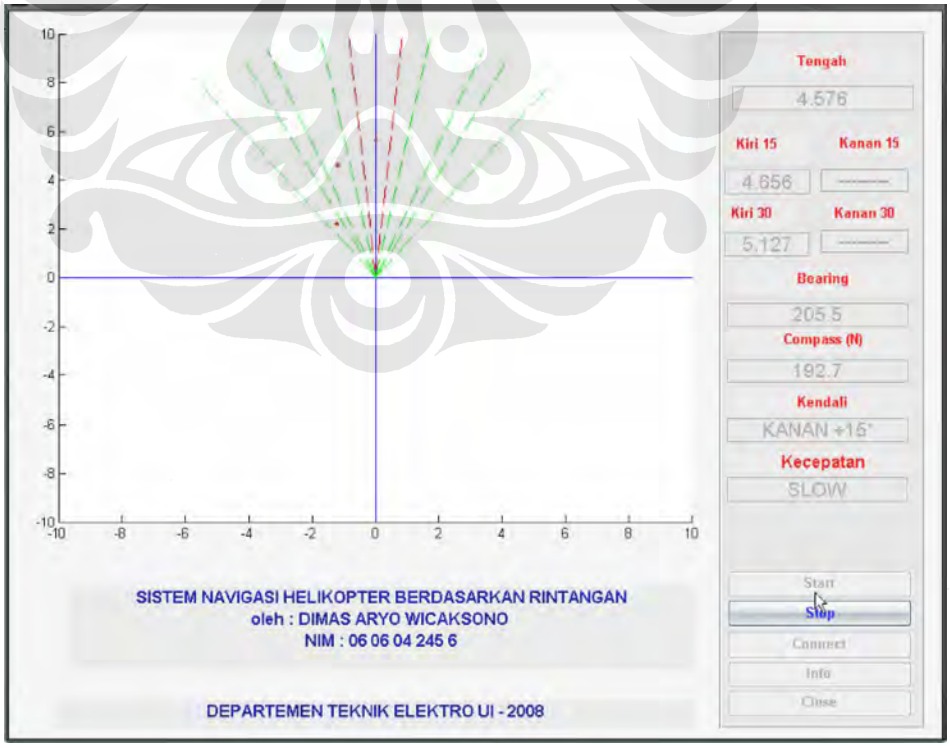


Gambar 4.9 Skema pengujian 4

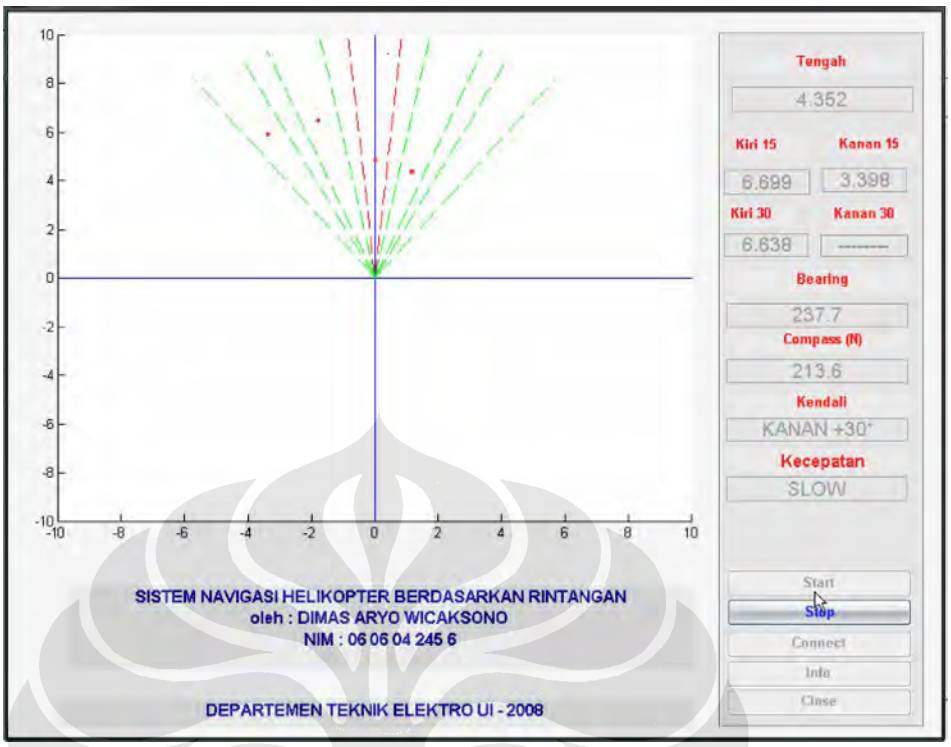
- Hasil – hasil pengujian skema 4



Gambar 4.10 Hasil pengujian 1 dari skema 4



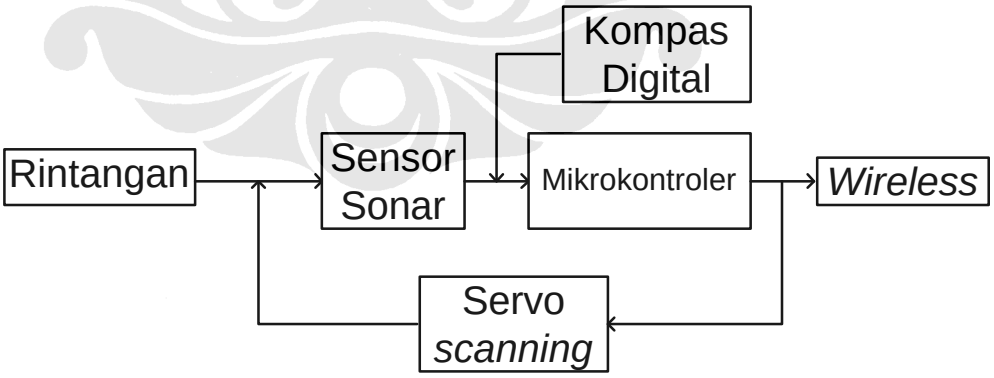
Gambar 4.11 Hasil pengujian 2 dari skema 4



Gambar 4.12 Hasil pengujian 3 dari skema 4

4.2 ANALISIS SISTEM

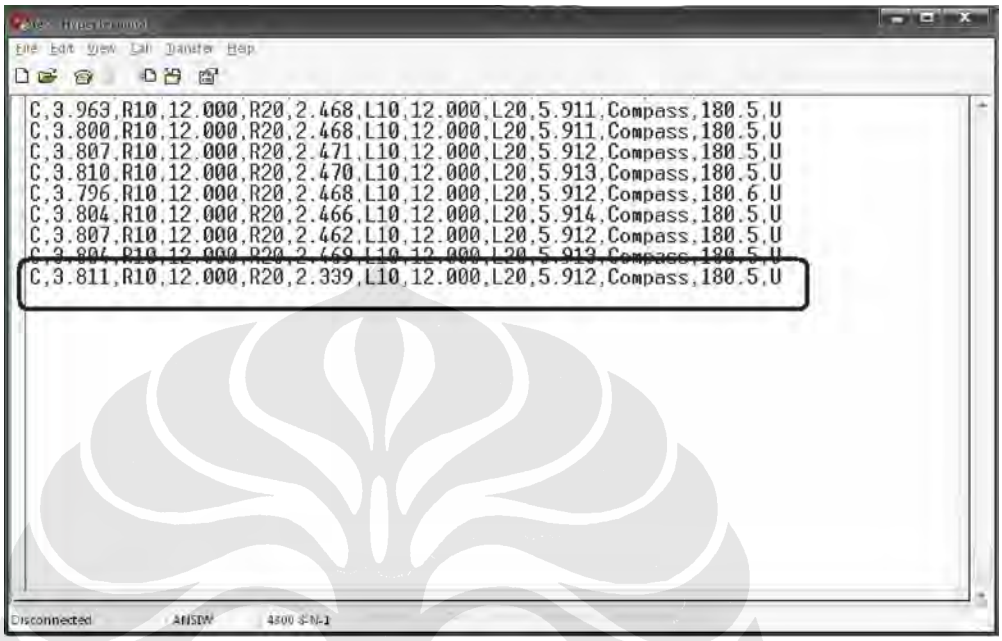
Setelah melakukan pengujian, maka dilakukan analisis dari sistem yang dirancang.



Gambar 4.13 Perangkat keras sistem navigasi

Berdasarkan perancangan yang dibuat, sistem terdiri dari perangkat keras yang berfungsi untuk mengambil data rintangan yang dideteksi oleh sensor sonar serta data arah posisi dari kompas digital. Setelah data didapat, maka

mikrokontroler mengirimkan data tersebut ke komputer secara *wireless.* Seperti yang didapat dari hasil pengujian pertama pada sub bagian 4.1.1 terlihat bahwa komputer dapat menerima data yang dikirimkan oleh mikrokontroler.



Gambar 4.14 Hasil penerimaan data pada komputer

Pembacaan data diatas (yang dilingkari oleh garis hitam) adalah

- C,3.811, : pada daerah tengah terdeteksi objek dengan jarak 3,811 meter.
- R.10,12.000 : pada daerah kanan 15^o tidak terdeteksi objek.
- R.20,2,339 : pada daerah kanan 30^o terdeteksi objek dengan jarak 2,339 meter.
- L.10,12.000 : pada daerah kiri 15^o tidak terdeteksi objek.
- L.20,5.912 : pada daerah kiri 30^o terdeteksi objek dengan jarak 5,912 meter.
- Compass,180.5,U : hasil pembacaan pada kompas digital

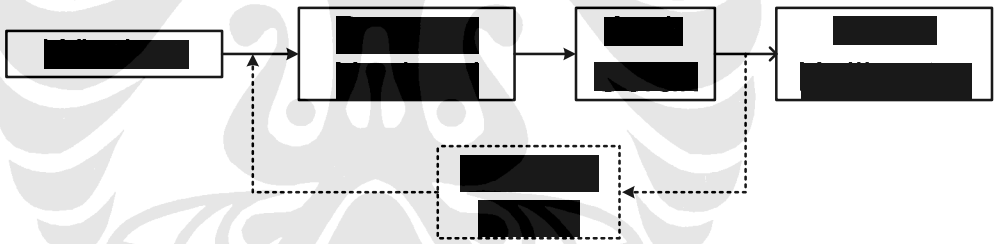
Dari data yang didapat tersebut, terlihat bahwa hasil pendeteksian sesuai dengan skema yang diujikan dimana terdapat objek pada bagian tengah, kanan 30^o dan kiri 30^o dan tidak terdapat objek pada bagian kanan 15^o dan kiri 15^o. Sedangkan jarak yang didapat dari pendeteksian terdapat perbedaan, seperti yang terlihat pada tabel 4.1 berikut ini :

Tabel 4.1. Hasil pembacaan sensor sonar

Rintangan	Jarak yang diukur	Hasil pembacaan sensor	Rintangan	Jarak yang diukur	Hasil pembacaan sensor	Rintangan	Jarak yang diukur	Hasil pembacaan sensor
Daerah tengah	4 meter	3.963	Daerah kanan 30	2,5 meter	2.468	Daerah kiri 30	6 meter	5.911
		3.8			2.468			5.911
		3.807			2.471			5.912
		3.81			2.47			5.913
		3.796			2.468			5.912
		3.804			2.466			5.914
		3.807			2.462			5.912
		3.804			2.469			5.913
		3.811			2.339			5.912
	Rata - rata	3.8224444		Rata - rata	2.4534444		Rata - rata	5.9122222
	% kesalahan	4.4388889		% kesalahan	1.8622222		% kesalahan	1.462963

Dari data yang terdapat pada tabel 4.1, dapat terlihat bahwa ketelitian pembacaan sensor sonar terdapat kesalahan sebesar ± 1 – 5 %. Sedangkan dari beberapa pengambilan data yang didapat terlihat bahwa sensor memiliki *repeatablity* yang baik dalam mendeteksi objek di posisi yang sama.

Setelah pengujian penerimaan data pada komputer dilakukan, maka selanjutnya dilakukan pengujian dengan menggunakan program navigasi yang telah dibuat seperti terlihat pada gambar di bawah ini :



Gambar 4.15 Pengolah data dari Sistem navigasi

Dari gambar diatas terlihat data yang diterima langsung diolah oleh program navigasi. Setelah itu, program akan menghasilkan arah gerak tujuan untuk helikopter. Arah pergerakan ini akan dibandingkan dengan pembacaan kompas digital. Setelah program dijalankan (hasil pada gambar 4.3), terlihat bahwa program dapat mengolah data yang dikirim dan menghasilkan kendali arah gerak helikopter sesuai dengan logika kondisi yang dibuat sesuai dengan skema deteksi objek yang diujikan. Kondisi skema yang diujikan adalah sesuai dengan tabel logika kondisi berikut ini :

Tabel 4.2. Kondisi untuk Kanan 15 dan Kiri 15 tidak ada rintangan

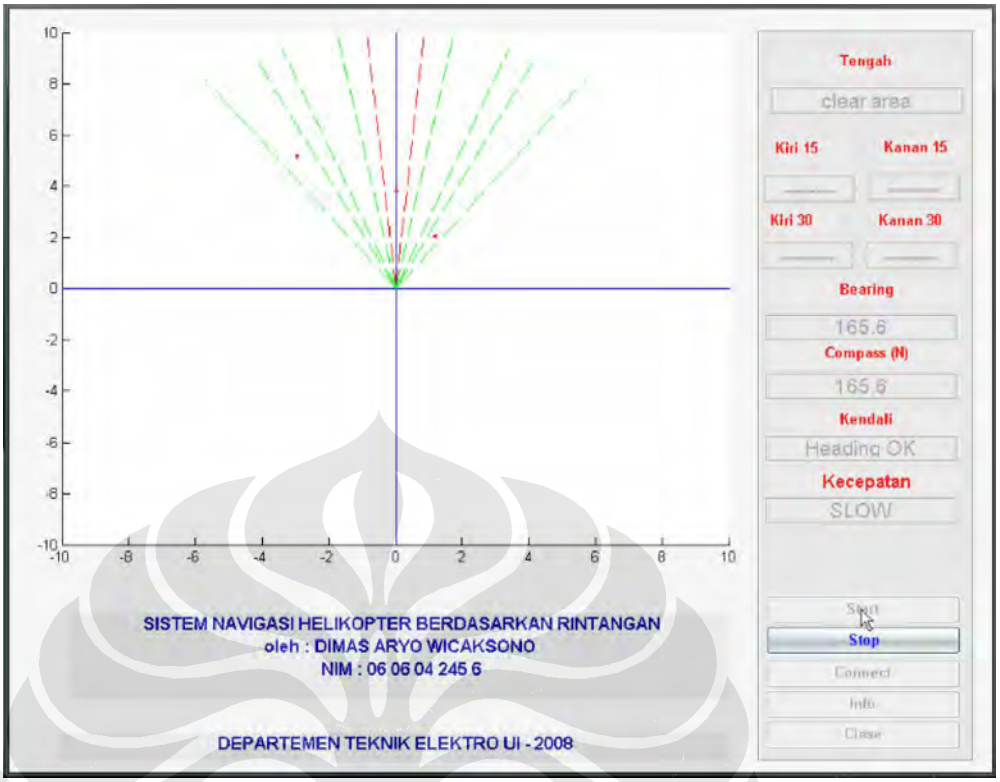
Input				Ouput	
Kiri 30	**Kiri 15**	**Kanan 15**	**Kanan 30**	**Control**	**Kecepatan**
nc	c	c	nc	???	---
dekat			dekat	Naik	Stop
sedang			dekat	Kiri +15	Slow
jauh			dekat	Kiri +15	Normal
dekat			sedang	Kanan +15	Slow
sedang			sedang	Kanan +15	Slow
jauh			sedang	Kiri +15	Normal
x			jauh	Kanan +15	Normal

Tabel 4.3. Kondisi rintangan

Kondisi		Input	Ouput
		Tengah	**Kecepatan**
C		$s \geq 10$ m	Fast
nc	Jauh	$7 \leq s < 10$	Normal
	Sedang	$3 < s < 7$	Slow
	Dekat	$s \leq 3$	Stop

Kondisi untuk skema pengujian 1 adalah terdapat objek pada bagian tengah, kanan 30^o dan kiri 30^o dan tidak terdapat objek pada bagian kanan 15^o dan kiri 15^o. Bila dilihat pada tabel 4.1, maka objek pada bagian kanan 15^o berada pada daerah dekat dan objek pada bagian kiri 15^o berada pada daerah sedang. Sehingga keluaran kondisi untuk skema ini adalah **Kiri +15** untuk kendalinya dan **Slow** untuk kecepatannya. Hal tersebut adalah sesuai yang ditampilkan oleh program navigasi.

Setelah kendali didapat, maka tujuan arah gerak helikopter untuk menghindari rintangan adalah sesuai dengan bearing yang ditampilkan pada program. Nilai bearing ini didapat dari perhitungan kompas digital dikurangi dengan besar derajat dari kendali yang diinginkan. Pada pengujian ini, arah kompas menunjukkan $180{,}4^0$ U. Berati nilai bearing yang dihasilkan adalah $165{,}6\ ^0$. Untuk itu, arah helikopter arah digerakkan ke arah bearing tersebut untuk menghindari rintangan yang ada didepannya. Hasil yang didapat ketika arah kompas sama dengan arah bearing adalah sebagai berikut :



Gambar 4. 16 Hasil pengujian program navigasi dengan skema 1

Setelah itu, program akan menampilkan keadaan di mana pada dearah tengah tidak terdapat rintangan seperti pada gambar 4.4 hasil pengujian.

Skema pengujian ke-2 yaitu terdapat rintangan pada daerah tengah dengan jarak 7,25 meter dan daerah kiri 15^0 dengan jarak 4,25 meter. Dari hasil pengujian didapat pembacaan jarak sensor sonar yaitu 6,984 meter untuk daerah tengah dan 4 meter untuk daerah kiri 15^0. Dari data ini terdapat terdapat kesalahan sebesar 3,75 %. Untuk kondisi skema yang diujikan adalah sesuai dengan tabel kondisi berikut ini :

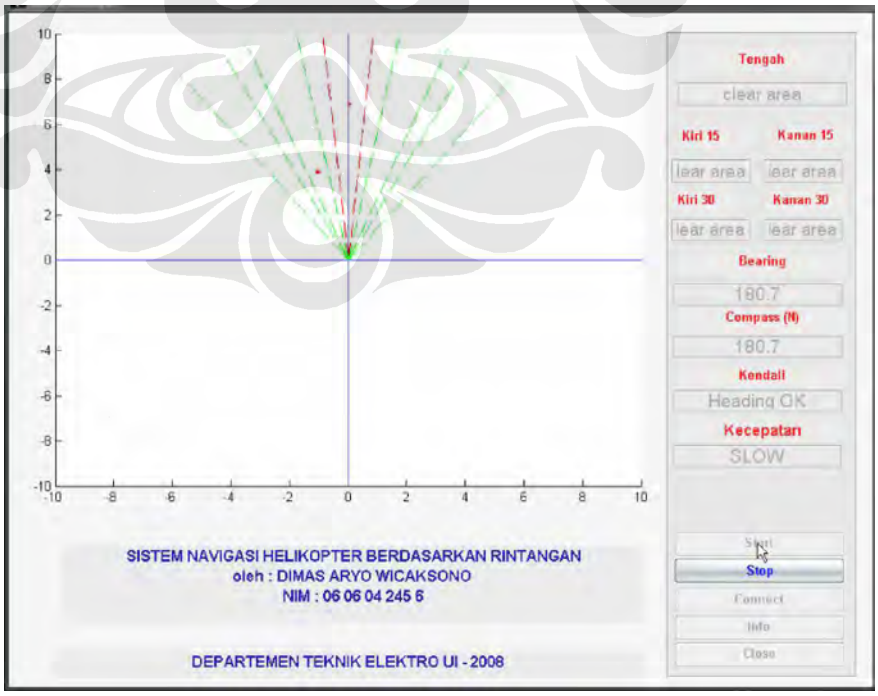
Tabel 4.4 Kondisi untuk daerah *scanning*

No	Input				Ouput	
	Kiri 30	Kiri 15	Kanan 15	Kanan 30	Control	Kecepatan
1	c	c	c	c	Kanan +15	Normal
2	nc	c	c	c	Kanan +15	Normal
3	c	nc	c	c	Kanan +15	Normal
4	nc	nc	c	c	Kanan +15	Normal
5	c	c	c	nc	Kiri +15	Normal
6	c	c	nc	c	Kiri +15	Normal
7	c	c	nc	nc	Kiri +15	Normal

Tabel 4.5. Kondisi rintangan

Kondisi		Input Tengah	Ouput Kecepatan
C		$s \geq 10$ m	Fast
nc	Jauh	$7 \leq s < 10$	Normal
	Sedang	$3 < s < 7$	Slow
	Dekat	$s \leq 3$	Stop

Pada skema 2 yang diujikan, tidak terdapat rintangan pada kedua daerah kanan. Sehingga didapat kendali **Kanan +15** dan kecepatan **Normal**. Dari hasil pengujian didapat kendali **Kanan +15** dan kecepatan **Slow**. Dari sini terlihat terdapat perbedaan kecepatan. Hal ini terjadi karena, logika kondisi membandingkan nilai kecepatan dengan jarak rintangan yang berada di tengah. Dari hasil pengujian di dapat jarak tengah sebesar 6,984 meter. Itu artinya, nilai kecepatan adalah **Slow**. Sehingga, kendali dan kecepatan yang didapat dari hasil pengujian tidak salah. Pada pengujian ini, arah kompas menunjukkan $165,7^0$ U. Berati nilai bearing yang dihasilkan adalah $180,7^0$. Untuk itu, arah helikopter arah digerakkan ke arah bearing tersebut untuk menghindari rintangan yang ada didepannya. Hasil yang didapat ketika arah kompas sama dengan arah bearing adalah sebagai berikut :



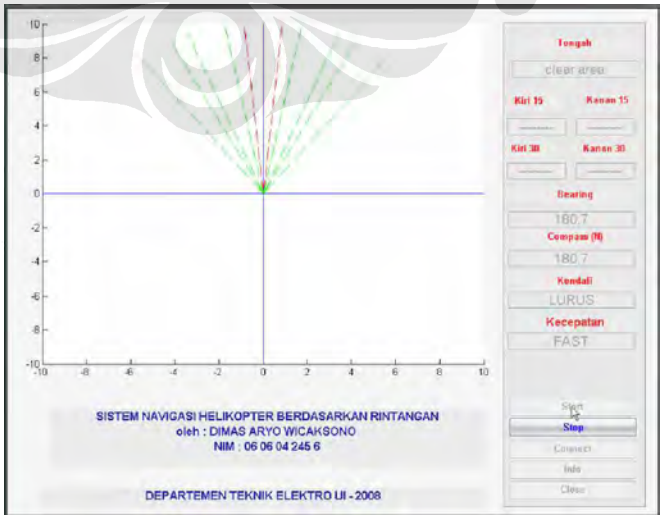
Gambar 4. 17 Hasil pengujian program navigasi dengan skema 2

Skema pengujian ke-3 yaitu terdapat rintangan pada daerah tengah dengan jarak 7,25 meter, daerah kanan 15^0 dengan jarak 2,75 meter, daerah kiri 15^0 dengan jarak 6,25 meter dan daerah kiri 30^0 dengan jarak 4 meter. Sedangkan daerah kanan 0^0, tidak terdapat rintangan. Dari hasil pengujian didapat pembacaan jarak sensor sonar yaitu 6,955 meter untuk daerah tengah, 2,625 meter untuk daerah kanan 15^0, 5,972 meter untuk daerah kiri 15^0 dan 3,709 meter untuk daerah kiri 30^0. Dari data ini terdapat tersebut kesalahan pembacaan jarak oleh sensor sonar sebesar 4 – 7 %. Untuk kondisi skema yang diujikan adalah sesuai dengan tabel logika kondisi berikut ini :

Tabel 4.6 Kondisi untuk Kanan 30 tidak ada rintangan

Input				**Ouput**	
Kiri 30	**Kiri 15**	**Kanan 15**	**Kanan 30**	**Control**	**Kecepatan**
nc	nc	nc	c	???	---
X	X	dekat		Naik	Stop
		sedang		Kanan +30	Slow
		jauh		Kanan +30	Normal

Dari tabel 4.5, terlihat bahwa kondisi skema yang diujikan untuk kanan 15 adalah berada pada range dekat. Dengan begitu kendali dan kecepatan yang dihasilkan **Naik** dan **Stop**. Hal ini sesuai dengan hasil yang ditampilkan pada program. Sedangkan untuk bearing untuk skema pengujian ini adalah sama dengan arah kompas. Sehingga cukup dengan menaikkan perangkat maka akan didapat kondisi berikut :



Gambar 4. 18 Hasil pengujian program navigasi dengan skema 3

Untuk skema pengujian terakhir, yaitu skema pengujian 4 dilakukan dengan cara sambil bergerak untuk menghindari beberapa rintangan. Hasil pengujian skema 4 dapat dilihat pada tabel berikut :

Tabel 4.7 Hasil pengujian skema 4

	Rintangan	Jarak	Hasil deteksi	% kesalahan
1	daerah tengah	9	8.516	5.37
	daerah kiri 15	6	5.686	5.23
2	daerah tengah	5	4.576	8.48
	daerah kiri 15	5	4.656	6.88
	daerah kiri 30	5.5	5.127	6.78
3	daerah tengah	4.5	4.352	3.28
	daerah kanan 15	3.5	3.398	2.91
	daerah kiri 15	7	6.699	4.3
	daerah kiri 30	7	6.638	5.17

Dari tabel diatas terlihat bahwa % kesalahan pembacaan sensor sonar yaitu berkisar 2 – 8,5 %.

Untuk kondisi skema yang diujikan ditunjukkan tabel logika kondisi berikut ini :

Tabel 4.8 Logika Kondisi untuk skema 4

	Input				**Ouput**	
	Kiri 30	**Kiri 15**	**Kanan 15**	**Kanan 30**	**Control**	**Kecepatan**
1 dan 2	c	c	c	c	Kanan +15	Normal
	nc	c	c	c	Kanan +15	Normal
	c	nc	c	c	Kanan +15	Normal
	nc	nc	c	c	Kanan +15	Normal
	c	c	c	nc	Kiri +15	Normal
	c	c	nc	c	Kiri +15	Normal
	c	c	nc	nc	Kiri +15	Normal
	Input				**Ouput**	
	Kiri 30	**Kiri 15**	**Kanan 15**	**Kanan 30**	**Control**	**Kecepatan**
3	nc	nc	nc	c	???	---
	X	X	dekat		Naik	Stop
			sedang		Kanan +30	Slow
			jauh		Kanan +30	Normal

Dari hasil pengujian yang didapat, kendali dan kecepatan yang dihasilkan sesuai dengan tabel logika kondisi yang direncanakan. Untuk keadaan 1, terdapat

rintangan pada daerah tengah dan kiri 15^0, sehingga kendali dan kecepatan yang dihasilkan adalah **Kanan 15** dan **Normal**. Untuk keadaan 2, terdapat rintangan pada daerah tengah, kiri 15^0 dan kiri 30^0. Karena jarak pada daerah tengah berada pada range 3 – 7 meter, sehingga kendali dan kecepatan yang dihasilkan adalah **Kanan 15** dan **Slow**. Dan untuk keadaan 3, terdapat rintangan pada daerah tengah, kanan 15^0, kiri 15^0 dan kiri 30^0, sehingga kendali dan kecepatan yang dihasilkan adalah **Kanan 30** dan **Normal**. Untuk bearing arah tujuan yang ditampilkan untuk menghindari rintangan pada skema pengujian ini sesuai dengan kendali yang diharapkan.

Dari setiap hasil pengujian yang dilakukan dari skema 1 sampai skema 4 terdapat kesalahan pembacaan sensor sonar yaitu ± 1 – 9 %. Hal ini disebabkan oleh bentuk objek yang dideteksi berbeda - beda serta posisi dari objek yang dideteksinya. Namun, persentase kesalahan ini tidak membuat kendali dan kecepatan untuk navigasi penghindar rintangan menjadi salah. Sehingga, untuk jarak rintangan sampai 10 meter, sensor sonar yang digunakan dalam skripsi ini masih bisa dipakai dalam implementasi sistem navigasi helikopter berdasarkan rintangan pada keadaan yang sebenarnya. Sedangkan untuk kompas digital, dari setiap hasil pengujian memberikan hasil pembacaan yang baik sehingga setiap kendali dan bearing arah tujuan yang diharapkan, kompas dapat memberi arah yang baik untuk menghindari rintangan yang diberikan.

BAB V

KESIMPULAN

Setelah sistem dianalisa dan diuji, maka dapat disimpulkan sebagai berikut:

1. Sensor sonar dan kompas digital dapat memberikan data pembacaan yang baik sehingga sistem navigasi berdasarkan rintangan dapat memberikan panduan yang tepat untuk menghindari rintangan.
2. Jarak minimum deteksi sensor sonar adalah 0,17 meter dengan waktu proses baca sebesar 18,442 ms dan jarak maksimum deteksi adalah 10,33 meter dengan waktu proses baca sebesar 79,7 ms.
3. Sensor sonar yang digunakan memiliki persentase kesalahan ± 1 – 9 %. Hal ini disebabkan oleh bentuk objek yang dideteksi berbeda - beda serta posisi dari objek yang dideteksinya. Namun, persentase kesalahan ini tidak membuat kendali dan kecepatan untuk navigasi penghindar rintangan menjadi salah. Sehingga, untuk jarak rintangan sampai 10 meter, sensor sonar yang digunakan dalam skripsi ini masih bisa dipakai dalam implementasi sistem navigasi helikopter berdasarkan rintangan pada keadaan yang sebenarnya.
4. Kompas digital yang digunakan memiliki ketelitian resolusi sampai $0{,}1^0$ pada setiap perubahannya. Kompas ini mengirimkan pulsa PWM secara kontinyu dengan pulsa 1 sebagai informasi arah sudut dan pulsa 0 diantara pulsa 1. Lebar pulsa 1 bervariasi antara 1 ms (0^0) sampai 36.99 ms (359.9^0). Sedangkan lebar pulsa 0 adalah 65 ms. Sehingga total periodenya adalah 65 ms + lebar pulsa 1 (antara 66 ms sampai 102 ms).
5. Logika kondisi yang dibuat dapat membaca data yang diperoleh dari pendeteksian sensor sonar sehingga hasil kendali dan kecepatan yang diharapkan dapat berguna untuk menghindari rintangan.

DAFTAR ACUAN

[1] Springer Handbook of Robotic, Bab 21 Sonar Sensing, hal 1. Diakses 18 Februari 2008 www.springer.com/cda/content/document/cda_downloaddocument/9783540239574-c1.pdf?SGWID=0-0-45-434998-p5208384

[2] Springer Handbook of Robotic, Bab 21 Sonar Sensing, hal 2. Diakses 18 Februari 2008 www.springer.com/cda/content/document/cda_downloaddocument/9783540239574-c1.pdf?SGWID=0-0-45-434998-p5208384

[3] Paulus Andi Nalwan, "Jenis – jenis motor", hal 3, Diakses tanggal 17 April 2008 dari Robotindonesia.com http://www.robotindonesia.com/article/an0012.pdf

DAFTAR PUSTAKA

SensComp 6500 Ranging Module, diakses 11 Februari 2008
http://www.acroname.com/robotics/parts/R11-6500.html

Atmel 8051Microcontrollers Manual. Dari data sheet ATMEL, diakses 3 Maret 2008
http://www.atmel.com/dyn/resources/prod_documents/DOC4316.PDF

Data sheet Atmel Mikrokontroler 89252 , diakses 3 Maret 2008
http://www.atmel.com/dyn/resources/prod_documents/doc1919.pdf

LISTING PROGRAM

```
'= = =Program ini digunakan untuk
mengontrol modul sensor sonar, kompas
digital dan
'= = = motor servo.


$large
$regfile = "8052.dat"
$romstart = &H0000  'Alamat awal
dari ROM
$crystal = 11059200
$baud = 4800
Dim Pc As Byte , Pa As Byte
Dim Waktu As Word
Dim Jarak As Single , R As Byte ,
Jarak2 As Single
Dim Halangan(5) As String * 6
Dim A As Word
Dim S As Single
Dim Y As String * 8
Echo Alias P3.2
'Sonar
Cacah Alias P3.4
'Compas
Config Timer0 = Timer , Mode = 1 ,
Gate = Internal
Enable Interrupts
Enable Timer0
On Timer0 Overflow
Gosub Inisialisasi_ppi
R = 1
Halangan(1) = ""
Halangan(2) = ""
Halangan(3) = ""
Halangan(4) = ""
Halangan(5) = ""
Gosub Putar_tengah
Gosub Putar_tengah
Gosub Putar_tengah
Gosub Putar_tengah
Do
Jarak = 0
Counter0 = 0
Waktu = 0
Pc = &B00000000                'Reset
Init
 Out &H2002 , Pc
Pc = &B00000001                'Set Init
 Out &H2002 , Pc
 Start Timer0
Waitms 1
Bitwait Echo , Set
 Stop Timer0
Pc = &B00000000        'Reset Init
 Out &H2002 , Pc
Waktu = Counter0
Jarak = Waktu / 1000
Jarak= Jarak* 0.344
Jarak = Jarak / 2
Halangan(r) = Fusing(jarak , ##.###)
'C
Incr R
If R = 2 Then
Gosub Putar_kanan
Gosub Putar_kanan
Gosub Putar_kanan
Gosub Putar_kanan
Elseif R = 3 Then
Gosub Putar_kanan
Gosub Putar_kanan
Gosub Putar_kanan
Gosub Putar_kanan
Gosub Putar_kanan
Elseif R = 4 Then
Gosub Putar_tengah
Gosub Putar_tengah
Gosub Putar_tengah
Gosub Putar_tengah
Waitms 20
Gosub Putar_tengah
Gosub Putar_tengah
Gosub Putar_tengah
Gosub Putar_tengah
Gosub Putar_kiri
Gosub Putar_kiri
Elseif R = 5 Then
Gosub Putar_kiri
Elseif R = 6 Then
Gosub Putar_tengah
Gosub Putar_tengah
Gosub Putar_tengah
Gosub Putar_tengah
Gosub Putar_tengah
Waitms 20
Gosub Putar_tengah
Gosub Putar_tengah
Gosub Putar_tengah
Gosub Putar_tengah
Gosub Compass
Print "C" ; "," ; Halangan(1) ; "," ;
"R10" ; "," ; Halangan(2) ; "," ; "R20" ; "," ;
Halangan(3) ; "," ; "L10" ; "," ;
Halangan(4) ; "," ; "L20" ; "," ;
```

```
Halangan(5) ; "," ; "Compass" ; "," ; Y ; "," ;
"U"
86.  R = 1
87.  Halangan(1) = ""
88.  Halangan(2) = ""
89.  Halangan(3) = ""
90.  Halangan(4) = ""
91.  Halangan(5) = ""
92.  End If
93.  Loop
94.  End
95.  Inisialisasi_ppi:
96.   Out &H2003 , &B10000000
'all Port = Output
97.  Return
98.  Overflow:
99.  Bitwait Echo , Reset
100. Stop Timer0
101. If R = 1 Then
102. Gosub Compass
103. Print "C" ; "," ; "12.000" ; "," ; "R10" ;
"," ; "12.000" ; "," ; "R20" ; "," ; "12.000" ;
"," ; "L10" ; "," ; "12.000" ; "," ; "L20" ; "," ;
"12.000" ; "," ; "Compass" ; "," ; Y ; "," ;
"U"
104. R = 1
105. Elseif R = 2 Then          'kanan ke 1st
106. Incr R                     'R =2
107. Gosub Putar_kanan
108. Gosub Putar_kanan
109. Gosub Putar_kanan
110. Gosub Putar_kanan
111. Halangan(2) = "12.000"
112. Elseif R = 3 Then          'kanan 2nd
113. Incr R
'R=3
114. Gosub Putar_tengah
115. Gosub Putar_tengah
116. Gosub Putar_tengah
117. Gosub Putar_tengah
118. Waitms 20
119. Gosub Putar_tengah
120. Gosub Putar_tengah
121. Gosub Putar_tengah
122. Gosub Putar_tengah
123. Gosub Putar_kiri
124. Gosub Putar_kiri
125. Halangan(3) = "12.000"
126. Elseif R = 4 Then          'kiri 1st
127. Incr R
'R=4
128. Gosub Putar_kiri
129. Halangan(4) = "12.000"
130. Elseif R = 5 Then
'R = 5
131. Halangan(5) = "12.000"
132. Gosub Putar_tengah
133. Gosub Putar_tengah
134. Gosub Putar_tengah
135. Gosub Putar_tengah
136. Waitms 20
137. Gosub Putar_tengah
138. Gosub Putar_tengah
139. Gosub Putar_tengah
140. Gosub Putar_tengah
141.  Gosub Compass
142. Print "C" ; "," ; Halangan(1) ; "," ;
"R10" ; "," ; Halangan(2) ; "," ; "R20" ; "," ;
Halangan(3) ; "," ; "L10" ; "," ;
Halangan(4) ; "," ; "L20" ; "," ;
Halangan(5) ; "," ; "Compass" ; "," ; Y ; "," ;
"U"
143. R = 1
144. Halangan(1) = ""
145. Halangan(2) = ""
146. Halangan(3) = ""
147. Halangan(4) = ""
148. Halangan(5) = ""
149. End If
150.  Reset Tcon.5
151. Disable Interrupts
152. Enable Interrupts
153. Pc = &B00000000
154. Out &H2002 , Pc
155. Pc = &B00000001      'Set Init
156. Out &H2002 , Pc
157. Start Timer0
158. Waitms 1
159. Return
160. Putar_kiri:
161. Pa = &B00000000
162. Out &H2000 , Pa
163. Delay
164. Delay
165. Delay
166. Pa = &B00000001
167. Out &H2000 , Pa
168. Delay
169. Delay
170. Delay
171. Delay
172. Delay
173. Delay
174. Delay
175. Delay
176. Pa = &B00000000
177. Out &H2000 , Pa
178. Waitms 2
179. Return
180. Putar_kanan:
181. Pa = &B00000000
182. Out &H2000 , Pa
183. Delay
184. Delay
```

```
Delay
Pa = &B00000001
Out &H2000 , Pa
Waitms 2
Delay
Delay
Pa = &B00000000
Out &H2000 , Pa
Waitms 2
'2ms
Return
Putar_tengah:
Pa = &B00000000
Out &H2000 , Pa
Delay
Delay
Delay
Pa = &B00000001
Out &H2000 , Pa
Waitms 1
Delay
Delay
Delay
Delay
Delay
Delay
Delay
Delay
Pa = &B00000000
Out &H2000 , Pa
Waitms 20
'2ms
Return
Compass:
Reset Tcon.4
Reset Tcon.5
Set Tcon.4
Counter0 = 0
Bitwait Cacah , Reset
Bitwait Cacah , Set
Start Timer0
Bitwait Cacah , Reset
Waitms 2
Stop Timer0
A = Counter0
S = 12 / 11.0592
S = A * S
S = S – 1000
S = S / 100
Y = Fusing(s , ###.#)
return
```