DT-AVR

AN205 – *Line Follower Robot* berbasis kamera HaViMo 2.0 dengan sistem kendali PD

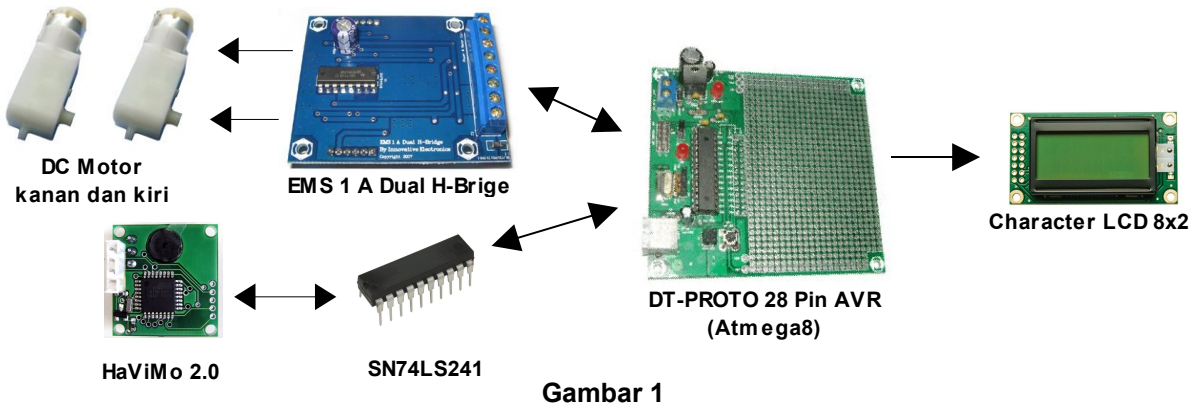
Oleh: Tim IE

Modul kamera pengolah citra belakangan ini semakin populer digunakan pada aplikasi robotika. Modul kamera tersebut biasa digunakan pada robot sepakbola *humanoid,* robot penjejak warna, kontes robot cerdas, dan aplikasi lain yang memerlukan pengolahan citra warna. Salah satu modul kamera yang cukup terkenal yaitu **HaViMo 2.0** (Hamid *Vision Module* versi 2.0) dari HaViSys. HaViMo 2.0 merupakan modul kamera *CMOS* yang terintegrasi dengan *chip processor* untuk melakukan *image processing* berbasis warna seperti *Region Growing* dan *Griding* secara langsung sehingga pengguna bisa mendapatkan hasil pengolahan citra yang siap digunakan. HaViMo 2.0 memiliki format protokol yang sama dengan protokol dynamixel.
Pada aplikasi ini telah dibuat suatu robot pengikut garis (gelap diatas terang) menggunakan **HaViMo 2.0** dengan sistem kendali PD (Proposional Derivatif). Proses kalibrasi dan pemilihan warna yang akan dideteksi dilakukan terlebih dahulu melalui komputer dengan software HaViMoGUI. Sebagai kontroler utama digunakan **DT-PROTO 28 Pin AVR** yang berbasis Atmega8. Sebagai aktuator robot digunakan satu buah modul **EMS 1 A Dual H-Bridge** untuk men-*drive* dua buah **GMX022 DC Gear Motor**. Sama dengan AN203, pada aplikasi ini digunakan IC **74LS241** sebagai jembatan antarmuka *UART half duplex TTL* dengan *UART TTL* mikrokontroler.

Berikut adalah perlengkapan yang diperlukan dalam aplikasi ini :

- 1x DT-PROTO 28 Pin AVR (dengan modifikasi Crystal 8MHz)
- 1x HaViMo 2.0
- 1x Character LCD 8x2
- 1x IC SN74LS241 dan 1x Resistor 10K Ohm
- 1x EMS 1 A Dual H-Bridge
- 2x GMX022 DC Gear Motor 3-12V; 1:120; 100rpm; 90 Degree shaft
- 2x RodaPenggerak Diameter 6,7 cm
- 1x Ball Caster
- 1x Baterai LiPo 7,4V; 800mAH
- USB2Dynamixel (untuk kalibrasi HaViMo)
- Arena Line Follower dengan garis hitam dan latar belakang putih
- 1x kabel 3 pin Dynamixel dan Beberapa kabel jumper
- Acrylic untuk bodi robot, lem, mur dan baut secukupnya.

Adapun blok diagram dari AN205 adalah sebagai berikut.



Gambar 1
Blok Diagram AN205

Hubungan antar modul adalah sebagai berikut :

DT-PROTO 28 Pin AVR	SN74LS241
GND (+5V)	GND (Pin 10)
VCC (Gnd)	VCC (Pin 20)
PORTD.0/RXD (PD0) **	1Y4 (Pin 12)***
PORTD.1/TXD (PD1) **	2A1 (Pin 11)***
PORTD.7/Direction Control (PD7)*	1G (Pin 1)**
	2G (Pin 19)**

*) Pin tidak mutlak, dapat diganti dengan pin lain dengan penyesuaian program
**) Pin ini mutlak, tidak dapat diganti dengan pin yang lain.
***) Pin tidak mutlak, dapat diganti dengan pin lain yang sesuai fungsinya.

Tabel 1
Hubungan DT-AVR LCMS dengan SN74LS241

SN74LS241	HaViMo 2.0
GND (Gnd)	GND
1A4 (Pin 8)*	DATA
2Y1 (Pin 9)*	
V Baterai	VDD

*) Pin tidak mutlak, dapat diganti dengan pin lain yang sesuai fungsinya.

Tabel 2
Hubungan SN74LS241 dengan HaViMo 2.0

DT-PROTO 28 Pin AVR	EMS 1 A Dual H-Bridge
GND (Gnd)	GND (J3 pin 1)
VCC (+5V)	+5V (J3 pin 2)
PORTB.1/OC1A (PB1)**	M1EN
PORTB.2/OC1B (PB2)**	M2EN
PORTB.3 (PB3)*	M2IN1
PORTB.0 (PB0)*	M2IN2
PORTB.5 (PB5)*	M1IN1
PORTB.4 (PB4)*	M1IN2

*) Pin tidak mutlak, dapat diganti dengan pin lain dengan penyesuaian program
**) Pin ini mutlak, tidak dapat diganti dengan pin yang lain

Tabel 3
Hubungan DT-PROTO 28 Pin AVR dengan EMS 1 A Dual H-Bridge

DT-PROTO 28 Pin AVR	Character LCD 8x2
GND (Gnd)	VSS (pin 1)
VCC (+5V)	VDD (pin 2)
V kontras (Variabel Resistor)	VO (pin 3)
PORTC.0 (PC0)*	RS (pin 4)
Gnd (Low)	R/W (pin 5)
PORTC.1 (PC1)*	E (pin 6)
Tidak Terhubung	DB0 (pin 7)
Tidak Terhubung	DB1 (pin 8)
Tidak Terhubung	DB2 (pin 9)
Tidak Terhubung	DB3 (pin 10)
PORTC.5 (PC5)*	DB4 (pin 11)
PORTC.2 (PC2)*	DB5 (pin 12)
PORTC.4 (PC4)*	DB6 (pin 13)
PORTC.3 (PC3)*	DB7 (pin 14)

*) Pin tidak mutlak, dapat diganti dengan pin lain dengan penyesuaian program

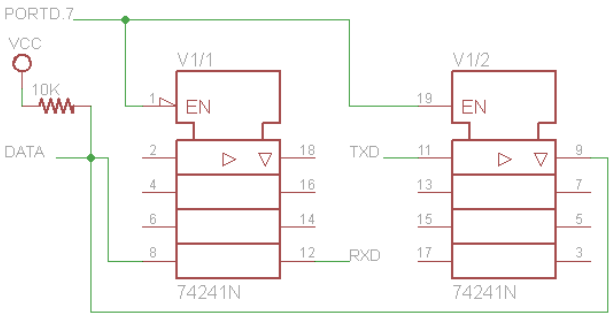
Tabel 4
Hubungan DT-PROTO 28 Pin AVR dengan Character LCD 8x2

EMS 1 A Dual H-Bridge (J3)	Motor dan Baterai
PGND	GND (DT-PROTO 28 Pin AVR)
VCC	+5V (DT-PROTO 28 Pin AVR)
MGND	Gnd Baterai
V Mot	V Baterai
M2Out1	Pin1 Motor kiri
M2Out2	Pin2 Motor kiri
M1Out1	Pin1 Motor kanan
M1Out2	Pin2 Motor kanan

*) Pin tidak mutlak, dapat diganti dengan pin lain dengan penyesuaian program
**) Pin ini mutlak, tidak dapat diganti dengan pin yang lain

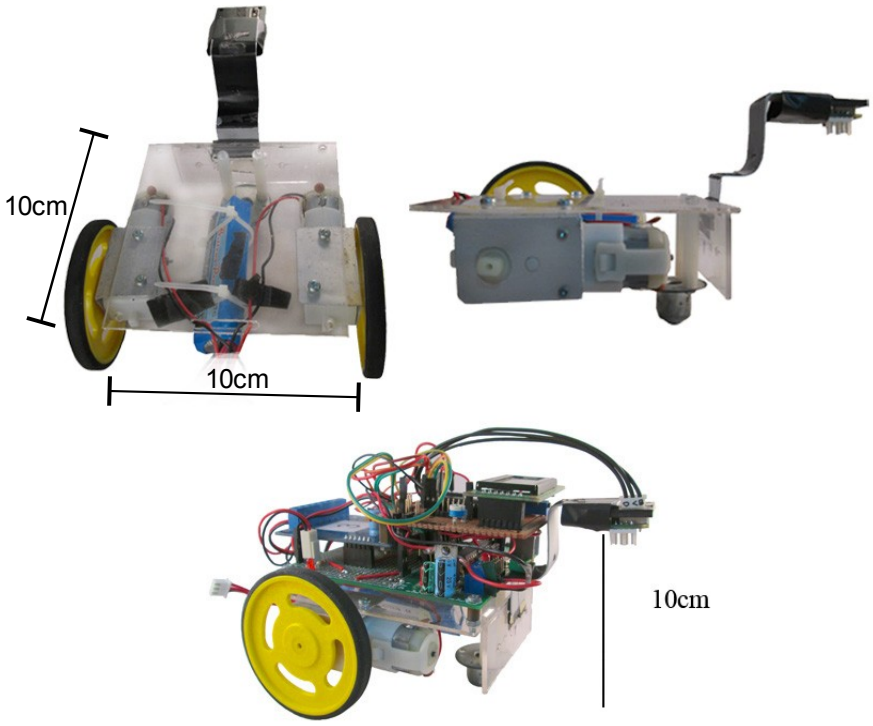
Tabel 5
Hubungan EMS 1 A Dual H-Bridge (J3) dengan Motor dan Baterai

Skema koneksi SN74LS241 dengan modul-modul yang lain adalah sebagai berikut :



Gambar 2
Skema koneksi SN74LS241 dengan HaViMo 2.0 dan DT-PROTO 28 Pin AVR

Bentuk konfigurasi bodi robot *Line Follower* menggunakan bahan akrilik dan aluminium.



Gambar 3
Konfigurasi mekanik robot

Modul – modul diatas perlu dikonfigurasi terlebih dahulu agar dapat bekerja sesuai dengan yang diharapkan. Berikut ini adalah langkah – langkah konfigurasi yang perlu dilakukan :

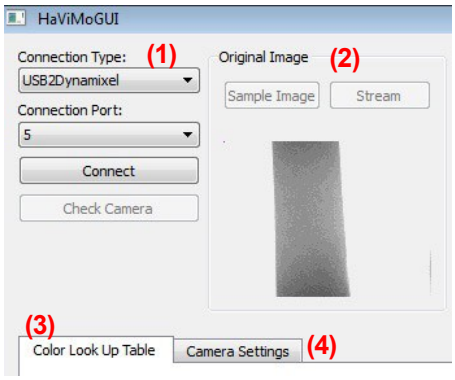
- DT-AVR PROTO 28 Pin AVR
  - Mikrokontroler Atmega8 pada DT-PROTO 28 Pin AVR menggunakan osilator eksternal berupa crystal dengan nilai frekuensi 4 Mhz. Untuk mendukung baudrate komunikasi 1Mbps yang dibutuhkan untuk berkomunikasi dengan HaViMo 2.0, maka diperlukan penggantian *Crystal Oscillator* dengan frekuensi 8MHz atau 16MHz, pada AN ini digunakan XTAL 8MHz.

- Apabila Atmega8 yang digunakan masih menggunakan sumber clock internal atau tidak sesuai dengan kebutuhan kristal 8MHz, silakan melakukan pengaturan *fusebit* pada ATmega8 dahulu agar dapat bekerja dengan kristal 8MHz. Informasi lebih detil mengenai pengaturan fusebit terdapat pada AN177.
- Komunikasi antara Atmega8 dengan HaViMo 2.0 dilakukan menggunakan komunikasi serial UART TTL. Maka dari itu perlu dilakukan pengaturan jumper J5 dan J6 pada posisi tidak terhubung. Untuk menjembatani antara komunikasi UART TTL dengan UART Half Duplex TTL, digunakan IC SN74LS421.

J5	J6
1	1
2	2

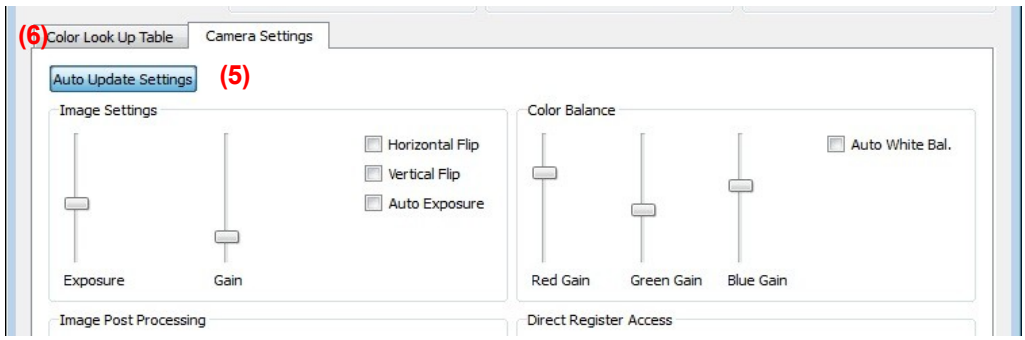
Gambar 4
Pengaturan Jumper J5 dan J6 pada DT-PROTO 28 Pin AVR

- HaViMo 2.0
  - Untuk bisa mengenali warna-warna tertentu, maka modul HaViMo 2.0 perlu terlebih dahulu dikalibrasi dan diprogram melalui software komputer HaViMoGUI.
  - Langkah-langkah pemilihan warna yang akan dideteksi adalah sebagai berikut :
    1. Hubungkan modul HaViMo dengan USB2Dynamixel, pastikan *switch* selektor pada USB2Dynamixel telah berada pada posisi *TTL*.
    2. Berikan catu daya pada modul HaViMo 2.0.
    3. Tancapkan USB2Dynamixel pada port USB komputer (diasumsikan driver telah terinstal).
    4. Buka program HaViMoGUI.
    5. Pada *Connection Type*, pilih USB2Dynamixel (1).
    6. Pada *Connection Port*, pilih port COM serial dari USB2Dynamixel.
    7. Klik *Connect*, apabila berhasil maka pada sudut kiri bawah dari HaViMoGUI akan ada keterangan status “Port Open”
    8. Kemudian klik *Check Camera*, apabila berhasil maka akan ada keterangan status “HaViMo Found”



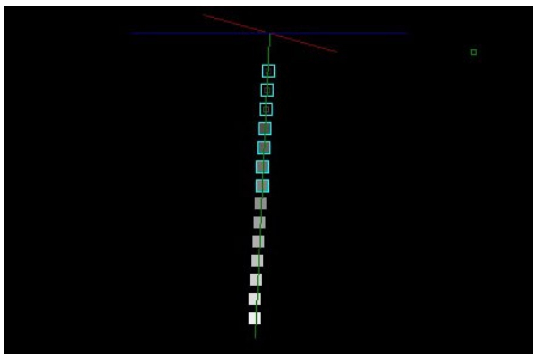
Gambar 5
Tampilan HaViMoGUI

9. Persiapkan kondisi target/garis warna hitam, kondisi pencahayaan lingkungan, serta latar belakang target, hal ini penting karena proses kalibrasi warna untuk *tracking* akan dilakukan.
10. Kik *Sample Image,* maka akan didapatkan gambar *Original Image (2)* serta Grafik Warna 3D pada halaman *Color Look Up Table (3)*.
11. Langkah selanjutnya yang perlu dilakukan adalah menyesuaikan parameter pada tab *Camera Settings* (4). Pada Tab *Camera Settings* dapat dilakukan pengaturan standard hasil capture dari HaViMo 2.0 sebelum dilakukan *Image Processing.* Hal-hal yang perlu diperhatikan pada tahap ini antara lain, kerataan pencahayaan, keseimbangan warna, dan tingkat kontras. Tahapan ini termasuk bagian sangat penting, karena dengan kualitas gambar yang baik dan merata akan mempermudah proses selanjutnya yaitu pemilihan warna yang akan dideteksi.
12. Pada *box Original Image* (2) Klik *Stream* agar gambar ditampilkan secara update terus menerus.
13. Klik *Auto Update Settings* untuk mulai mengatur berbagai macam *setting*-nya (5). Setelah selesai Klik kembali *Auto Update Settings* untuk menghentikan fungsi pengaturan secara langsung tersebut dan Klik *Stream* (2) sekali lagi untuk menghentikan proses *streaming image*.



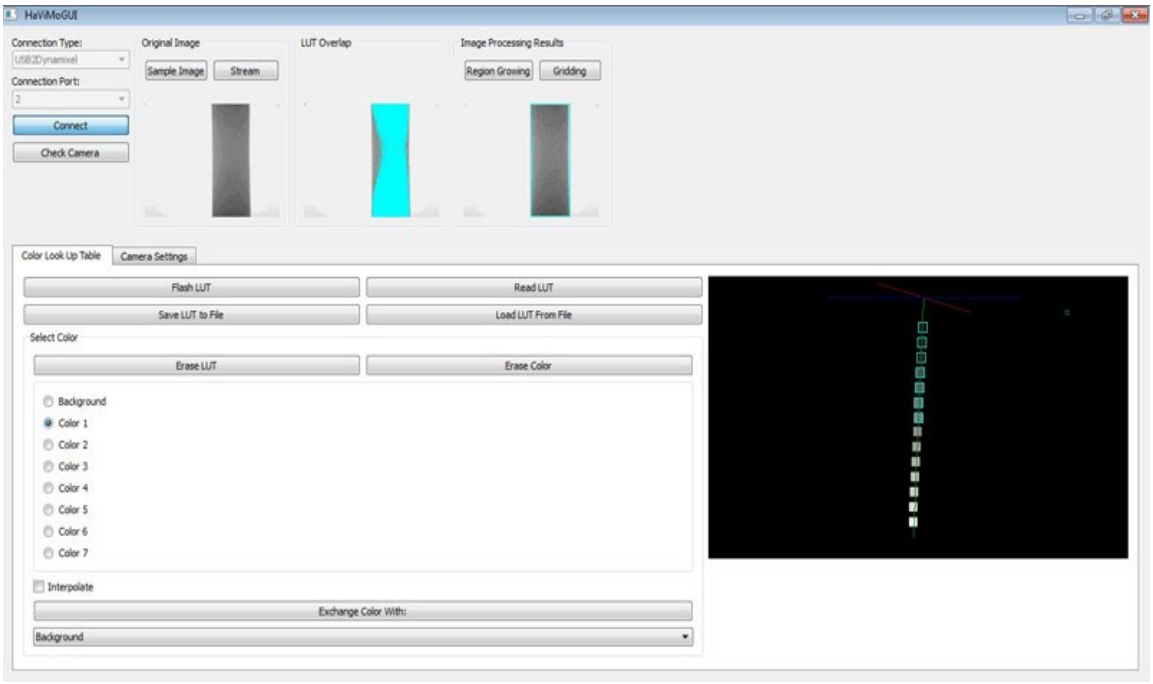
Gambar 6
Tampilan *Camera Settings* HaViMoGUI

14. Langkah selanjutnya yaitu pemilihan warna yang akan dideteksi. Kembalikan tampilan pada Tab Color Look Up Table (6). Klik *Sample Image* (2) untuk mendapatkan hasil gambar terbaru.
15. Klik *Erase LUT* dan *Erase Color* untuk menghapus setting sebelumnya (jika ada) pada HaViMo 2.0.
16. Klik pada *radio button Color 1* (7), maka saat ini kita telah siap memilih warna mana yang akan dideteksi oleh HaViMo.



Gambar 7
Tampilan Grafik Warna 3D HaViMoGUI

17. Klik pada warna-warna pada Grafik Warna 3D yang mirip dengan warna garis pada tampilan *Original Image*. Secara otomatis tampilan citra pada *LUT Overlap* (8) akan menunjukkan warna mana yang kita pilih, dengan tanda warna hijau cerah. Semakin banyak warna hijau cerah yang terlihat pada badan garis, maka itu berarti semakin banyak warna garis telah masuk dalam *Look Up Table*.
18. Grafik Warna 3D (Gambar 8) dapat diputar-putar untuk melihat dan memilih warna-warna yang sesuai agar nilai *LUT* semakin baik.
19. Pada proses seleksi warna ini akan lebih baik jika dilakukan beberapa kali dengan mensimulasikan kondisi pencahayaan yang mungkin terjadi. Misalkan pada *Sample Image* dan pemilihan warna pertama, kondisi dikondisikan ideal dimana sumber cahaya seperti lampu ruangan dapat langsung menerangi lintasan. Kemudian dilakukan *Sample Image* dan pemilihan warna kedua dengan kondisi lain yang mungkin terjadi seperti ketika lintasan terkena bayangan robot itu sendiri dan lain sebagainya.
20. Jika dirasa sudah cukup baik, maka LUT yang sudah tersusun dapat disimpan dengan cara Klik *Save LUT to File* (10).
21. Langkah selanjutnya adalah menuliskan *LUT* yang telah disusun ke HaViMo 2.0, hal ini dapat dilakukan dengan cara langsung dengan Klik *Flash LUT* (10) apabila *LUT* yang dimaksud telah/sedang dibuka. Apabila *LUT* yang diinginkan masih tersimpan, maka dapat dibuka terlebih dahulu dengan *Load LUT From File* (10).
22. Setelah proses *Flash LUT* selesai, klik *Sample Image* sekali lagi, Klik *Read LUT*, kemudian klik *Region Growing*. Apabila nilai *LUT* untuk deteksi garis sudah baik, maka posisi garis akan bisa dideteksi dengan tanda bingkai persegi empat berwarna hijau muda (9).



Gambar 8
Tampilan HaViMoGUI sukses mendeteksi *object*.

Setelah menghubungkan modul-modul tersebut menggunakan kabel jumper, lakukan pengecekan kembali menggunakan *multimeter* untuk memastikan koneksi antar modul telah terhubung dengan baik. Pastikan juga bahwa tidak terjadi hubungan singkat antara VCC dan GND sebelum memberikan catu daya.
Apabila konfigurasi di atas telah selesai, lakukan langkah – langkah berikut ini :

1. Berikan catu daya baterai LiPo +7,4V pada ..... DT-PROTO 28 Pin AVR (**perhatikan polaritas catu daya, Jika polaritas terbalik dapat menyebabkan kerusakan modul**).
2. Hubungkan *programmer* mikrokontroler AVR yang mendukung fitur ISP dengan DT-PROTO 28 Pin AVR, seperti DT-HiQ AVR In System Programmer, DT-HiQ AVR USB ISP, atau *programmer* lainnya.
3. *Download* file dengan ekstensi .hex (Havimo_Line_Follower_pwm.hex) yang berada di dalam folder (AN205\Havimo_Line_Follower_pwm\Release) ke DT-PROTO 28 Pin AVR.

Protokol komunikasi Dynamixel

Protokol komunikasi pada HaViMo 2.0 serupa dengan protokol komunikasi Dynamixel. Protokol ini terbagi menjadi dua jenis, yaitu data yang dikirimkan dari kontroler ke servo *(Instruction Packet)* dan data respon yang dikirim balik oleh servo kepada kontroler *(Status packet)*. Keterangan lengkap mengenai paket *Instruction*, *Status*, dan *Parameter* dapat dilihat pada manual dari produk servo Dynamixel seperti AX-12+.
Kedua paket data tersebut memiliki format sebagai berikut.

- **Instruction Packet**

Merupakan paket yang dikirimkan oleh kontroler utama kepada servo untuk mengirimkan perintah-perintah. Struktur dari *Instruction packet* adalah sebagai berikut:

0XFF	0XFF	ID	LENGTH	INSTRUCTION	PARAMETER1	...	PARAMETER N	CHECK SUM

Keterangan :

0xFF 0xFF : Merupakan indikator awal dari sebuah paket.

ID : Merupakan nomor ID/Identitas unique dari servo dynamixel.
Terdapat 254 nilai ID yang tersedia mulai dari 0x00 sampai dengan 0xFD. Ada nomor ID khusus yang disebut broadcasting ID yaitu nomor 0xFE. Paket yang dikirmkan dengan ID Broadcasting akan diterima dan dilaksanakan oleh semua servo yang terhubung. Paket yang terkirim dengan Broadcasting ID tidak akan menghasilkan Respon Status Packet.

LENGTH : Merupakan panjang dari paket dimana nilainya adalah “Jumlah parameter(N) + 2”.
INSTRUCTION : Instruksi untuk aktuator Dynamixel yang akan dilakukan.
PARAMETER0...N : Digunakan jika ada informasi tambahan yang butuh untuk dikirmkan selain dari Instruksi itu sendiri.
CHECK SUM : Merupakan data yang dihitung sebagai error checking. Metode penghitungan check sum adalah sebagai berikut :

Check Sum = ~(ID +Length +Instruction + Parameter1+... Parameter N)

Jika nilai yang terhitung lebih besar dari 255, maka byte terbawahlah yang ditentukan sebagai nilai check sum.
~ mewakili operasi logic NOT.

- **Status Packet/Return Packet**

Merupakan paket respon dari servo kepada kontroler utama setelah menerima sebuah *Instruction Packet*. Struktur dari *Status Packet* adalah sebagai berikut.

Keterangan :
0xFF 0xFF : Merupakan indikator awal dari sebuah paket.
ID : Merupakan nomor ID/Identitas unique dari servo dynamixel.
Terdapat 254 nilai ID yang tersedia mulai dari 0x00 sampai dengan 0xFD. Ada nomor ID khusus yang disebut broadcasting ID yaitu nomor 0xFE. Paket yang dikirmkan dengan ID Broadcasting akan diterima dan dilaksanakan oleh semua servo yang terhubung. Paket yang terkirim dengan Broadcasting ID tidak akan menghasilkan Respon Status Packet.
LENGTH : Merupakan panjang dari paket dimana nilainya adalah “Jumlah parameter(N) + 2”.
ERROR : Merupakan *Byte* yang yang merepresentasikan error yang dikirimkan oleh Servo.
INSTRUCTION : Instruksi untuk aktuator Dynamixel yang akan dilakukan.
PARAMETER0...N : Digunakan jika ada informasi tambahan yang butuh untuk dikirmkan selain dari Instruksi itu sendiri.
CHECK SUM : Merupakan yang dihitung sebagai error checking. Metode penghitungan 'Check Sum' adalah sebagai berikut :

Check Sum = ~(ID +Length +Instruction + Parameter1+... Parameter N)

Jika nilai yang terhitung lebih besar dari 255, maka byte terbawahlah yang ditentukan sebagai nilai checksum.
~ mewakili operasi logic NOT.

Instruksi HaViMo 2.0

Struktur protokol dan hardware komunikasi HaViMo 2.0 dengan protokol Dynamixel adalah sama, namun ada beberapa hal yang berbeda pada alamat ID dan alamat r*egister*. Alamat ID HaViMo 2.0 telah diatur secara permanen pada ID 100 (desimal) atau 0x64 (heksadesimal). HaViMo 2.0 juga memiliki daftar set instruksi-instruksi khusus berkaitan dengan pengaturan pengolahan citra yang hanya dapat digunakan dengan HaViMo 2.0. Instruksi-instruksi tersebut antara lain :

Instruction	Value	No. of Parameter	Function
PING	0x01	0	No action. Used for obtaining a Status Packet
READ_REGION	0x02	2	Read Results of Region Detection
WRITE	0x03	2	Equivalent to CAP_REGION for Compatibility
READ_REG	0x0C	2	Read Camera Chip Registers
WRITE_REG	0x0D	2	Write Camera Chip Registers (1)
CAP_REGION	0x0E	0	Capture and Find Color Regions (1)
RAW_SAMPLE	0x0F	0	Sample the Raw Image (used by GUI) (2)
LUT_MANAGE	0x10	0	Enter LUT Manage Mode (used by GUI) (2)

RD_FILTHR	0x11	2	Read Noise Filter Thresholds
WR_FILTHR	0x12	2	Write Noise Filter Thresholds (1)
RD_REGTHR	0x13	2	Read Region Filter Thresholds
WR_REGTHR	0x14	2	Write Region Filter Thresholds (1)
CAP_GRID	0x15	0	Capture and Compress using Griding algorithm (1)
READ_GRID	0x16	2	Read Results of the Griding Algorithm (3)
SAMPLE_FAST	0x17	0	Fast Sample (used by GUI) (2) (4)

1) Tidak ada return paket yang dihasilkan dari Instruksi ini.
2) Respon berbeda dengan standar paket *ROBOTIS* maupun *RoboBuilder.*
3) Address yang diberikan dikali dengan 16 secara internal.
4) Tidak didukung pada mode *RoboBuilder.*

Tabel 6
Daftar Instruksi HaViMo 2.0

Pengolahan citra dengan *Region Growing*

Instruksi yang digunakan untuk melacak posisi warna pada Aplikasi ini adalah CAP_REGION dan READ_REGION. CAP_REGION berfungsi untuk menangkap suatu citra untuk diproses menggunakan teknik *Region Growing*. Hasil dari prioses *Region Growing* ini dibaca dengan menggunakan instruksi READ _REGION. Pada AN205 ini proses pembacaan register ini telah dimasukkan dalam *library* Dynamixel_Havimo.

Hasil pengolahan Region Growing memiliki format sebagai berikut :

Index	Color	Pixels		SumX				SumY				MaxX	MinX	MaxY	MinY
XX	XX	XX	XX	XX	XX	XX	XX	XX	XX	XX	XX	XX	XX	XX	XX

*) seluruh tipe data dalam heksadesilmal

Tabel 5
Format Hasil Region Growing

Hasil pembagian antara 'jumlah piksel-piksel yang terdeteksi dalam *region'* (**Pixels**) dengan 'jumlah dari koordinat X pada piksel-piksel yang terdeteksi' (**SumX**) akan menghasilkan nilai 'titik tengah' (*center point)* dari koordinat sumbu X terhadap kotak *region*, begitu pula untuk pembagian **SumY** dengan **Pixels** akan menghasilkan kordinat titik tengah sumbu Y terhadap kotak *region*.

Posisi obyek (x,y) terhadap kamera HaViMo 2.0, diperoleh melalui perhitungan sebagai berikut :

Kordinat obyek pada sumbu X = **MinX* + (SumX / Pixels)**
Kordinat obyek pada sumbu Y = **MinY* + (SumY / Pixels)**

*) Nilai batas sisi kiri kotak *region* (**MinX**) dan nilai batas sisi atas kotak *region* (**MinY**) perlu ditambahkan untuk memperoleh posisi sebenarnya dari obyek terhadap keseluruhan citra.

Kendali Proporsional Derivatif

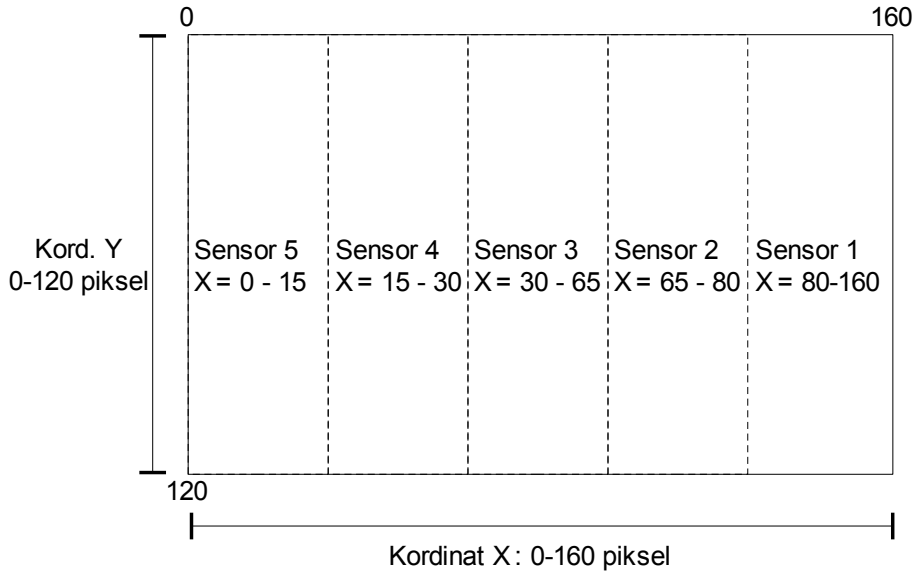
Kendali Proporsional Integral Derivatif (PID kontroler) merupakan mekanisme umum *control loop* dengan umpan balik yang digunakan secara luas pada sistem kontrol untuk industri. Kontroler PID akan menghitung nilai *error* sebagai selisih antara *Process Variable (PV)* yang diukur dengan S*et Point (SP)* yang diharapkan.

Perhitungan algoritma PID kontroler melibatkan tiga variabel yaitu nilai proporsional (P), integral (I), dan derivatif (D). Nilai variabel P bergantung pada nilai *error* saat ini, I merupakan nilai akumulasi *error* sebelumnya, dan D merupakan nilai prediksi dari *error* yang akan datang.

Aplikasi algoritma *Region Grow* pada HaViMo 2.0 menghasilkan kordinat X dan Y posisi garis terhadap sensor kamera, maka hal tersebut bisa diformulasikan agar menyerupai pola deteksi sensor garis.

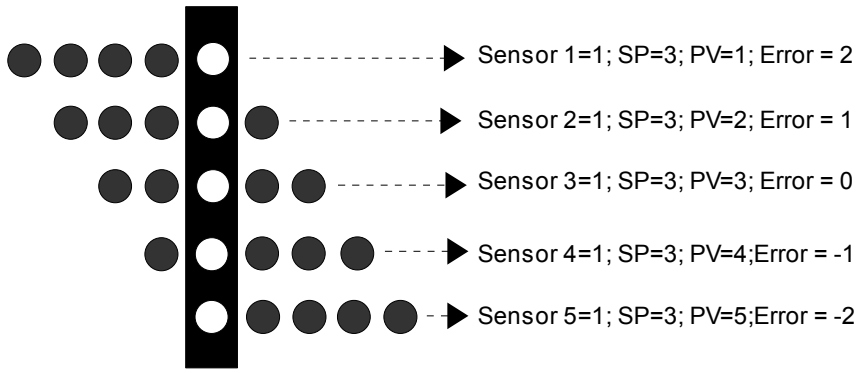
Resolusi *image* HaViMo 2.0 yang berukuran 120 piksel x 160 piksel dibagi menjadi 5 bagian daerah deteksi. Masing-masing daerah deteksi mewakili tiap-tiap sensor garis *virtual*. Besar *range* untuk tiap-tiap sensor *virtual* didapatkan dari hasil pengukuran langsung pada garis. Kordinat Y dari garis diabaikan pada aplikasi ini karena tidak memberikan efek terhadap pembacaan garis.

Nilai Kordinat X dipetakan mewakili 5 buah sensor sebagai berikut.



Gambar 9
Pembagian daerah deteksi *(virtual sensor)* pada *image* yang ditangkap oleh HaViMo 2.0

Pembobotan *Present Value (PV)* sensor-sensor garis *(virtual)* dapat digambarkan sebagai berikut.



Gambar 11
Pembobotan *Present Value*

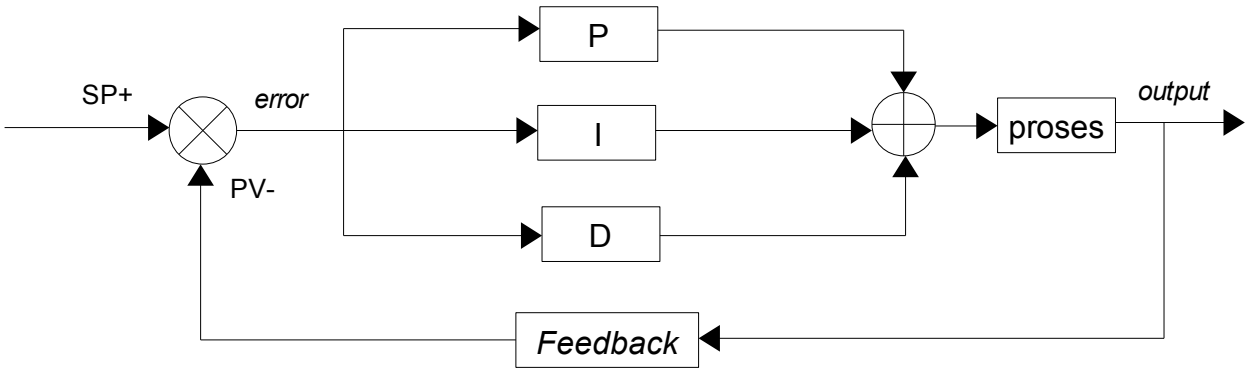
Nilai kesalahan dapat dinyatakan sebagai berikut :

Error = SP – PV

Keterangan :
SP = Set Point
Merupakan nilai/parameter yang diinginkan atau dijadikan acuan.
PV = Present Value
Merupakan nilai yang biasa disebut nilai sensor saat ini atau variabel terukur yang di umpan balikkan oleh sensor.
Error = nilai kesalahan
Merupakan nilai yang bisa disebut selisih antara nilai yang diinginkan (SP) dengan nilai yang terukur saat ini (PV)

Nilai *Set Point (SP)* yang diharapkan adalah *SP*=3 (saat sensor 3 =1) sehingga posisi robot tepat berada pada garis, apabila nilai *PV* terukur juga bernilai 3, maka pada kondisi ini nilai *Error (SP - PV)* sama dengan nol. Apabila kondisi *PV* sama dengan satu, maka nilai *Error* adalah 2. Nilai Error inilah yang jadi masukan untuk formula PID menghitung dan menghasilkan sinyal kontrol agar nilai *Error* terus mendekati nol.

Perhitungan nilai kontrol PID erat kaitannya dengan pengolahan *error*. Nilai *error* yang didapat akan diolah dengan rumus PID yang kemudian menghasilkan sinyal kontrol yang akan mengendalikan aktuator robot *line follower.* Berikut blok diagram kontroler PID.



Gambar 10
Blok Diagram PID kontroler

Formula kontrol PID dapat dinyatakan sebagai berikut.

PID = (Kp * Error) + (Ki * Integral_Error) + (Kd * (Error - Error_Sebelumnya))

Keterangan :
Kp = Konstanta proporsional *gain*
Ki = Konstanta integral *gain*
Kd = Konstanta derivatif *gain*
Integral_Error = hasil penjumlahan nilai-nilai error sebelumnya

Langkah-langkah menetukan nilai Kp dan Kd dilakukan melalui *trial* dan *error* sebagai berikut.

1. Sebagai awalan dipergunakan kecepatan motor yang tidak terlalu cepat yaitu nilai PWM 50 dari nilai PWM penuh (255). Hal tersebut juga mempertimbangkan kecepatan HaViMo 2.0 dalam membaca dan mengolah data yang berkisar antara 10-11 Hz.
2. Pertama-tama set semua nilai Kp, Kd, dan Ki menjadi 0 (nol). *Tuning* pertama dilakukan pada nilai Kp, tambahkan nilai Kp dari 0 hingga nilai yang membuat robot dapat mengikuti garis secara baik meskipun pergerakan robot masih goyang (berosilasi). Pada AN205 ini nilai Kp yang didapat yaitu 60.
3. Ketika robot sudah bisa mengikuti garis dengan baik namun masih bergoyang, nilai Kd dapat ditambahkan dengan membagi dua nilai Kp yang didapatkan. Lakukan pengamatan hasil penerapannya. Nilai Kd dapat ditambah maupun dikurangi agar robot lebih stabil. Pada AN205 ini nilai Kd yang didapat yaitu 30.
4. Pada AN205 ini nilai Ki dibiarkan nol sehingga, Ki dianggap tidak termasuk dalam hasil perhitungan.
5. Apabila dirasa masih kurang stasbil dapat dilakukan *tuning* kembali sampai didapatkan hasil yang diinginkan.

Library Dynamixel_Havimo

Kumpulan dari *instruction-instruction* Dynamixel servo dan kamera HaViMo yang telah dirangkum kedalam library program dengan nama *Dynamixel_Havimo.c dan* file header *Dynamixel_Havimo.h*. Library ini disusun berdasarkan *library* Arduino *Dynamixel.cpp* dan *Dynamixel.h* yang dibuat oleh Josué Alejandro Savage.
Beberapa fungsi-fungsi yang ada didalamnya sebagai contoh :

- **Hvm_ping()**
  Deskripsi : Melakukan ping ke HaViMo 2.0
  Sintaks : Hvm_ping();

Parameter: -
Contoh :
Hvm_ping();
Akan mengirimkan perintah untuk melakukan ping ke HaViMo 2.0
Respon Status Packet :
Jawaban ping HaViMo 2.0

- **Hvm_CapRegion()**
  Deskripsi : Menangkap gambar dan kemudian dilakukan *Image Processing Region Growing*.
  Sintaks : Hvm_CapRegion();
  Parameter: -
  Contoh :
  Hvm_CapRegion();
  Akan mengirimkan perintah untuk menangkap gambar dan melakukan *Region Growing*.
  Respon Status Packet :
  Command ini tidak memberikan respon status apapun.

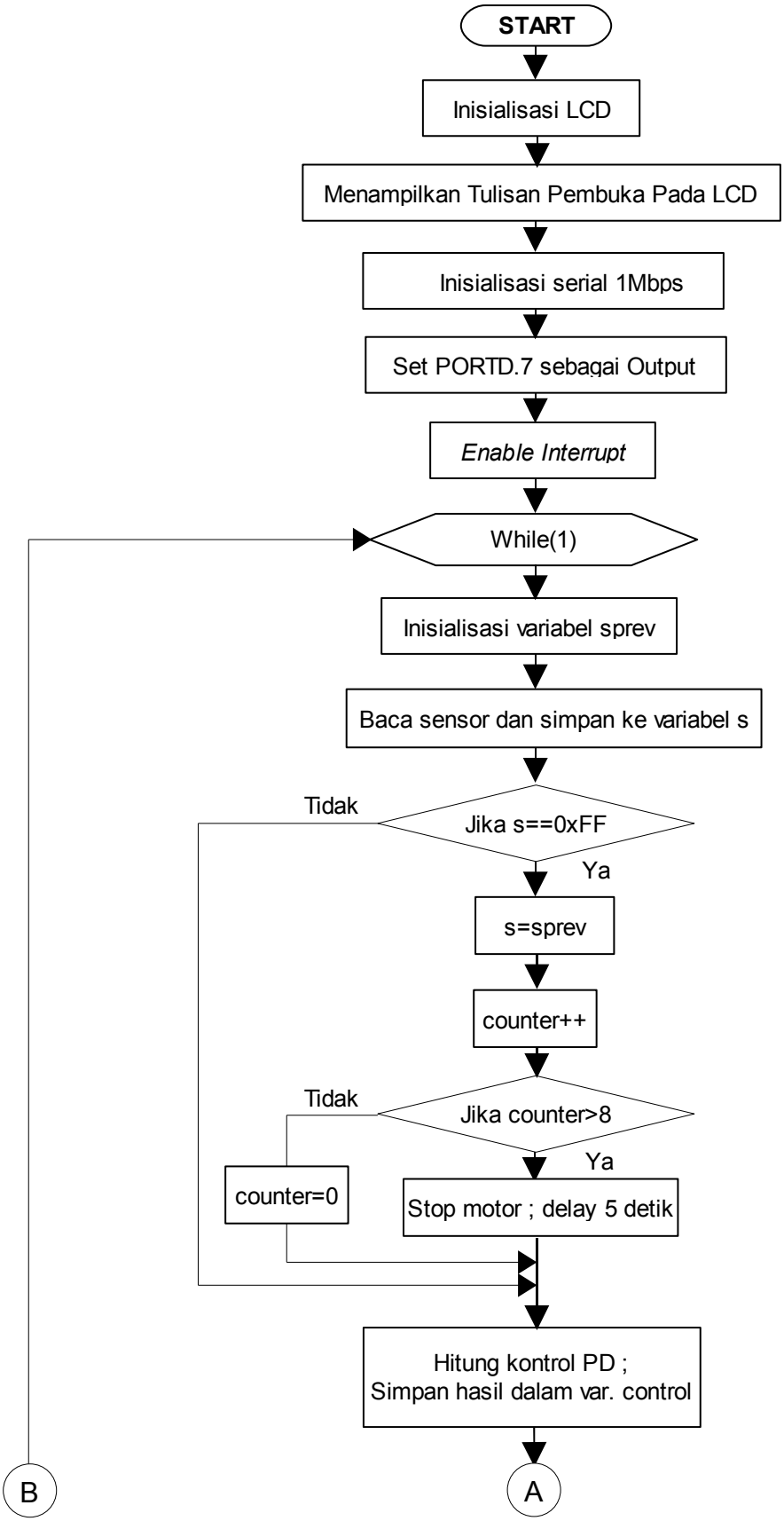
- **Hvm_ReadRegion()**
  Deskripsi : Membaca hasil dari Region Growing.
  Sintaks : Hvm_ReadRegion(unsigned char reg_address);
  Parameter: Register address hasil pembacaan pada alamat 0x10 s.d. 0xFF.
  Contoh :
  Hvm_ReadRegion(0x10);
  Akan mengirimkan perintah untuk membaca nilai register 0x10 hasil dari *Region Growing*.
  Respon Status Packet :
  Nilai register yang ingin diketahui nilainya.

Tambahan pada *library UART*

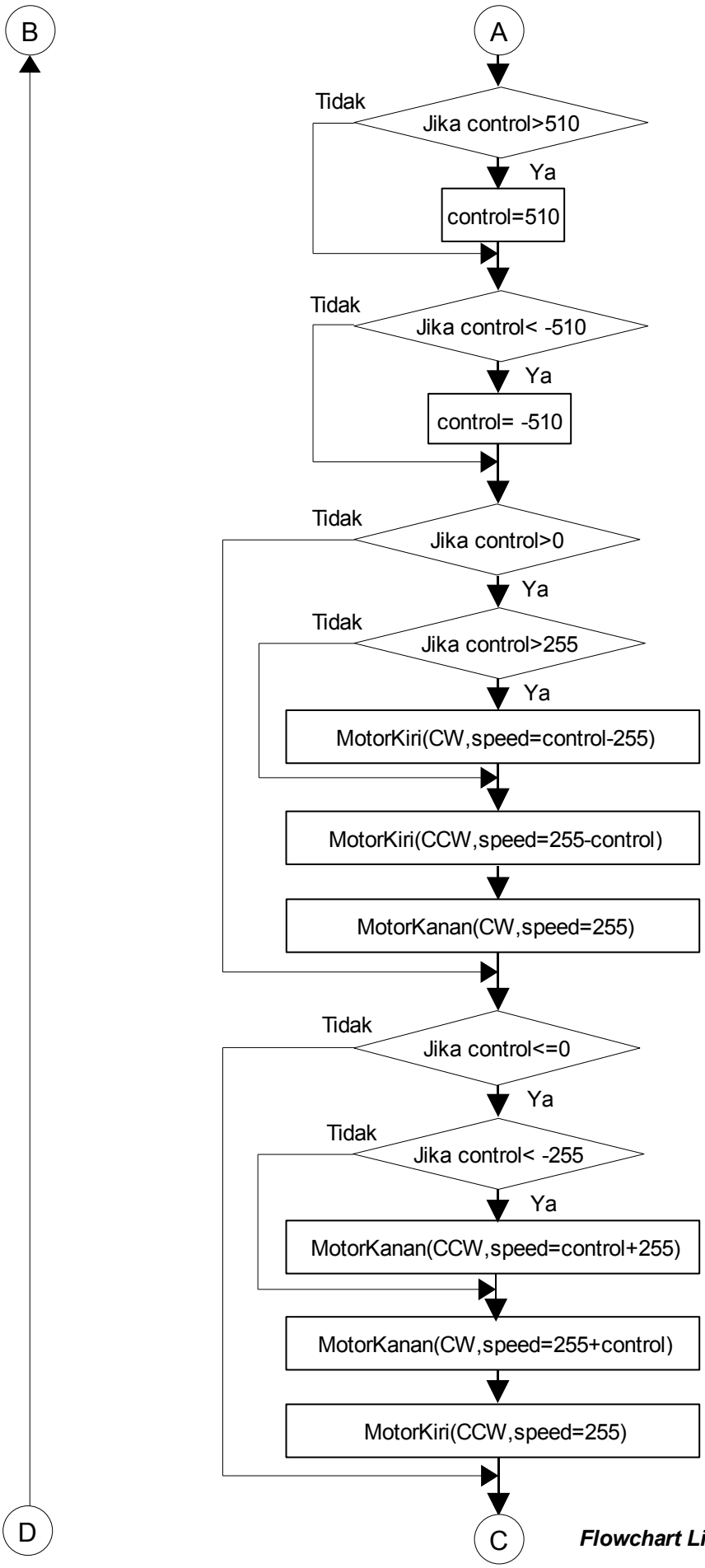
Untuk mendukung *library Dynamixel_Havimo* yang sumber aslinya berjalan sebagai *library* Arduino diperlukan fungsi khusus pada program *UART*, beberapa fungsi tersebut antara lain:

- **uart_available()** = berfungsi untuk melihat jumlah byte yang diterima oleh receiver buffer
- **uart_peek()** = berfungsi untuk mengintip *(peek) byte* buffer teratas dari data diterima yang belum dibaca.
- **uart_flush()** = berfungsi membersihkan *Rx buffer* dari data diterima yang belum dibaca.

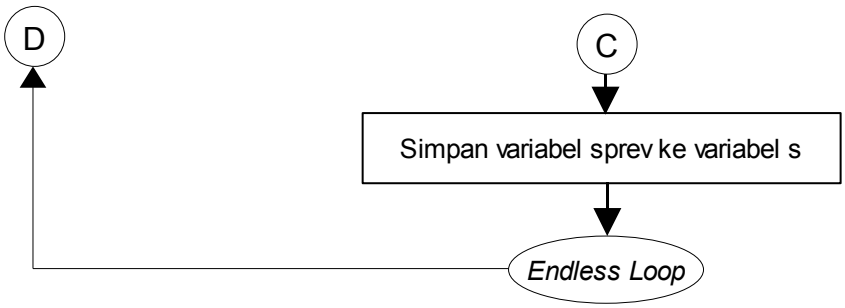
Adapun alur program dari Havimo_Line_Follower_pwm.exe adalah sebagai berikut:



Gambar 12
***Flowchart Line Follower Robot* (bagian 1)**



Gambar 13
***Flowchart Line Follower Robot* (bagian 2)**

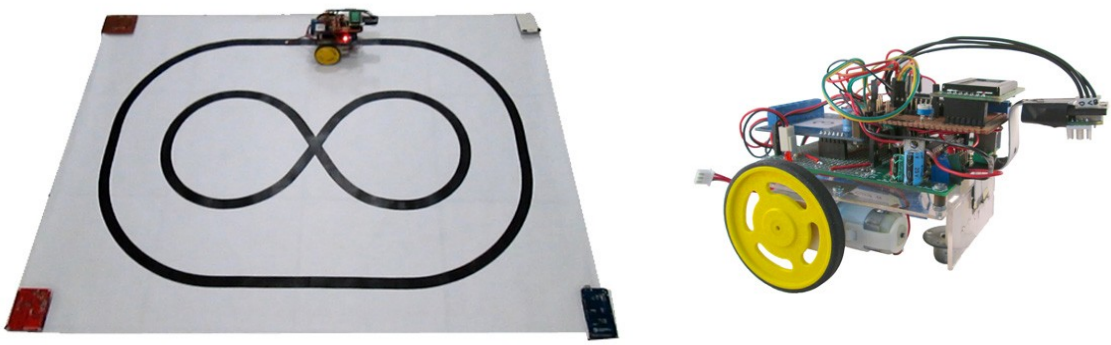


Gambar 14
***Flowchart Line Follower Robot* (bagian 3)**

Penjelasan urutan kerja dari program diatas adalah sebagai berikut :

1. Program melakukan inisialisasi LCD 8x2 pada PORTC.
2. Menampilkan Tampilan pembuka pada LCD.
3. Inisialisasi komunikasi Serial UART dengan baudrate 1Mbps.
4. Set PORTD.7 sebagai output untuk pengontrol pin *data direction.*
5. Meng-*enable* kan *Global Interrupt.*
6. Masuk kedalam *Superloop.*
7. Menginisialisasi variabel 'sprev' sebagai variabel yang akan menampung nilai sensor sebelumnya.
8. Membaca nilai sensor dan menyimpannya ke dalam variabel 's'.
9. Jika nilai 's' adalah 0xFF maka nilai 's' dimasukkan ke dalam variabel 'sprev'
10. tambah nilai 'counter', jika 'counter' bernilai lebih dari delapan stop motor selama 5 detik kemudian melalukan perhitungan kontrol PID. Jika counter adalah 0 maka langsung melakukan perhitungan kontrol PID. Hasil perhitungan PID disimpan pada variabel 'control'.
11. Melakukan normalisasi nilai 'control'. Jika nilai lebih besar dari 510, maka nilai 'control' adalah 510. Jika 'control' lebih kecil dari -510, maka nilai 'control' adalah -510.
12. Jika nilai 'control' lebih besar dari 0, dan kemudian nilai 'control' lebih besar dari pada 255, maka motor kiri bergerak searah jarum jam dengan kecepatan sama dengan nilai 'control' dikurangi dengan 255.
13. Jika nilai 'control' lebih besar dari 0 dan nilai 'control tidak lebih besar dari 255, maka motor kiri bergerak berlawanan jarum jam dengan kecepatan sama dengan 255 dikurangi dengan 'control'.
14. Motor kanan bergerak searah jarum jam dengan kecepatan penuh 255.
15. Jika nilai 'control' kurang dari atau sama dengan 0, dan kemudian nilai 'control' kurang dari pada -255, maka motor kanan bergerak berlawanan jarum jam dengan kecepatan sama dengan nilai 'control' ditambah dengan 255.
16. Jika nilai 'control' kurang dari atau sama dengan 0 dan nilai 'control tidak kurang dari -255, maka motor kanan bergerak searah jarum jam dengan kecepatan sama dengan 255 ditambah dengan 'control'.
17. Motor kiri bergerak berlawanan jarum jam dengan kecepatan penuh 255.
18. Menyimpan hasil pembacaan sensor 's' ke dalam variabel 'sprev'.
19. Program kembali ke langkah nomor 6 (Superloop).

Gambar koneksi keseluruhan modul dapat dilihat pada **Gambar 15**.



Gambar 15
Rangkaian antar modul pada AN205

Listing program aplikasi ini terdapat pada **AN205.ZIP**

Selamat berinovasi!