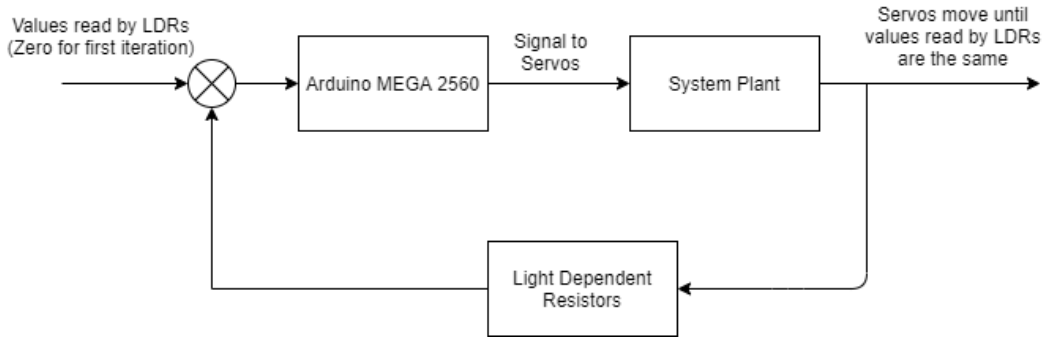
BAB III

METODE DAN PERANCANGAN SISTEM

3.1 Diagram Blok

Melalui Gambar 3.1, bisa dilihat bahwa sistem memiliki beberapa komponen diantaranya adalah *controller* yang berupa Arduino MEGA 2560 dimana semua *logic* sistem berjalan, blok *System Plant* berisi aktuatornya, lalu blok *Light Dependent Resistors* berisi empat LDR sebagai sensor.



Gambar 3.1 Diagram Blok Sistem

Blok *controller* atau *Arduino MEGA 2560* merupakan *controller* dari sistem. Artinya, dalam blok inilah perhitungan, perbandingan, dan seluruh logika *software* sistem berjalan. Masukan dari blok ini berupa nilai yang diberikan oleh blok *Light Dependent Resistors* dalam bentuk intensitas cahaya, lalu di dalam blok ini akan dilakukan operasi matematis di sistem kendali yang pada akhirnya akan mengirimkan sinyal PWM ke *System Plant* sebagai keluaran blok ini.

Blok *System Plant* merupakan *plant* sistem secara keseluruhan, tidak termasuk sensor. Blok ini sendiri memiliki dua blok lagi jika sistem dilihat lebih spesifik, yaitu dua motor servo sebagai aktuator sistem yang mewakilkan sumbu vertikal dan horizontal. Masing-masing motor servo memilki beban yang ditanggung berbeda. Oleh karena itu, torsinya akan berbeda juga yang

mengakibatkan fungsi transfernya berbeda juga. Dua blok motor servo ini disusun secara bersebelahan di dalam blok *System Plant*. Blok ini akan menerima input dari *controller* yang berisi perintah untuk menggerakkan kedua motor servo sesuai dari sistem kendali di *controller*. Keluaran dari blok ini berupa gerakan kedua motor servo sampai perintah yang dikirim *controller* tidak memiliki PWM yang menggerakkan motor.

Blok *Light Dependent Resistors* merupakan komponen sensor dari sistem yang terdiri dari empat *Light dependent resistor*. Blok ini akan jalan setelah blok *System Plant* berjalan. Masukan dari blok ini berupa perintah untuk menjalankan keempat sensor. Lalu, di dalam blok ini keempat LDR akan membaca nilai intensitas cahaya dari lingkungan dan menjadikan nilai ini menjadi keluaran dari blok sebagai umpan balik untuk *controller*.

Adapun kronologi singkat dari sistem. Sistem diasumsi berjalan pada iterasi kedua karena sensor baru akan memberikan bacaan pada iterasi kedua. Setelah keempat sensor LDR memberikan bacaan intensitas, Arduino akan melakukan operasi matematis untuk mengetahui apakah nilai bacaan dari keempat LDR sama, jika tidak sama maka Arduino akan menjadikan selisihnya sebagai masukan sistem kendali. Sistem kendali lalu akan menggerakkan dua motor servo sampai bacaan keempat LDR sama. Keadaan dimana nilai keempat LDR sama ini merupakan momen dimana panel surya menghadap ke matahari secara tegak lurus.

3.2 Perancangan Sistem Kendali

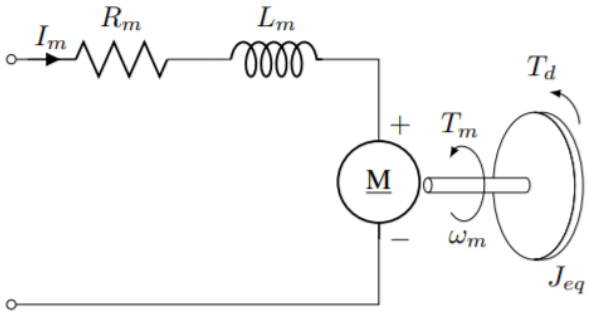
Untuk perancangan sistem kendali *full state observer*, akan dibagi menjadi beberapa bagian, yaitu:

1. Mencari dan menentukan fungsi alih *plant*

2. Menentukan dan merancang spesifikasi kontroler
3. Menentukan dan merancang spesifikasi estimator
4. Menggabungkan keduanya dalam satu diagram blok *full state observer*

3.2.1 Penentuan Fungsi Alih Motor Servo

Gambar 3.2 menunjukkan konstruksi motor servo. Umumnya, motor servo merupakan motor DC yang ditambah dengan *gearbox*, potensiometer, dan rangkaian jembatan H.



Gambar 3.2 Rangkaian Elektrik Motor Servo

Torsi motor (T_m) merupakan hasil kali anatara konstanta torsi motor dengan *anchor current* atau seperti pada persamaan 9.

$$T_m = K_{tn} i_a \quad (9)$$

Pada rangkaian terdapat juga *back emf* yang nilainya akan selalu sebanding dengan kecepatannya sehingga didapatkan persamaannya pada persamaan 10.

$$e_b = K_b \frac{d\theta}{dt} \quad (10)$$

Dengan K_b adalah konstanta *back emf*.

Berikutnya, bisa didapatkan persamaan diferensial dan persamaan torsinya seperti pada persamaan 11 dan 12.

$$L_a \frac{di_a}{dt} + R_a i_a + e_b = e \quad (11)$$

$$J \frac{d\theta^2}{dt^2} f_o \frac{d\theta}{dt} = T_m = K_{tn} i_a \quad (12)$$

Persamaan 10 sampai persamaan 12 lalu akan dicari transformasi *Laplace* nya sehingga menjadi persamaan 13 sampai 15.

$$E_b(s) = K_b s\theta(s) \quad (13)$$

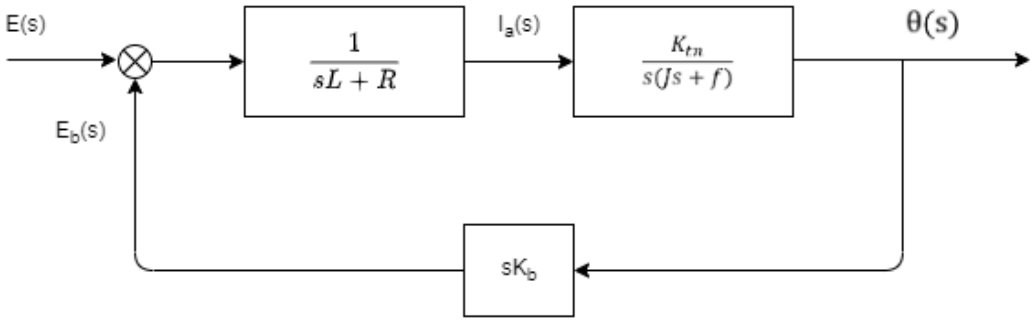
$$(L_a s + R_a)I_a(s) = E(s) - E_b(s) \quad (14)$$

$$(Js^2 + f_o s)\theta(s) = T_m(s) = K_{tn} I_a(s) \quad (15)$$

Berikutnya, mensubstitusikan persamaan 13 sampai 15 untuk mendapatkan fungsi alih motor servo seperti pada persamaan berikut dan bisa direpresentasikan dalam diagram blok seperti pada Gambar 3.3

$$G(s) = \frac{\theta(s)}{Ea(s)} = \frac{K_{tn}}{s[(R_a + sL_a)(Js + f_o) + K_T K_b]} \quad (16)$$

Persamaan 11 juga bisa dirancang dalam bentuk diagram blok seperti pada Gambar 3.3.



Gambar 3.3 Diagram Blok Motor Servo

Untuk merancang sistem kendali, digunakan fungsi alih motor servo yang ada di persamaan 16. Persamaan 16 bisa ditulis lagi menjadi persamaan 17.

$$\frac{\theta(s)}{Ea(s)} = \frac{Ktn}{s[La*Js^2 + (La*f + Ra*J)s + Ra*f + Ktn*Kb]} \quad (17)$$

Dengan:

$R_a = Armature\ resistor = 2.5\ \Omega$

$L_a = Armature\ Inductance = 0.062\ H$

$I_a = Armature\ current$

$V_a = Applied\ armature\ voltage$

$\tau = Motor\ torque$

$J = Moment\ of\ inertia = 0.00004 \frac{Kg}{m^2}$

$f = Motor\ and\ load\ coeffient\ of\ friction = 0.001 \frac{N}{\frac{rad}{s}}$

$K_{tn} = 0.026 \frac{Nm}{A}$

$K_b = 0.02 \frac{v}{rad/s}$

Bisa didapatkan fungsi alih motor servo tanpa mempertimbangkan perhitungan gir di dalam motor servo dengan memasukkan nilai pada variabel-variabel di persamaan 16 sehingga didapatkan fungsi alih servo di persamaan 18.

$$\frac{\theta(s)}{Ea(s)} = \frac{0.026}{0.00000248s^3 + 0.000162s^2 + 0.00302s} \quad (18)$$

Namun, fungsi alih diinginkan mewakili *plant* dari motor servo secara keseluruhan. Oleh karena itu, harus mempertimbangkan semua komponennya, termasuk beban yang digerakkan oleh aktuator. Fungsi alih akan berubah menjadi persamaan 19.

$$Total\ torque = (TF\ of\ Servo\ Motor * gear\ ratio) + \tau_d \quad (19)$$

Dengan τ_d adalah torsi dari masing-masing motor servo. Motor servo yang digunakan di penelitian ini ada dua, masing-masing mewakilkan sumbu yang berbeda; vertikal dan horizontal. Motor servo vertikal perlu mengangkat beban yang memiliki massa 5 kg dan panjang 50 cm. Sementara motor servo horizontal perlu mengangkat beban motor servo vertikal dan beban tambahan yang memiliki

massa 5 kg dan panjang 5 cm. Oleh karena itu, masing-masing motor servo mempunyai beban torsi yang berbeda-beda.

Torsi yang ada pada kedua motor servo bisa dicari dengan menggunakan persamaan 20 dan 21.

$$\tau_{Vertikal} = 2\ kg * 9.8\ m/s^2\ * 0.5\ m = 9.8\ Nm \quad (20)$$

$$\tau_{Horizontal} = 9.8\ Nm + (4\ kg * 9.8\ m/s^2 * 0.05\ m) = 11.76\ Nm \quad (21)$$

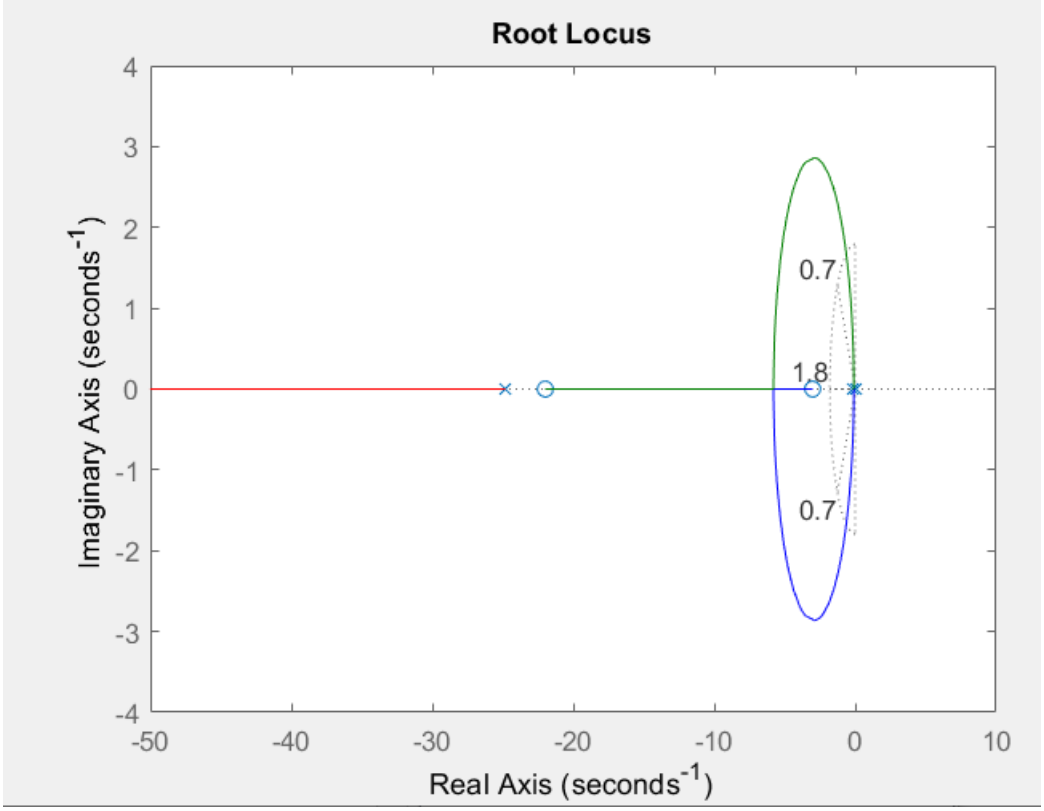
Lalu, persamaan disubstitusikan ke persamaan 19 untuk menghitung fungsi alih servo horizontal dan vertikal.

$$TF_{vertikal} = \frac{0.001497s^2+0.03742s+0.09927}{0.0001103s^3+0.004255s^2+0.03794s+0.009927} \quad (22)$$

$$TF_{horizontal} = \frac{0.001169s^2+0.02922s+0.06461}{0.0001119s^3+0.003966s^2+0.02974s+0.06461} \quad (23)$$

3.2.2 Penentuan Spesifikasi dan Perancangan Kontroler

Sistem kendali akan dirancang untuk masing-masing motor. Sistem diinginkan memiliki *maximum overshoot* dibawah 5% dan *rise time* maksimal 1 sekon. Dari persyaratan ini, bisa didapatkan nilai ω_n dan ζ sebesar 1.8 dan 0.7 secara berturut-turut. Setelah mendapatkan parameter kontrolernya, fungsi alih motor pada persamaan 22 dan 23 dipetakan dalam diagram *root locus* untuk ditentukan *pole* kendalinya. Hasil pemetaan di diagram *root locus* untuk motor servo vertikal ada di Gambar 3.3.



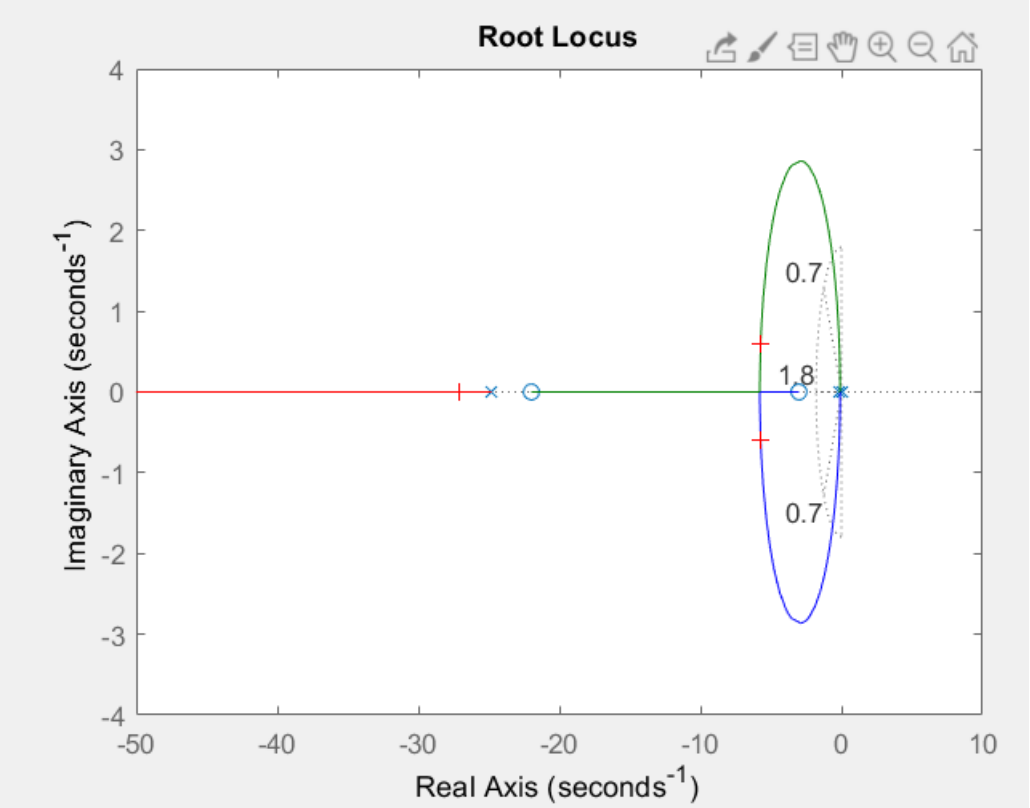
Gambar 3.4 Diagram *Root Locus* Sistem Belum Terkompensasi untuk Servo Vertikal

Mengacu kepada Gambar 3.4, garis putus-putus yang terbentuk dari sumbu imajiner 0.7 menandai lokasi *pole* dengan $\zeta = 0.7$, diantara garis ini, lokasi *pole* akan memiliki nilai $\zeta < 0.7$. Lalu, garis putus-putus berbentuk setengah lingkaran yang terbentuk dari sumbu *real* 1.8 menandai lokasi *pole* dengan frekuensi natural $\omega_n = 1.8$. Di dalam lingkaran, nilai $\omega_n < 1.8$ sementara diluar lingkaran nilai $\omega_n > 1.8$.

Kembali ke spesifikasi kontrolernya, diinginkan *maximum overshoot* kurang dari 5%, untuk memenuhi spesifikasi ini letak *pole* harus diantara sumbu imajiner positif dan negatif 0.7, dan untuk memenuhi spesifikasi *rise time* kurang dari 1 sekon, *pole* harus diletakkan diluar dari setengah lingkaran yang terbentuk dari sumbu *real* 1.8. Oleh karena itu, letak *pole* dipilih pada persamaan 24, sesuai

dengan Gambar 3.5. Didapatkan *Gain close loop* nya sebesar 3.8179 dan *gain feedforward* atau $\bar{N}$ sebesar 0.9998.

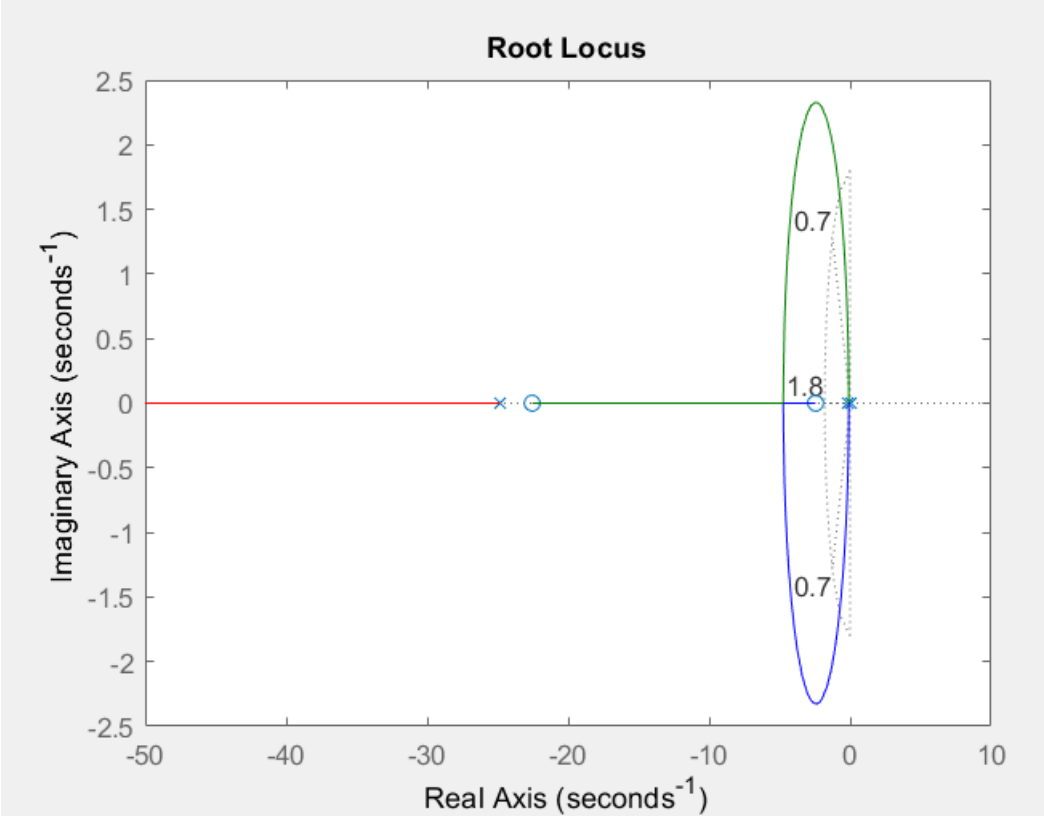
$$[-27.1041 \quad -5.7311 + 0.5939i \quad -5.7311 - 0.5939i] \qquad (24)$$



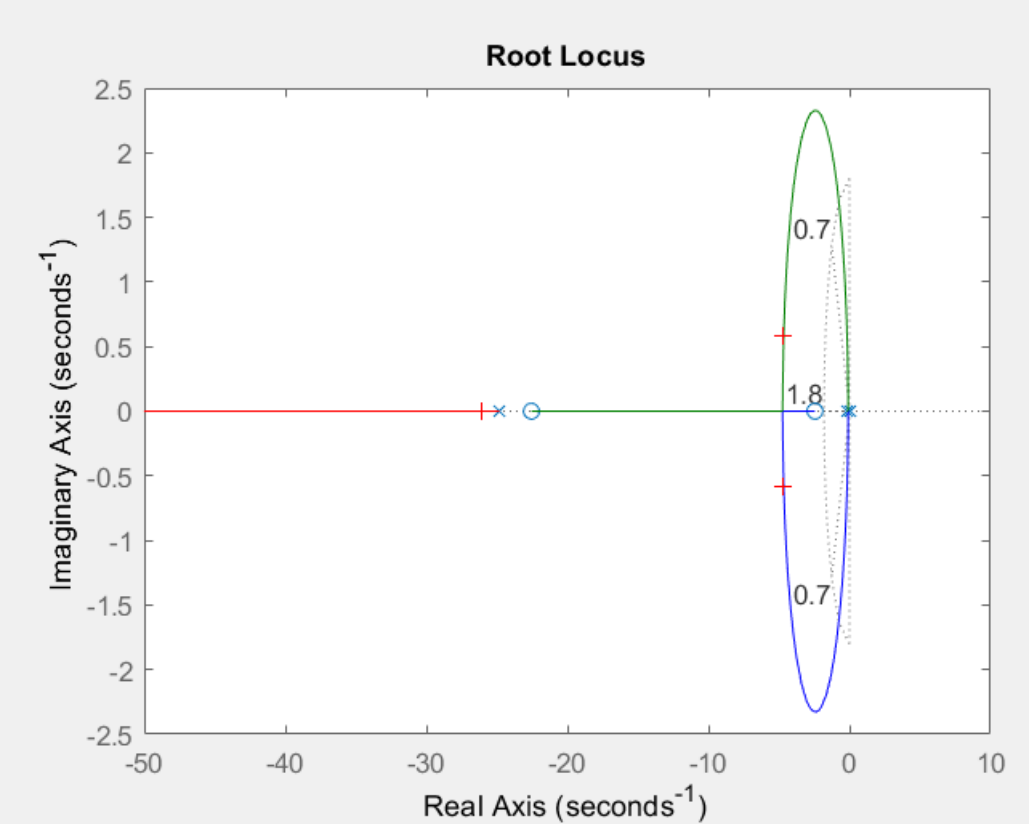
Gambar 3.5 Pemilihan Lokasi *Pole* Kendali di Diagram R*oot Locus* untuk Servo Vertikal

Setelah itu, hal yang sama akan dilakukan untuk motor servo horizontal. Didapatkan *pole* kendali pada persamaan 25 dengan *gain close* loop sebesar 2.4848 dan *gain feedforward* atau $\bar{N}$ sebesar 1. Hasil pemilihan *pole* kendali ada pada Gambar 3.7 sementara Gambar 3.6 merupakan pemetaan fungsi alih motor servo horizontal pada diagram *root locus*.

$$[-26.1146 \quad -4.666 + 0.5819i \quad -4.666 - 0.5819i] \qquad (25)$$



Gambar 3.6 Diagram *Root Locus* Sistem Belum Terkompensasi untuk Servo Horizontal



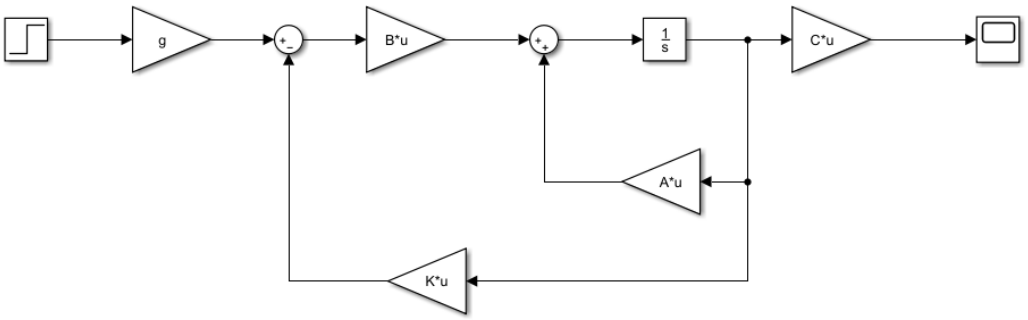
Gambar 3.7 Pemilihan Lokasi *Pole* Kendali di Diagram *Root Locus* untuk Servo Horizontal

Dari *pole* kendali yang sudah ditentukan, bisa dicari nilai *gain* kendali dalam bentuk matrix dengan menggunakan metode *pole placement.* Hasil *gain* kendali untuk motor servo horizontal dan vertikal ada di persamaan 26 dan 27.

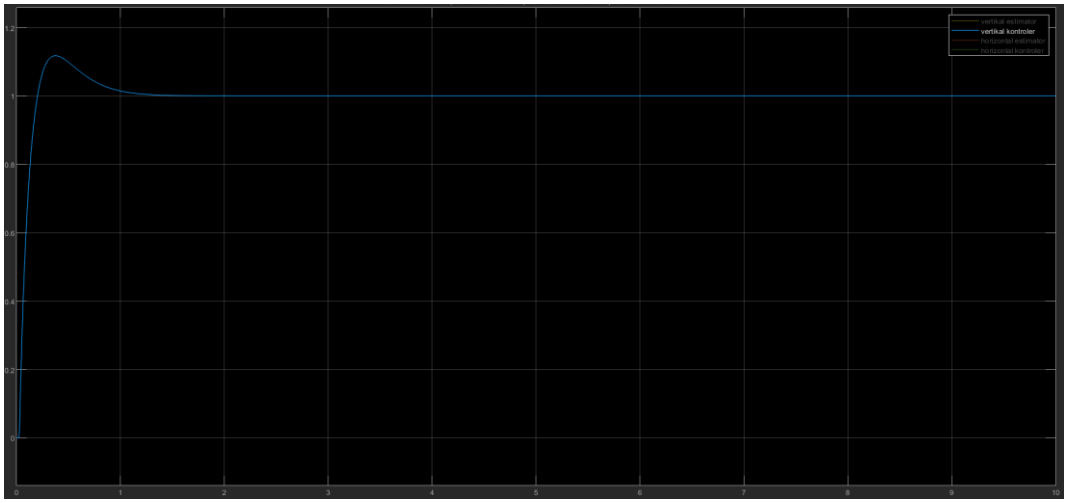
$$K_{vertikal} = [-0.0103 \quad -0.1001 \quad -0.1920] \tag{26}$$

$$K_{horizontal} = [0.0042 \quad 0.0386 \quad 0.0076] \tag{27}$$

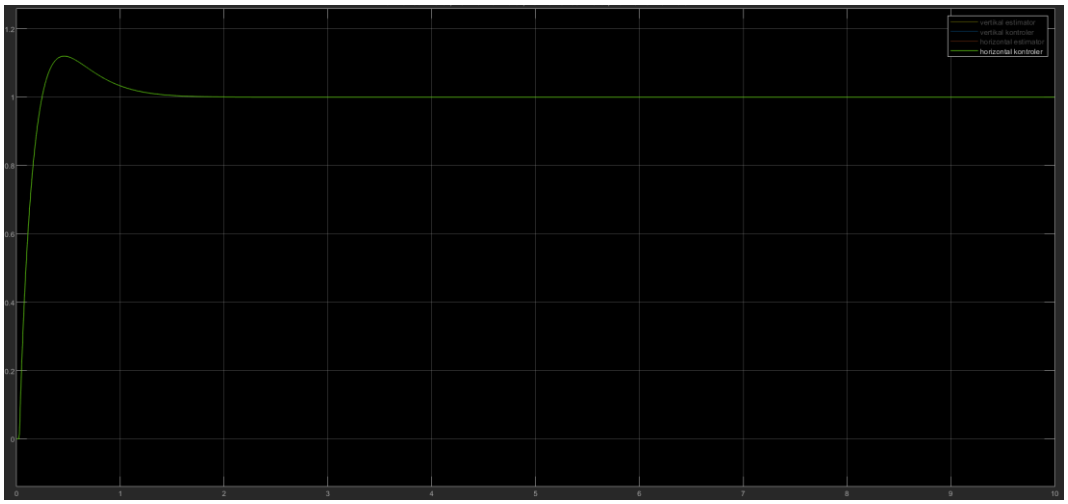
Setelah mendapatkan *gain* kendali untuk kedua motor servo, respon sistemnya akan dilihat melalui perancangan diagram blok seperti pada Gambar 3.8. Dari diagram blok ini, sistem disimulasikan dan didapatkan responnya pada Gambar 3.9 dan 3.10.



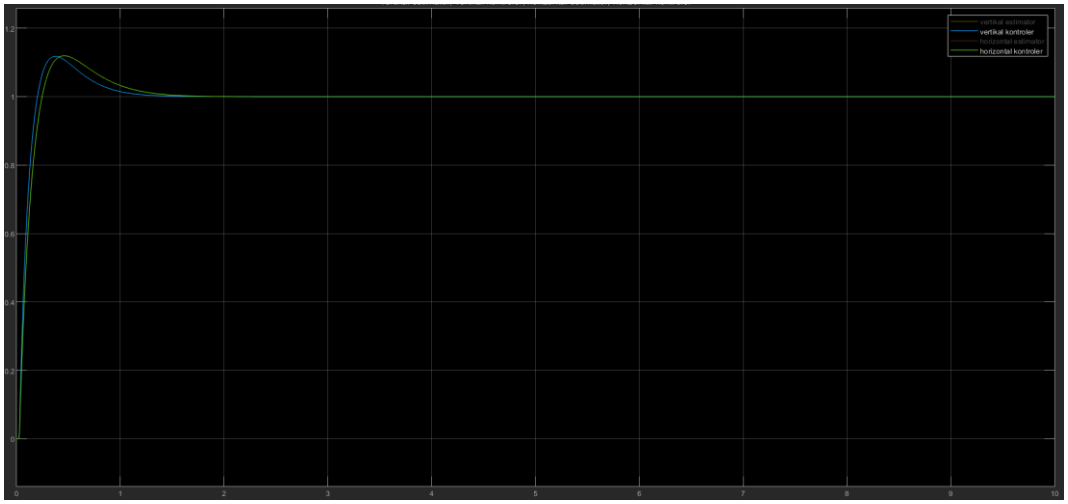
Gambar 3.8 Diagram Blok Motor Servo Vertikal dan Horizontal dengan Kontroler



Gambar 3.9 Respon Motor Servo Vertikal dengan Kontroler



Gambar 3.10 Respon Motor Servo Horizontal dengan Kontroler



Gambar 3.11 Respon Motor Servo Horizontal dan Vertikal dengan Kontroler

Mengacu pada Gambar 3.11, respon yang berwarna biru adalah respon motor servo vertikal dan warna hijau adalah motor servo horizontal. Terlihat bahwa motor servo horizontal memiliki *rise time* dan *transient time* yang lebih lama daripada motor servo vertikal.

3.2.3 Penentuan dan Perancangan Spesifikasi Estimator

Pada penelitian ini, estimator akan dirancang dengan spesifikasi bisa melakukan estimasi sepuluh kali lebih cepat daripada sistem. Oleh karena itu, letak *pole* estimator dipilih dari letak *pole* kendali dikali sepuluh seperti pada persamaan 28 dan 29 untuk *pole* estimator motor servo vertikal dan horizontal berturut-turut.

$$[-275.03 \quad -57.862 + 7.046i \quad -57.862 - 7.046i] \qquad (28)$$

$$[-264.512 \quad -46.949 + 7.816i \quad -46.949 - 7.816i] \qquad (29)$$

Lalu dari *pole-pole* estimator, dicari persamaan karakteristiknya dengan menggunakan persamaan 30. Berikutnya mencari persamaan 31 yang di dalamnya memiliki matrix G. Matrix ini merupakan matrix *gain* estimator dalam bentuk variabel seperti pada persamaan 32. Setelah itu, persamaan 30 dan 31 dibandingkan

untuk mendapatkan *gain* estimator yang berbentuk angka dalam susunan matrix seperti pada persamaan 33 dan 34.

$$(s - pole1) * (s - pole2) * (s - pole3) \quad (30)$$

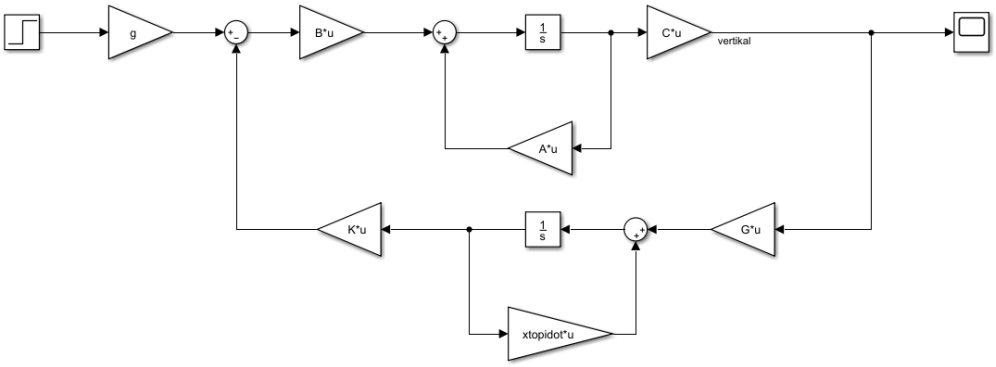
$$\det(sI - A + G * C) \quad (31)$$

$$G = \begin{pmatrix} g1 \\ g2 \\ g3 \end{pmatrix} \quad (32)$$

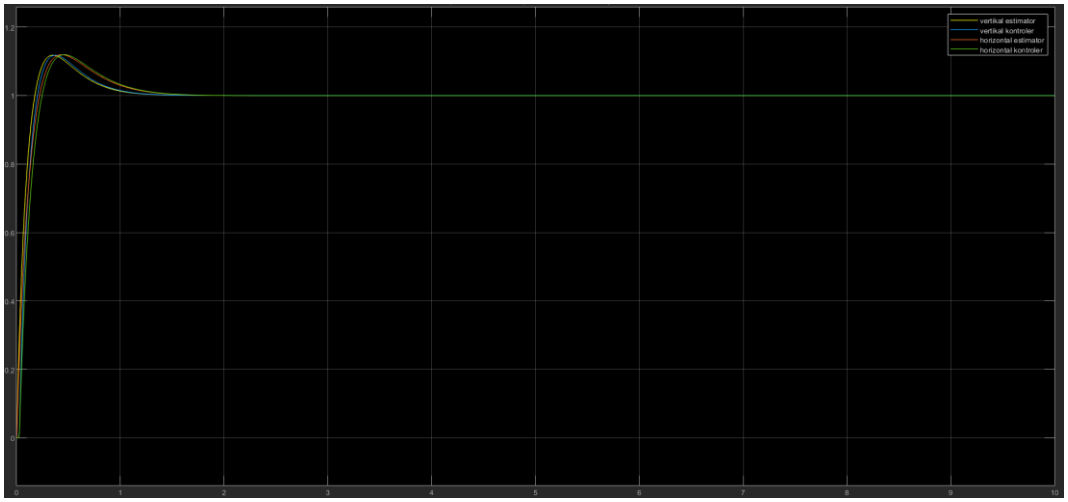
$$G_{vertikal} = \begin{pmatrix} 313.6233 \\ 30.37797 \\ -15.7955 \end{pmatrix} \quad (33)$$

$$G_{horizontal} = \begin{pmatrix} 221.0499 \\ 34.6809 \\ -19.1315 \end{pmatrix} \quad (34)$$

Setelah mendapatkan semua parameter kontroler dan estimator, bisa dirancang diagram blok sistem terkompensasi dengan kontroler dan estimator seperti pada Gambar 3.12.



Gambar 3.12 Diagram Blok Motor Servo Vertikal dan Horizontal dengan Estimator



Gambar 3.13 Respon Sistem Motor Servo Vertikal dan Horizontal Terkendali dan Terestimasi

Gambar 3.13 menunjukkan perbandingan respon motor servo vertikal dan horizontal menggunakan kontroler dan estimator. Jika dilihat respon sistemnya dengan estimator, respon motor servo vertikal dan horizontal dengan estimator memiliki *rise time* yang lebih bagus atau lebih kecil daripada tanpa menggunakan estimator (menggunakan kontroler saja). Namun peningkatan *rise time* tidak sepuluh kali lebih cepat daripada respon sistem dengan kontroler yang pada awal dispesifikasikan. Ini karena dengan menggunakan kontroler sudah didapatkan *rise time* yang kecil (dibawah satu sekon), maka dengan menambahkan estimator akan ada *diminishing return* sebab *rise time* sudah tidak bisa lebih cepat lagi.

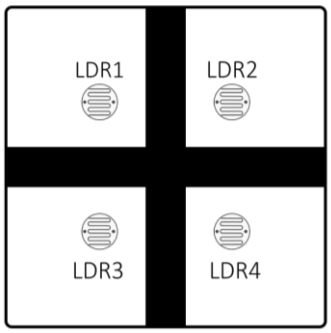
3.3 Perancangan Hardware

Perancangan *hardware* dari sistem meliputi semua komponen yang berbentuk fisik. Diantaranya adalah sensor dan kerangka sistem.

3.3.1 Perancangan Sensor

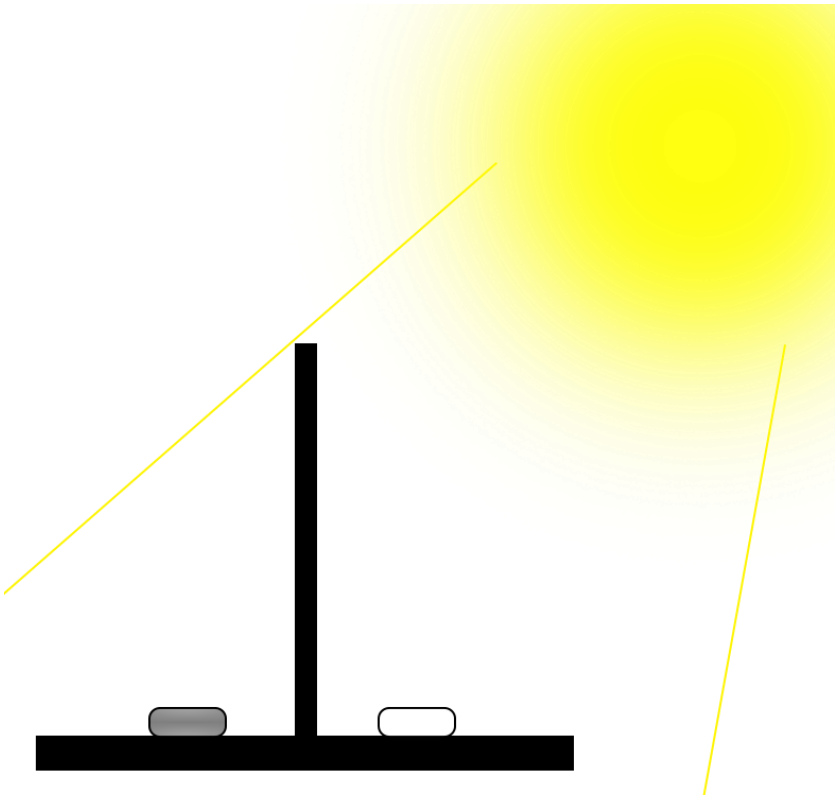
Seperti yang sudah dijelaskan pada bab sebelumnya, sensor dalam sistem ini menggunakan *light dependent resistor.* Sistem menggunakan sebanyak empat

buah LDR untuk dapat mencapai spesifikasinya. Keempat LDR ini akan digabungkan menjadi suatu sensor *fusion* dengan konfigurasi pada gambar berikut.



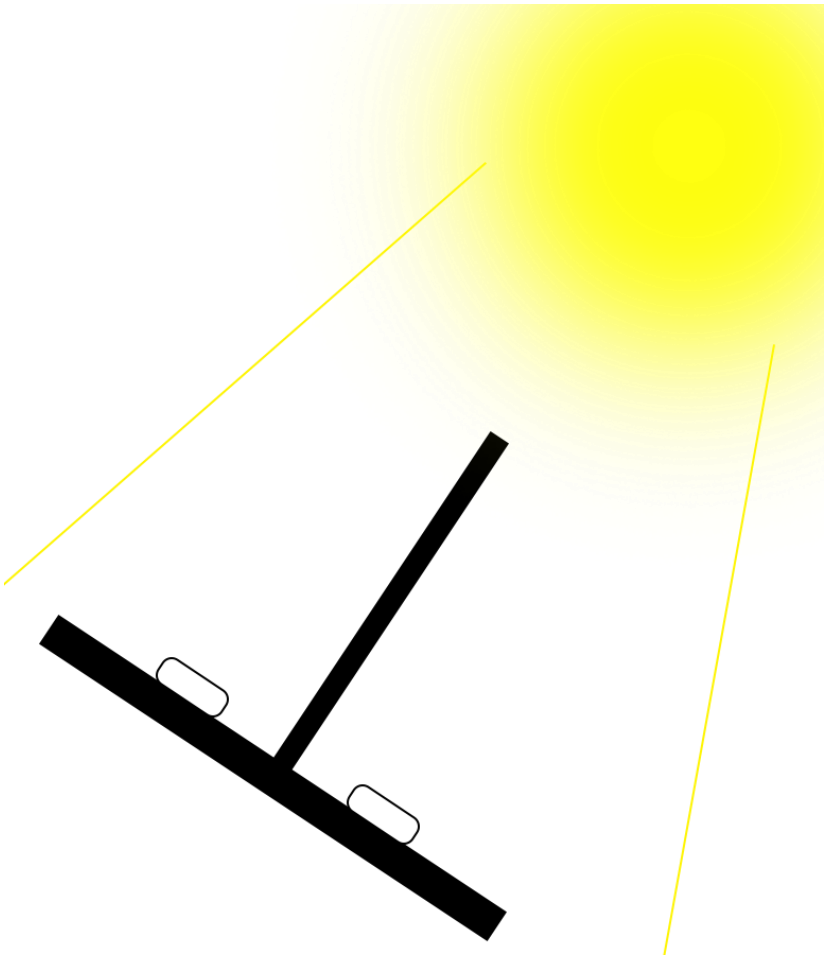
Gambar 3.14 Konfigurasi Sensor

Adapun proses sensor *fusion* nya. Akan dilakukan pengambilan rata-rata dari setiap dua LDR di setiap sisi sistem. LDR1 dan LDR2 akan diambil nilai rata-ratanya untuk mewakilkan sisi atas, LDR1 dan LDR3 akan diambil nilai rata-ratanya untuk mewakilkan sisi kiri, LDR3 dan LDR4 akan diambil nilai rata-ratanya untuk mewakilkan sisi bawah, LDR2 dan LDR4 akan diambil nilai rata-ratanya untuk mewakilkan sisi kanan. Berikutnya, akan diambil nilai representasi untuk setiap sumbu sistem yaitu vertikal dan horizontal. Untuk representasi nilai vertikal, akan diambil selisih dari nilai atas dan bawah sementara untuk representasi nilai horizontal, akan diambil selisih dari nilai kiri dan kanan. Nilai untuk sumbu vertikal dan horizontal ini lalu akan masuk ke masing-masing sistem kendali motor servo sebagai *error*.



Gambar 3.15 Ilustrasi Konfigurasi Sensor Keadaan Tidak Optimal

Gambar 3.15 memberikan ilustrasi lebih lanjut dari penjelasan sebelumnya. Keadaan pada Gambar 3.15 akan menghasilkan nilai *error* karena terdapat nilai pada horizontal atau vertikal yang berarti ada selisih antara sisi atas dengan bawah atau sisi kiri dengan kanan pada sistem. Jika hal ini terjadi, maka sistem kendali akan berjalan karena mendeteksi ada *error* untuk menggerakkan motor servo sampai keadaan pada Gambar 3.16 tercapai.



Gambar 3.16 Ilustrasi Konfigurasi Sensor Keadaan Optimal

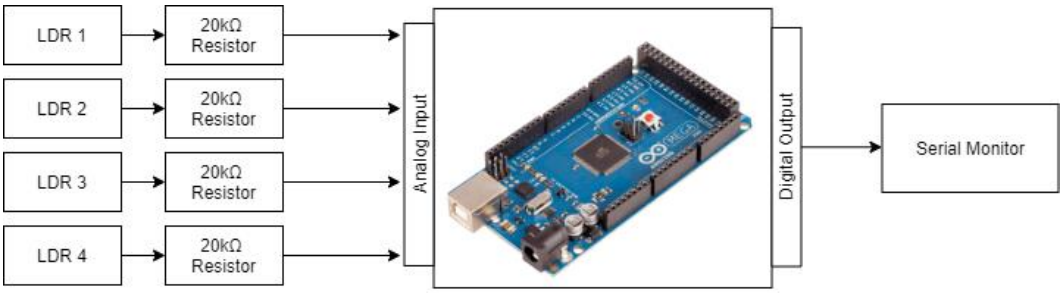
Jika keadaan pada Gambar 3.16 sudah tercapai, maka sistem dalam keadaan optimal dan sistem kendali tidak akan mengubah gerakan motor servo karena tidak mendeteksi ada nilai pada vertikal atau horizontal yang berarti tidak ada selisih antara sisi atas dengan bawah atau sisi kiri dengan kanan (*error)*.

3.3.2 Perancangan Akuisisi Data

Sebagai kebutuhan akuisisi data, sistem memerlukan resistor dan ADC sebagai pengkondisi sinyal yang diilustrasikan pada Gambar 3.17. ADC yang digunakan adalah ADC 10-bit yang ada di dalam Arduino.

Untuk data yang akan diakuisisi adalah data dari sensor saja Keluaran dari blok LDR 1 sampai LDR 4 dan blok resistor merupakan bacaan tegangan antara

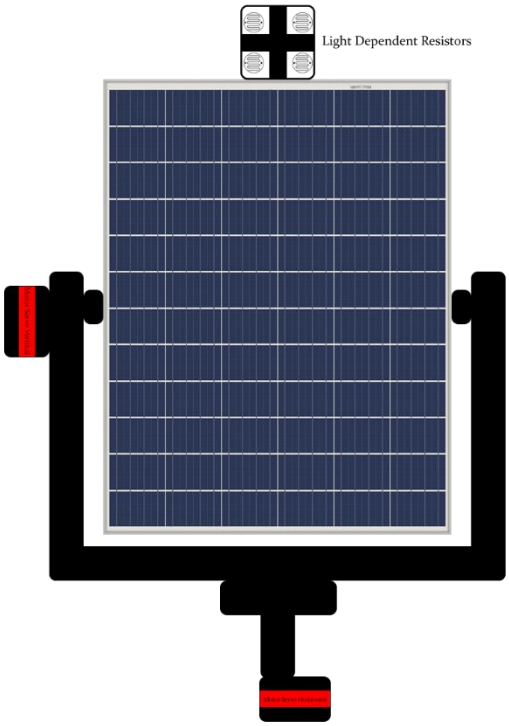
light dependent resistor dan resistor yang berbentuk suatu rangkaian *voltage divider*. Berikutnya, bacaan tegangan ini akan masuk ke pin analog di Arduino. Lalu, nilai analog dari empat sensor ini akan dikonversi oleh ADC di Arduino untuk ditampilkan di *output* dalam tampilan *serial monitor*.



Gambar 3.17 Diagram Akuisisi Data

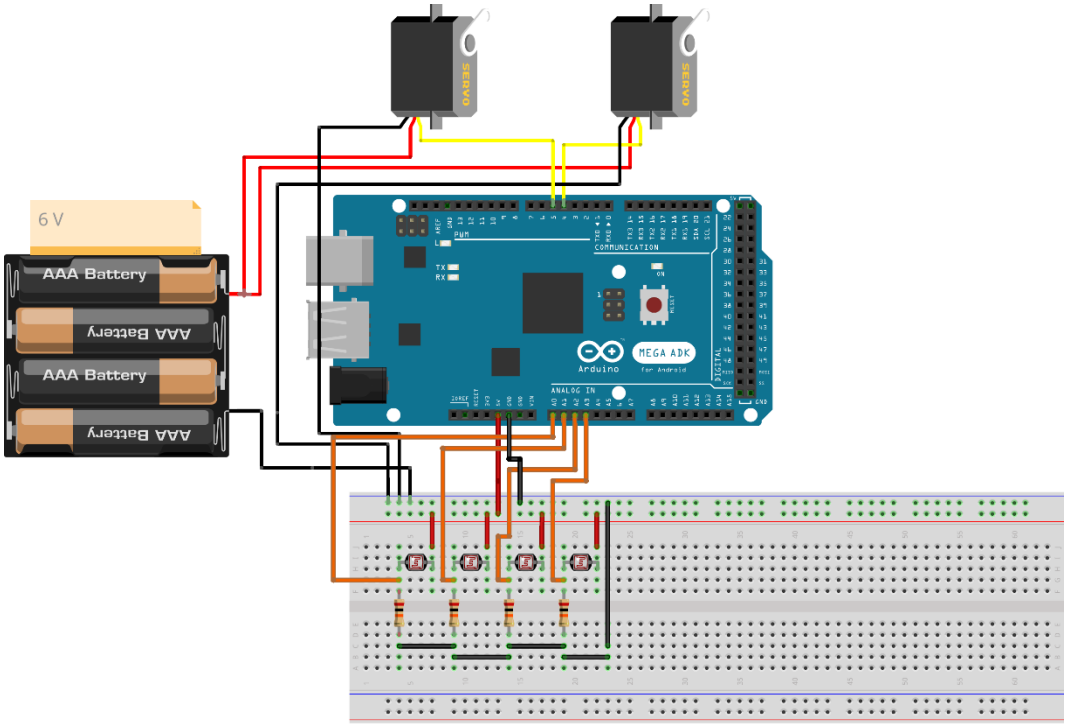
3.3.3 Perancangan Fisik Sistem

Melalui Gambar 3.18, bisa dilihat sistem memiliki beberapa komponen fisik, diantaranya adalah sensor yang berupa konfigurasi dari empat LDR, aktuator yang berupa motor servo yang disusun untuk mewakilkan pergerakan sumbu vertikal dan horizontal, kerangka sistem dan panel suryanya sendiri. Ada empat *Light dependent resistors* yang disusun berdasarkan spesifikasi sistem dan diletakkan di bagian terpisah. Motor servo vertikal berada di samping sistem dan berguna untuk menggerakkan kerangka sistem dalam sumbu vertikal, sementara motor servo horizontal yang berada di bawah sistem berguna untuk menggerakkan sistem dalam sumbu horizontal.



Gambar 3.18 Ilustrasi Perancangan Sistem Tampak Depan

Sistem juga dirancang sirkuitnya melalui diagram *wiring* pada Gambar 3.19. Pin PWM motor servo dihubungkan dengan pin digital 4 dan 5 di Arduino, pin analog A0 sampai A3 dihubungkan ke LDR 1 sampai LDR 4 untuk mendapatkan bacaan sensor dalam nilai analog. Keempat LDR diberikan tegangan 5 V sebagai tegangan sumber dari pin 5 V di Arduino. Kedua motor servo menggunakan sumber daya eksternal dengan tegangan sebesar 6 V.



Gambar 3.19 Diagram *Wiring* Sistem

3.4 Perancangan Software

Perancangan perangkat lunak dari sistem meliputi dua hal, pemrograman sistem dalam Arduino dengan bahasa C dan alur kerja sistem dalam diagram alir.

3.4.1 Pemrograman Sistem

Sistem dimulai dengan inisialisasi pin sensor LDR dan batas motor servo seperti pada gambar dibawah ini. Pin analog A0 sampai A3 digunakan untuk LDR pertama sampai LDR keempat. Untuk aktuator, digunakan *library* motor servo dari Arduino. Kedua motor servo juga diberi batas dari 10 sampai 100 derajat untuk motor servo vertikal dan diberi batas dari 10 sampai 170 derajat untuk motor servo horizontal. Diberi juga batas toleransi sensor sebesar 50 untuk mencegah terjadinya *error* histeresis.

```
#include <Servo.h>
Servo vertikal;
Servo horizontal;
int ldr1 = A0;
int ldr2 = A1;
int ldr3 = A2;
int ldr4 = A3;
int tolerance = 50;
int interval = 3600000;
int vertikalval = 30;
int horizontalval = 30;
int vertikalLimitUp = 100;
int vertikalLimitDown = 10;
int horizontalLimitUp = 170;
int horizontalLimitDown = 10;
```

Gambar 3.20 Inisialisasi Sensor dan Batas Aktuator

Berikutnya, akan diinisialisasi parameter sistem kendali *full state observer* yang sudah dirancang. Variabel-variabel pada Gambar 3.21 didapatkan dari matrix Ac dan Bc yang dapat dicari menggunakan persamaan 35 dan 36. Setelah mendapatkan matrix ukuran 3x3 dan matrix 2x3 dari kedua persamaan, isi dari matrix akan dimasukkan ke dalam program untuk digunakan di sistem kendali.

$$Ac = A - G * C \quad (35)$$

$$Bc = [B\ G] \quad (36)$$

```
int a11v = 1;
int a12v = 0;
int a13v = 0;
int a21v = -410;
int a22v = -10310;
int a23v = -27340;
int a31v = 210;
int a32v = 5360;
int a33v = 14220;

int b11v = 0;
int b12v = 0;
int b21v = 313.6233;
int b22v = 30.3797;
int b31v = 0;
int b32v = -15.7955;

int a11h = 1;
int a12h = 0;
int a13h = 0;
int a21h = -360;
int a22h = -9060;
int a23h = -20020;
int a31h = 200;
int a32h = 5000;
int a33h = 11050;

int b11h = 0;
int b12h = 0;
int b21h = 221.0499;
int b22h = 34.6809;
int b31h = 0;
int b32h = -19.1315;
```

Gambar 3.21 Inisialisasi Parameter Sistem Kendali

Pada bagian *setup* di Gambar 3.22, dilakukan komunikasi serial dengan *baud rate* 9600 bps untuk komunikasi dengan *serial monitor*. Sesuai dengan diagram *wiring*, motor servo vertikal juga akan dihubungkan dengan pin digital 4 di Arduino dan motor servo horizontal dihubungkan dengan pin digital 5.

```
void setup() {
  Serial.begin(9600);
  vertikal.attach(4);
  horizontal.attach(5);
}
```

Gambar 3.22 *Setup* Program

Gambar 3.23 menunjukkan bagian *loop* program. Pertama, akan mengambil bacaan dari keempat LDR. Untuk *sensor fusion*, seperti yang sudah dijelaskan sebelumnya, akan diambil nilai rata-rata LDR 1 dan 2 untuk mendapatkan bagian atas sistem, LDR 1 dan 3 untuk mendapatkan bacaan bagian kiri sistem, LDR 2 dan 4 untuk mendapatkan bagian kanan sistem serta LDR 3 dan 4 untuk mendapatkan bagian bawah sistem.

```
void loop() {
  int val1 = analogRead(ldr1);
  int val2 = analogRead(ldr2);
  int val3 = analogRead(ldr3);
  int val4 = analogRead(ldr4);
  int atas = (val1 + val2) / 2;
  int kiri = (val1 + val3) / 2;
  int kanan = (val2 + val4) / 2;
  int bawah = (val3 + val4) / 2;
  int difver = atas - bawah;
  int difhor = kiri - kanan;
```

Gambar 3.23 *Loop* Program

Lalu, bagian atas dan bawah sistem akan dicari selisihnya untuk mendapatkan nilai perbedaan vertikal. Bagian kiri dan kanan sistem juga akan dikurangi untuk mendapatkan nilai perbedaan horizontal.

Berikutnya adalah perancangan sistem kendali *full state observer* di Arduino. Mengacu kepada Gambar 3.24, perbedaan bacaan sensor bagian atas dan bawah dijadikan keluaran sistem kendali untuk motor servo vertikal, sementara perbedaan bacaan sensor bagian kiri dan kanan dijadikan keluaran sistem kendali untuk motor servo horizontal. Lalu, akan dibuat tiga persamaan status yang sesuai dengan persamaan status *full state observer* untuk masing-masing motor servo.

```
//-------------------------------------------------------------------------------------------
//                                Full State Observer Begins
//-------------------------------------------------------------------------------------------
y_v = difver;
y_h = difhor;
x1_hat_v = (a11v * x1_hat_v) + (a12v * x2_hat_v) + (a13v * x3_hat_v) + (b11v * u_v) + (b12v * y_v);
x2_hat_v = (a21v * x1_hat_v) + (a22v * x2_hat_v) + (a23v * x3_hat_v) + (b21v * u_v) + (b22v * y_v);
x3_hat_v = (a31v * x1_hat_v) + (a32v * x2_hat_v) + (a33v * x3_hat_v) + (b31v * u_v) + (b32v * y_v);

x1_hat_h = (a11h * x1_hat_h) + (a12h * x2_hat_h) + (a13h * x3_hat_h) + (b11h * u_h) + (b12h * y_h);
x2_hat_h = (a21h * x1_hat_h) + (a22h * x2_hat_h) + (a23h * x3_hat_h) + (b21h * u_h) + (b22h * y_h);
x3_hat_h = (a31h * x1_hat_h) + (a32h * x2_hat_h) + (a33h * x3_hat_h) + (b31h * u_h) + (b32h * y_h);

u_v = (k1v * x1_hat_v + k2v * x2_hat_v + k3v * x3_hat_v) +  g_h * r;
u_h = (k1h * x1_hat_h + k2h * x2_hat_h + k3h * x3_hat_h) + g_v * r;

//  Serial.print("u_v: ");
//  Serial.println(u_v);
//  Serial.print("u_h: ");
//  Serial.println(u_h);

//-------------------------------------------------------------------------------------------
//                                Full State Observer Ends
//-------------------------------------------------------------------------------------------
```

Gambar 3.24 Bagian Sistem Kendali *Full State Observer*

Dibuat juga persamaan masukan sistem kendali yang sesuai dengan persamaan masukan *full state observer*. Bagian berikutnya ada pada Gambar 3.25. Jika perbedaan bacaan sensor (keluaran sistem kendali) melebihi nilai toleransi yang telah ditentukan sebelumnya, sistem akan memberikan u_v dan u_h sebagai masukan sistem kendali kepada kedua motor servo untuk menggerakkannya. Diberikan juga batas yang akan mengembalikan posisi motor servo jika terdeteksi posisi melebihi batas yang telah ditentukan.

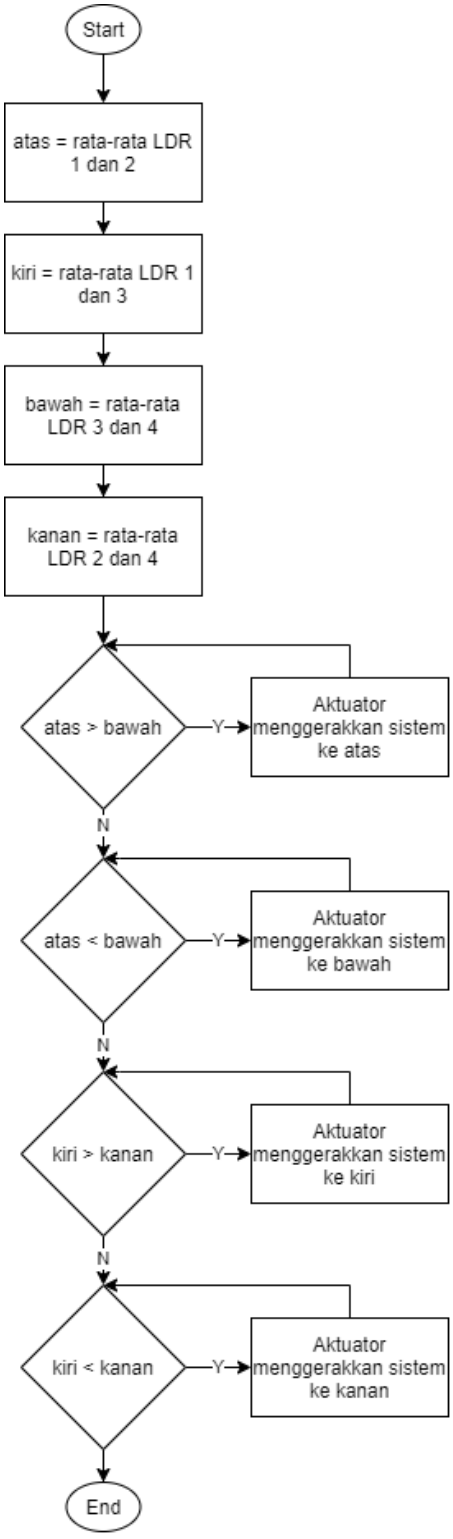
```
if (-1 * tolerance > difver || difver > tolerance)
{
  if (atas > bawah)
  {
    vertikalval = vertikalval + u_v;
    if (vertikalval > vertikalLimitUp)
    {
      vertikalval = vertikalLimitUp;
    }
  }
  else if (atas < bawah)
  {
    vertikalval = vertikalval - u_v;
    if (vertikalval < vertikalLimitDown)
    {
      vertikalval = vertikalLimitDown;
    }
  }
  vertikal.write(vertikalval);
  delay(100);
}
if (-1 * tolerance > difhor || difhor > tolerance)
{
  if (kiri > kanan)
  {
    horizontalval = horizontalval - u_h;
    if (horizontalval > horizontalLimitUp)
    {
      horizontalval = horizontalLimitUp;
    }
  }
  else if (kiri < kanan)
  {
    horizontalval = horizontalval + u_h;
    if (horizontalval < horizontalLimitDown)
    {
      horizontalval = horizontalLimitDown;
    }
  }
  horizontal.write(horizontalval);
  delay(100);
}
delay(interval);
```

Gambar 3.25 Lanjutan Bagian Sistem Kendali

3.4.2 Flow Chart Sistem

Diagram alir tertera pada Gambar 3.26 dan terdiri dari beberapa blok. Mengacu kepada empat blok *process* pertama, LDR 1 diletakkan di bagian kiri atas, LDR 2 diletakkan di bagian kanan atas, LDR 3 diletakkan di kiri bawah, dan LDR 4 diletakkan di kanan bawah. Lalu dari setiap dua LDR, diambil nilai rata-ratanya untuk mewakilkan satu bagian sistem, yaitu atas, bawah, kiri, dan kanan seperti yang sudah dibahas di perancangan sensor.

Setelah itu, pada blok *decision* sistem mulai bekerja dengan melihat dan melakukan komparasi untuk mengetahui apakah ada perbedaan bacaan sensor yang berfungsi sebagai *error* antara keempat bagian sistem tadi, jika ditemukan perbedaan bacaan, maka di blok *process* yang berada di sebelah kanan blok *decision,* sistem akan menggerakkan aktuator yang berupa dua motor servo dengan sistem kendali ke arah yang memiliki bacaan yang lebih besar sampai bacaan bagian yang dikomparasi sama nilainya.



Gambar 3.26 *Flow Chart* Sistem