BAB III
PEMBUATAN DAN SIMULASI PLATFORM IMU

3.1 Inertial Measurement Unit (IMU)

3.1.1 Hardware

A. Razor 9 DOF dan FTDI *breakout board*

Razor 9 *Degree of Freedom* (DOF) merupakan hardware IMU yang menjadi obyek pada penelitian tugas akhir, lihat gambar 3.1. Razor memiliki empat sensor yaitu satu sensor akselerometer, dua sensor gyroskop, dan satu sensor magnetometer. Seluruh sensor tersebut dihubungkan pada satu buah *board* sehingga menjadi satu kesatuan.

Selanjutnya pengaturan dan pengolahan data keempat sensor menggunakan *mikrocontroller* ATmega328. *Upload sketch* pada ATmega328 membutuhkan FTDI basic breakout board, lihat gambar 3.2, yang dihubungkan dengan FTDI connector yang telah disediakan pada Razor 9 DOF.

Gambar 3.1 IMU Razor 9 DOF [Ref. 20]

Gambar 3.2 FTDI basic breakout board [Ref. 8]

Spesifikasi keempat sensor IMU razor diambil dari *datasheet* masing-masing produk sensor. Tabel 3.1 memberikan informasi spesifikasi dasar dari masing-masing sensor meliputi jenis sensor voltase *input*, *measurement range*, temperatur kerja, dan berat sensor.

Tabel 3.1 Spesifikasi sensor

Sensor / **Spesifikasi**	**ADXL345 3-*Axis* Digital Accelerometer**	**LY530ALH ±300 ^{0}/s Analog *Yaw*-Rate Gyroscope**	**LPR530AL Dual *Axis* *Pitch* and *Roll* ±300 ^{0}/s Analog Gyroscope**	**HMC5843 3-*Axis* Digital Compass IC**
Tipe Sensor	Digital	Analog	Analog	Digital
V *Input*	2.0 V - 3.6 V	2.7 V - 3.6 V	2.7 V - 3.6 V	2.5 V - 3.3 V
Measurement Range	±2, ±4, ±8, ±16 g	±300^0	±300^0	Min :-4 gauss Max : 4 gauss
Temperature	-40 - +85 ^{0}C	-40 - +85 ^{0}C	-40 - +85 ^{0}C	-30 – 85 ^{0}C
Weight	20mg	-	-	50mg

B. XBee dan XBee *Explorer* USB

XBee seperti pada gambar 3.3 berfungsi untuk menyalurkan data secara *wireless* ke sesama XBee. Gambar 3.4 menunjukkan XBee *explorer* yang berfungsi untuk pengambilan data dan penyetingan XBee.

Gambar 3.3 XBee [Ref. 23]

Gambar 3.4 XBee *explorer* [Ref. 24]

Percobaan dilakukan dengan menjauhkan XBee IMU dari XBee PC sejauh 50 meter, dan IMU masih bisa mengirim data. Hasil ini didalam range yang diberikan oleh vendor, di mana jarak maksimal komunikasi antar XBee menggunakan *wireless* mencapai 1,5 km.

C. Arduino Fio

Arduino Fio seperti pada gambar 3.5 berfungsi untuk menjembatani antara baterai sebagai sumber daya, Razor 9 DOF sebagai sumber data, dan XBee sebagai pengirim data dari IMU ke PC.

Gambar 3.5 Arduino Fio [Ref. 1]

Arduino Fio memiliki empat keunggulan yaitu memiliki *mounting* XBee sehingga tidak membutuhkan peralatan tambahan; berukuran ringkas

atau kecil; *input* baterai 3.3 volt dengan jst connector sehingga dapat dihubungkan dengan baterai yang berukuran kecil; dan USB merupakan fitur untuk melakukan *charging* baterai. Arduino Fio membutuhkan FTDI *breakout board* untuk *upload sketch* pada *mikrocontroller*.

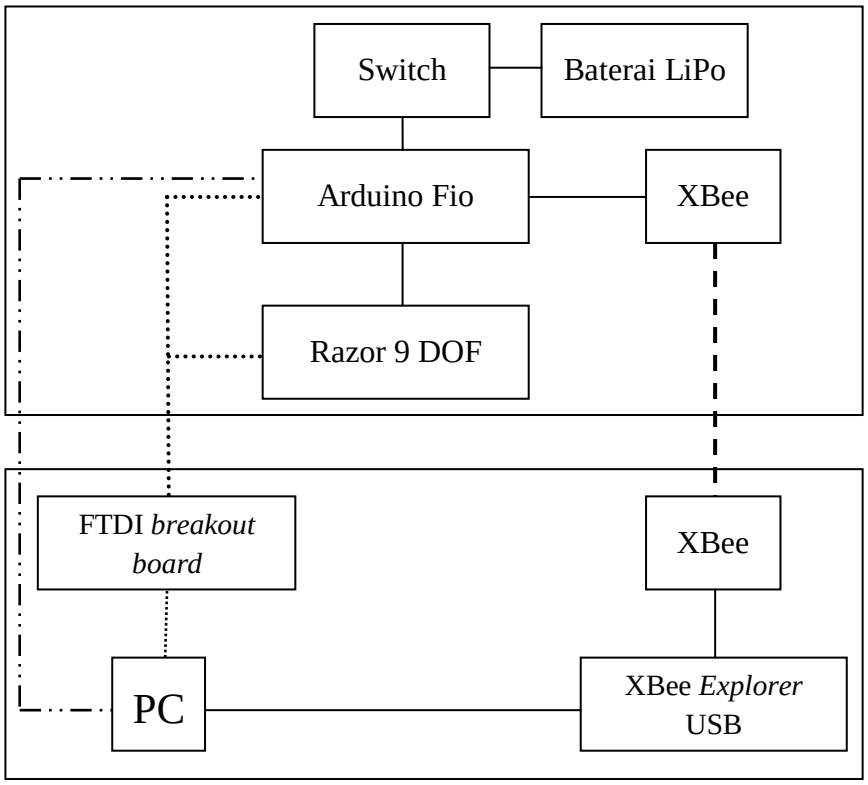
D. Baterai LiPo

Baterai menggunakan jenis *Lithium Polymer single cell* dengan voltase 3,7 volt dan 1000 mAh. Baterai menggunakan konektor *jst* seperti pada gambar 3.6. Percobaan dilakukan untuk menguji ketahanan baterai menyuplai daya IMU setelah di *fully charged*. Hasil uji menemukan baterai dapat bertahan selama kurang lebih 45 menit.

Gambar 3.6 Baterai [Ref. 16]

E. Rangkaian IMU

Skema rangkaian IMU dan hubungan antar komponen dijelaskan pada gambar 3.7. Pada gambar terlihat komponen ftdi dihubungkan hanya pada saat melakukan upload sketch dari PC. Terdapat dua xbee pada IMU dan PC yang saling berkomunikasi untuk mengirimkan data. Kabel mini-USB digunakan dari PC - FTDI *breakout board* dan PC Arduino Fio.



Gambar 3.7 Sketsa rangkaian IMU

Keterangan Garis :

——— Hubungan antar hardware

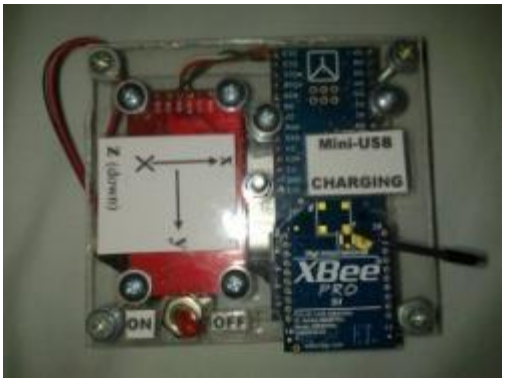
– – – Hubungan *wireless*

····· Hubungan pada saat *upload sketch*

—·· Hubungan pada saat *charging* baterai

F. *Assembly*

Assembly seperti pada gambar 3.8 menggunakan plastik bening yang berupa *mounting* ke komponen IMU. Keterangan pada IMU adalah sumbu razor, on-off *switch*, dan cara me-*recharge* baterai, yaitu dengan menghubungkan mini USB ke Arduino Fio.



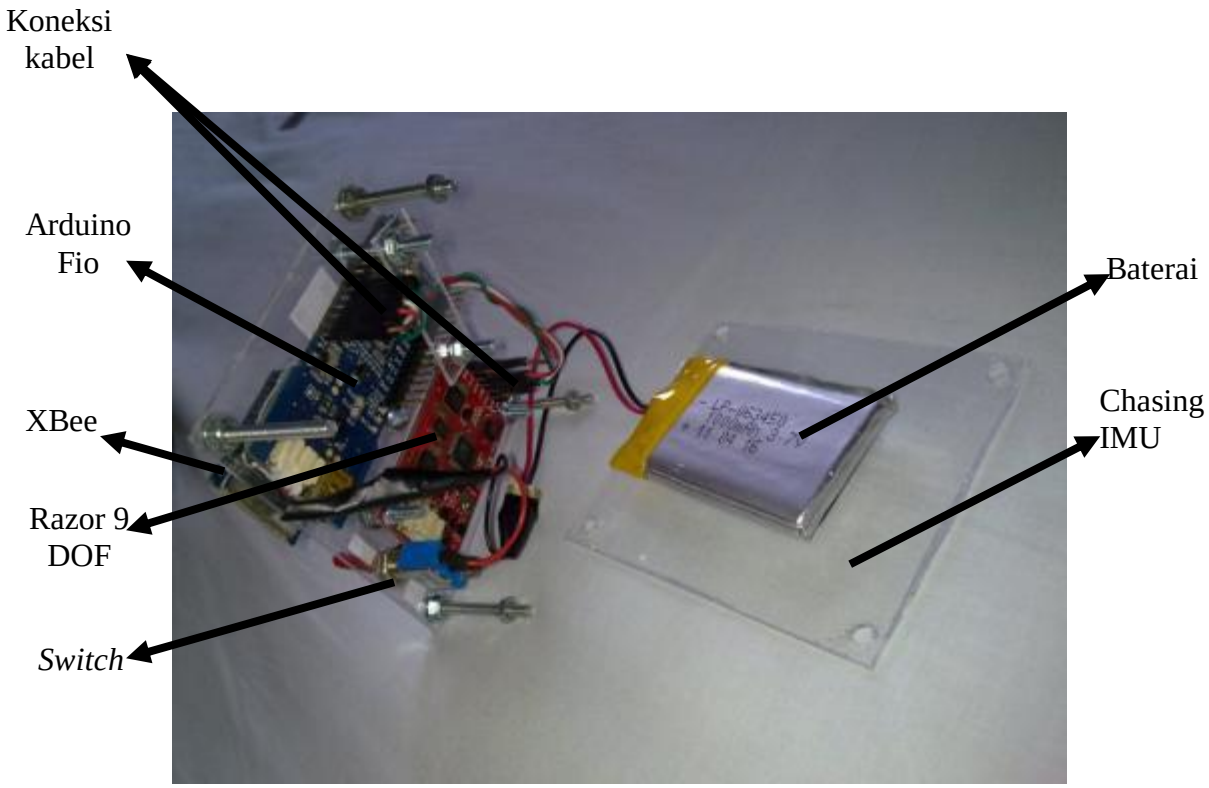
Gambar 3.8 *Assembly* IMU

Arduino Fio dan Razor 9 DOF dihubungkan pada FTDI masing-masing dengan tujuan untuk mengalirkan data serial. FTDI juga menyalurkan daya pada Razor 9 DOF melalui kabel yang dihubungkan pada Razor 9 DOF dan Arduio Fio, seperti yang dapat dilihat pada tabel 3.2.

Tabel 3.2 Koneksi Arduino Fio dan Razor 9 DOF

Arduino Fio	Razor 9DOF
3V3	3.3V
GND	GND
RX1	TX0
TX0	RX1

Pada gambar 3.9 dapat terlihat bagian-bagian IMU yang ada didalam rangkaian IMU. Baterai pada IMU ini terpasang pada *velcro* dan posisinya dapat dipindah mendekati pusat massa IMU.



Gambar 3.9 *Sub-assembly* IMU

3.1.2 Software

A. IMU Razor 9 DOF (Arduino 0022)

IMU Razor menggunakan software versi Arduino 0022. *Sketch* Razor 9 DOF dapat dilihat pada file lampiran A *Mod of SF9DOF_AHRS_1_0*. Sketch tersebut diubah beberapa bagian untuk kebutuhan tugas akhir ini. Tutorial dan *sketch* IMU dapat dibaca di lampiran A. Perubahan *sketch* digaris bawahi, *sketch* pertama yang dirubah adalah pada SF9DOF_AHRS_1_0 baris 38.

```
#define Accel_Scale(x) x*(GRAVITY/9.81)//Scaling the raw data of the accel to actual acceleration in meters for seconds square
```

Menjadi

```
#define Accel_Scale(x) x*(9.81/GRAVITY)//Scaling the raw data of the accel to actual acceleration in meters for seconds square
```

Perubahan tersebut mengubah keluaran akselerometer yang berupa gaya *G* menjadi nilai gravitasi m/s^2. Perubahan kedua adalah pada

```
#if PRINT_EULER == 1
    Serial.print("ANG:");
    Serial.print(ToDeg(roll));
    Serial.print(",");
    Serial.print(ToDeg(pitch));
    Serial.print(",");
    Serial.print(ToDeg(yaw));
    //Serial.print(",");
    //Serial.print(ToDeg(MAG_Heading));
    #endif
```

Menjadi

```
#if PRINT_EULER == 1
    Serial.print("ANG:");
    Serial.print(ToDeg(roll));
    Serial.print(",");
    Serial.print(ToDeg(pitch));
    Serial.print(",");
    Serial.print(ToDeg(yaw));
    Serial.print(",");
    Serial.print( Accel_Scale(accel_x) );
    Serial.print(",");
    Serial.print( Accel_Scale(accel_y) );
    Serial.print(",");
    Serial.print( Accel_Scale(accel_z) );
    //Serial.print(“,”);
    //Serial.print(ToDeg(MAG_Heading));
    //Serial.print( Accel_Scale(m/s^2) );
    #endif
```

Penambahan kode tersebut mengakibatkan nilai akselerasi hasil *scaling* tercetak pada serial, sehingga data dapat diakuisisi menggunakan Python.

B. XBee (X-CTU)

Pemrograman dan penghubungan antar XBee menggunakan software X-CTU. Hal utama yang perlu diperhatikan adalah pengaturan *baud rate*, di mana *baud rate* adalah sejumlah simbol yang ditransfer perdetik. Penyetelan *baud rate* bertujuan agar tiap komponen dapat saling berkomunikasi sehingga data dapat ditransfer ke PC. Proses penyetingan XBee menggunakan software XCTU dapat dilihat pada lampiran A.

C. Arduino Fio (Arduino 0022)

Sketch yang di-*upload* pada Arduino Fio melalui FTDI *breakout board* dapat dilihat pada file lampiran A *ArduFioRazor9DOF*. Urutan kerja

untuk melakukan proses *upload* juga dijelaskan pada lampiran A. Program pada Arduino Fio hanya bertujuan agar *baud rate* dari Arduino Fio adalah 57600 sehingga memiliki *baud rate* yang sama dengan Razor 9DOF dan XBee.

D. Data Aquisisi

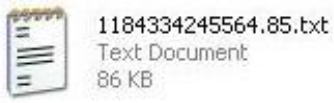
Data aquisisi atau pengambilan data dari serial dapat menggunakan beberapa software seperti Processing, Windows Visual Studio, LabVIEW, dan Python. Penggunaan Python karena banyak tersedia di internet sehingga mempersingkat waktu kerja. Hasil dari pengambilan data Python adalah file berformat *note txt*, yang kemudian diubah menjadi plot di MATLAB. Tutorial setting Python tersedia pada lampiran tergabung pada tutorial Razor 9 DOF.

Aquisisi data dilakukan secara otomatis oleh Python dengan format *file.txt* pada folder yang sama dengan *IMU_Razor9DOF.py*. Berikut ini langkah proses aquisisi data dan *screenshot*:

1. Memasang IMU pada platform dan XBee *explorer* pada pc
2. Menyalakan program X-CTU dan masukkan data pada platform, kemudian klik *IMU_Razor9DOF.py* pada pc, bentuk file dapat dilihat pada gambar 3.10.

Gambar 3.10 File IMU_Razor9DOF.py

3. Menyalakan IMU dan platform
4. Menunggu platform menyelesaikan satu siklus gerakan
5. Mematikan IMU dan platform
6. Mencari file hasil pengambilan data di folder yang sama dengan *IMU_Razor9DOF.py*, lihat gambar 3.11.



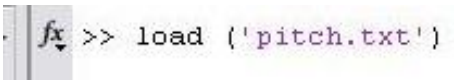
Gambar 3.11 File hasil pengambilan data

Kemudian mengubah nama file sesuai dengan percobaan yang sedang dilakukan misal *pitch*, lihat gambar 3.12. Selanjutnya menyusun data agar tiap baris memiliki format setipe.

Gambar 3.12 Pengubahan nama file

7. Memindah file yang sudah diubah namanya ke folder MATLAB, selanjutnya memasukkan *input* pada *command* MATLAB, lihat gambar 3.13, dibawah ini langkah meng*input*:
   a. Melakukan load pada MATLAB dengan *command* :

      >> load ('file.txt')



Gambar 3.13 *Command load* pada MATLAB

   b. Memerintahkan MATLAB mengambil data pada satu kolom. *Command* pada MATLAB adalah :

      >> nama=[file(:,1)]

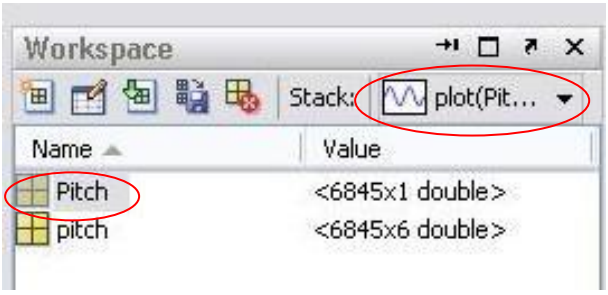
      Di mana (:,1) memiliki arti mengambil data pada kolom pertama pada *file.txt*, lihat gambar 3.14. Platform memiliki kemampuan untuk melakukan empat tes, di mana keempat tes tersebut berhubungan dengan kolom pertama yang berupa sudut

pitch, kolom ke dua adalah *roll*, kolom ke tiga adalah *yaw*. Kolom ke enam adalah akselerasi sumbu Z.

```
>> Pitch=[pitch(:,1)]
```

Gambar 3.14 *Command* pengambilan kolom data pada MATLAB

c. Mengklik data dan melakukan *plotting* dengan mengklik perintah *plot* pada *workspace*, seperti yang dapat dilihat pada gambar 3.15.



Gambar 3.15 *Workspace* MATLAB

3.2 Platform

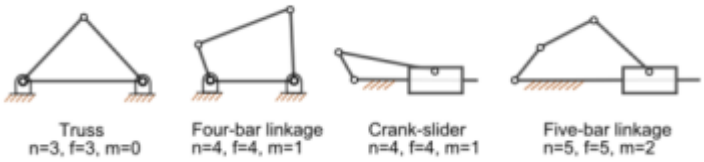
3.2.1 Hardware

A. Platform Inertial Measurement Unit (IMU)

Pada Tugas akhir ini platform IMU dirancang dengan referensi dari sistem yang dikembangkan di Ohio State University. Pada sistem tersebut platform kalibrasi IMU berdasarkan *Carpal Wrist Device*, lihat gambar 3.16. Gerakan *pitch* dan *roll* diperoleh dari gerakan yang terkoordinasi dari penggerak linier. Tiga derajat kebebasan hasil dari gerakan linier dapat menghasilkan gerakan rotasi pada platform.

Gambar 3.16 Virginia *tech carpal wrist* [Ref. 11,h.4]

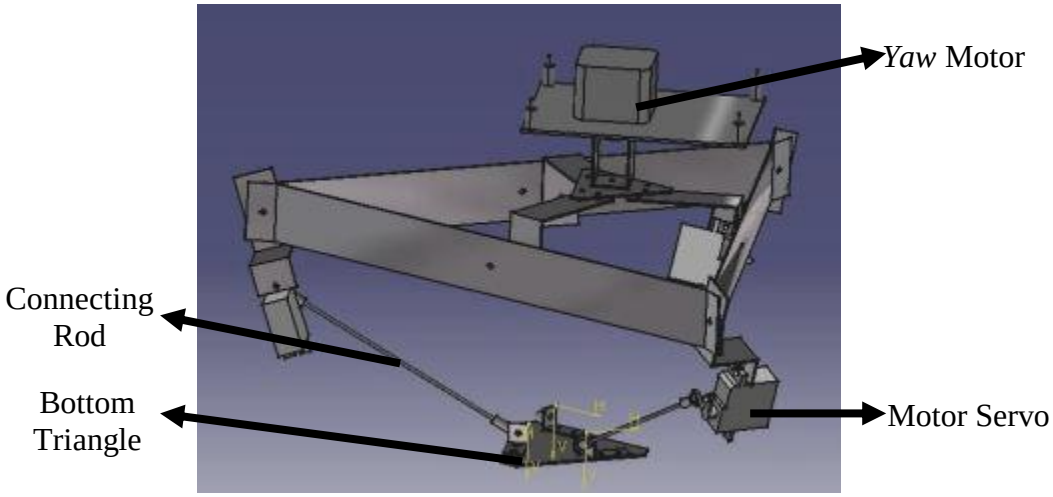
Gerakan *yaw* pada platform tidak terbatas dan diperoleh dari meja putar yang memutari seluruh *carpal wrist* dan *assembly*. Pada tugas akhir ini gerakan linier digantikan oleh servo motor yang menghasilkan gerakan rotasi. Melalui pandangan 2 dimensi, gerakan yang terjadi akan membentuk gerakan dasar *linkage*. Gambar 3.17 adalah contoh beberapa jenis *linkage* yang ada.



Gambar 3.17 Contoh *linkage* [Ref. 13]

B. Desain Platform IMU

Sumber gerak pada penelitian ini menggunakan tiga buah motor servo, yang berfungsi sebagai penggerak segitiga yang dihubungkan dengan IMU. Melalui ketiga servo tersebut didapatkan gerak *pitch*, *roll*, dan sumbu Z. Satu buah servo lagi digunakan untuk mendapatkan gerakan *yaw*. Hasil desain dapat dilihat pada gambar 3.18.



Gambar 3.18 Desain platform IMU

C. Penentuan Spesifikasi Estimasi Platform

Penentuan spesifikasi didasarkan pada kemampuan *hardware* yang didapatkan dan digunakan pada platform. Desain awal berguna untuk menentukan kemungkinan gerakan yang dapat terjadi dan digunakan sebagai estimasi spesifikasi awal. Untuk mendapatkan estimasi *pitch* dan *roll*, desain disimulasikan pada SimMechanic dengan input yang sederhana. Berikut ini estimasi spesifikasi platform:

- Massa IMU : 500gram
- *Pitch* dan *roll* : $\pm 10^0$
- *Yaw* : $\pm 90^0$

D. Penjabaran Pembuatan Platform IMU

1. Desain awal.

Pembuatan desain platform IMU dengan sumber gerak menggunakan motor servo. Pemilihan sumber gerak motor servo karena sumber ini mudah didapatkan. Komponen lain dalam platform meliputi alumunium, akrilik, mur dan baut, di mana komponen tersebut mudah didapatkan di pasaran. Melalui desain yang sederhana memungkinkan untuk menguji dan menganalisa platform dengan mudah. Tujuan dari desain platform yang diharapkan adalah platform mampu melakukan unjuk kerja yang bisa digunakan untuk mengkalibrasi IMU.

2. Pemilihan Bahan

Komponen platform yang digunakan menggunakan bahan-bahan yang mudah ditemukan di pasaran meliputi:

a. Alumunium plat

Alumunium plat, lihat pada gambar 3.19, yang digunakan memiliki ukuran ketebalan 0,2 milimeter. Alasan menggunakan bahan ini karena ringan sehingga motor yang digunakan tidak memerlukan torsi yang besar.

b. Mur dan Baut

Pemilihan bahan ini karena mur dan baut, lihat pada gambar 3.19, merupakan jenis sabungan yang mudah dipasang dan dilepas sehingga mudah dimodifikasi, dan apabila terjadi kesalahan pemasangan mudah untuk memperbaiki.

c. Akrilik

Akrilik, lihat pada gambar 3.19, yang digunakan memiliki ketebalan 5 milimeter. Bahan ini digunakan karena mudah didapat di pasaran.

Gambar 3.19 Bahan platform

d. *Ball joint*

Penggunaan *ball joint* karena bahan ini dapat menyediakan koneksi yang mampu bergerak secara tiga rotasi derajat kebebasan , pada gambar 3.20 *ball joint* yang belum terpasang.

Gambar 3.20 *Ball joint*

e. Motor servo

Pada gambar 3.21 dapat dilihat motor servo yang merupakan sebuah motor dengan sistem closed feedback di mana posisi dari motor akan diinformasikan kembali ke rangkaian kontrol yang ada di

dalam motor servo. Motor ini terdiri dari sebuah motor, serangkaian gear, potensiometer dan rangkaian kontrol. Penggunaan motor servo dikarenakan mudah dikontrol.

Gambar 3.21 Servo

f. Arduino Uno dan motor shield

Arduino Uno dan motor shield, dapat dilihat pada gambar 3.22 berfungsi sebagai *microcontroller* untuk mengatur gerakan servo yang terprogram melalui PC. Pengaturan menggunakan *pin pulse width modulation* (pwm) pada nomor 5, 6, 9, dan 10. Sumber daya menggunakan 12 volt dari *power supply*.

Gambar 3.22 Arduino Uno dan motor shield

g. *Power Supply*

Power Supply atau sumber daya, lihat gambar 3.23 adalah sumber listrik AC yang dikonversi menjadi listrik DC dengan bantuan modifikasi PSU PC desktop. PSU PC desktop yang digunakan merupakan PSU yang umum sehingga mudah didapat. Kabel pada PSU memiliki standar yaitu kabel hijau *switch*, kabel kuning 12v, kabel merah 5v, kabel oranye 3,3v, kabel biru -12v, kabel abu-abu -5v, kabel coklat *sense*, dan kabel hitam *ground*. Kabel yang digunakan adalah kabel hijau, kabel kuning, kabel merah, dan kabel hitam.

Gambar 3.23 *Power supply*

3. Pengolahan Bahan

Dalam penelitian ini, pengolahan bahan melalui dua langkah yang mencakup pengukuran menggunakan alat ukur dan pengolahan bahan itu sendiri. Pengolahan bahan yang pertama adalah pengukuran menggunakan alat ukur. Dibawah ini alat ukur yang digunakan:

a. *Waterpass*

Waterpass atau alat pengukur kedataran permukaan berfungsi untuk mengukur datar atau tidaknya permukaan tempat platform diletakkan. Alat ini juga bertujuan untuk meminimalisir kesalahan peletakkan sambungan baut pada hasil *assembly*. Memodifikasi *waterpass* dapat dilihat gambar 3.24, dengan menambahkan skala 0

hingga 3 pada kanan dan kiri gemlembung, di mana 1 mendekati penyimpangan 1^0 dari kedataran ($1 \sim 1^0$), 2 mendekati penyimpangan $1{,}5^0$ ($2 \sim 1{,}5^0$), dan 3 mendekati penyimpangan 2^0 ($3 \sim 2^0$) .

Gambar 3.24 Modifikasi *waterpass*

b. *Vernier caliper*

Vernier caliper, lihat gambar 3.25, digunakan untuk menggantikan mistar ukur pada pengukuran yang membutuhkan ketelitian tinggi dan geometri yang sulit diukur dengan mistar ukur seperti diameter lingkaran.

c. Mistar ukur

Mistar, lihat gambar 3.25, adalah alat ukur linier. Pada perancangan ini mistar ukur paling sering digunakan karena mudah digunakan. Mistar ukur yang digunakan memiliki ketelitian 0,5 mm.

Gambar 3.25 Alat ukur

Pengolahan kedua adalah Pengolahan bahan itu sendiri yang menggunakan perkakas sederhana seperti bor tangan, tang, dan obeng, perkakas dapat dilihat pada gambar 3.26. Salah satu pembuatan *part* diberikan contoh sebagai berikut, yaitu *part* yang mengandung seluruh proses yang mencakup pengerjaan *part* pemegang servo (*servo holder*). Meliputi pengukuran (*measurement*), pengeboran (*drilling*) seperti terlihat pada gambar 3.27, dan penekukan (*bending*) seperti pada gambar 3.28.

Gambar 3.26 Bor tangan, tang, dan obeng

Gambar 3.27*Drilling*

Gambar 3.28 *Bending*

4. Pemilihan software

Software Arduino-1.0 yang merupakan software terbaru digunakan untuk mengolah *sketch* Arduino. Bentuk dasar sketch yang dipakai pada gerakan servo diambil dari contoh gerakan servo sweep. Gerakan ini memiliki getaran yang dapat dieliminir dengan penahan sederhana. Tabel 3.3 menyajikan keseluruhan software beserta kegunaannya dalam pengujian platform IMU.

Tabel 3.3 Sofware-software dalam pengujian platform

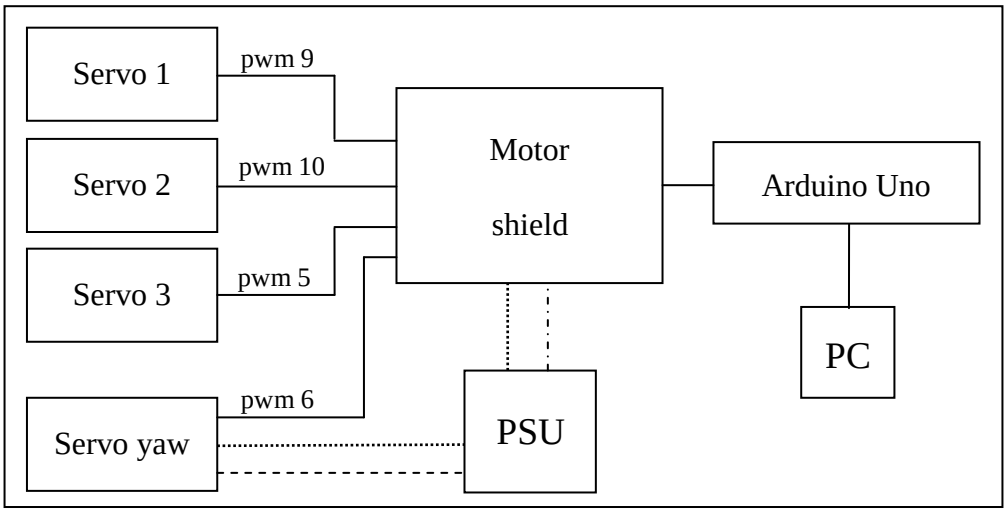
No	Software	Kegunaan
1.	Arduino-0022	Program Razor 9DOF dan Arduino Fio
2.	Arduino 1.0	Program platform
3.	MATLAB Simulink	Pemisahan data IMU dan simulasi
4.	Python	Penyimpan data hasil serial

5. Pengolahan software

   Pengolahan Software akan dijelaskan pada subbab 3.3.

6. Pemasangan atau *Assembly*

   Langkah selanjutnya adalah memasangkan semua komponen platform menjadi satu kesatuan, baik komponen hardware maupun hasil pengolahan software. Hubungan PC - Arduino - motor servo dijelaskan melalui gambar 3.29.



Gambar 3.29 Sketsa rangkaian platform

Keterangan Garis :

——— Hubungan antar hardware

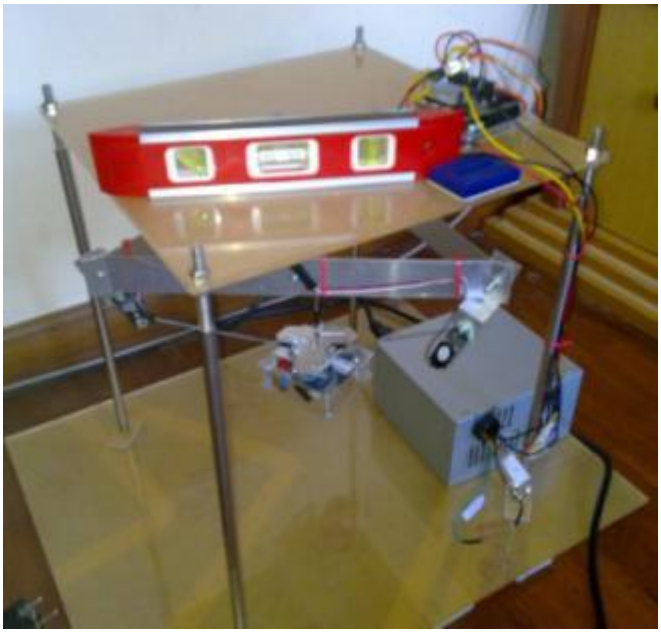
– – – Kabel daya 5v

— · – Kabel daya 12v

····· Kabel ground

7. Pemeriksaan ukuran, geometri, dan software

   Pemeriksaan bertujuan untuk meminimalkan eror-eror yang mungkin terjadi sekecil mungkin. Pemeriksaan dilakukan dengan mengecek kedataran segitiga motor servo yang telah dipasang atau *assembly* dengan menggunakan *waterpass*. Pada gambar 3.30 menunjukkan pengetesan kedataran pada tahap akhir platform. Langkah ini bersifat iterasi dengan langkah b, e, dan f untuk meminimalkan segala eror yang mungkin terjadi.



Gambar 3.30 Pengetesan kedataran dengan *waterpass*

8. Pengetesan dan pengambilan data

   Apabila langkah pertama hingga ketujuh telah dilakukan, selanjutnya dilakukan pengetesan dengan IMU untuk memperoleh data pengukuran dan unjuk kerja platform.

3.2.2 Sofware

A. *Hardware Testing*

Pengetesan sederhana pada servo mendapatkan beberapa karakteristik servo, seperti nilai maksimum dan minimum derajat yang aman untuk digunakan pada platform yaitu 10 derajat sampai 170. Pada contoh *sketch* terdapat *sketch sweep* yang memerintahkan servo untuk bergerak satu derajat, sehingga nilai diambil dari servo terkecil yang dapat dimanfaatkan yaitu satu derajat.

B. *Sketch* servo

Berdasarkan beberapa pengetesan diatas maka dibuat program dengan nilai minimum sudut servo 10^o dan nilai maksimum 170^o . *Input* adalah jenis *ramp* dari contoh program servo *sweep*. *Sketch* yang di-*upload* pada Arduino Uno terlampir pada file lampiran B. Potongan *sketch* di bawah ini merupakan penyesuaian servo dengan pin Arduino.

```
void setup()
{
  servo1.attach(9);   //pitch
  servo2.attach(10);  //roll_1
  servo3.attach(5);   //roll_2
  servo4.attach(6);   //yaw
}
```

Potongan *sketch* di bawah ini merupakan setting nol dan waktu inisialisasi pada semua servo.

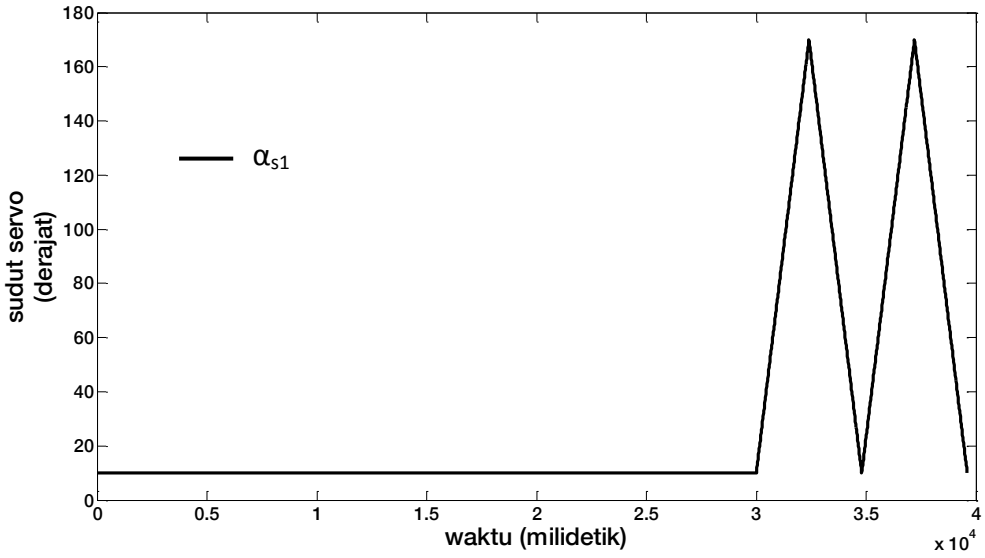
```
servo1.write(10);
servo2.write(10);
servo3.write(10);
servo4.write(10);
delay(30000);
```

Potongan *sketch* di bawah ini merupakan perintah gerak pada servo.

```
for(pos = 10; pos < 170; pos += 1)  // goes from 10 degrees to 170 degrees
{                                   // in steps of 1 degree
  servo1.write(pos);                // tell servo to go to position in variable 'pos'
  delay(15);                        // waits 15ms for the servo to reach the position
}
```

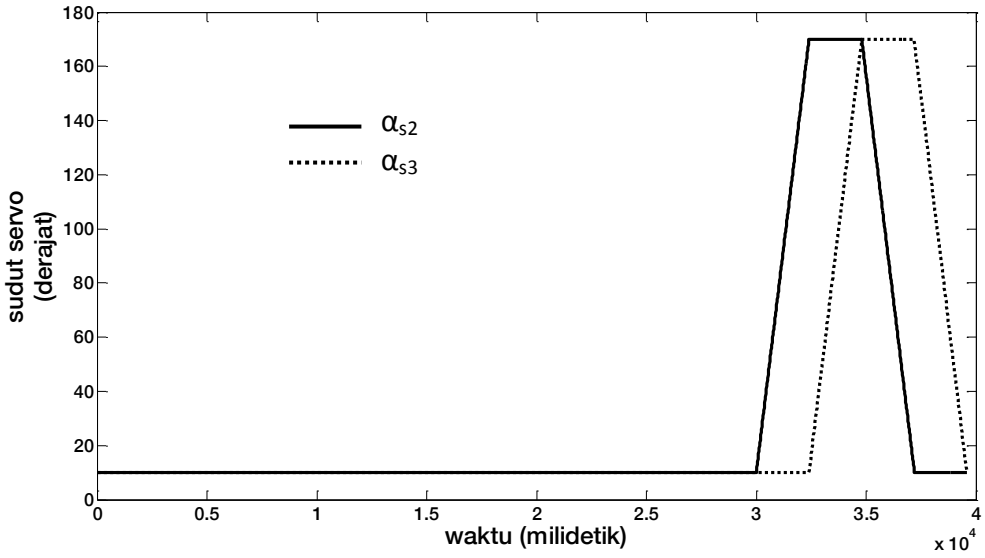
Input pitch diberikan pada gambar 3.31. Grafik dibuat ideal dari *sketch* servo *pitch*. Jenis *input* ramp adalah sudut servo bergerak secara *step* dengan perubahan sudut 1 derajat dengan jarak antar perubahan tersebut adalah 15

milidetik. Pada 30 detik pertama adalah waktu untuk menyalakan IMU dan IMU melakukan inisialisasi awal.



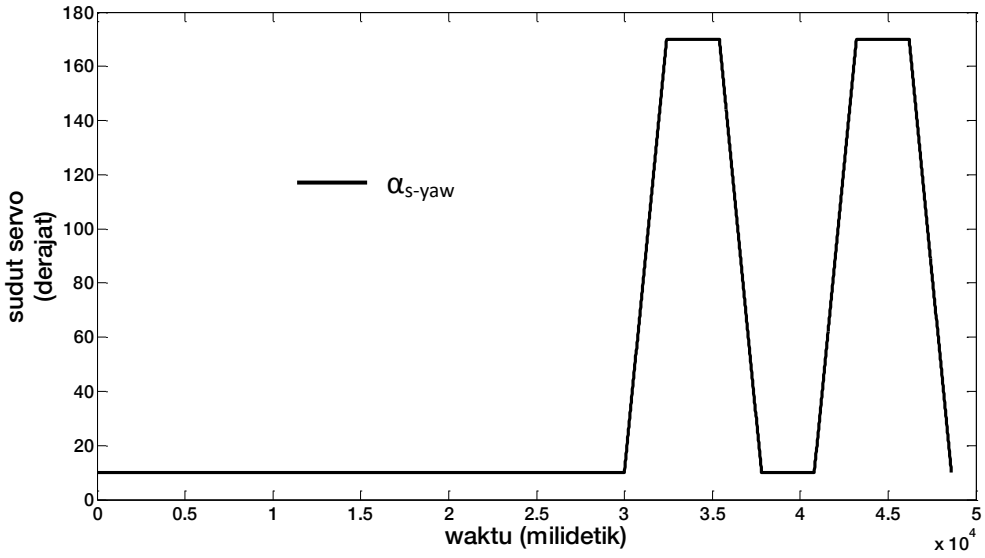
Gambar 3.31 Grafik *input pitch*

Pada gerakan *roll* menggunakan 2 buah servo yang gerakannya saling berurutan, *input roll* diberikan pada gambar 3.32. Servo yang digunakan adalah servo nomor 2 dan nomor 3, dimana servo nomor 3 adalah servo yang melakukan gerakan setelah servo 2. Sudut dan waktu delay sama seperti gerakan servo *pitch*



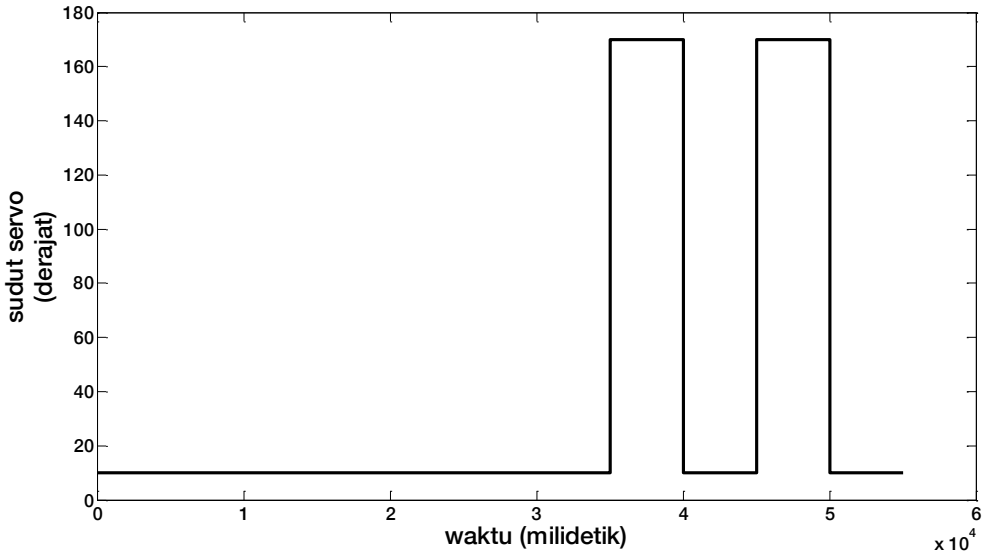
Gambar 3.32 Grafik *input roll*

Gerakan *yaw* merupakan gerakan sistem segitiga secara keseluruhan. Sumber gerak *yaw* adalah servo *yaw*. *Input* pada servo *yaw* diberikan pada gambar 3.33 dengan nilai yang sama seperti servo *pitch* dan *roll*. Waktu delay diberikan pada puncah gerakan servo untuk pengamatan nilai dan pengamatan sistem peredaman sistem segitiga.



Gambar 3.33 Grafik *input yaw*

Pada gerakan sumbu Z. Servo 1, servo 2, dan servo 3 digerakkan secara spontan ketiga-tiganya dengan gerakan berupa step seperti pada gambar 3.34. Gerakan step bertujuan agar akselerasi yang terjadi maksimal.

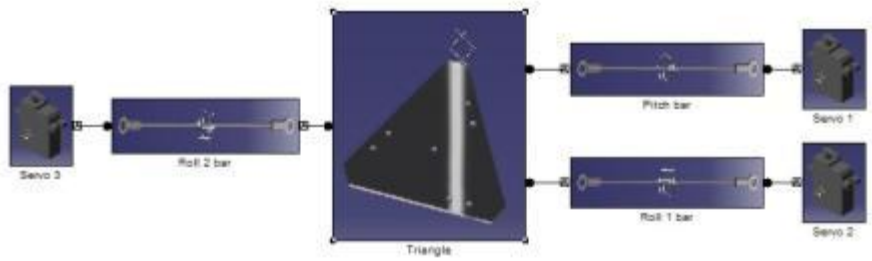


Gambar 3.34 Grafik *input* Sumbu Z

3.3 Simulink

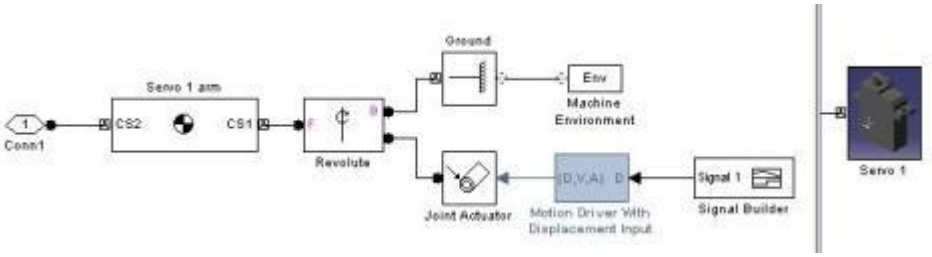
3.3.1 SimMechanic

Gerakan dihasilkan oleh tiga motor servo dan cukup sulit dianalisa dengan kinematika dan dinamika sederhana sehingga digunakan alat bantu SimMechanic untuk mensimulasikan gerakan yang terjadi. Berikut ini hasil akhir SimMechanic dari platform, seperti ditunjukkan gambar 3.35. Hasil akhir dibuat dengan *subsystem* yang mewakili kerja masing-masing *part. Part* yang diwakilkan adalah motor servo, batang penghubung, dan segitiga pemegang IMU.



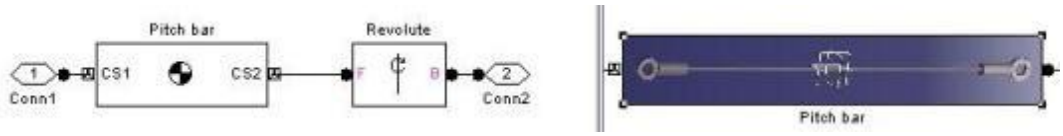
Gambar 3.35 Hasil akhir SimMechanic

Subsystem yang pertama adalah servo motor, sistem yang ada di dalam servo dapat dilihat di gambar 3.36. Pada *subsystem* servo digunakan sumber gerak *Joint Actuator* dengan aktuasi berupa gerakan, *input* dibantu dengan blok *Motion Driver.* Blok tersebut berfungsi membantu sistem SimMechanic untuk menghasilkan kebutuhan *Joint Actuator* pada gerakan yang membutuhkan kecepatan dan akselerasi. Isi blok pada *Motion Driver* diberikan pada lampiran D.



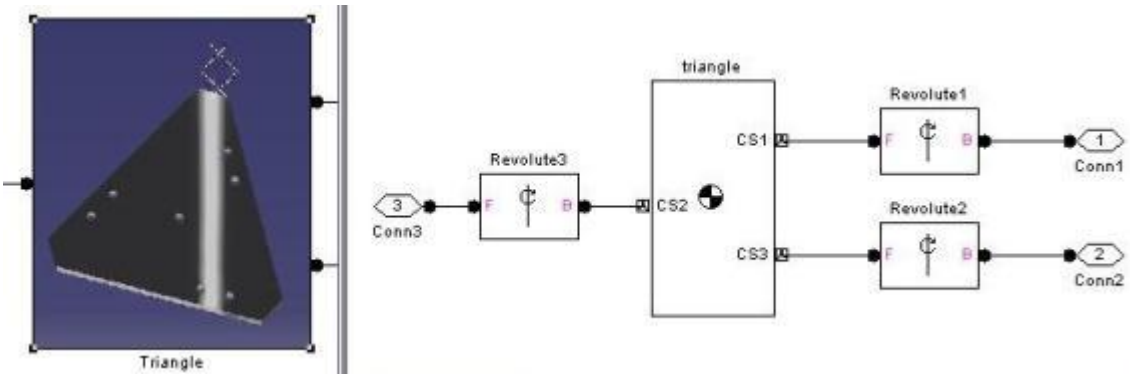
Gambar 3.36 *Subsystem* servo

Subsystem batang penghubung memiliki sistem seperti pada gambar 3.37. *Subsystem* ini berjumlah 3 buah yang semuanya terhubung dengan segitiga dan servo.



Gambar 3.37 *Subsystem* batang penghubung

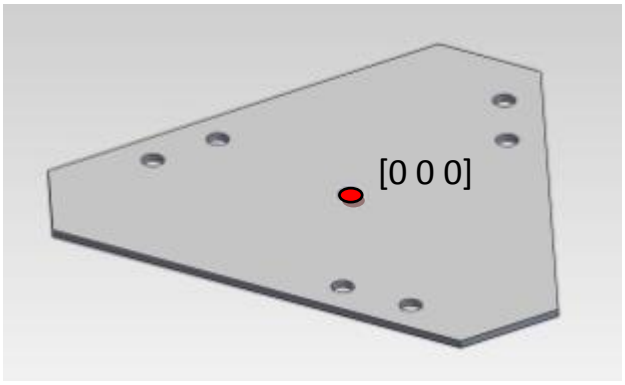
Subsystem segitiga memiliki sistem yang dapat dilihat pada gambar 3.38. Didalam *subsystem* terdapat sensor yang terpasang pada *body* segitiga untuk medapatkan pembacaan data hasil simulasi.



Gambar 3.38 *Subsystem* segitiga

A. Koordinat awal

Pada SimMechanic ini titik [0 0 0] berada di salah satu sumbu servo, seperti yang dapat di gambar 3.39.



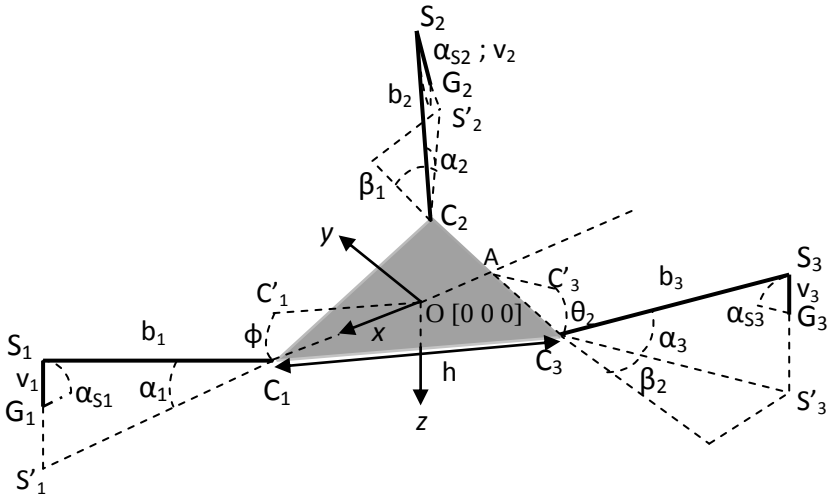
Gambar 3.39 Koordinat [0 0 0]

B. Data SimMechanic

Data, Blok spesifik, dan asumsi yang digunakan pada SimMechanic Platform adalah sebagai berikut:

1. Geometri

Geometri segitiga dilihat dari pandangan atas dan depan pada gambar 3.40. Pengolahan data [x y z] merupakan analisa geometri sederhana untuk dimasukkan ke dalam Koordinat Sistem *Body*. Tabel rincian data *input* dapat dilihat pada lampiran D.



Gambar 3.40 Geometri platform

Keterangan gambar 3.38 :

A : proyeksi titik O pada garis C_2C_3

C_1 : titik segitiga pada batang *pitch*

C_2 : titik segitiga pada batang *roll* 1

C_3 : titik segitiga pada batang *roll* 2

$C'_i = C'_1 = C'_2 = C'_3$: perubahan titik segitiga i

$G_i = G_1 = G_2 = G_3$: titik ground i

O : titik origin *world* [0 0 0]

$S_i = S_1 = S_2 = S_3$: titik ujung servo i

$S'_i = S'_1 = S'_2 = S'_3$: proyeksi b_i pada bidang horizontal xy

b_1 : batang *pitch*

b_2 : batang *roll* 1

b_3 : batang *roll* 2

h : $C_1C_2 = C_2C_3 = C_3C_1$

$v_i = v_1 = v_2 = v_3$: batang servo i

x : sumbu x *world*

y : sumbu y *world*

z : sumbu z *world*

$\alpha_i = \alpha_1 = \alpha_2 = \alpha_3$: sudut batang (b_i) dengan bidang horizontal

$\alpha_{si} = \alpha_{s1} = \alpha_{s2} = \alpha_{s3}$: sudut servo i

β_1 : sudut antara garis $C_3S'_3$ dengan perpanjangan garis AC_3

β_2 : sudut antara garis $C_2S'_2$ dengan perpanjangan garis AC_2

ϕ : sudut *pitch* antara OC_1 dengan bidang horizontal xy *world*

θ_1 : sudut *roll* 1 antara AC'_2 dengan bidang horizontal xy *world*

θ_2 : sudut *roll* 2 antara AC'_3 dengan bidang horizontal xy *world*

Ukuran geometri gambar 3.38:

b_i = 14.4 cm

h = C_1C_2 = C_2C_3 = C_3C_1 =4 cm

v_i = 1.5 cm

$\alpha_i = 40^o$

$\beta_1 = \beta_2 = 30^o$

2. Inersia dan Massa

   Inersia pada uji coba platform ini diberi nilai sebagai berikut:

   a) Segitiga

      Massa : 90 gram

      Inersia : [356.73 0 0;0 271.11 0;0 0 664.37] g.cm^2

   b) Batang penghubung dan servo (asumsi sebagai batang silinder)

      - Servo

        Massa : 3 gram

        Inersia : [0.5625 0 0;0 0.5625 0;0 0 0]g.cm^2

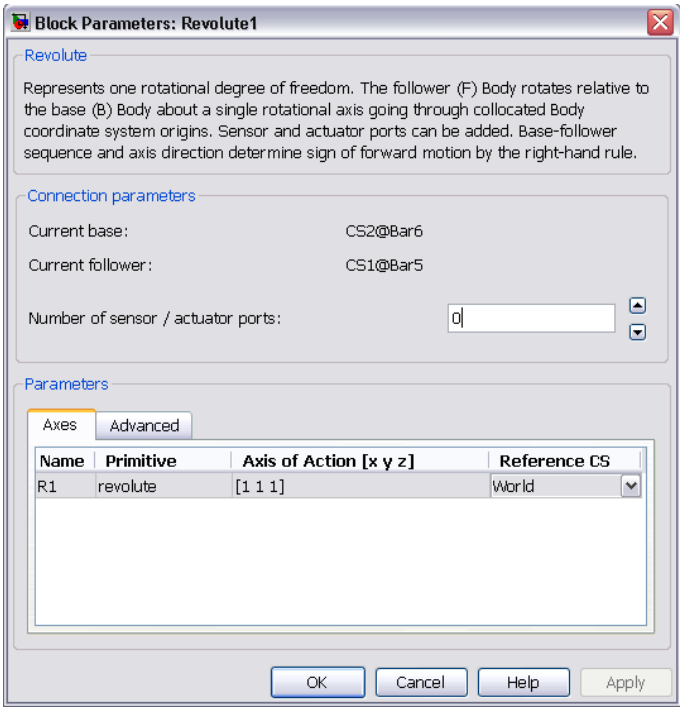
      - Batang Penghubung

        Massa : 4 gram

        Inersia : [69.12 0 0;0 69.12 0;0 0 0] g.cm^2

3. *Joint*

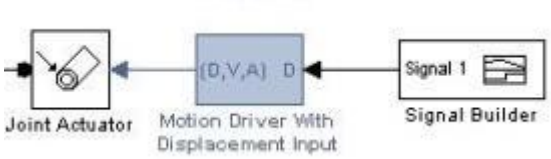
   Joint yang digunakan adalah jenis *revolute*. *Joint revolute* merupakan *joint* yang derajat kebebasannya adalah rotasi. Derajat kebebasan pada GUI *revolut* diatur dengan nilai 0 adalah tidak dapat berputar dan 1 adalah dapat berputar. Sebagai contoh Nilai *revolute* adalah [1 0 0] maka *joint revolute* tersebut dapat berputar pada sumbu X. Pada project ini menggunakan ball joint sehingga *revolute* dapat bergerak bebas semua sumbu. GUI ditunjukkan dengan gambar 3.41.



Gambar 3.41 *GUI revolute*

4. *Actuator*

Actuator berguna untuk memberikan *input* gaya atau gerakan pada sistem Simmechanic. Penggunaan *Joint Actuator* dengan *input* berupa *motion* dikarenakan gerakan servo merupakan gerakan dengan hasil berupa sudut, sehingga simulasi menggunakan pendekatan seperti pada servo. Penggunaan Joint Actuator dapat dilihat pada gambar 3.42.

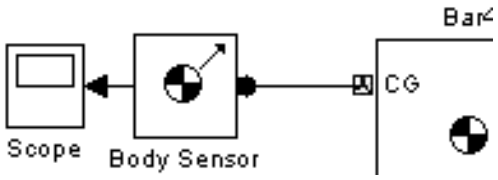


Gambar 3.42 *Joint actuator* [Ref. 19]

5. Sensor

Sensor berguna untuk dihubungkan dengan blok simulink "*scope*' yang berguna untuk mencatat semua gerakan yang terjadi pada

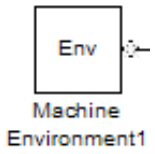
body/joint, sehingga data yang dihasilkan dapat dianalisa. *Body sensor* dapat dihubungkan dengan titik koordinat atau pada *center gravity* seperti pada gambar 3.43.



Gambar 3.43 *Body* sensor [Ref. 19]

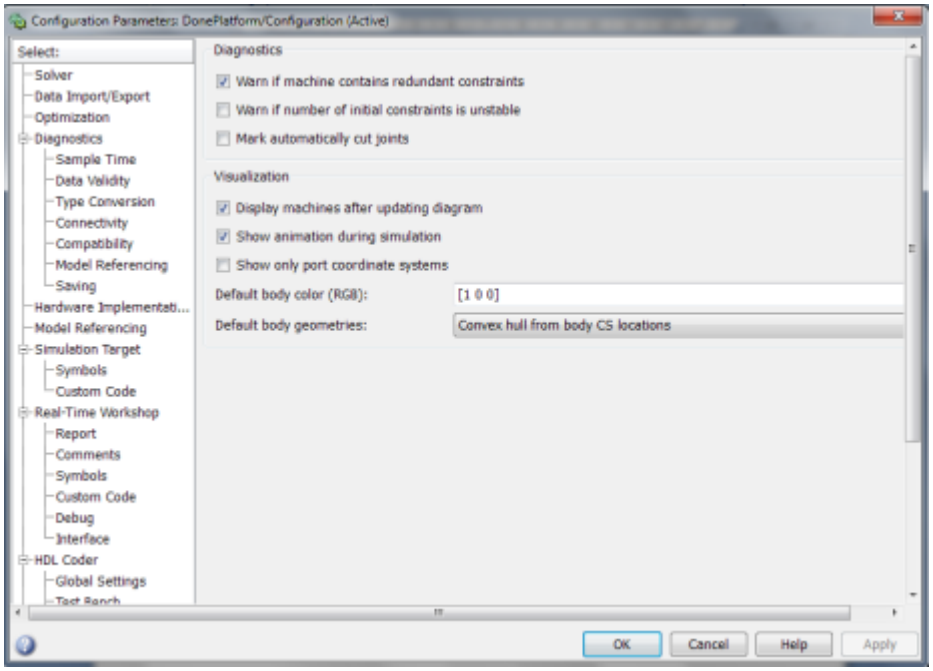
6. Visualisasi

Untuk menunjukkan visualisasi menggunakan blok "*Machine Environment*" seperti pada gambar 3.44, dimana blok tersebut terhubung dengan *ground*.



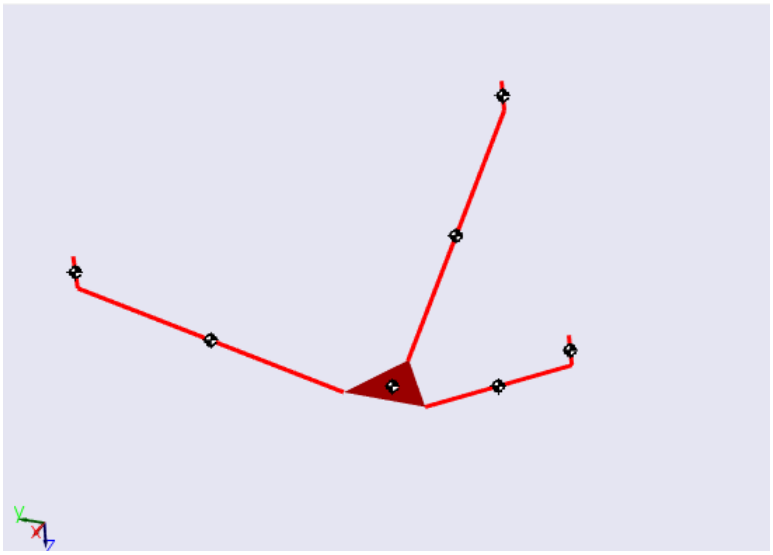
Gambar 3.44 *Machine environment* [Ref. 19]

Setelah masuk kedalam *option* pada Blok "*Machine Environment*", memilih "*Visualization*" dan mengklik pilihan "*Configuration Parameters*", maka akan muncul GUI untuk mengkonfigurasi parameter. Klik pada bagian *option* "*Display machines after updating diagram*" dan "*Show animation during simulation*" seperti pada gambar 3.45.



Gambar 3.45 *Configuration parameters*

Jika sudah maka dengan memulai simulasi otomatis simulink akan menghasilkan body dari SimMechanic seperti pada gambar 3.46.



Gambar 3.46 Visualisasi SimMechanic

7. Asumsi

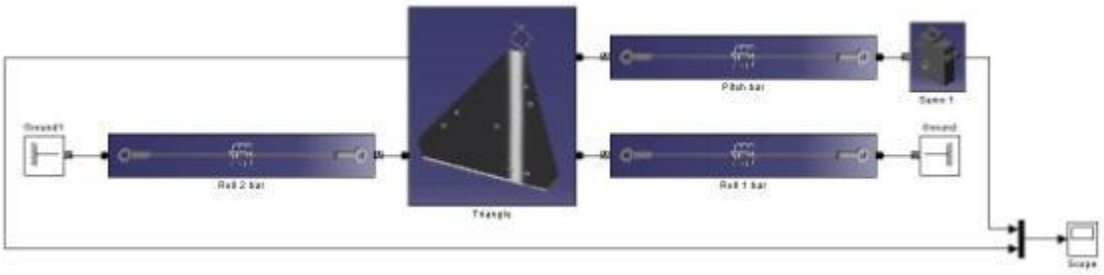
Asumsi yang digunakan pada simulasi platform ini adalah joint yang tidak bermassa dan tidak memiliki friksi. Beban dan Inersia segitiga diasumsikan adalah IMU, asumsi digunakan dimana beban dan nilai inersia IMU relatif dominan dibandingkan segitiga pemegang IMU.

C. Hasil Simulasi SimMechanic

Terdapat dua gerakan yang akan dianalisa dengan SimMechanic yaitu *pitch* dan *roll*.

1. *Pitch*

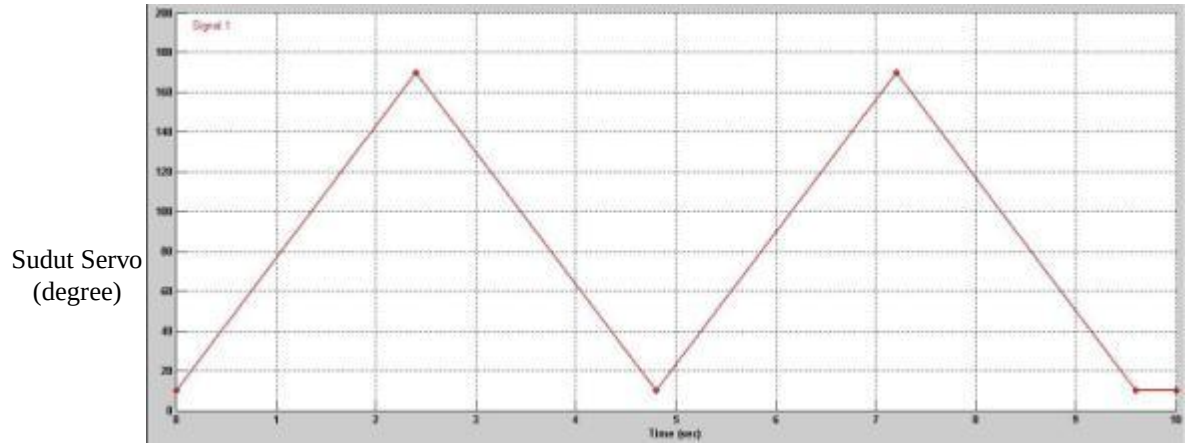
Pada SimMechanic *pitch*, model ditunjukkan seperti pada gambar 3.47. Pengurangan komponen digunakan untuk mempermudah perhitungan dengan software SimMechanic. Pengurangan komponen juga mengurangi kemungkinan eror yang mungkin terjadi pada sistem.



Gambar 3.47 Model SimMechanic *pitch*

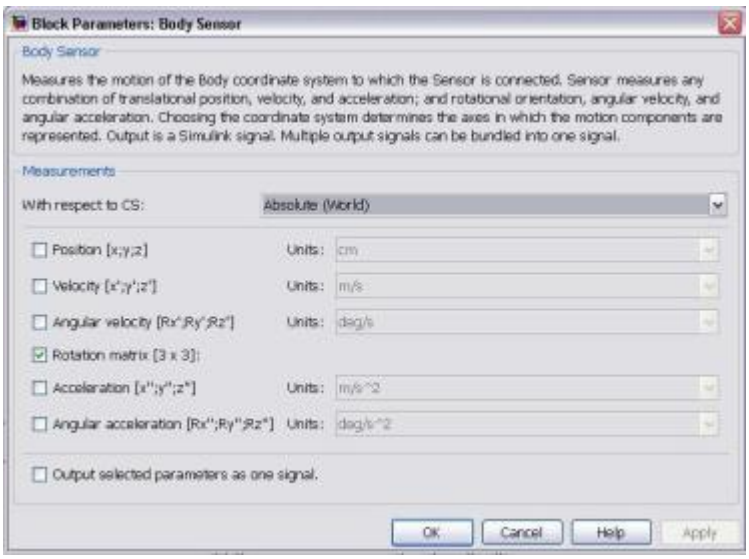
Perubahan lain dari bentuk dasar model SimMechanic dasar adalah nilai matriks revolute yang dirubah dari matriks [1 1 1] menjadi [0 1 0]. Perubahan tersebut bertujuan untuk mengarahkan servo pada 1 gerak yaitu sumbu Y.

Input berupa gerakan pada *joint* dengan nilai *input* yang sama seperti pada *sketch* servo. *Input* berupa grafik *input* ideal pada gambar 3.48



Gambar 3.48 *Input pitch*

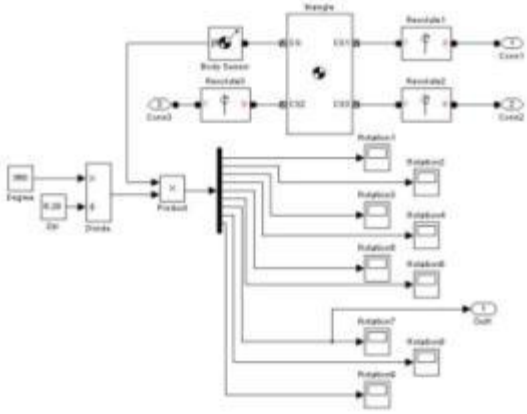
Untuk mendapatkan data, model diberi *body sensor* pada CG blok triangle seperti pada gambar 3.49 dan di*setting* agar mengeluarkan data rotasi segitiga IMU yang berupa matrix dan bernilai radian. Orientasi dari pengambilan sudut oleh body sensor adalah *Absolute World*, pengambilan asumsi tersebut menyesuaikan dengan kondisi sensor IMU.



Gambar 3.49 *Body sensor*

Pengolahan body sensor pada gambar 3.50 adalah pengonversi sudut yang mengeluarkan *output* derajat. Hasil body sensor merupakan matrix

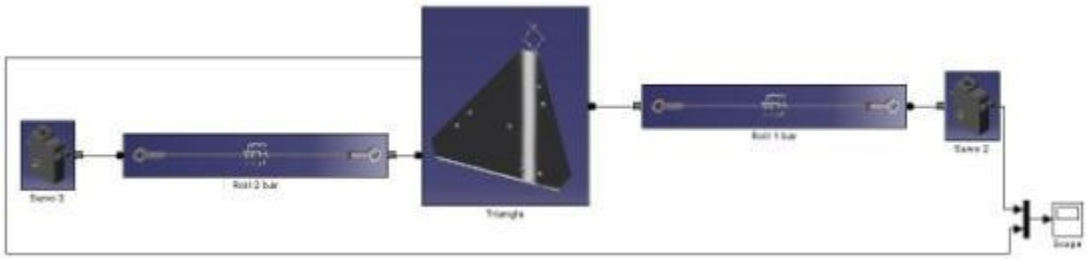
[3x3] sehingga digunakan demux untuk memisahkan data yang diperlukan. Pada simulasi *pitch* data yang dibutuhkan adalah pada scope rotasi nomor 7.



Gambar 3.50 Pengolahan *output body sensor*

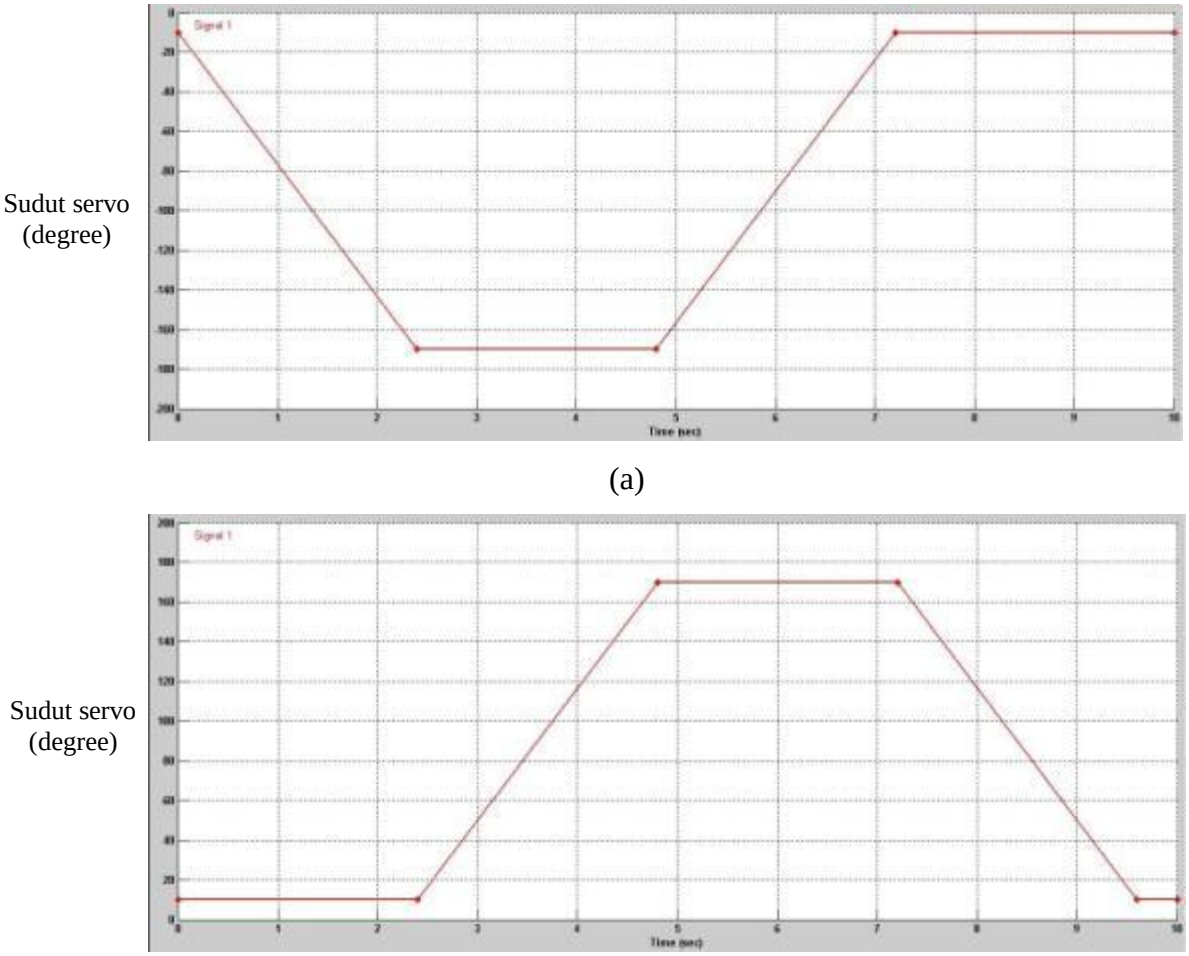
2. *Roll*

Roll disederhanakan dengan menghilangkan hubungan pada CS1 secara menyeluruh seperti terlihat pada gambar 3.51. Pada simulasi mengganti *joint* dengan penyesuaian prismatic untuk memberikan gerak translasi tidak mencegah terjadinya eror.



Gambar 3.51 Model SimMechanic *roll*

Input pada SimMechanic mirip seperti pada *pitch* dengan kondisi yang terbagi pada 2 servo, nilai pada servo 2 negatif agar arah sesuai yang diinginkan. *Input* dapat dilihat pada gambar 3.52.



Gambar 3.52 (a) *Input roll* servo2
(b) *Input roll* servo3

Body sensor diletakkan pada CG seperti pada simulasi *pitch*. Pada simulasi *roll* data yang dibutuhkan adalah pada scope rotasi nomor 6. Kesalahan pada simulasi *roll* adalah servo 3 tidak bergerak seperti pada pengujian sebenarnya. Rotasi servo berada pada sumbu Y, hal ini belum sesuai pengujian sebenarnya dimana arah rotasi servo seharusnya mengarah 30 derajat menghadap segitiga pemegang IMU.