# IMPLEMENTASI *INVERSE KINEMATICS* UNTUK ROBOT *QUADRUPED* MENGGUNAKAN SENSOR *ACCELEROMETER*

**SKRIPSI**

Oleh:

**Ahmad Iqbal Nasrudin**
**NIM 131910201007**

**PROGRAM STUDI STRATA I TEKNIK ELEKTRO**
**JURUSAN TEKNIK ELEKTRO**
**FAKULTAS TEKNIK**
**UNIVERSITAS JEMBER**
**2017**

# IMPLEMENTASI *INVERSE KINEMATICS* UNTUK ROBOT *QUADRUPED* MENGGUNAKAN SENSOR *ACCELEROMETER*

**SKRIPSI**

diajukan guna melengkapi tugas akhir dan memenuhi syarat-syarat
untuk menyelesaikan Program Studi Strata 1 Teknik Elektro (SI)
dan mencapai gelar Sarjana Teknik (ST)

Oleh:
**Ahmad Iqbal Nasrudin**
**NIM 131910201007**

**PROGRAM STUDI STRATA I TEKNIK ELEKTRO**
**JURUSAN TEKNIK ELEKTRO**
**FAKULTAS TEKNIK**
**UNIVERSITAS JEMBER**
**2017**

## PERSEMBAHAN

Dengan puja dan puji syukur kepada Allah SWT. Berkat rahmat dan karunia-Mu akhirnya Karya Tulis Ilmiah yang sederhana ini dapat terselesaikan tepat pada waktunya. Karya Tulis Ilmiah ini ku persembahkan untuk:

1. Ibunda Suriyah dan ayahanda Bisri Mustofa yang telah memberikan kasih sayang, segala dukungan, serta do'a yang tiada henti untuk kesuksesan anaknya. Ucapan terimakasih saja takkan pernah cukup untuk membalas kebaikan beliau, karena itu terimalah persembahan bakti dan cinta ku untuk Ibu Bapakku;
2. Semua sanak saudaraku yang selalu setia menyemangati serta memberikan dukungan baik dari segi mental maupun materi terhadap semua aktifitas perkuliahan dari awal sampai saat ini;
3. Guru-guruku sejak taman kanak-kanak sampai dengan perguruan tinggi;
4. Semua Dosen Jurusan Teknik Elektro Fakultas Teknik Universitas Jember yang senantiasa memberikan ilmunya. Semoga ilmu yang Bapak/Ibu berikan bermanfaat dan barokah untukku dan untuk pribadi masing-masing serta menjadi amalan penolong Bapak/Ibu kelak;
5. Saudara-saudaraku Teknik Elektro 2013 Universitas Jember;

## MOTTO

"Apabila seorang keturunan Adam meninggal dunia maka terputuslah amalnya kecuali dari tiga hal: shadaqah jariyyah, atau ilmu yang bermanfaat, atau seorang anak shalih yang mendo'akannya"
(HR. Muslim no.1631)

"Barang siapa yang menghendaki kehidupan dunia maka wajib baginya memiliki ilmu, dan barang siapa yang menghendaki kehidupan akhirat maka wajib baginya memiliki ilmu, dan barang siapa yang menghendaki keduanya maka wajib baginya memiliki ilmu"
(HR. Turmudzi)

"Tak perlu kawatir akan menjadi apa di masa depan nanti, entah itu akan berhasil atau gagal, namun yang pasti apa yang Kita lakukan sekarang akan membentuk Kita di masa depan nanti"
(Anonim)

# PERNYATAAN

Saya yang bertanda tangan di bawah ini:

Nama : Ahmad Iqbal Nasrudin

NIM : 131910201007

Menyatakan dengan sesungguhnya bahwa proyek akhir yang berjudul: "Implementasi *Inverse Kinematics* Untuk Robot *Quadruped* Menggunakan Sensor *Accelerometer"* adalah benar-benar hasil karya sendiri, kecuali jika dalam pengutipan substansi disebutkan sumbernya dan belum pernah diajukan pada institusi mana pun serta bukan karya jiplakan. Saya bertanggung jawab atas keabsahan dan kebenaran isinya sesuai dengan sikap ilmiah yang harus dijunjung tinggi.

Demikian pernyataan ini saya buat dengan sebenarnya, tanpa adanya tekanan dan paksaan dari pihak manapun serta bersedia mendapat sanksi akademik jika ternyata di kemudian hari pernyataan ini tidak benar.

Jember, 25 Juli 2017

Yang menyatakan,

Ahmad Iqbal Nasrudin
NIM 131910201007

**SKRIPSI**

# IMPLEMENTASI *INVERSE KINEMATICS* UNTUK ROBOT *QUADRUPED* MENGGUNAKAN SENSOR *ACCELEROMETER*

Oleh

Ahmad Iqbal Nasrudin

NIM 131910201007

Pembimbing :

Dosen Pembimbing Utama : M. Agung Prawira N, S.T., M.T.

Dosen Pembimbing Anggota : Khairul Anam, S.T., M.T., Ph.D.

**PENGESAHAN**

Skripsi berjudul ” Implementasi *Inverse Kinematics* Untuk Robot *Quadruped* Menggunakan Sensor *Accelerometer*” karya Ahmad Iqbal Nasrudin telah diuji dan disahkan pada:

Hari, Tanggal : Selasa, 25 Juli 2017
Tempat : Fakultas Teknik Universitas Jember

Tim penguji:

Ketua,

Mohamad Agung Prawira, S.T., M.T.
NIP 19871217 201212 1 003

Anggota I,

Khairul Anam, ST.,MT., Ph.D.
NIP 19780405 200501 1 002

Anggota II,

Sumardi, S.T., M.T.
NIP 19670113 199802 1 001

Anggota III,

Dodi Setiabudi, S.T., M.T.
NIP 198405312008121004

Mengesahkan
Dekan,

Dr. Ir. Entin Hidayah, M.U.M.
NIP 19661215 199503 200 1

# RINGKASAN

**Implementasi *Inverse Kinematics* Untuk Robot *Quadruped* Menggunakan Sensor *Accelerometer***; Ahmad Iqbal Nasrudin, 131910201007; 2017: 70 halaman; Jurusan Teknik Elektro Fakultas Teknik Universitas Jember.

Robot merupakan alat yang sudah tidak asing lagi bagi mahasiswa elektro khususnya konsentrasi Kontrol dan Instrumentasi. Pengembangan robot sangat pesat dengan berbagai jenis penggerak dan dengan *system* baru yang ditemukan, salah satunya yaitu Robot *Quadruped*. Dimana robot ini sering difungsikan sebagai robot penjelajah pada medan yang terjal.

Robot *Quadruped* merupakan robot yang bergerak dengan menggunakan 4 kaki yang disusun oleh beberapa motor servo sebagai penggeraknya di masing-masing kaki tergantung DOF yang digunakan. Pada robot *Quadruped* banyaknya n-DOF yang biasa digunakan ada dua, yaitu robot 2 DOF dan juga robot 3 DOF. Perbedaan dari kedua jenis itu adalah gerak robot berkaki yang memiliki kaki 2 DOF hanya bisa bergerak maju-mundur dan naik-turun saja, sedangkan robot berkaki yang memiliki 3 DOF gerak dari robot akan menjadi lebih kompleks yaitu robot dapat bergerak maju-mundur, naik-turun, dan juga menyamping kanan-kiri.

Pada penelitian ini robot *Quadruped* menggunakan metode *inverse kinematics* dan *forward kinematics*. Dengan metode ini robot dapat diatur ketinggian dan jarak langkahnya sesuai keinginan tanpa harus mengatur manual sudut-sudut pada motor servo karena sudut-sudut motor servo akan diperoleh dari keluaran metode *inverse kinematics*. Dengan penambahan sensor *accelerometer* dan kontrol PID robot *Quadruped* dapat menyeimbangkan *body* pada bidang miring statis maupun dinamis. Penelitian ini dilakukan di Laboratorium Sistem Kendali Jurusan Teknik Elektro Universitas Jember.

Dari penelitian robot *Quadruped* dalam menstabilkan *body* pada bidang miring dengan kontrol PID metode *Ziegler-Nichols* yang diperoleh konstanta Kp = 1,8, Ti = 0,05, dan Td = 0,0125 didapatkan respon robot sebagai berikut. Pengujian pertama yaitu pengujian statis robot terhadap sudut *pitch* dalam keadaan *standby*

pada sudut *pitch* 15° diperoleh *Settling Time* (Ts) = 200 ms dan *Error Stady State* (Ess) = 1, pada sudut *pitch* 5° diperoleh *Settling Time* (Ts) = 240 ms dan *Error Stady State* (Ess) = 3, pada sudut *pitch* -5° diperoleh *Settling Time* (Ts) = 320 ms dan *Error Stady State* (Ess) = 4, pada sudut *pitch* -15° diperoleh *Settling Time* (Ts) = 200 ms dan *Error Stady State* (Ess) = 2. Pengujian kedua yaitu pengujian statis robot terhadap sudut *roll* dalam keadaan *standby* pada sudut *roll* 15° diperoleh *Settling Time* (Ts) = 180 ms dan *Error Stady State* (Ess) = 2, pada sudut *roll* 5° diperoleh *Settling Time* (Ts) = 320 ms dan *Error Stady State* (Ess) = 2, pada sudut *roll* -5° diperoleh *Settling Time* (Ts) = 220 ms dan *Error Stady State* (Ess) = 1, pada sudut *roll* -15° diperoleh *Settling Time* (Ts) = 340 ms dan *Error Stady State* (Ess) = 2. Pengujian ketiga yaitu pengujian dinamis robot terhadap *roll* dan *pitch* dalam keadaan *standby* pada saat sudut *roll* 7° dan sudut *pitch* -10° robot memiliki respon yang lumayan cepat dalam mencapai stabil, namun setelah itu langsung diuji pada sudut *roll* 15° dan gabungan sudut *roll* dan *pitch* -10° robot memiliki respon cukup lama dalam menstabilkan *body*. Pengujian keempat yaitu pengujian statis robot menstabilkan *body* terhadap sudut *roll* dalam keadaan berjalan pada sudut *roll* -15° dalam menstabilkan *body* terdapat osilasi atau *error* maksimal 9° dikarenakan getaran yang ditimbulkan oleh robot yang sedang berjalan, pada sudut *roll* 7° dalam menstabilkan *body* terdapat osilasi atau *error* maksimal -10° dikarenakan getaran yang ditimbulkan oleh robot yang sedang berjalan. Pengujian kelima yaitu pengujian dinamis robot menstabilkan *body* terhadap sudut *roll* dalam keadaan berjalan pada gangguan *roll* sebesar 7° detik ke 1540 ms robot mampu menstabilkan *body* pada detik ke 2120 ms dengan *Error Stady State* (Ess) = 6°, pada gangguan *roll* sebesar -7° detik ke 1740 ms robot mampu menstabilkan *body* pada detik ke 2140 ms dengan *Error Stady State* (Ess) = 6°.

## SUMMARY

**Implementation of Inverse Kinematics For Quadruped Robot Using Accelerometer Sensor**; Ahmad Iqbal Nasrudin, 131910201007; 2017: 70 page; Department Of Electrical Engineering Faculty Of Engineering University Of Jember.

Robots are tools that are familiar to students of the electrical and Instrumentation Control concentration in particular. Development of robot very rapidly with different types of movers and with the new system were found, one of them namely Quadruped Robot. Where a robot is often being utilized as a robot explorers on steep terrain.

Quadruped robot is a robot that moves by using the 4 legs are made from some of the servo motor as a driving force on each foot fit the DOF is used. Quadruped robot on the number n-DOF is used there are two, namely 2 DOF robot and also robot 3 DOF. The difference of both types it is the motion of legged robots that have a 2 DOF can only moved forward-backward and up-down, whereas a legged robot has 3 DOF motion of robots will become more complex, namely robot can move forward-backward, up-down, and also sideways right-left.

On the research of this Quadruped robot using inverse kinematics method and forward kinematics. With this method the robot can set altitude and distance stride as you wish without having to manually set the corners on a servo motor because the corners of the servo motor will be retrieved from the output method of inverse kinematics. With the addition of sensor accelerometer and PID control of Quadruped robot can balance the body on the inclined plane static or dynamic. This research was conducted in the laboratory of the Department of Control Systems of Electrical Engineering University of Jember.

Research of robot Quadruped in stabilizing the body on the incline with PID control method of Ziegler-Nichols obtained constants Kp = 1.8, Ti = 0.05, and Td = 0.0125 robotic response is obtained as follows. The first static test of the robot against the corner of the pitch in standby at an angle of the pitch 15° obtained Settling Time (Ts) = 200 ms and Error Stady State (Ess) = 1, at an angle of the pitch

5° obtained Settling Time (Ts) = 240 ms and Error Stady State (Ess) = 3, at the angel of the pitch -5° obtained Settling Time (Ts) = 320 ms and Error Stady State (Ess) = 4, at the angle of the pitch 15° obtained Settling Time (Ts) = 200 ms and Error Stady State (Ess) = 2. Second, static testing robot against the angle of roll in standby at the angle of the roll 15° obtained Settling Time (Ts) = 180 ms and Error Stady State (Ess) = 2, at the angle of the roll 5° obtained Settling Time (Ts) = 320 ms and Error Stady State (Ess) = 2, at the angle of the roll 5° obtained Settling Time (Ts) = 220 ms and Error Stady State (Ess) = 1, at the angle of the roll 15° obtained Settling Time (Ts) = 340 ms and Error Stady State (Ess) = 2. Third, dynamic testing robot against the roll and pitch in standby at the time angle roll 7° and 10° angle of pitch, robot have a response fairly quickly in achieving stable, however after that directly tested at the angle of 15 ° combined roll and roll and the angle of 10° pitch robot have a long enough in response to stabilize the body. Fourth, static testing robot to stabilize the body roll angle against the State in running at the angle of te roll 15° for stabilize the body there is oscillations or maximum error 9° due to vibration caused by robot that is currently running, at the angle of the roll 7° for stabilize the body there is oscillations or maximum error -10° due to vibration caused by the robot is running. Fifth, dynamic testing robot for stabilize the body against the roll angel in running interference was given the roll 7° on the second to 1540 ms, robot able to stabilize the body in seconds to 2120 ms with Error Stady State (Ess) = 6°, and interference was given the roll of-7° on the second to 1740 ms, robot able to stabilize the body in seconds to 2140 ms with Error Stady State (Ess) = 6 °.

# PRAKATA

*Bismillahirrohmanirrohim*

Puji syukur ke hadirat Allah SWT atas segala rahmat dan hidayah-Nya sehingga laporan skripsi yang berjudul "*Implementasi Inverse Kinematics Pada Robot Quadruped Menggunakan Sensor Accelerometer"* dapat terselesaikan dengan baik. Laporan skripsi ini disusun untuk memenuhi salah satu syarat dalam menyelesaikan pendidikan Strata Satu (S1) pada Jurusan Teknik Elektro Fakultas Teknik Universitas Jember.

Terselesaikannya laporan skripsi ini tidak terlepas dari bantuan berbagai pihak, oleh karena itu disampaikan ucapan terima kasih kepada:

1. Allah Swt yang telah melimpahkan rahmat dan rizki-Nya serta memberi kelancaran dan kemudahan sehingga terselesaikannya skripsi ini.
2. Nabi Muhammad SAW yang telah membawa kita ke peradaban manusia yang lebih baik.
3. Ibu dan Bapak serta saudara-saudaraku tercinta, atas jasa-jasanya, kesabaran, do'a, dan tidak pernah lelah dalam mendidik dan memberi cinta yang tulus dan ikhlas kepada penulis semenjak kecil.
4. Ibu Dr. Ir. Entin Hidayah M.U.M selaku Dekan Fakultas Teknik Universitas Jember.
5. Bapak Dr. Ir. Bambang Sri Kaloko, S.T., M.T. selaku Ketua Jurusan Fakultas Teknik Elektro Universitas Jember.
6. Bapak Dedi Kurnia Setiawan, S.T., M.T. selaku Ketua Prodi S1 Fakultas Teknik Elektro Universitas Jember.
7. Bapak M. Agung Prawira N, S.T., M.T. selaku dosen pembimbing utama dan Bapak Khairul Anam, ST., MT., Ph.D. selaku dosen pembimbing anggota yang telah meluangkan waktu dan pikiran guna memberikan bimbingan dan pengarahan dalam penyusunan proyek akhir ini.
8. Seluruh Dosen yang ada di Fakultas Teknik khususnya Teknik Elektro beserta karyawan.

9. Keluarga kecil kontrakan pojok tercinta yang telah banyak memberikan dorongan, semangat, kasih sayang, dan bantuan baik secara moral maupun materi demi lancarnya penyusunan skripsi ini
10. Keluarga besar Teknik Elektro seperjuangan khususnya angkatan 2013 INTEL UNEJ, terimakasih atas dukungan dan motivasi yang kalian berikan selama menjalani masa kuliah sampai terlaksananya skripsi ini.
11. Semua pihak yang tidak dapat saya sebutkan satu persatu, terima kasih atas dukungan dan motivasi kalian dalam penyusunan skripsi ini.

Penulis menyadari bahwa sebagai manusia biasa tidak terlepas dari keterbatasan, yang biasanya akan mewarnai kadar ilmiah dari skripsi ini. Oleh karena itu penulis selalu terbuka terhadap masukan dan saran dari semua pihak yang sifatnya membangun untuk mendekati kesempurnaan. Tidak lupa penulis menyampaikan permohonan maaf yang sebesar-besarnya jika terdapat kesalahan dan kekeliruan. Akhir kata penulis berharap laporan ini dapat memberikan manfaat bagi pembaca dan dapat menjadi bahan acuan yang bermanfaat di kemudian hari.

Jember, 25 Juli 2017

Penulis

# DAFTAR ISI

# DAFTAR TABEL

# DAFTAR GAMBAR

# DAFTAR LAMPIRAN

# BAB 1. PENDAHULUAN

## 1.1 LATAR BELAKANG

Robot merupakan barang yang sudah tidak asing lagi bagi mahasiswa elektro khususnya konsentrasi Kontrol dan Instrumentasi. Pengembangan robot sangat pesat dengan berbagai macam *system* baru yang ditemukan. Salah satu jenis robot yang juga banyak dikembangkan adalah Robot Berkaki. Robot ini bergerak berdasarkan kinematik gerak yang diterapkan pada tiap-tiap kaki yang disusun oleh motor-motor servo. (Asrofi, dkk. 2015)

Robot berkaki merupakan robot yang menggunakan beberapa servo sebagai aktuatornya di masing-masing kaki tergantung berapa kaki dan DOF yang digunakan. Pada robot berkaki banyaknya n-DOF yang biasa digunakan yaitu ada dua, robot 2-DOF dan juga robot 3-DOF. Perbedaan dari kedua jenis itu adalah gerak robot berkaki yang memiliki kaki 2 DOF hanya bisa bergerak maju-mundur dan naik-turun saja, sedangkan robot berkaki yang memiliki 3 DOF gerak dari robot akan menjadi lebih kompleks yaitu robot dapat bergerak maju-mundur, naik-turun, dan juga menyamping kanan-kiri. Sedangkan dari banyaknya kaki jenisnya robot berkaki yaitu ada robot berkaki dua, berkaki empat, berkaki enam, dan berkaki delapan, semakin banyak kaki yang digunakan maka akan semakin seimbang pula beban yang dipikul oleh servo, sebaliknya semakin sedikit kaki yang digunakan maka akan semakin berkurang keseimbangan robot dan beban yang dipikul servo akan lebih berat, robot berkaki delapan akan lebih seimbang daripada robot berkaki enam, robot berkaki enam akan lebih seimbang daripada robot berkaki empat, dan robot berkaki empat akan lebih seimbang daripada robot berkaki dua yang rawan akan ketidakseimbangan, pasalnya beban yang dipikul haruslah rata disetiap kakinya untuk memperoleh keseimbangan. Kekurangan tersebut juga sering terjadi ketika robot menemui permukaan bidang yang tidak rata (miring). Hal ini disebabkan bidang yang tidak rata akan mempengaruhi titik beban badan robot, Perpindahan titik beban badan robot ini akan mengakibatkan pembebanan pada salah satu motor servo. Hal ini yang akan menyebabkan motor servo yang diberikan beban tertinggi mengalami kerusakan yang lebih cepat.

Dari permasalahan tersebut di atas akan dilakukan penelitian robot berkaki empat 3-DOF yang membahas tentang "***Implementasi Inverse Kinematics untuk Robot Quadruped Menggunakan Sensor Accelerometer***". Dengan menggunakan sensor *accelerometer* dan algoritma *inverse kinematics* diharapkan akan mampu mengoptimalkan keseimbangan robot *quadruped*, dengan demikian tidak ada motor servo yang bekerja paling berat atau terbebani lebih antara servo satu dengan servo di kaki yang lainnya sehingga umur pakai servo akan lebih panjang dan akan memperoleh keseimbangan robot yang maksimal.

## 1.2 Rumusan Masalah

Dari permasalahan yang ada, yang menjadi fokus utama dari latar belakang adalah sebagai berikut:

1. Bagaimana mendesain robot berkaki empat (*quadruped*).
2. Bagaimana membuat algoritma *inverse kinematics* agar robot dapat seimbang dengan menggunakan sensor *accelerometer*.

## 1.3 Batasan Masalah

Pada penelitian implementasi *invers kinematics* untuk memperoleh keseimbangan pada robot *quadruped* dengan menggunakan sensor *accelerometer* memiliki batasan masalah sebagai berikut:

1. Penelitian ini hanya akan menganalisa optimalisasi pada bagian kaki.
2. Pada penelitian ini robot tidak didesain untuk *mapping* lintasan.
3. Jenis kaki yang digunakan pada robot ini adalah jenis kaki yang memiliki 3 sendi (3 DOF).
4. Lintasan adalah bidang miring dengan sudut kemiringan tertentu.

## 1.4 Tujuan Penelitian

1. Membuat desain robot berkaki empat (*quadruped*).
2. Membuat robot *quadruped* seimbang dengan menggunakan algoritma *invers kinematics* dan sensor *accelerometer*.

### 1.5 Manfaat Penelitian

Manfaat dari penelitian ini adalah:

1. Metode *inverse kinematics* pada robot *quadruped* dengan sensor *accelerometer* ini diharapkan menjadi metode yang efektif untuk memperoleh keseimbangan robot.
2. Kedepannya diharapkan juga dapat dimanfaatkan pada kontes robot devisi pemadam api berkaki.
3. Dengan menggunakan kaki empat dapat mengurangi biaya, karena kebanyakan peserta KRPAI kebanyakan masih menggunakan kaki enam dimana membutuhkan motor servo lebih banyak.
4. Rujukan bagi peneliti lain yang akan melakukan pengembangan atau penelitian selanjutnya.

### 1.6 Sistematika Penulisan

Sistematika penuliasan skripsi ini dibagi menjadi 3 bagian utama yaitu bagian pendahuluan, bagian isi, dan bagian penutup.

**BAB 1. PENDAHULUAN**, berisi: latar belakang, rumusan masalah, batasan masalah, tujuan dan manfaat penelitian, dan sistematika penulisan.

**BAB 2. TINJAUAN PUSTAKA**, berisi: Robot, Arduino Nano, Motor Servo, Sensor *Accelerometer*, dan *Inverse Kinematics*.

**BAB 3. METODOLOGI PENELITIAN**, berisi: Tempat dan Waktu Penelitian, Tahapan Pelaksanaan, Diagram Alir Penelitian, Rencana Jadwal Pelaksanaan Penelitian, Alat dan Bahan, Blok Diagram Sistem, *Flowchart*.

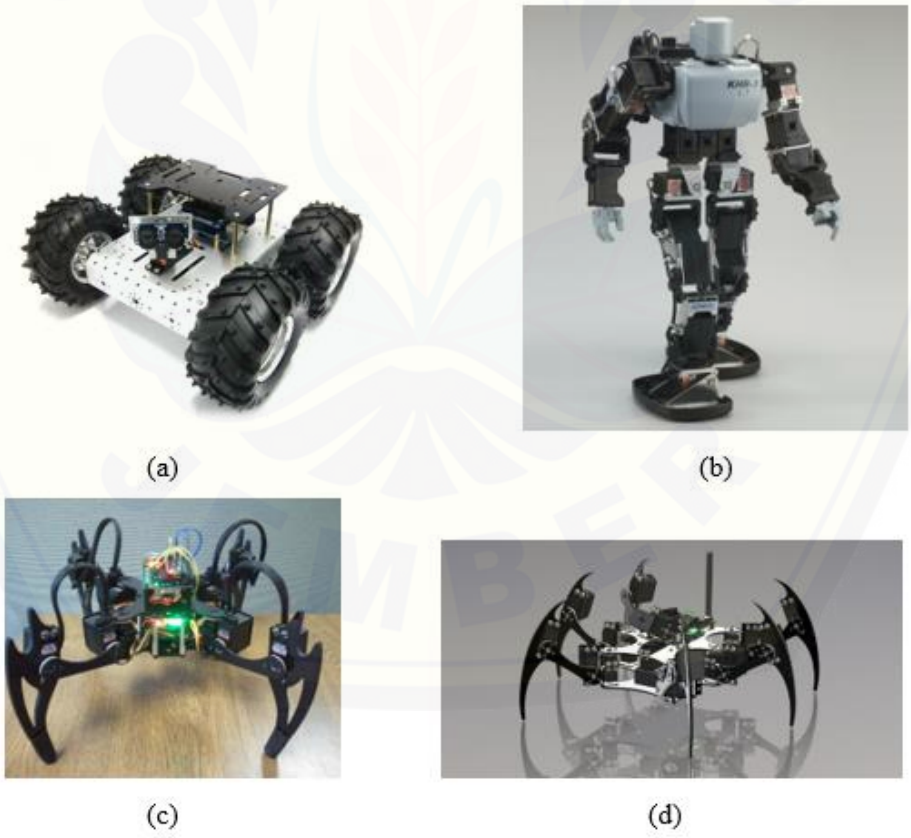
**BAB 4. PENGUJIAN DAN HASIL PENELITIAN**, berisi: Pengujian Alat Perbagian, Tuning Kontrol PID dengan Teori *Ziegler-Nichols*, dan Pengujian Kinerja Robot Quadruped secara Keseluruhan dengan Kontrol PID.

**BAB 5. PENUTUP**, berisi: Kesimpulan, dan Saran.

# BAB 2 TINJAUAN PUSTAKA

## 2.1 Robot

*Mobile* robot merupakan alat mekanik yang mampu bergerak pada suatu lingkungan dengan tingkat kemampuan tertentu (Carelli dan Freire, 2003). *Mobile* robot dapat dikendalikan secara manual maupun secara otomatis. *Mobile* robot yang dikendalikan secara manual dapat menggunakan *remote control*, *stick game* maupun yang langsung terhubung dengan perangkat komputer, sedangkan *mobile* robot yang dikendalikan secara otomatis dapat menggunakan sensor sebagai masukan untuk robot. *Mobile* robot terbagi dalam dua jenis yaitu *wheeled* robot atau robot beroda dan *legged* robot atau robot berkaki. Jumlah robot berkaki bermacam-macam seperti robot berkaki dua (*humanoid*), robot berkaki empat (*Quadruped*), dan robot berkaki enam (*Hexapod*).




Gambar 2.1 (a) *Mobile Robot*, (b) *Biped Robot*, (c) *Quadruped Robot*, (d) *Hexapod Robot*

(Sumber: https://www.google.com)

*Mobile* robot yang dikendalikan secara otomatis harus mempunyai sensor yang baik dan juga pemasangan sensor yang tepat. Sensor pada robot digunakan sebagai navigasi. Navigasi pada robot merupakan masalah untuk memerintahkan robot berinteraksi dengan lingkungannya. Deteksi batas-batas pada ruang dan estimasi posisi merupakan dua peranan penting dan mendasar dari navigasi. Deteksi batas diperlukan agar robot tidak bertabrakan dengan objek lain, sementara estimasi posisi diperlukan agar robot dapat menempatkan posisi sesuai dengan lingkungannya. (Crowley, 1989).

Sensor jarak pada robot berfungsi untuk medeteksi batas dan mengetahui letak dari sebuah objek yang berada di sekitarnya. Sensor jarak umumnya menggunakan sensor *infrared* dan sensor *ultrasonic*. Sensor *infrared* memanfaatkan cahaya *infrared* sebagai pendeteksi jaraknya, sementara sensor *ultrasonic* memanfaatkan gelombang *ultrasonic* sebagai pendeteksi jaraknya. Pengukuran kesalahan sensor *ultrasonic* bergantung pada beberapa faktor. Faktor yang mendasar adalah kondisi lingkungan dimana propagasi gelombang *ultrasonic* berada seperti suhu, kelembaban, dan pergerakan udara. Kondisi fungsional yang dapat menyebabkan gangguan adalah adanya pemancar gelombang aktif pada frekuensi yang sama, pantulan dari objek lain misalnya fenomena *multi-echo* dari objek tertentu (Majchrzak *et al*, 2009).

Kaki *hexapod* digerakan menggunakan servo. Jumlah servo yang digunakan berbeda-beda mulai dari 3 servo hingga 18 servo. Perbedaan jumlah servo tentunya membedakan pula algoritme pergerakan kaki, peletakan servo, serta *design* kerangka *hexapod*. *Design* kerangka *hexapod* terdiri atas tubuh yang kaku dengan enam kaki yang sesuai, masing-masing kaki memiliki pergerakan yang bebas (Saranli *et al*, 2001).

Selain dari *design* kerangka robot yang berbeda, algoritme pergerakan kaki antara cara berjalan *tripod* dengan cara berjalan lambat atau berjalan lainnya berbeda-beda, hal ini dikarenakan kontrol algoritme berjalan biasanya tidak terpusat, yang berarti bahwa pergerakan setiap kaki relatif bebas (Thirion dan Thiry, 2002).

Penelitian terdahulu yang terkait ialah Implementasi *Inverse Kinematics* untuk Koordinasi Gerak Robot Berkaki Enam, penelitian ini menggunakan 18 servo (Wulandari, 2013). Penelitian tersebut berhasil mengimplementasikan algoritme *Inverse Kinematics* untuk menggerakan kaki robot dan koordinasi kaki dengan baik. Namun, pergerakan robot masih dirasakan kurang cepat dan penggunaan 18 servo menghabiskan daya yang cukup besar. Meminimalisir jumlah servo diharapkan dapat menghemat daya, namun dapat pula mempercepat pergerakan kaki.

## 2.2 Arduino Mega 2560

Arduino Mega 2560 adalah jenis mikrokontroler yang berbasis Arduino dengan menggunakan *chip* ATmega2560. Pada *board* ini memiliki pin Input/Output yang lumayan banyak, sejumlah 54 buah digital I/O pin (15 pin diantaranya adalah PWM), 16 pin *input* analog, 4 pin UART (*serial port hardware*). Arduino Mega 2560 dilengkapi dengan sebuah oscilator 16 Mhz, 1 buah *port* USB, *power jack* DC, ICSP *header*, dan tombol reset. Board mokrokontroler ini sudah sangat lengkap, dimana sudah memiliki segala yang dibutuhkan untuk sebuah mikrokontroler. Dengan penggunaan yang cukup sederhana sama halnya dengan Arduino jenis lainnya, kita tinggal menghubungkan *power* dari USB ke PC atau melalui adaptor DC ke jack DC.


Gambar 2.2 Arduino Mega 2560

(Sumber: *www.arduino.cc*)

- Spesifikasi:

| | |
|---|---|
| *Chip* Mikrokontroler | ATmega2560 |
| Tegangan Operasi | 5V |
| Tegangan *input* (dianjurkan) | 7-12V |
| Tegangan *input* (batas) | 6-20V |
| Digital I/O Pin | 54 buah, 15 diantaranya *output* PWM) |
| Analog *Input* Pin | 16 |
| Arus DC Pin I/O | 20 mA |
| Arus DC Pin 3.3V | 50 mA |
| Memori Flash | 256 KB, 8 KB telah digunakan untuk bootloader |
| SRAM | 8 KB |
| EEPROM | 4 KB |
| Clock Speed | 16 MHz |
| Panjang | 101,52 mm |
| Lebar | 53,3 mm |
| Berat | 37 g |

- Pemrograman

Untuk melakukan pemrograman *board* Arduino Mega 2560 dilakukan dengan menggunakan *Software* Arduino IDE yang bisa kita dapatkan secara gratis di *website resminya*. *Chip* ATmega2560 pada Arduino Mega 2560 telah terisi program awal yang biasa disebut dengan *bootloader*. *Bootloader* sendiri adalah *firmware* yang gunanya untuk memudahkan kita melakukan pemrograman yang lebih sederhana dengan menggunakan Arduino *Software*, tanpa harus menggunakan tambahan *hardware* lain (*downloader*). Cukup dengan menghubungkan Arduino menggunakan kabel USB ke PC atau Mac/Linux, jalankan *Software* Arduino IDE, dan dapat langsung mulai memprogram *chip* ATmega2560. Lebih mudahnya lagi, pada *Software* Arduino IDE telah diberikan banyak contoh program bagi yang masih dalam tahap pembelajaran.

Bagi pengguna mikrokontroler yang sudah mahir, bisa juga memrogram Arduino Mega 2560 dengan tanpa menggunakan program awal (*bootloader*) dan

melakukan pemrograman langsung via pin ICSP (*In Circuit Serial Programming*) dengan menggunakan Arduino ISP.

Pada Arduino Mega 2560 dilengkapi dengan chip ATmega16U2 yang di dalamnya telah terisi program yang fungsinya sebagai konverter USB *to Serial*. *Firmware* ATmega16U2 di *load* oleh DFU *bootloader*, dan kita juga dapat merubahnya yaitu dengan menggunakan *software* Atmel Flip untuk pengguna Windows atau DFU *programmer* untuk pengguna Mac OSX dan Linux.

- Proteksi

*Board* Arduino Mega 2560 telah dilengkapi dengan fitur proteksi yaitu *polyfuse* yang dapat direset gunanya untuk melindungi *port* USB komputer/laptop kita dari *short* atau arus yang berlebih. Walaupun sebenarnya pada kebanyakan komputer mempunyai perlindungan *port* tersebut, namun *polyfuse* pelindung pada Arduino Mega 2560 memberikan lapisan perlindungan tambahan yang membuat kita merasa aman ketika menghubungkan Arduino ke komputer. Jika arus lebih dari 500mA, maka sirkuit proteksi akan secara otomatis memutus, dan akan menyambung kembali ketika batasan arus telah aman.

- *Power Supply*

*Board* Arduino Mega 2560 sudah akan mendapatkan sumber tegangan ketika terkoneksi kabel USB dengan komputer, atau juga bisa dengan menggunakan *power supply* eksternal.

*Power supply* eksternal dapat diperoleh dari adaptor DC atau baterai, melalui jack DC yang tersedia pada Arduino, atau menghubungkan langsung GND dan pin Vin pada *board. Power supply* eksternal yang dapat digunakan adalah yang memiliki tegangan antara 6V sampai 20V. Namun ada catatan jika diberi tegangan kurang dari 7V, maka pin 5V akan memberikan nilai tegangan yang tidak murni 5V, sehingga akan memungkinkan rangkaian bekerja dengan tidak sempurna. Dan jika diberi tegangan yang lebih dari 12V, regulator tegangan pada *board* bisa *overheat* yang bisa merusak PCB. Dengan demikian, rentang tegangan yang direkomendasikan adalah 7V sampai 12V

Pin *power* pada Arduino Mega 2560:

- **GND**, adalah *ground* (negatif).
- **Vin**, adalah pin yang digunakan jika kita ingin memberikan catu daya langsung ke *board* Arduino dengan rentang tegangan yang disarankan 7V - 12V.
- **Pin 5V**, adalah pin *output* dimana pin tersebut memberikan sumber tegangan 5V yang telah melalui regulator tegangan.
- **3V3**, adalah pin *output* dimana pin tersebut memberikan sumber tegangan 3.3V yang telah melalui regulator tegangan.
- **IOREF**, adalah pin yang memberikan referensi tegangan mikrokontroler. Biasanya digunakan pada *board shield* untuk mendapatkan tegangan yang sesuai, yaitu 5V atau 3.3V.

➢ Memori

Pada Arduino Mega 2560 memiliki kapasitas memori sebesar 256 KB, sedangkan 8 KB dari memori tersebut telah digunakan untuk *bootloader*. Jumlah SRAM berkapasitas sebesar 8 KB, dan EEPROM sebesar 4 KB, yang dapat digunakan dengan menggunakan *EEPROM library* saat melakukan pemrograman.

➢ *Input* dan *Output* (I/O)

*Input* dan *output* pada Arduino Mega 2560 memiliki jumlah pin paling banyak dari semua *board* Arduino. Arduino Mega 2560 memiliki sebanyak 54 buah pin digital yang bekerja pada tegangan 5V, dan setiap pin dapat memberi maupun menerima arus sebesar 20mA, yang dapat difungsikan sebagai *input* maupun *output*, dengan menggunakan sintak pemrograman `pinMode()`, `digitalWrite()`, dan `digital(Read)`.

Pin-pin memiliki fungsi khusus:

- ***Serial***, memiliki 4 pin *serial* yang masing-masing terdiri dari 2 pin RX dan TX. *Serial* 0 : pin 0 (RX) dan pin 1 (TX). *Serial* 1 : pin 19 (RX) dan pin 18 (TX). *Serial* 2 : pin 17 (RX) dan pin 16 (TX). *Serial* 3 : pin 15 (RX) dan pin 14 (TX). RX sebagai penerima dan TX sebagai pengirim data *serial*. Pin 0 dan pin 1 adalah pin *serial* yang digunakan oleh USB *to* TTL ATmega16U2.

- ***External Interrups***, yaitu pin 2 (*interrupt* 0), pin 3 (*interrupt* 1), pin 18 (*interrupt* 5), pin 19 (*interrupt* 4), pin 20 (*interrupt* 3), dan pin 21 (*interrupt* 2). Dengan demikian Arduino Mega 2560 memiliki jumlah *interrupt* yang cukup banyak yaitu 6 buah. Untuk mengaktifkannya dengan sintak `attachInterrupt()`.
- **PWM**: Pin 2 sampai 13 dan 44 sampai 46, adalah pin yang menyediakan *output* PWM 8 bit dengan menggunakan sintak `analogWrite()`.
- **SPI** : Arduino Mega 2560 mendukung komunikasi SPI yang terdapat pada Pin 50 (MISO), 51 (MOSI), 52 (SCK), dan 53 (SS) dengan menggunakan SPI *Library* untuk mengaktifkannya.
- **LED** : Pada pin 13 terdapat *built-in* LED yang dikendalikan oleh pin 13. Set *HIGH* untuk menyalakan, dan set *LOW* untuk mematikannya.
- **I2C** : Arduino Mega 2560 juga mendukung komunikasi I2C yang terdapat pada Pin 20 (SDA) dan pin 21 (SCL) dengan menggunakan *Wire Library* untuk mengaktifkannya.
- **AREF**: Sebagai tegangan refernsi untuk *input* analog.
- ***Reset***: Sama dengan tombol *reset*. Pada Arduino Mega terdapat pin yang berfungsi sebagai *reset* dengan dihubungkan ke *LOW* maka Arduino akan melakukan *reset*.

➢ Komunikasi

Arduino Mega 2560 memiliki beberapa fasilitas komunikasi dengan komputer, Arduino lainnya, maupun *board* mikrokontroler lainnya. IC Atmega2560 terdapat komunikasi *serial* UART TTL (5V) pada pin 0 (RX) dan pin 1 (TX). IC ATmega16U2 yang ada pada *board* berfungsi untuk menerjemahkan komunikasi ini melalui USB dan akan tampil sebagai *Virtual Port* di komputer. *Firmware* 16U2 menggunakan *driver* USB standar sehingga tidak membutuhkan *driver* tambahan.

Pada *Software* Arduino IDE terdapat *serial monitor* yang memudahkan data textual untuk dikirim menuju Arduino atau yang keluar dari Arduino. indikator LED TX dan RX akan menyala berkedip-kedip ketika ada data yang ditransmisikan

melalui *chip* USB *to Serial* dengan kabel USB ke komputer. Untuk menggunakan komunikasi *serial* digunakan *Serial library*.

IC ATmega2560 juga mendukung komunikasi I2C dan SPI. Pada *Software* Arduino IDE sudah dilengkapi *Wire Library* untuk memudahkan kita menggunakan komunikasi I2C. Dan begitu juga untuk menggunakan komunikasi SPI, digunakan SPI *library*.

- *Reset* Otomatis

Berbeda dengan *board* mikrokontroler lainnya yang biasanya harus menekan tombol *reset* sesaat sebelum *upload* program, Pada Arduino Mega 2560 dilengkapi dengan fitur *auto reset* yang dikendalikan oleh *software* pada komputer yang terkoneksi. Jadi kita tidak lagi direpotkan ketika melakukan pemrograman Arduino Mega 2560.

## 2.3 Motor Servo

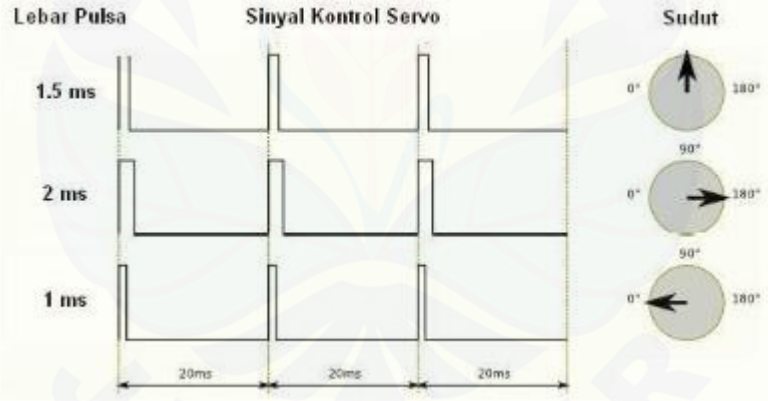
Motor servo adalah suatu aktuator (motor) yang geraknya adalah berputar yang dirancang dengan sistem kontrol umpan balik (*close loop*), sehingga dapat diatur untuk menentukan dan memastikan posisi sudut dari porosnya. Motor servo merupakan perangkat yang di dalamnya terdiri dari motor DC, serangkaian *gear*, rangkaian kontrol dan potensiometer. Serangkaian *gear* yang terhubung dengan poros motor DC gunanya untuk memperlambat putaran dan meningkatkan torsi dari motor servo, sedangkan potensiometer berfungsi sebagai penentu batas posisi putaran poros motor servo dengan perubahan resistansinya saat motor berputar.

Pengendalian motor servo hanya dilakukan dengan menggunakan metode PWM (*Pulse Width Modulation*). Motor servo akan berputar searah arah jarum jam jika jumlah pulsa yang dimasukkan lebih kecil dari 1600 pulsa (1,6 ms pulsa). Sedangkan motor servo akan berputar berlawanan jarum jam jika jumlah pulsa yang dimasukkan lebih besar dari 1600 pulsa (1,6 ms pulsa).

Secara umum motor servo memiliki 2 jenis. Yaitu motor servo standar dan motor servo *Continous*. Motor servo jenis standar hanya akan mampu berputar sebesar 180 derajat. Motor servo jenis ini sering digunakan untuk sistem robotika

misalnya pada "Robot *Arm*" (Robot Lengan). Sedangkan motor Servo jenis *continuous* mampu berputar sebesar 360 derajat. motor servo c*ontinous* jenis ini sering digunakan untuk *Mobile* Robot. Teknik PWM (*Pulse Width Modulation*) ini menggunakan sistem lebar pulsa untuk mengendalikan putaran motor servo. Tampak pada gambar ketika motor servo diberikan pulsa 1.5 ms dengan periode selebar 20 ms, maka gerakan motor servo akan berada pada posisi tengah atau 90 derajat. Semakin lebar pulsa *OFF* maka akan semakin besar gerakan sumbu ke arah jarum jam dan semakin kecil pulsa *OFF* maka akan semakin besar gerakan sumbu ke arah yang berlawanan dengan jarum jam. (Sujarwata, 2013)

Untuk menggerakkan motor servo ke kanan atau ke kiri, berdasarkan dari nilai *delay* yang kita berikan. Untuk membuat motor servo pada posisi *center* diberikan pulsa sebesar 1.5ms. Dan untuk membuat servo berputar ke kanan diberikan pulsa lebih kecil dari 1.3ms, dan pulsa lebih besar dari 1.7ms untuk berputar ke kiri dengan delay 20ms, seperti pada gambar berikut:




Gambar 2.3 Pulsa kendali motor servo

(Sumber: *Sujarwata. 2013*)

**2.4 Sensor MPU-6050**

Modul MPU-6050 adalah modul berinti IC MPU-6050 yang merupakan 6 axis *Motion Processing Unit* dengan penambahan regulator tegangan dan beberapa komponen pelengkap lainnya, sehingga membuat modul ini dapat langsung dipakai dengan tegangan DC sebesar 3V sampai 5V. Modul ini menggunakan komunikasi

I2C yang dapat langsung disambungkan ke mikrokontroler yang mempunyai fasilitas I2C.

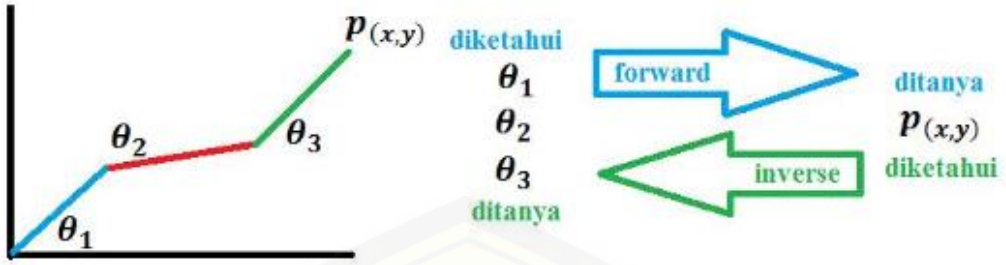
Sensor MPU-6050 ini berisikan sebuah sensor *Accelerometer* dan sebuah sensor *Gyroscope* yang masing-masing adalah 3 axis dan saling terintegrasi. Dengan fasilitas *hardware internal* 16 bit ADC untuk setiap kanalnya membuat sensor ini sangat akurat dalam pembacaannya. Sensor menangkap nilai kanal axis X, Y, dan Z secara bersamaan dalam satu waktu.

Spesifikasi modul MPU-6050:

- Berbasis IC MPU-6050
- Sumber tegangan berkisar 3-5VDC
- *Gyroscope range* + 250 500 1000 2000 ° /s
- *Accelerometer range* ± 2 ± 4 ± 8 ± 16 g
- Komunikasi I2C
- *Built-in* 16 bit ADC, 16 bits data output
- Jarak antar pin 2,54 mm
- Dimensi modul 20,3 mm x 15,6 mm

## 2.5 *Invers Kinematics*

*Inverse Kinematics* merupakan kebalikan dari Forward Kinematics, jika Forward Kinematik adalah metode untuk menentukan orientasi dan posisi *end-effector* dari besarnya sudut sendi dan panjang link lengan. Persamaan forward kinematics didapatkan berdasarkan jumlah DOF dan jenis kinematic chain dari kaki-kaki robot berkaki. Jadi inverse kinematics yaitu menentukan besarnya sudut sendi dari orientasi dan posisi end-effector saat panjang link telah ditetapkan. Metode ini diperlukan untuk mengetahui besarnya sudut pada persendian yang diperlukan agar end-effector dapat mencapai posisi yang dikehendaki.



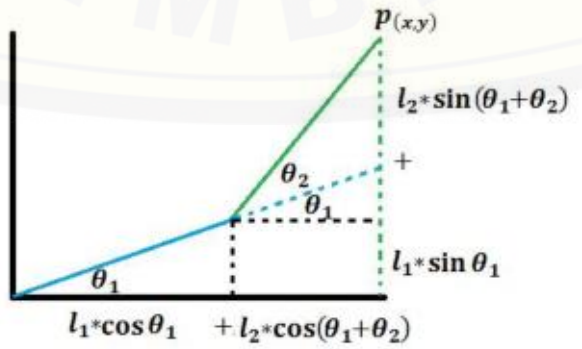

Gambar 2.4 Model Kontrol Kinematika Robot
(https://handritoar.files.wordpress.com/2013/08/untitled.gif)

Inverse Kinematics lebih banyak diaplikasikan, namun memiliki kerumitan dalam perhitungannya karena beberapa hal, diantaranya:

1. Melibatkan persamaan non-linier, Solusi yang dihasilkan bisa banyak dan bahkan menjadi tak hingga.
2. Kemungkinan tidak mendapatkan solusi terjadi ketika posisi end-effector berada di luar wrokspace atau configuration space.
3. Solusi perhitungan akan semakin rumit ketika jumlah link dan sendi semakin banyak namun posisi end-effector menjadi semakin akurat.

Ada beberapa metode untuk memecahkan permasalahan inverse kinematics, yang metode yang dirasa paling sederhana adalah dengan pendekatan persamaan trigonometri.

Berikut ini adalah gambar grafik representasi sendi gerak lengan robot yang digunakan dalam perhitungan *inverse kinematics* pada robot berkaki:




Gambar 2.5 Grafik Representasi Lengan Robot Berkaki

Dalam inverse kinematics 2 DOF memiliki kesulitan jika diselesaikan dengan hanya menggunakan persamaan trigonometri, tetapi masih dapat diselesaikan dengan cara berikut,

1. Berdasarkan forward kinematics 2 DOF

$$Px = l_1.cos\theta_1 + l_2.\cos(\theta_1 + \theta_2) \quad \text{.....(2.1)}$$

$$Py = l_1.sin\theta_1 + l_2.\sin(\theta_1 + \theta_2) \quad \text{.....(2.2)}$$

2. Pertama, harus mencari $\theta_2$ terlebih dahulu, $\theta_2$ dapat diperoleh dari peng-kuadrat-an dikedua sisi persamaan baik Px maupun Py,

$$P_x^2 = [l_1.cos\theta_1 + l_2.\cos(\theta_1 + \theta_2)]^2$$

$$P_x^2 = l_1^2.cos^2\theta_1 + l_2^2.cos^2(\theta_1 + \theta_2) + 2l_1l_2.cos\theta_1.\cos(\theta_1+\theta_2)$$

$$P_y^2 = [l_1.sin\theta_1 + l_2.sin(\theta_1 + \theta_2)]^2$$

$$P_y^2 = l_1^2.sin^2\theta_1 + l_2^2.sin^2(\theta_1 + \theta_2) + 2l_1l_2.sin\theta_1.\sin(\theta_1+\theta_2)$$

$$P_x^2 + P_y^2 = l_1^2cos^2\theta_1 + l_2^2cos^2(\theta_1 + \theta_2) + 2l_1l_2cos\theta_1.\cos(\theta_1 + \theta_2) + l_1^2sin^2\theta_1 + l_2^2sin^2(\theta_1 + \theta_2) + 2l_1l_2sin\theta_1.\sin(\theta_1 + \theta_2)$$

$$P_x^2 + P_y^2 = l_1^2(sin^2\theta_1 + cos^2\theta_1) + l_2^2(sin^2(\theta_1 + \theta_2)+cos^2(\theta_1 + \theta_2)) + 2l_1l_2(cos\theta_1.\cos(\theta_1 + \theta_2) + sin\theta_1.\sin(\theta_1 + \theta_2))$$

3. Berdasarkan persamaan $cos^2\theta + sin^2\theta = 1$

$$P_x^2 + P_y^2 = l_1^2 + l_2^2 + 2l_1l_2(cos\theta_1.\cos(\theta_1 + \theta_2) + sin\theta_1.\sin(\theta_1 + \theta_2))$$

4. Berdasarkan

$\cos(a + b) = \cos a\,.\cos b - \sin a\,.\sin b$ & $\sin(a + b) = \sin a\,.\cos b + \cos a\,.\sin b$

Maka:

$$P_x^2 + P_y^2 = l_1^2 + l_2^2 + 2l_1l_2(cos\theta_1.\cos(\theta_1 + \theta_2) + sin\theta_1.\sin(\theta_1 + \theta_2))$$

$$P_x^2 + P_y^2 = l_1^2 + l_2^2 + 2l_1l_2(cos^2\theta_1.cos\theta_2 - cos\theta_1.sin\theta_1.sin\theta_2 + sin^2\theta_1.cos\theta_2 - sin\theta_1.cos\theta_1.sin\theta_2)$$

$$P_x^2 + P_y^2 = l_1^2 + l_2^2 + 2l_1l_2(cos\theta_2[cos^2\theta_1 + sin^2\theta_1] + sin\theta_1.cos\theta_1.sin\theta_2 - sin\theta_1.cos\theta_1.sin\theta_2)$$

$$P_x^2 + P_y^2 = l_1^2 + l_2^2 + 2l_1l_2cos\theta_2$$

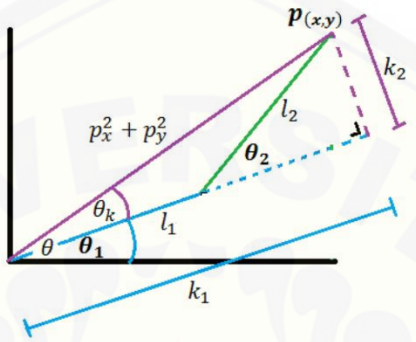
$$2l_1l_2cos\theta_2 = P_x^2 + P_y^2 - l_1^2 - l_2^2$$

$$cos\theta_2 = \frac{P_x^2+P_y^2-l_1^2-l_2^2}{2l_1l_2} \quad \text{.....(2.3)}$$

5. Maka dari cos tetha2 dapat diperoleh yaitu:

$$\theta_2 = acos\left(\frac{Px^2 + Py^2 - l_1^2 - l_2^2}{2l_1 l_2}\right) \quad \text{.....(2.4)}$$

6. Untuk mencari $\theta_1$ maka dapat menggunakan referensi dari gambar berikut:




Gambar 2.6 Grafik Referensi untuk Mencari $\theta_1$

$$k_1 = l_1 + l_2.cos\theta_2$$

$$k_2 = l_2.sin\theta_2$$

$$\theta = atan\left(\frac{Py}{Px}\right)$$

$$\theta = \theta_k + \theta_1$$

$$\theta_1 = \theta - \theta_k$$

$$\theta_1 = atan\left(\frac{Py}{Px}\right) - atan\left(\frac{k_2}{k_1}\right) \quad \text{.....(2.5)}$$

# BAB 3. METODE PENELITIAN

## 3.1 Tempat Dan Waktu Penelitian

Adapun tempat dan waktu penelitian, pengujian dan analisis dilakukan secara umum dilakukan di :

Tempat : Lab. Sistem Kendali Fakultas Teknik Universitas Jember

Alamat : Jl. Slamet Riyadi No. 62, Patrang, Kabupaten Jember

Waktu : 28 Desember 2016 – 31 Juli 2017

## 3.2 Tahapan Pelaksanaan

Untuk mencapai hasil penelitian pada tugas akhir ini, maka akan dilakukan beberapa tahapan sesuai diagram kerja yang telah ada diatas, yakni sebagai berikut:

1. Tahap Studi Literatur

   Tahap ini merupakan tahapan untuk mencari sumber referensi terkait penelitian yang akan dilakukan, dengan informasi yang didapat maka akan menjadi acuan untuk mencapai hasil penelitian.

2. Tahap Pemodelan (Desain) Robot Berkaki Empat

   Setelah didapatkan informasi mengenai bahan, komponen, serta penataan *body* robot yang baik dan benar, langkah selanjutnya adalah pembuatan desain robot berkaki empat.

3. Tahap Pembelian Alat dan Bahan

   Dengan didapatkannya referensi dari beberapa literature pendukung, serta dari data hasil pemodelan, maka akan direncanakan tahap pembelian alat dan bahan yang tentunya sesuai kebutuhan.

4. Tahap Pembuatan/Perakitan Robot Berkaki Empat

   Setelah dilakukan pembuatan desain dengan ukuran-ukuran yang telah ditentukan, maka langkah selanjutnya adalah pembuatan robot berkaki empat.

5. Tahap Pengujian Alat dan Pengambilan Data

   Setelah dibuat robot berkaki empat maka langkah selanjutnya melakukan eksperimen berupa pengujian keseimbangan robot berkaki empat dengan

bidang yang kurang rata. Dengan pengujian tersebut akan didapat data untuk selanjutnya dianalisa.

6. Tahap Analisa Data

   Setelah tahap pengujian dilakukan dan didapatkan data, maka langkah selanjutnya yaitu melakukan analisa terhadap data yang telah diperoleh. Analisa tersebut berfungsi untuk mengetahui berapa besar pengaruh sensor *accelerometer* dengan metode *inverse kinematics* untuk keseimbangan robot berkaki empat.

### 3.3 Alat Dan Bahan

Peralatan yang diperlukan untuk melakukan penelitian ini adalah:

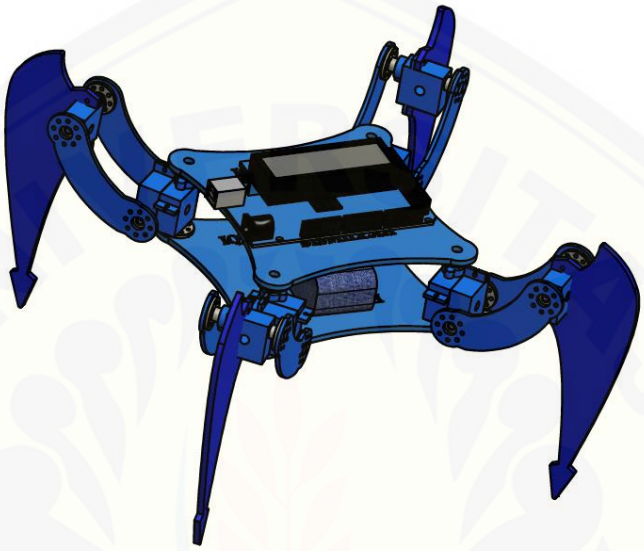
1. Laptop/PC
2. 1 set *Toolbox*
3. Setrika
4. *Multitester*
5. Solder
6. Atraktor
7. Mesin Bor
8. Mesin Gerinda
9. Gergaji
10. Gunting
11. *Cutter*
12. Penggaris
13. Wadah plastik

Bahan-bahan yang diperlukan dalam penelitian ini antara lain adalah:

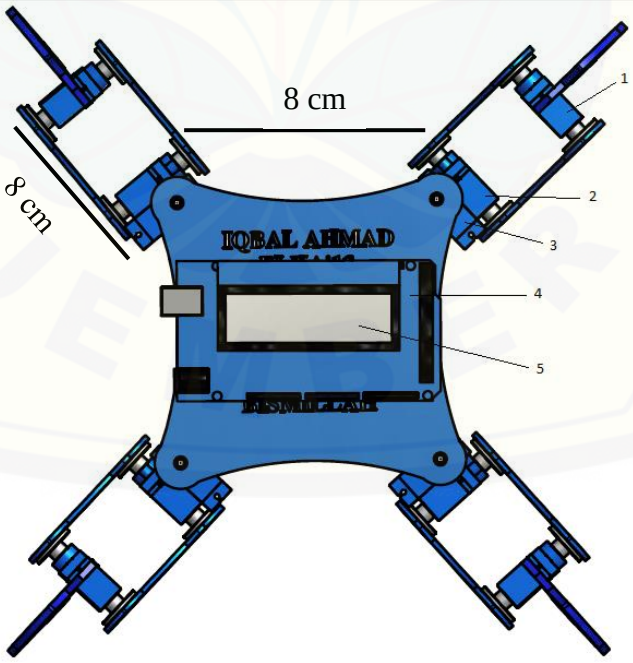
1. Arduino Mega 2560
2. Motor Servo
3. Sensor Accelerometer
4. Catu daya (*Battery*)
5. LCD
6. PCD polos

7. Mika *Acrylic*
8. Kabel Secukupnya
9. Timah

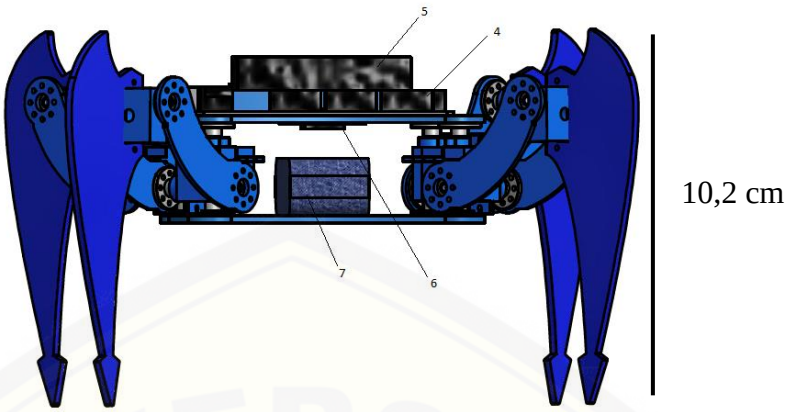
### 3.4 Perancangan *Hardware*



Gambar 3.1 Desain Robot Tampak Belakang Atas




Gambar 3.2 Desain Robot Tampat Atas




Gambar 3.3 Desain Robot Tampak Depan

Keterangan:

1. Servo 1, Aktuator yang fungsinya adalah untuk menggerakkan kaki robot *quadruped* bagian *Tibia.*
2. Servo 2, Aktuator yang fungsinya adalah untuk menggerakkan kaki robot *quadruped* bagian *Femur*.
3. Servo 3, Aktuator yang fungsinya adalah untuk menggerakkan kaki robot *quadruped* bagian *Coxa*.
4. Arduino Mega 2560, Mikrokontroler yang digunakan robot *quadruped* sebagai kontrolernya.
5. LCD, fitur robot *quadruped* untuk menampilkan data sensor juga hasil sudut-sudut yang diperoleh dari proses *inverse kinematics*.
6. Sensor MPU-6050, sensor *accelerometer* yang digunakan untuk *setpoint* robot *quadruped* untuk memperoleh keseimbangan.
7. *Battery* LiPo 12V, sebagai catu daya robot *quadruped*.

Spesifikasi Ukuran Desain:

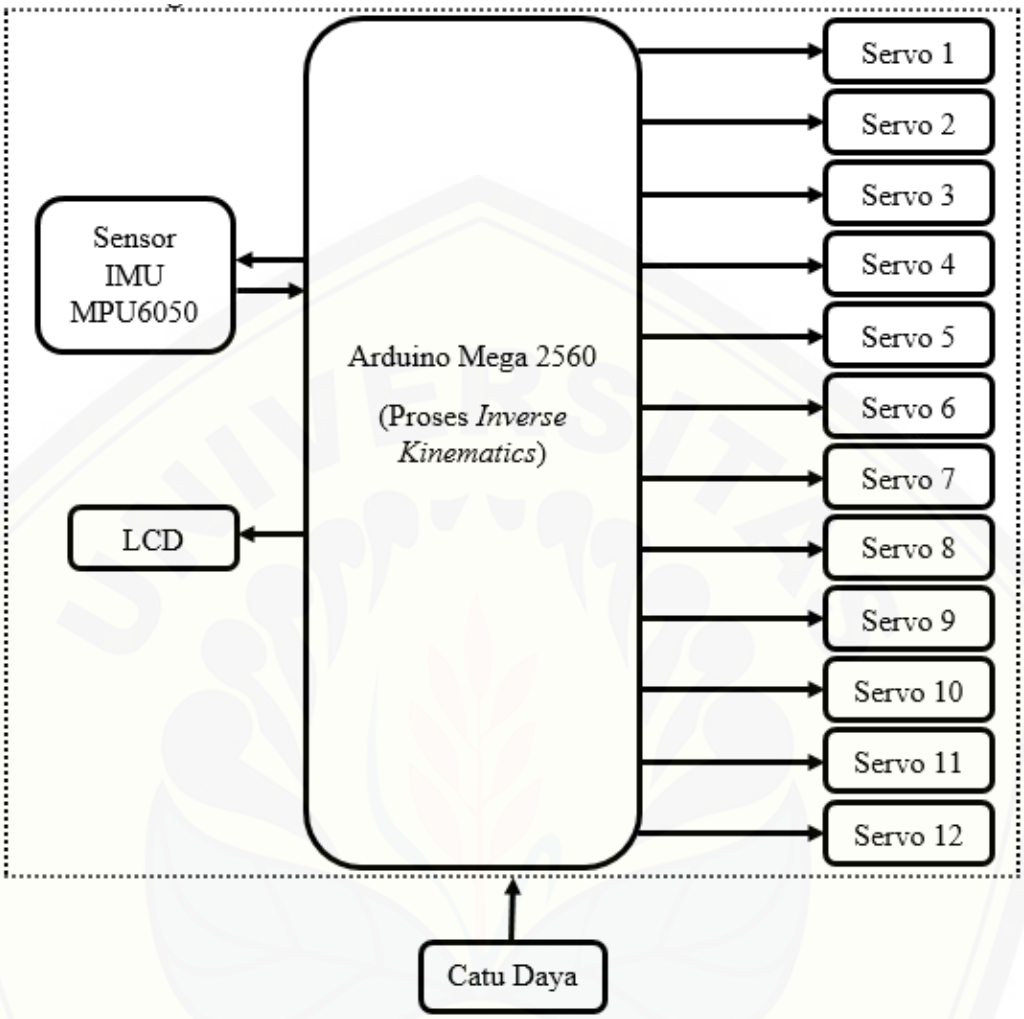
*Body* = 20 cm x 20 cm x 15,05 cm

Kaki *Coxa* = 2 cm

Kaki *Femur* = 8 cm

Kaki *Tibia* = 10,2 cm

### 3.5 Perancangan Elektronika




Gambar 3.4 Perancangan Diagram Blok Harware

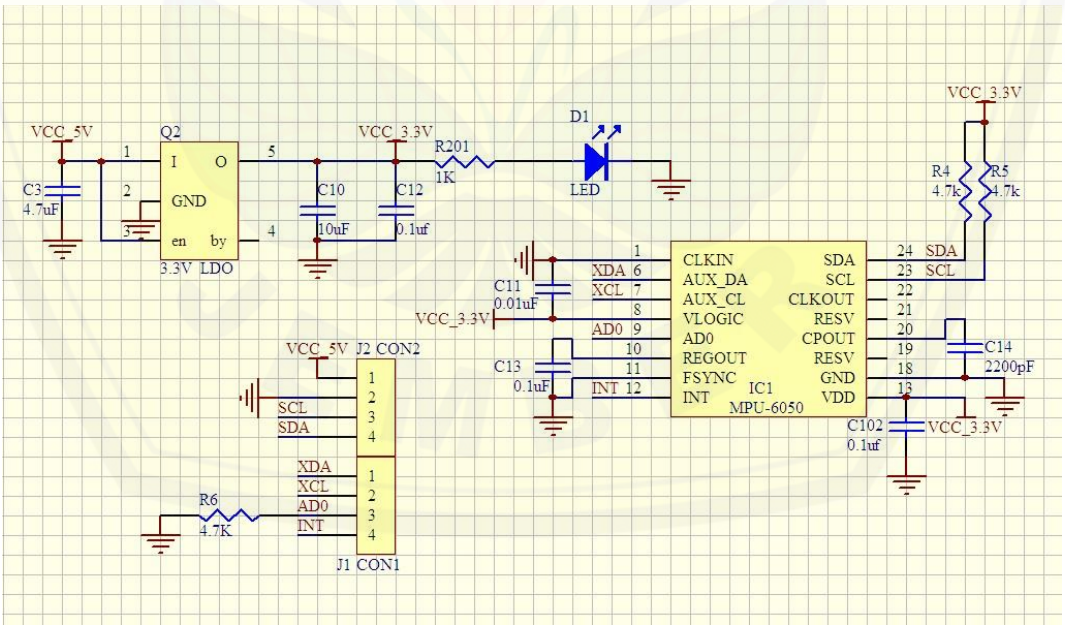
Dari blok diagram pada Gambar 3.4 dapat dilihat bahwa pada sistem ini memiliki 1 nilai *input* sensor yaitu sensor *accelerometer* yang fungsinya adalah sebagai *feedback* atau pengkoreksi dari sistem kestabilan robot, dimana sensor tersebut sebelum menjadi masukan diproses terlebih dahulu yaitu dengan *kalman filter* yang semulanya sensor tersebut kurang stabil nilainya setelah diproses *kalman filter* akan menjadi lebih stabil, sehingga untuk proses selanjutnya atau *feedback* untuk sistem robot diharapkan akan menjadi lebih *smooth*.

Metode yang digunakan dalam robot berkaki empat yang akan dibuat adalah kontrol PID dan juga *inverse kinematics* yaitu dengan nilai error masuk pada proses PID yang kemudian diperoleh nilai $\Delta x$ dan $\Delta y$ sebagai masukan *inverse kinematics*

akan dapat diketahui sudut-sudut servo pada setiap sendi robot berkaki, dengan menggunakan sensor *accelerometer* sebagai nilai *feedback* atau mengetahui posisi robot pada saat itu. Setelah proses *inverse kinematics* selesai dilakukan oleh Arduino Mega 2560, maka akan diperoleh nilai sudut-sudut servo yang diharapkan dapat membuat robot kembali ke titik stabilnya. Sedangkan untuk catu daya (*power supply*) dari robot berkaki empat ini akan menggunakan *battery* LIPO dan masing-masing komponen akan terdapat *regulator* sendiri, hal ini ditujukan agar tidak ada pembatasan arus atau salah satu komponen mengalami kekurangan arus.

1. Rangkaian Sensor MPU-6050

Perancangan robot *quadruped* ini akan menggunakan sensor *accelerometer* jenis MPU-6050 yang merupakan 6-*axis motion processing unit*, karena data keluaran dari sensor ini sangat akurat dengan fasilitas *hardware* internal 16-bit ADC pada setiap kanalnya dan harganya pula juga cukup murah. Dimana untuk komunikasi dengan Arduino Mega 2560 menggunakan pin I2C yaitu SDA (pin 20) dan SCL (pin 21).

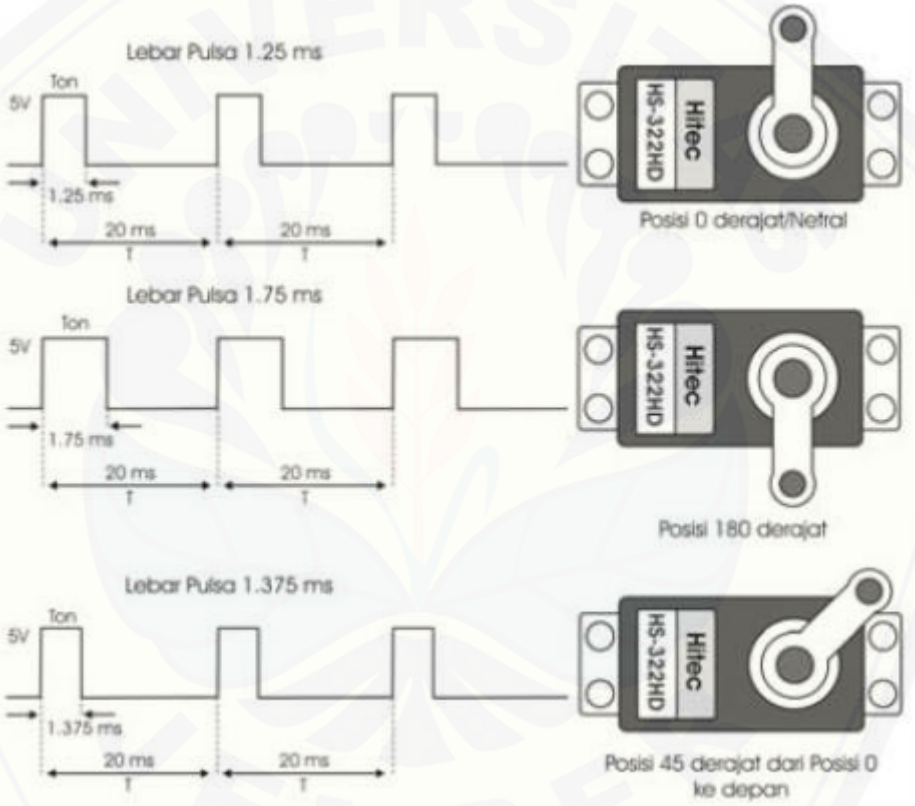



Gambar 3.5 Rangkaian Sensor MPU6050

(*https://www.google.com*)

2. Rangkaian Motor Servo

Dalam perancangan robot *quadruped* ini menggunakan motor servo sebanyak 12 buah karena setiap kaki membutuhkan 3 motor servo (3 DOF), untuk kontrol motor servo membutuhkan pin PWM (*Pulse Wide Modulation*) yaitu pembentuk sinyal kotak, dengan menggunakan Arduino Mega 2560 maka tidak perlu menggunakan driver PWM tambahan karena pada Arduino Mega 2560 memiliki fasilitas pin PWM yang cukup untuk mengontrol motor servo sebanyak 12 buah.

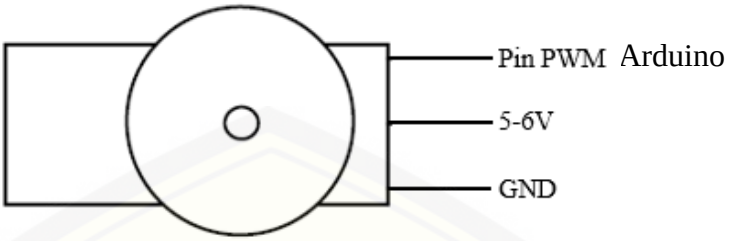



Gambar 3.6 Kendali PWM Motor Servo

(*https://www.google.com*)

Dapat dilihat pada gambar 3.6 di atas merupakan teknik pemberian sinyal PWM untuk mengendalikan motor servo, dengan periode kurang lebih 20 ms, dimana lebar pulsa *HIGH* antara 0,5 ms sampai 2 ms. Apabila motor servo diberikan pulsa dengan besar 1,5 ms maka motor servo akan berputar menujut 90°, jika diberikan pulsa kurang dari 1,5 ms maka motor servo akan berputar mendekati

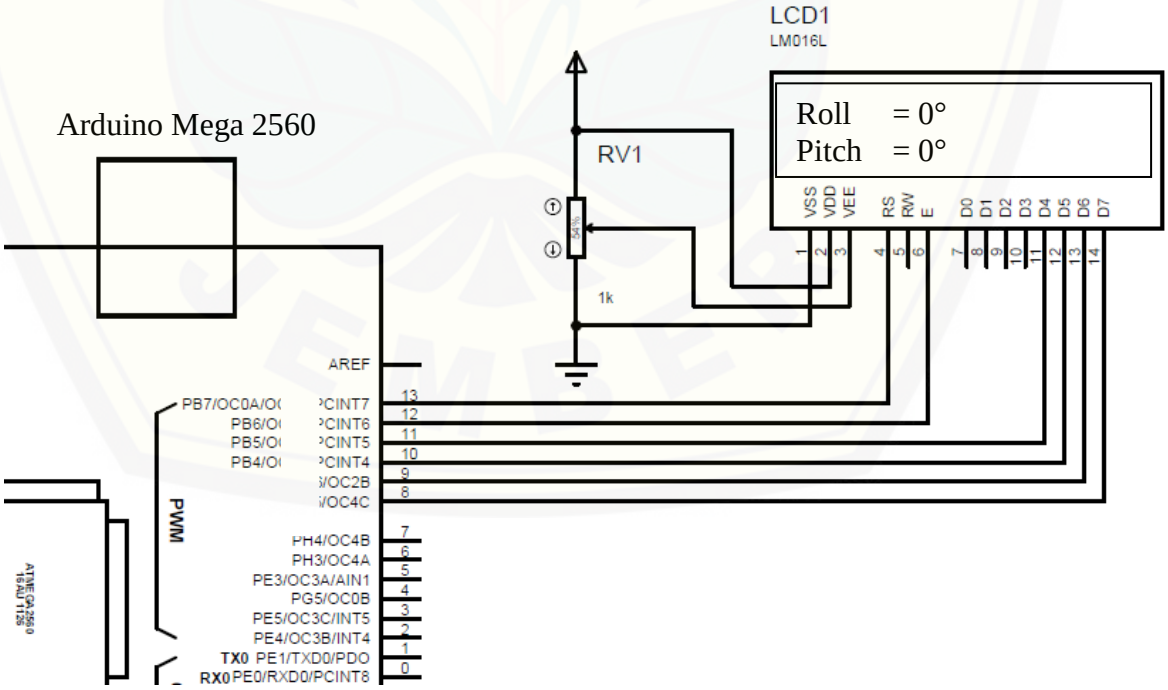
posisi 0°, dan jika diberikan pulsa lebih dari 1,5 ms maka motor servo akan berputar mendekati posisi 180°.




Gambar 3.7 Rangkaian Motor Servo

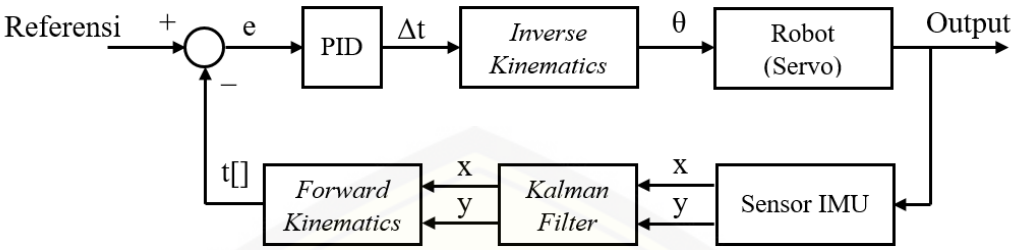
3. Rangkaian LCD

Rangkaian LCD pada perancangan robot *quadruped* ini difungsikan sebagai penampil hasil pembacaan sensor *accelerometer* sudut *roll* dan sudut *pitch*, yaitu menampilkan data pembacaan sensor dan hasil pemrosesan data dari metode *inverse kinematics*. Untuk komunikasi dengan Arduino menggunakan pin digital dan terdapat rangkaian *variable resistor* yang fungsinya yaitu untuk mengatur kecerahan LCD.




Gambar 3.8 Rangkaian LCD

### 3.6 Desain Sistem Kontrol (*Software*)




Gambar 3.9 Diagram Blok Kontroler

t_*error* = t_ref – t ..... (3.1)

Pada diagram blok kontroler diatas menunjukkan bahwa sistem kontrol yang akan digunakan diatas adalah sistem *close loop*, dimana mulai dari sebuah nilai referensi (*SetPoint*) yang telah ditentukan akan menjadi konstanta yang tetap nilainya dan *feedback* menjadi pengurangnya akan diperoleh nilai *error* masuk pada kontrol PID untuk mencari nilai Δt yang akan menjadi masukan pada *inverse kinematics* sebagai metode penggerak motor servo bagian *femur* dan *tibia* pada masing-masing kaki yaitu sudut theta1 untuk *femur* dan theta2 untuk *tibia*. Setelah itu robot akan bergerak sesuai sudut-sudut theta yang diperoleh yang kemudian posisi keseimbangan robot akan dibaca oleh sensor IMU dengan parameter keluaran sensor berupa x dan y. Selanjutnya karena hasil pembacaan dari sensor kurang *linier* atau kurang stabil maka akan dilakukan pemrosesan data yaitu *kalman filter* yang hasilnya tersebut akan menjadi masukan pada proses *forward kinematics* yang gunanya adalah untuk mendapatkan nilai t yang akan menjadi *feedback* untuk nilai referensi. Matematis *inverse kinematics* serta Kontrol PID dalam program Arduino sebagai berikut.

```
t =r*sin(Acc*3.1412/180);
lasterror(t)=error(t);
error(t)=0-t;
Integral=Integral+error(t);
```

```
delta_t=(KP*error(t))+(KI*Integral)+(KD*(error(t)-
lasterror(t)));
tinggi=tinggi+t;
L0=sqrt(pow(tinggi,2) + pow((M-coxa),2));
a1=acos(y[a]/L[a]);
a1=a1*57296/1000;
double tib=pow(tibia,2);
double fem=pow(femur,2);
double L=pow(L0,2);
a2=acos((tib-fem-LL[a])/(-2*femur*L[a]));
a2=a2*57296/1000;
a=a1+a2;
B=acos((L-tib-fem)/(-2*tibia*femur));
B=B*57296/1000;
```

### 3.7.1 *Kalman Filter*

*Kalman Filter* merupakan suatu metode yang fungsinya adalah digunakan untuk melakukan estimasi suatu nilai atau hasil pembacaan sensor IMU pada penelitian ini, karena pengukuran data hasil sensor IMU yang akan saya gunakan terus berubah-ubah nilainya (tidak linier), maka dengan menggunakan *Kalman Filter* akan dapat mengestimasi *state* dari sistem dinamis dengan tipe kebiasaan acak tertentu dengan menggunakan informasi statistik. Konsep dari kalman filter yang saya pakai pada penelitian ini adalah sebagai berikut:

*Prediction Update*

$$Pk = Pk + Q \quad \text{.....(3.2)}$$

*Measurement Update*

$$Kk = \frac{Pk}{Pk+R} \quad \text{.....(3.3)}$$

$$Xk = Xk + Kk * (Zk - Xk) \quad \text{.....(3.4)}$$

$$Pk = (1 - Kk) * Pk \quad \text{.....(3.5)}$$

Keterangan:

Pk = *Estimated Error*

Q = *Process Noise*

Kk = *Kalman Gain*

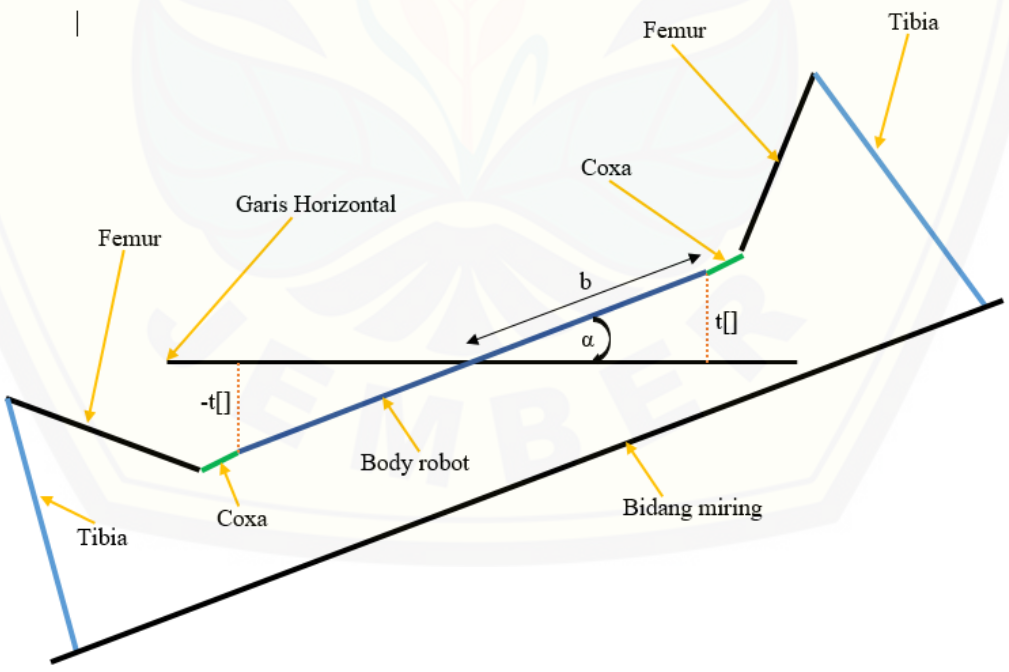
R = *Measurement Noise*

Xk = *Filtered Value*

Zk = *Measurement Sensor*

Pada persamaan rumus di atas diketahui bahwa pada metode kalman filter terdapat dua proses yaitu *Prediction Update* (*update* waktu) yang tugasnya adalah untuk mendapatkan nilai *pra*-estimasi untuk waktu step selanjutnya. Dan *Measurement Update* (*update* pengukuran) yang tugasnya adalah untuk keperluan umpan balik, yaitu memadukan hasil pengukuran terbaru dengan nilai *pra*-estimasi untuk memperoleh nilai *pasca*-estimasi yang lebih baik.

### 3.7.2 *Forward Kinematics*




Gambar 3.10 *Body* robot pada bidang miring

Dalam bab 2 sebelumnya telah dijelaskan bahwa fungsi dari metode ini adalah untuk mencari nilai titik yang diperoleh dengan perumusan matematika

dimana panjang lengan serta sudut *body* terhadap bidang horizontal yang diperoleh dari sensor *accelerometer* sebagai *input* dari *forward kinematics*. Pada robot yang akan dibuat metode ini digunakan untuk mencari t yaitu perubahan tinggi pada ujung *body* terhadap bidang horizontal dengan menggunakan persamaan sebagai berikut:

$$\text{Sin } \alpha = \frac{t}{b}$$

$$t = b \text{ x sin } \alpha \quad \text{.....(3.6)}$$

Keterangan:

α = Sudut lengan terhadap garis horizontal diperoleh dari pembacaan sensor *accelerometer*

t = Perubahan tinggi pada ujung *body* terhadap bidang horizontal

b = Setengah panjang *body* robot

### 3.7.3 Kontrol PID

PID (*Proportional - Integral - Derivative*) merupakan suatu kontroler untuk menentukan kepresisian sistem dengan adanya umpan balik pada sistem tersebut. Kontrol PID sendiri terdiri dari *Proportional* (P), *Integral* (I), dan *Derivative* (D), sehingga pada sistem yang akan dibuat dengan menggunakan kontrol PID dimana dari nilai *SetPoint* yang telah ditentukan yaitu sudut 0° dengan dikurangi nilai *feedback* dari sensor IMU dan metode *forward kinematics* berupa *output* t menjadi nilai *error* sebagai masukan kontrol PID, disinilah fungsi kontrol PID yaitu mempertahankan nilai *SetPoint* pada posisi nol atau keadaan *body* robot seimbang dengan keluaran Δt untuk setiap kaki robot dengan metode *invers kinematics.* Dengan menggunakan perumusan sebagai berikut:

$$\text{LastError\_t} = \text{Error\_t}$$

$$\text{Error\_t} = \text{SP} - \text{t}$$

$$\text{Integral\_t} = \text{Integral\_t} + \text{Error\_t}$$

$$\text{Derivative\_t} = \text{Error\_t} - \text{LastError\_t}$$

Kontrol *Proportional*:

$$P = K_p * Error\_t \quad (3.7)$$

Kontrol *Integral*

$$I = K_i * Integral\_t \quad (3.8)$$

Kontrol *Derivative*

$$D = K_d * Derivative\_t \quad (3.9)$$

Kontrol PID

$$PID = K_p * Error\_t + K_i * Integral\_t + K_d * Derivative\_t$$

Sehingga pada sistem yang akan dibuat diperoleh perumusan sebagai berikut:

Rumus Δt

$$\Delta t = K_p * Error\_t + K_i * Integral\_t + K_d * Derivative\_t \quad (3.10)$$

Keterangan :

LastError_t = nilai *Error* terakhir

Error_t = Perbedaan *setpoint* dengan pembacaan sensor (*feedback*)

SP = konstanta dari *setpoint* yang telah ditentukan

t = Perubahan tinggi pada ujung *body* terhadap bidang horizontal

Integral_t = Akumulator atau penjumlahan Error_t

P = *output* Kontrol *Proportional*

I = *output* kontrol *Integral*

D = *output* kontrol *Derivative*

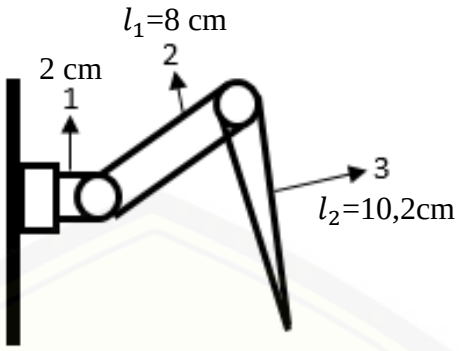
$K_p$ = Konstanta *Proportional*

$K_i$ = Konstanta *Integral*

$K_d$ = Konstanta *Derivative*

PID/Δt = nilai *output* kontrol PID

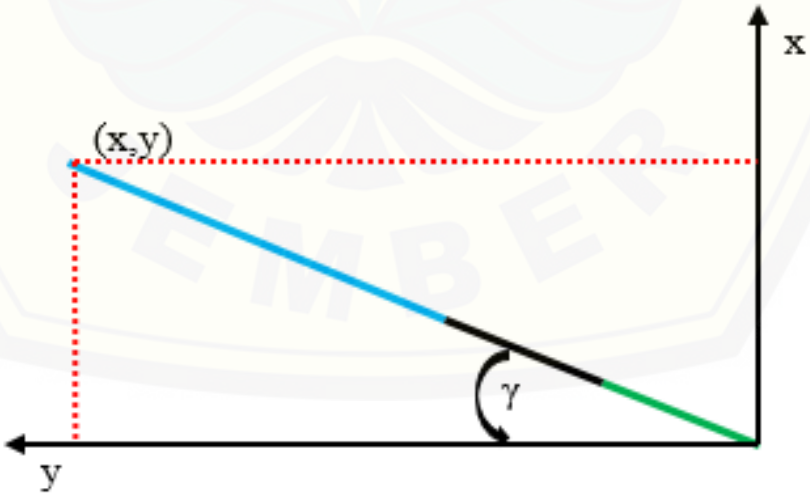
### 3.7.4 *Inverse Kinematics*




Gambar 3.11 Kaki Robot, (1) *coxa*, (2) *femur*, (3) *tibia*

Dalam bab 2 sebelumnya telah dijelaskan bahwa fungsi *Inverse Kinematics* pada robot berkaki adalah sebagai sistem penggerak motor servo di tiap-tiap sendi secara otomatis sehingga tidak perlu mengatur secara manual, pada penelitian ini robot yang akan dibuat akan menggunakan 2 *inverse kinematics* yaitu *inverse kinematics* untuk robot berjalan dan *inverse kinematics* untuk keseimbangan robot dimana untuk keseimbangan ada penambahan sensor IMU sebagai *feedback* atau pengkoreksi.

a. *Inverse Kinematics* Jalan




Gambar 3.12 Grafik Representasi untuk mencari sudut jalan kaki robot

Pada Gambar 3.12 dapat dilihat bahwa *inverse kinematics* yang digunakan adalah 1-DOF dengan nilai x adalah jarak langkah robot dalam satuan cm, sedangkan nilai y adalah jarak titik tumpu kaki robot dari bidang sampai *body* robot, dengan menggunakan rumus trigonometri maka akan diperoleh nilai sudut langkah untuk pergerakan pada motor servo menggunakan masukan yaitu jarak langkah yang kita inginkan dan jarak titik tumpu kaki sampai dengan *body* robot, dengan perumusan sebagai berikut:

$$\gamma = tan^{-1}\frac{x}{y} \quad \text{.....(3.11)}$$

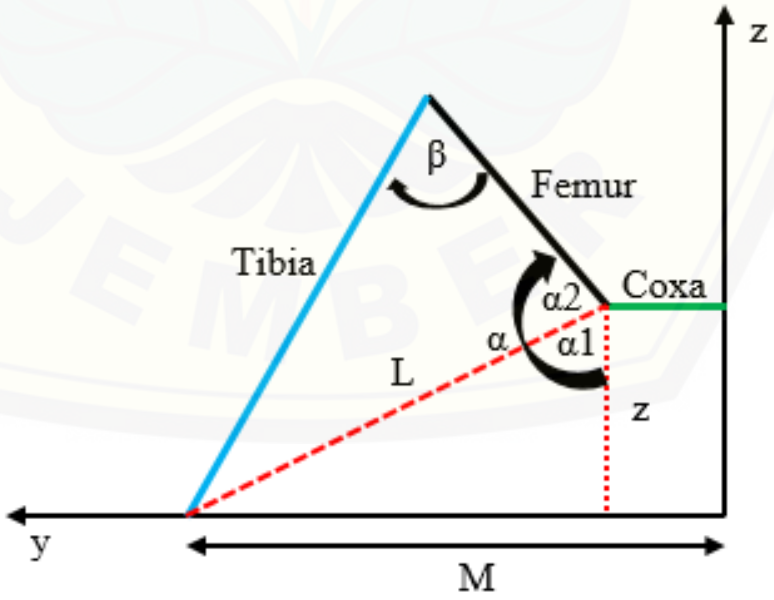
Keterangan:

γ = Sudut langkah robot

x = Jarak langkah robot (cm)

y = Jarak titik tumpu sampai *body*

b. *Inverse Kinematics* Kestabilan




Gambar 3.13 Grafik Representasi untuk mencari sudut keseimbangan robot

Pada *inverse kinematics* kestabilan ini digunakan untuk mencari nilai sudut lengan *femur* (α) dan lengan *tibia* (β) yang diperoleh dengan masukan berupa ketinggian robot, panjang lengan *femur*, panjang lengan *tibia*. Untuk memperoleh kestabilan parameter z atau ketinggian robot akan memperoleh masukan dari perhitungan *forward kinematics* berupa t dapat dilihat pada Gambar 3.9, dengan penggabungan ini akan diperoleh kestabilan robot dimana nilai referensi yang telah ditentukan yaitu robot seimbang ketika:

Sudut X = 0

Sudut Y = 0

Keadaan tersebut menyatakan bahwa robot dalam keadaan seimbang, sedangkan jika nilai sudut X dan sudut Y diatas tidak sama dengan nol maka akan masuk dalam proses *forward kinematics* sebagai *feedback* untuk kontrol PID yang diperoleh keluaran berupa Δt sebagai masukan *inverse kinematics*, sehingga akan diperoleh sudut terbaru untuk masukan motor servo yaitu bagian *femur* dan *tibia*. Dengan mengacu rumus pada landasan teori bab 2 dapat diperoleh perumusan sebagai berikut:

z = z + t

$$L = \sqrt[2]{z^2 + (M - coxa)^2}$$

$$\alpha_1 = cos^{-1}\frac{z}{L}$$

$$\alpha_2 = cos^{-1}\frac{tibia^2 - femur^2 - L^2}{-2(femur)(L)}$$

$$\alpha = \alpha_1 + \alpha_2 \quad \text{.....(3.12)}$$

$$\beta = cos^{-1}\frac{L^2 - tibia^2 - femur^2}{-2(tibia)(femur)} \quad \text{.....(3.13)}$$

Keterangan :

z = tinggi body robot

t = Perubahan tinggi pada ujung *body* terhadap bidang horizontal

L = Jarak titik tumpu robot terhadap lengan *coxa*

M = Jarak titik tumpu sampai *body* robot

Coxa = Panjang lengan *coxa*

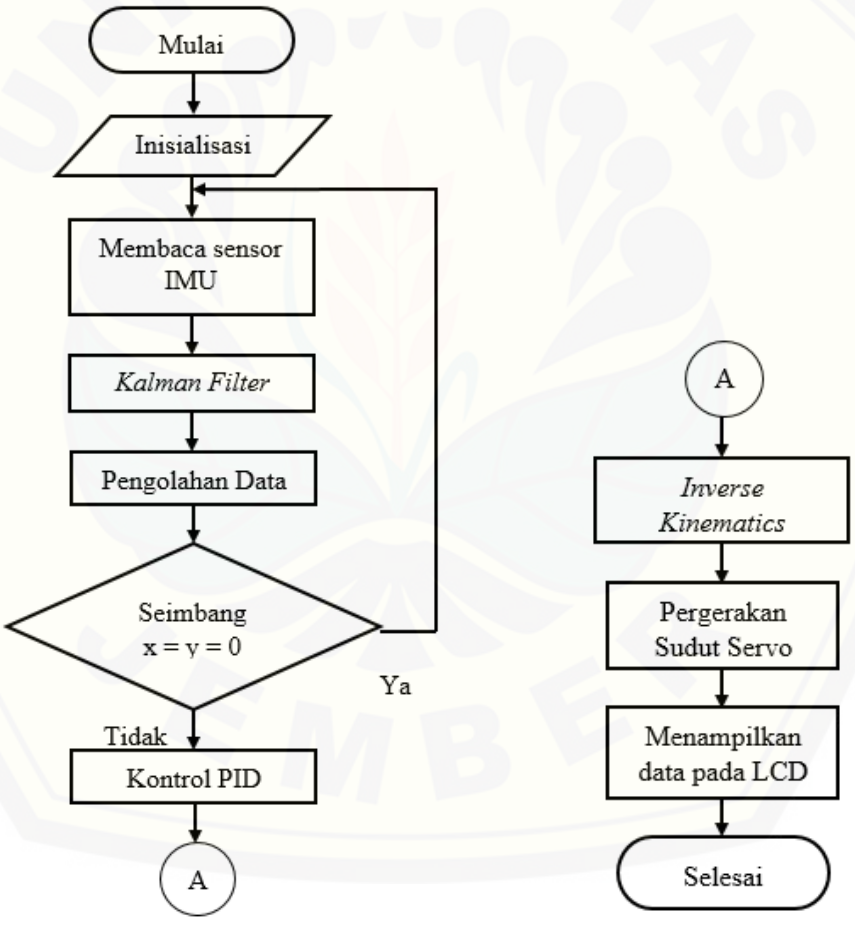
Femur = Panjang lengan *femur*

Tibia = Panjang lengan *tibia*

α = Sudut total lengan *femur*

β = Sudut lengan *tibia*

### 3.7.5 *Flowchart*

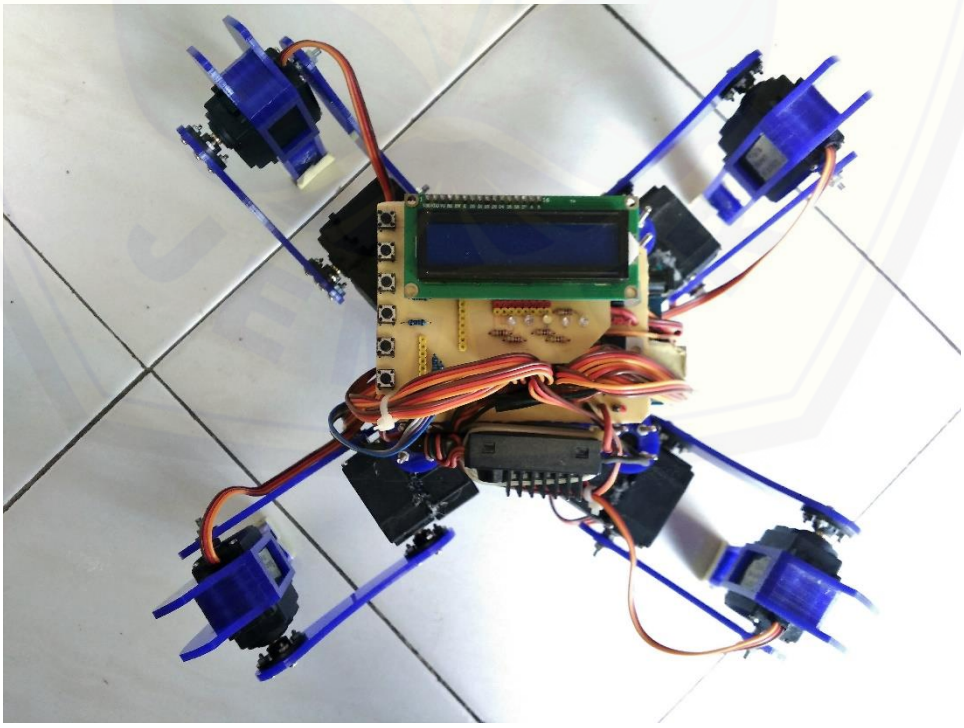



Gambar 3.14 *Flowchart* sistem robot berkaki empat

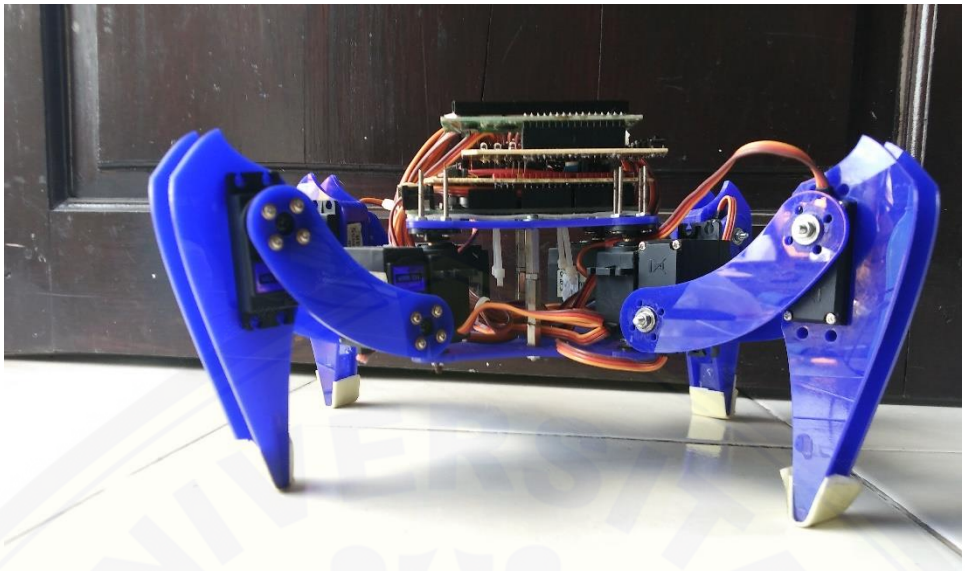
Pada diagram alir (*Flowchart*) di atas yaitu Gambar 3.14 adalah sistem dari robot berkaki empat dari Mulai (*Start*) yang kemudian masuk proses inisialisasi

*variable* dan nilai konstanta yang selanjutnya akan masuk pada blok *input* dimana pada proses tersebut akan ada pengambilan data sensor *accelerometer* yang selanjutnya masuk pada proses *kalman filter* dimana dalam proses tersebut pembacaan sensor yang naik turun yang kurang stabil akan dilakukan pem-*filter*-an sehingga naik turunnya sensor akan menjadi konstan. Setelah itu akan diketahui posisi robot sudah stabil apa belum, jika sudah stabil maka akan kembali ke proses pembacaan sensor (*looping*), dan jika belum stabil maka akan masuk ke proses *inverse kinematics* dan nilai hasil dari filter sensor *accelerometer* akan menjadi nilai referensi atau *setpoint* pada proses tersebut. Pada proses *inverse kinematics* data referensi dari sensor *accelerometer* sebagai penentu posisi robot *quadruped* dan akan dilakukan proses perhitungan *inverse kinematics* sehingga akan didapatkan sudut-sudut yang dibutuhkan servo pada masing-masing sendi pada kaki robot *quadruped* yang akan membuat stabil pergerakan badan robot. Selanjutnya hasil pembacaan sensor accelerometer dan hasil dari proses *inverse kinematics* akan tertulis pada LCD dan proses selesai.

## 3.7 Hasil Jadi Robot *Quadruped*



Gambar 3.15 Tampilan atas Robot *Quadruped* yang telah dibuat



Gambar 3.16 Tampilan depan Robot *Quadruped* yang telah dibuat

Dari hasil pembuatan robot quadruped pada Gambar 3.15 dan Gambar 3.16 telah mengalami perubahan dari perancangan desain sebelumnya karena pada robot *quadruped* yang baru ini menggunakan motor servo yang lebih besar yaitu yang sebelumnya menggunakan MG90S dan yang sekarang menggunakan MG996R dengan ukuran yang lebih besar. Pergantian komponen aktuator ini dikarenakan kurangnya torsi atau kekuatan motor servo MG90S dalam menjalankan robot quadruped yang dibuat dengan beban yang cukup berat.

## 3.8 Proses Kalibrasi Sensor dan Aktuator

### 3.8.1 Kalibrasi Sensor MPU6050

Sensor MPU6050 merupakan salah satu jenis sensor IMU yang terdapat 6-*axis*. Namun sebelum sensor ini dapat digunakan pada robot *quadruped*, maka sensor *accelerometer* harus dilakukan proses kalibrasi untuk memperoleh nilai berupa sudut karena keluaran asli pada sensor *accelerometer* adalah nilai percepatan terhadap gravitasi bumi, maka diperlukan proses matematis untuk

mendapat nilai sudut kemiringan, untuk lebih jelasnya dapat dilihat rumus perhitungan sebagai berikut:

$$\text{Sudut X} = \text{Atan}\left(\frac{A_y}{\sqrt{A_x^2+A_z^2}}\right) \quad \text{.....(3.14)}$$

$$\text{Sudut Y} = \text{Atan}\left(\frac{-A_x}{\sqrt{A_y^2+A_z^2}}\right) \quad \text{.....(3.15)}$$

Dari proses perhitungan dengan menggunakan perumusan diatas dilakukan proses kalibrasi dengan menggunakan papan, dimana sensor *accelerometer* ditempelkan pada papan tersebut, setelah itu papan dimiringkan dari sudut -35° sampai dengan sudut 35° pada sumbu X dan sumbu Y setelah itu sudut yang ditampilkan pada LCD dibandingkan dengan hasil pembacaan dengan busur, maka diperoleh nilai *error* pembacaan sudut kemiringan dengan sensor *accelerometer*, dapat dilihat pada tabel 3.1.

Tabel 3.1 Pembacaan sensor MPU6050 sebelum proses kalibrasi

| No | (Busur) Sudut *Roll* | Keluaran Sensor | | | (Busur) Sudut *Pitch* | Keluaran Sensor | | | Hasil Perhitungan | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | AccX | AccY | AccZ | | AccX | AccY | AccZ | Sudut *Roll* | Sudut *Pitch* |
| 1 | -35 | 488 | -9160 | 12987 | -35 | 9243 | -324 | 12976 | -35,1 | -35,4 |
| 2 | -25 | 536 | -6712 | 13944 | -25 | 6516 | -300 | 13960 | -25,6 | -25,0 |
| 3 | -20 | 620 | -5292 | 14492 | -20 | 5360 | -324 | 14564 | -20,0 | -20,2 |
| 4 | -15 | 654 | -4048 | 14964 | -15 | 4116 | -216 | 14964 | -15,1 | -15,3 |
| 5 | -10 | 536 | -2716 | 15204 | -10 | 2872 | -260 | 15384 | -10,1 | -10,5 |
| 6 | -7 | 596 | -1988 | 15284 | -7 | 1926 | -56 | 15212 | -7,4 | -7,21 |
| 7 | -5 | 576 | -1444 | 15376 | -5 | 1491 | -20 | 15520 | -5,36 | -5,48 |
| 8 | -3 | 484 | -892 | 15484 | -3 | 936 | -92 | 15536 | -3,29 | -3,44 |
| 9 | 0 | 596 | 0 | 15516 | 0 | 74 | 5 | 15468 | 0 | -0,27 |
| 10 | 3 | 628 | 934 | 15432 | 3 | -836 | 12 | 15376 | 3,46 | 3,11 |
| 11 | 5 | 648 | 1412 | 15360 | 5 | -1359 | 124 | 15472 | 5,24 | 5,01 |
| 12 | 7 | 612 | 1994 | 15372 | 7 | -1880 | 84 | 15432 | 7,38 | 6,94 |
| 13 | 10 | 696 | 2812 | 15288 | 10 | -2652 | 188 | 15008 | 10,41 | 10,02 |
| 14 | 15 | 792 | 4056 | 14808 | 15 | -4059 | 32 | 14800 | 15,29 | 15,33 |
| 15 | 20 | 916 | 5320 | 14384 | 20 | -5298 | 40 | 14428 | 20,25 | 20,16 |
| 16 | 25 | 1024 | 6639 | 13944 | 25 | -6424 | 156 | 13744 | 25,4 | 25,05 |
| 17 | 35 | 1320 | 9451 | 13212 | 35 | -8924 | 80 | 12492 | 35,44 | 35,54 |

Dari data yang diperoleh pembacaan sensor MPU6050 pada tabel 3.1 bukanlah sudut *roll* atau sudut X dan sudut *pitch* sudut Y, melainkan nilai percepatan terhadap gravitasi bumi. Dengan perubahan sudut *roll* pada data keluaran *accelerometer* yang berubah secara linear adalah data AccY sedangkan data AccX dan AccZ cenderung tetap atau tidak ada perubahan, dan pada perubahan sudut *pitch* pada data keluaran *accelerometer* yang berubah secara linear adalah data AccX sedangkan data AccY dan AccZ cenderung tetap atau tidak ada berubah.

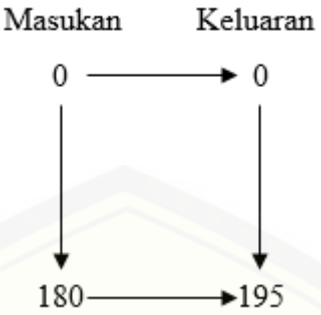
### 3.8.2 Servo MG996R

Servo MG996R merupakan motor servo yang memiliki torsi lumayan besar yaitu sampai 11KG dengan harga yang cukup terjangkau, dengan torsi yang besar tersebut dirasa cukup mumpuni untuk digunakan sebagai aktuator pada robot *quadruped*, namun ada kekurangan pada motor servo ini yaitu belum terdapat kontrol di dalamnya sehingga perlu adanya proses kalibrasi pada seluruh motor servo yang akan digunakan sebelum digunakan pada robot *quadruped* yaitu dengan mencoba memberikan masukan sudut pada servo dengan step sudut dari 0 sampai dengan 180 derajat. Lalu dilihat hasil putaran motor servo menggunakan alat pengukur sudut, dengan begitu akan diperoleh data sudut putar motor servo yang sesungguhnya dan selanjutnya data tersebut akan digunakan untuk proses kalibrasi motor servo dengan menggunakan rumus matematika. Untuk lebih jelasnya dapat dilihat proses perumusan matematika berikut:

Spesifikasi Motor Servo MG996R

| | |
|---|---|
| Torsi | = 4,8 V – 9,4 KGcm |
| | 6,0 V – 11 KGcm |
| Speed | = 4,8 V – 0,17 s/60° |
| | 6,0 V – 0,14 s/60° |
| Berat | = 55 gram |
| Dimensi | = Panjang – 40,6 mm |
| | Lebar – 19,8 mm |
| | Tinggi – 42,9 mm |
| Tipe Gear | = Metal |

Proses kalibrasi servo bagian *coxa* depan kanan:




Gambar 3.17 Hubungan sudut masukan dengan keluaran servo *coxa* depan kanan

Dapat dilihat pada gambar 3.17 nilai masukan servo dengan keluaran servo tidak sesuai, dimana masukan dari 0° sampai dengan 180° diperoleh keluaran putaran motor servo sebesar 0° sampai dengan 195°. maka diperlukan proses kalibrasi dengan menggunakan rumus matematika untuk penyesuaian antara masukan, keluaran dengan sudut yang diinginkan sesuai sebagai berikut:

$$\Delta\gamma_1 = \frac{180\ x\ \gamma_1}{195} \quad \text{.....(3.16)}$$

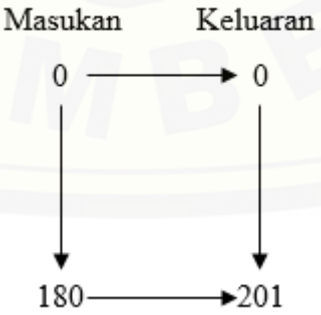
Keterangan:

195 = Sudut keluaran motor servo ketika diberi masukan maksimal

$\gamma_1$ = Sudut *coxa* hasil perhitungan *inverse kinematics* jalan (gambar 3.11)

$\Delta\gamma_1$ = Sudut hasil proses kalibrasi

Proses kalibrasi servo bagian *coxa* belakang kanan:




Gambar 3.18 Hubungan sudut masukan dan keluaran servo *coxa* belakang kanan

Dapat dilihat pada gambar 3.18 nilai masukan servo dengan keluaran servo tidak sesuai, dimana masukan dari 0° sampai dengan 180° diperoleh keluaran putaran motor servo sebesar 0° sampai dengan 201°. maka diperlukan proses kalibrasi dengan menggunakan rumus matematika untuk penyesuaian antara masukan, keluaran dengan sudut yang diinginkan sesuai sebagai berikut:

$$\Delta\gamma_2 = \frac{180\ x\ \gamma_2}{201} \quad \text{.....(3.17)}$$

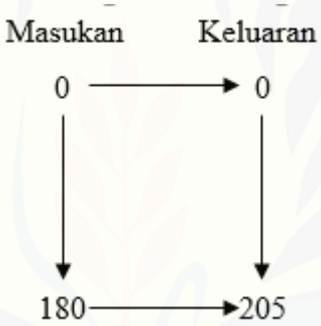
Keterangan:

201 = Sudut keluaran motor servo ketika diberi masukan maksimal

$\gamma_2$ = Sudut *coxa* hasil perhitungan *inverse kinematics* jalan (gambar 3.11)

$\Delta\gamma_2$ = Sudut hasil proses kalibrasi

Proses kalibrasi servo bagian *coxa* depan kiri:




Gambar 3.19 Hubungan sudut masukan dan keluaran servo *coxa* depan kiri

Dapat dilihat pada gambar 3.19 nilai masukan servo dengan keluaran servo tidak sesuai, dimana masukan dari 0° sampai dengan 180° diperoleh keluaran putaran motor servo sebesar 0° sampai dengan 205°. maka diperlukan proses kalibrasi dengan menggunakan rumus matematika untuk penyesuaian antara masukan, keluaran dengan sudut yang diinginkan sesuai sebagai berikut:

$$\Delta\gamma_3 = \frac{180\ x\ \gamma_3}{205} \quad \text{.....(3.18)}$$

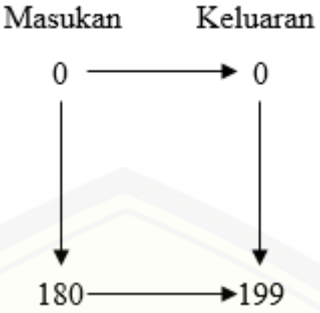
Keterangan:

205 = Sudut keluaran motor servo ketika diberi masukan maksimal

$\gamma_3$ = Sudut *coxa* hasil perhitungan *inverse kinematics* jalan (gambar 3.11)

$\Delta\gamma_3$ = Sudut hasil proses kalibrasi

Proses kalibrasi servo bagian *coxa* belakang kiri:




Gambar 3.20 Hubungan sudut masukan dan keluaran servo *coxa* belakang kiri

Dapat dilihat pada gambar 3.20 nilai masukan servo dengan keluaran servo tidak sesuai, dimana masukan dari 0° sampai dengan 180° diperoleh keluaran putaran motor servo sebesar 0° sampai dengan 199°. maka diperlukan proses kalibrasi dengan menggunakan rumus matematika untuk penyesuaian antara masukan, keluaran dengan sudut yang diinginkan sesuai sebagai berikut:

$$\Delta\gamma_4 = \frac{180\ x\ \gamma_4}{199} \quad \text{.....(3.19)}$$

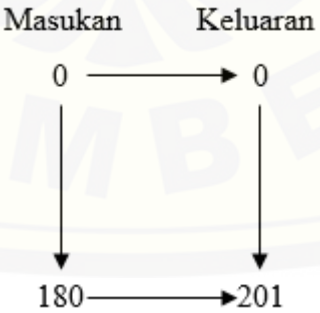
Keterangan:

199 = Sudut keluaran motor servo ketika diberi masukan maksimal

$\gamma_4$ = Sudut *coxa* hasil perhitungan *inverse kinematics* jalan (gambar 3.11)

$\Delta\gamma_4$ = Sudut hasil proses kalibrasi

Proses kalibrasi servo bagian *femur* depan kanan:




Gambar 3.21 Hubungan sudut masukan dan keluaran servo *femur* depan kanan

Dapat dilihat pada gambar 3.21 nilai masukan servo dengan keluaran servo tidak sesuai, dimana masukan dari 0° sampai dengan 180° diperoleh keluaran

putaran motor servo sebesar 0° sampai dengan 201°. maka diperlukan proses kalibrasi dengan menggunakan rumus matematika untuk penyesuaian antara masukan, keluaran dengan sudut yang diinginkan sesuai sebagai berikut:

$$\Delta\alpha_1 = \frac{180\ x\ \alpha_1}{201} \quad \text{.....(3.20)}$$

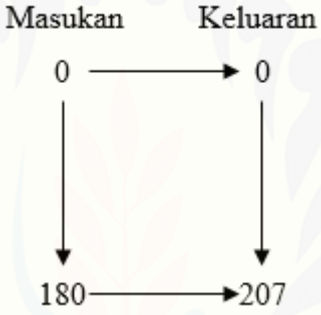
Keterangan:

201 = Sudut keluaran motor servo ketika diberi masukan maksimal

$\alpha_1$ = Sudut *femur* hasil perhitungan *inverse kinematics* kestabilan (gambar 3.12)

$\Delta\alpha_1$ = Sudut hasil proses kalibrasi

Proses kalibrasi servo bagian *femur* belakang kanan:




Gambar 3.22 Hubungan sudut masukan dan keluaran servo *femur* belakang kanan

Dapat dilihat pada gambar 3.22 nilai masukan servo dengan keluaran servo tidak sesuai, dimana masukan dari 0° sampai dengan 180° diperoleh keluaran putaran motor servo sebesar 0° sampai dengan 207°. maka diperlukan proses kalibrasi dengan menggunakan rumus matematika untuk penyesuaian antara masukan, keluaran dengan sudut yang diinginkan sesuai sebagai berikut:

$$\Delta\alpha_2 = \frac{180\ x\ \alpha_2}{207} \quad \text{.....(3.21)}$$

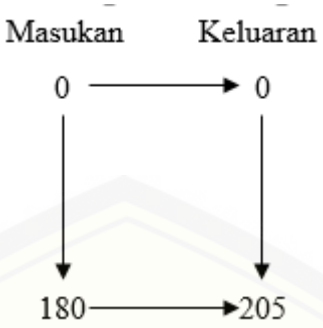
Keterangan:

207 = Sudut keluaran motor servo ketika diberi masukan maksimal

$\alpha_2$ = Sudut *femur* hasil perhitungan *inverse kinematics* kestabilan (gambar 3.12)

$\Delta\alpha_2$ = Sudut hasil proses kalibrasi

Proses kalibrasi servo bagian femur depan kiri:




Gambar 3.23 Hubungan sudut masukan dan keluaran servo *femur* depan kiri

Dapat dilihat pada gambar 3.23 nilai masukan servo dengan keluaran servo tidak sesuai, dimana masukan dari 0° sampai dengan 180° diperoleh keluaran putaran motor servo sebesar 0° sampai dengan 205°. maka diperlukan proses kalibrasi dengan menggunakan rumus matematika untuk penyesuaian antara masukan, keluaran dengan sudut yang diinginkan sesuai sebagai berikut:

$$\Delta\alpha_3 = \frac{180\ x\ \alpha_3}{205} \quad \text{.....(3.22)}$$

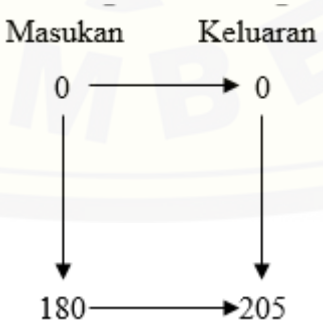
Keterangan:

205 = Sudut keluaran motor servo ketika diberi masukan maksimal

$\alpha_3$ = Sudut *femur* hasil perhitungan *inverse kinematics* kestabilan (gambar 3.12)

$\Delta\alpha_3$ = Sudut hasil proses kalibrasi

Proses kalibrasi servo bagian *femur* belakang kiri:




Gambar 3.24 Hubungan sudut masukan dan keluaran servo *femur* belakang kiri

Dapat dilihat pada gambar 3.24 nilai masukan servo dengan keluaran servo tidak sesuai, dimana masukan dari 0° sampai dengan 180° diperoleh keluaran putaran motor servo sebesar 0° sampai dengan 205°. maka diperlukan proses kalibrasi dengan menggunakan rumus matematika untuk penyesuaian antara masukan, keluaran dengan sudut yang diinginkan sesuai sebagai berikut:

$$\Delta\alpha_4 = \frac{180\ x\ \alpha_4}{205} \quad \text{.....(3.23)}$$

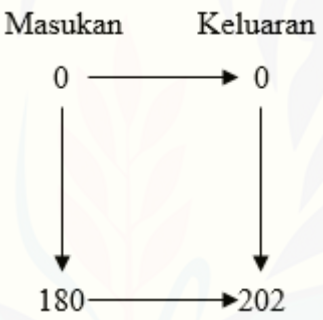
Keterangan:

205 = Sudut keluaran motor servo ketika diberi masukan maksimal

$\alpha_4$ = Sudut *femur* hasil perhitungan *inverse kinematics* kestabilan (gambar 3.12)

$\Delta\alpha_4$ = Sudut hasil proses kalibrasi

Proses kalibrasi servo bagian *tibia* depan kanan:




Gambar 3.25 Hubungan sudut masukan dan keluaran servo *tibia* depan kanan

Dapat dilihat pada gambar 3.25 nilai masukan servo dengan keluaran servo tidak sesuai, dimana masukan dari 0° sampai dengan 180° diperoleh keluaran putaran motor servo sebesar 0° sampai dengan 202°. maka diperlukan proses kalibrasi dengan menggunakan rumus matematika untuk penyesuaian antara masukan, keluaran dengan sudut yang diinginkan sesuai sebagai berikut:

$$\Delta\beta_1 = \frac{180\ x\ \beta_1}{202} \quad \text{.....(3.24)}$$

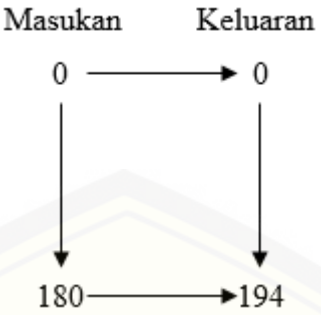
Keterangan:

202 = Sudut keluaran motor servo ketika diberi masukan maksimal

$\beta_1$ = Sudut *tibia* hasil perhitungan *inverse kinematics* kestabilan (gambar 3.12)

$\Delta\beta_1$ = Sudut hasil proses kalibrasi

Proses kalibrasi servo bagian *tibia* belakang kanan:




Gambar 3.26 Hubungan sudut masukan dan keluaran servo *tibia* belakang kanan

Dapat dilihat pada gambar 3.26 nilai masukan servo dengan keluaran servo tidak sesuai, dimana masukan dari 0° sampai dengan 180° diperoleh keluaran putaran motor servo sebesar 0° sampai dengan 194°. maka diperlukan proses kalibrasi dengan menggunakan rumus matematika untuk penyesuaian antara masukan, keluaran dengan sudut yang diinginkan sesuai sebagai berikut:

$$\Delta\beta_2 = \frac{180\ x\ \beta_2}{194} \quad \text{.....(3.25)}$$

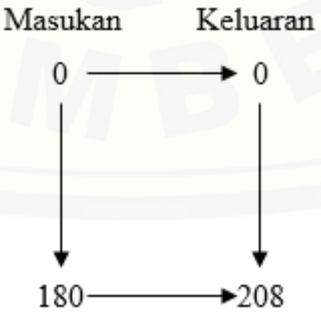
Keterangan:

194 = Sudut keluaran motor servo ketika diberi masukan maksimal

$\beta_2$ = Sudut *tibia* hasil perhitungan *inverse kinematics* kestabilan (gambar 3.12)

$\Delta\beta_2$ = Sudut hasil proses kalibrasi

Proses kalibrasi servo bagian *tibia* depan kiri:




Gambar 3.27 Hubungan sudut masukan dan keluaran servo *tibia* depan kiri

Dapat dilihat pada gambar 3.27 nilai masukan servo dengan keluaran servo tidak sesuai, dimana masukan dari 0° sampai dengan 180° diperoleh keluaran putaran motor servo sebesar 0° sampai dengan 208°. maka diperlukan proses kalibrasi dengan menggunakan rumus matematika untuk penyesuaian antara masukan, keluaran dengan sudut yang diinginkan sesuai sebagai berikut:

$$\Delta\beta_3 = \frac{180\ x\ \beta_3}{208} \quad \text{.....(3.26)}$$

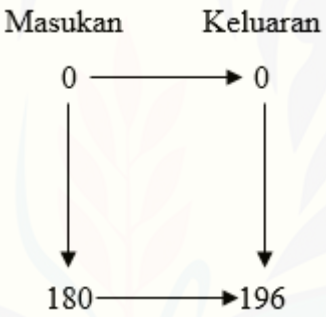
Keterangan:

208 = Sudut keluaran motor servo ketika diberi masukan maksimal

$\beta_3$ = Sudut *tibia* hasil perhitungan *inverse kinematics* kestabilan (gambar 3.12)

$\Delta\beta_3$ = Sudut hasil proses kalibrasi

Proses kalibrasi servo bagian *tibia* belakang kiri:




Gambar 3.28 Hubungan sudut masukan dan keluaran servo *tibia* belakang kiri

Dapat dilihat pada gambar 3.28 nilai masukan servo dengan keluaran servo tidak sesuai, dimana masukan dari 0° sampai dengan 180° diperoleh keluaran putaran motor servo sebesar 0° sampai dengan 196°. maka diperlukan proses kalibrasi dengan menggunakan rumus matematika untuk penyesuaian antara masukan, keluaran dengan sudut yang diinginkan sesuai sebagai berikut:

$$\Delta\beta_4 = \frac{180\ x\ \beta_4}{196} \quad \text{.....(3.27)}$$

Keterangan:

196 = Sudut keluaran motor servo ketika diberi masukan maksimal

$\beta_4$ = Sudut *tibia* hasil perhitungan *inverse kinematics* kestabilan (gambar 3.12)

$\Delta\beta_4$ = Sudut hasil proses kalibrasi

### 3.9 Rencana Pengujian

Pada robot penelitian yang akan dibuat nanti, rencana akan diuji dengan beberapa model bidang yang memiliki kemiringan yang berbeda-beda yaitu bidang kemiringan -15°, -12°, -10°, -5°, -3° , 0°, 3°, 5°, 10°, 12°, 15° terhadap sudut *pitch* dan juga bidang kemiringan -15°, -12°, -10°, -5°, -3° , 0°, 3°, 5°, 10°, 12°, 15° terhadap sudut *roll*, masing-masing keadaan bidang miring *roll* dan *pitch* tersebut robot juga diuji saat robot pada posisi diam dan saat robot berjalan sehingga dengan keterbatasan sudut motor servo dan desain yang telah dibuat akan diketahui respon bagaimana robot *quadruped* yang akan dibuat dalam mempertahankan keseimbangannya.

# BAB 5. PENUTUP

## 5.1 Kesimpulan

Dari hasil penelitian implimentasi *inverse kinematics* pada robot *quadruped* telah dilakukan, maka dapat ditarik beberapa kesimpulan sebagai berikut:

1. Kepresisian dari *design* atau ukuran robot *quadruped* sangat berpengaruh terhadap masukan pada metode *inverse kinematics*, semakin pendek dari *design* kaki maka jarak langkah yang dapat ditempuh juga akan semakin dekat. Dalam penelitian ini penulis menggunakan desain yaitu lengan coxa 2 cm, lengan femur 7 cm, dan lengan tibia 15,5 cm.
2. Dengan menggunakan persamaan *Inverse kinematics* dan kontrol PID *ziegler-nichols* pada robot *quadruped* diperoleh respon robot dalam menstabilkan *body* yaitu pada pengujian statis terhadap sudut *pitch* dalam keadaan *standby* diperoleh *settling time* terbesar yaitu 320 ms dengan Ess = 4 pada sudut *pitch* -5°. Pada pengujian statis terhadap sudut *roll* dalam keadaan berjalan di sudut *roll* -15° terdapat osilasi terbesar yaitu 9°, di sudut *roll* -7° terdapat osilasi terbesar yaitu 13°, di sudut *roll* 7° terdapat osilasi terbesar yaitu -10°, dan di sudut 15° terdapat osilasi terbesar yaitu -12°. Pada pengujian dinamis terhadap sudut *roll* dalam keadaan berjalan diberikan gangguan sudut *roll* 7° diperoleh respon dalam menstabilkan *body* dalam waktu 580 ms dan saat diberikan gangguan sudut *roll* -7° diperoleh respon dalam menstabilkan *body* dalam waktu 400 ms.

## 5.2 Saran

Dari hasil penelitian implimentasi *inverse kinematics* pada robot *quadruped* dan hasil pengujian yang telah dilakukan. Penulis mempunyai beberapa saran sehingga penelitian ini dapat dikembangkan.

1. Menggunakan aktuator yang memiliki tingkat kepresisian yang lebih tinggi atau aktuator yang didalamnya sudah memiliki kontrol yang baik.
2. Adanya penambahan sensor tekanan pada masing-masing kaki sehingga robot juga dapat menyeimbangkan body pada bidang yang tidak rata atau bidang yang berlubang maupun berbatu.

3. Menggunakan metode kecerdasan buatan yang lebih baik lagi sehingga robot dalam berjalan dapat melakukan pembelajaran sendiri.
4. Adanya penambahan sensor jarak dan kompas sehingga robot dapat melakukan proses mapping ruangan.

# DAFTAR PUSTAKA

Arduino Mega 2560. https://www.arduino.cc/en/Main/ArduinoBoardMega2560. Diakses pada 21 Juni 2016.

Asrofi, Sumardi, dan Budi. 2015. "*Stabilisasi Robot Berkaki 6 (Hexapod) Pada Bidang Miring Menggunakan 9 DOF IMU Berbasis Invers Kinematic*," Jurnal Teknik Elektro, Universitas Diponegoro Semarang.

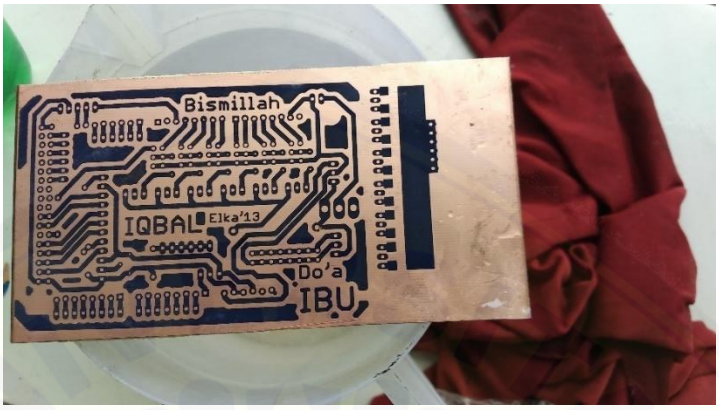
Darwison dan Wahyudi. 2015. "*Kontrol Kecepatan Robot Hexapod Pemadam Api Menggunakan Metode Logika Fuzzy,*" Jurnal Teknik Elektro, Universitas Andalas.

Rois, Kemalasari, Sumantri, dan Wijayanto. 2010. "*Pengaturan Posisi Motor Servo DC Dengan Metode Fuzzy Logic,*" Jurnal Teknik Elektronika, Politeknik Elektronika Negeri Surabaya.

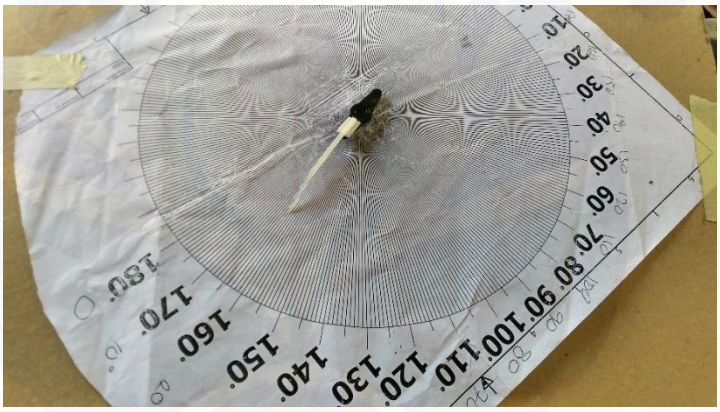
Sigit, Riyanto. 2007. "*Robotika, Sensor dan Aktuator"*. Yogyakarta: Graha Ilmu, Vol.63.

# LAMPIRAN

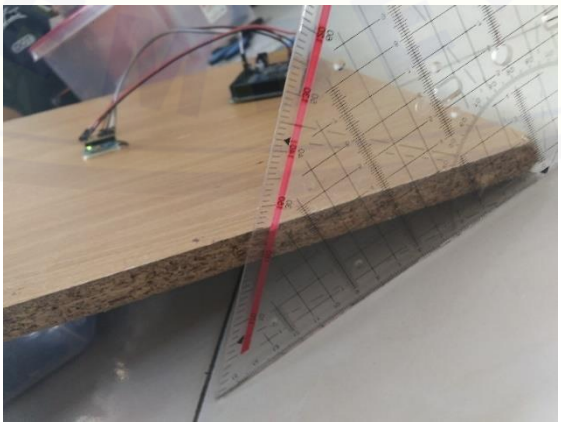
## A. Dokumentasi Penelitian



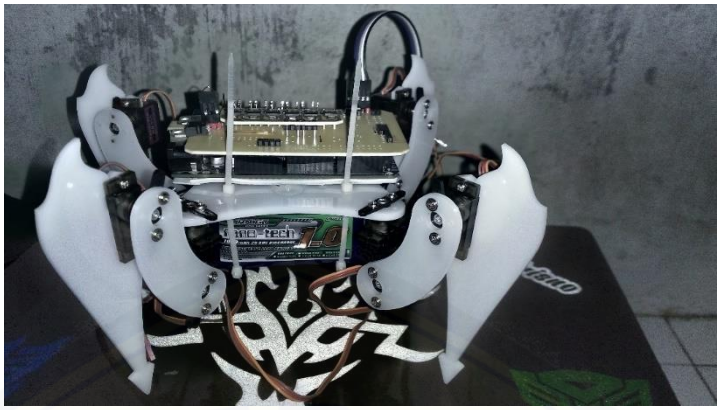

Gambar A.1 Proses Pembuatan Elektronika Robot *Quadruped*




Gambar A.2 Proses Kalibrasi Motor Servo

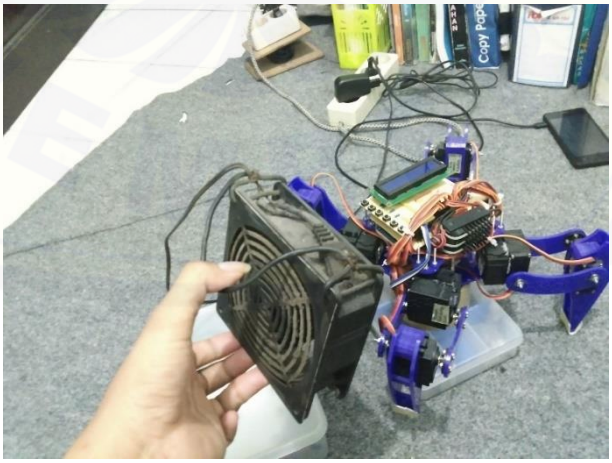


Gambar A.3 Proses Kalibrasi Sensor MPU6050



Gambar A.4 Hasil Pembuatan Robot *Quadruped* 1(*Failure*)

Gambar A.5 Proses Pengujian Robot *Quadruped* Pada Bidang Miring



Gambar A.6 Proses Pendinginan Motor Servo Pada Robot *Quadruped*

## B. Data Tuning PID *Ziegler-Nichols* aturan *Quarter-Decay*

Tabel B.1 Data Tuning PID *Ziegler-Nichols* aturan *Quarter-Decay*

| **Waktu (ms)** | ***Roll* (°)** | ***Pitch* (°)** |
|---|---|---|
| 20 | 0 | 2 |
| 40 | -1 | 12 |
| 60 | 2 | 0 |
| 80 | 4 | -9 |
| 100 | 0 | -5 |
| 120 | -2 | 0 |
| 140 | -3 | 3 |
| 160 | -1 | 1 |
| 180 | 1 | 0 |
| 200 | -2 | 2 |
| 220 | -2 | 1 |
| 240 | -1 | 1 |
| 260 | -1 | 1 |
| 280 | 0 | 0 |
| 300 | 0 | 0 |
| 320 | 0 | 0 |
| 340 | 0 | 0 |
| 360 | 0 | 0 |
| 380 | -1 | 0 |
| 400 | 0 | -1 |
| 420 | 0 | -1 |
| 440 | 0 | 0 |
| 460 | 0 | 0 |
| 480 | -1 | 0 |
| 500 | -1 | -1 |
| 520 | -1 | -1 |
| 540 | 0 | -1 |
| 560 | 0 | -1 |
| 580 | 0 | -1 |
| 600 | 0 | -1 |
| 620 | 0 | -1 |
| 640 | 0 | -1 |
| 660 | 0 | -1 |
| 680 | 0 | -1 |
| 700 | 0 | -1 |
| 720 | 0 | -1 |
| 740 | 0 | -1 |
| 760 | 0 | -1 |
| 780 | 0 | -1 |
| 800 | 0 | -1 |
| 820 | 0 | -1 |
| 840 | 0 | -1 |
| 860 | 0 | -1 |
| 880 | 0 | -1 |
| 900 | 0 | -1 |
| 920 | 0 | -1 |
| 940 | 0 | -1 |
| 960 | 0 | -1 |
| 980 | 0 | -1 |
| 1000 | 0 | -1 |
| 1020 | 0 | 0 |
| 1040 | 0 | 0 |
| 1060 | 0 | 0 |
| 1080 | 0 | 0 |
| 1100 | 0 | -1 |
| 1120 | 1 | -1 |
| 1140 | 1 | -1 |
| 1160 | 1 | -1 |
| 1180 | 1 | -1 |
| 1200 | 0 | 0 |
| 1220 | 0 | 0 |
| 1240 | 0 | 0 |
| 1260 | 1 | 0 |
| 1280 | 1 | 0 |

## C. Data Pengujian Robot *Quadruped*

Tabel C.1 Data Pengujian Statis Robot Terhadap Sudut *Pitch* dalam Keadaan *Standby*

| Waktu (ms) | 15° | | 13° | | 10° | | 7° | | 5° | | -5° | | -7° | | -10° | | -13° | | -15° | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | *Roll* (°) | *Pitch* (°) | *Roll* (°) | *Pitch* (°) | *Roll* (°) | *Pitch* (°) | *Roll* (°) | *Pitch* (°) | *Roll* (°) | *Pitch* (°) | *Roll* (°) | *Pitch* (°) | *Roll* (°) | *Pitch* (°) | *Roll* (°) | *Pitch* (°) | *Roll* (°) | *Pitch* (°) | *Roll* (°) | *Pitch* (°) |
| 20 | 9 | -14 | 6 | -5 | 2 | -8 | 1 | -1 | 1 | -2 | -1 | 2 | -5 | 2 | -4 | 3 | -8 | 6 | -7 | 11 |
| 40 | 7 | -12 | 11 | -2 | 13 | -7 | 6 | 1 | 5 | -3 | 5 | 6 | 13 | 16 | 2 | 11 | 4 | 25 | 3 | 20 |
| 60 | 20 | -13 | 7 | -15 | 11 | 0 | 11 | -3 | 5 | 1 | 4 | -3 | -1 | 4 | -7 | -6 | -16 | 16 | -3 | 3 |
| 80 | 5 | -10 | 1 | -9 | 5 | 2 | 5 | -1 | 0 | -4 | 1 | 1 | 13 | 8 | 3 | 9 | 12 | -6 | 8 | -8 |
| 100 | 0 | -1 | -2 | -5 | -9 | -14 | 0 | -6 | -5 | -3 | 2 | -12 | -10 | -22 | -5 | -18 | -2 | -18 | -1 | 1 |
| 120 | -8 | -2 | 10 | 9 | -3 | 5 | -9 | -1 | -3 | -5 | -2 | -1 | -7 | -2 | 0 | -2 | 11 | -5 | -4 | -7 |
| 140 | -7 | 0 | -1 | 8 | -9 | -2 | 7 | -15 | -7 | 0 | -4 | -14 | -19 | -22 | 0 | -10 | 0 | -4 | -12 | -1 |
| 160 | -10 | 4 | 1 | 9 | 15 | 10 | -15 | 2 | 11 | 0 | -7 | 22 | -3 | 29 | -2 | 22 | -1 | 7 | -9 | 0 |
| 180 | 4 | 0 | -6 | -17 | -1 | 0 | 14 | -9 | 1 | 4 | -11 | 7 | -12 | 3 | 0 | 12 | 0 | -1 | -13 | 0 |
| 200 | 1 | -1 | -4 | -3 | 9 | 1 | 3 | 10 | 6 | 5 | -7 | 14 | 9 | 14 | 4 | 14 | 0 | -1 | -1 | -3 |
| 220 | 2 | -1 | -13 | 1 | -2 | -11 | 21 | 8 | -3 | 3 | 0 | -24 | 4 | -23 | 3 | -8 | 1 | -2 | 0 | 0 |
| 240 | 0 | -1 | 3 | -2 | 0 | -5 | 3 | 10 | -2 | -3 | 5 | -6 | 3 | 4 | 1 | 0 | -1 | 0 | 0 | -1 |
| 260 | 0 | 0 | -7 | 5 | 0 | -3 | 12 | -9 | 0 | -2 | 8 | -12 | -2 | 0 | 0 | 0 | -2 | 0 | 2 | -1 |
| 280 | 0 | -1 | 11 | -5 | 0 | -3 | -13 | 2 | 0 | -1 | 0 | 21 | 4 | -4 | 0 | 0 | 0 | -1 | 0 | -2 |
| 300 | 0 | -2 | 14 | -2 | 0 | -4 | 2 | 2 | -3 | 0 | 10 | 3 | 2 | -1 | 0 | 1 | -3 | 1 | 0 | -2 |
| 320 | -1 | -2 | 9 | -3 | 0 | -3 | 0 | 6 | -2 | -2 | 4 | 2 | 3 | -2 | -1 | 0 | -1 | -1 | 0 | -2 |
| 340 | 0 | -1 | 21 | -2 | 0 | -3 | -1 | 0 | 0 | -1 | 0 | 4 | 3 | -2 | 0 | 0 | -2 | 0 | 0 | 0 |
| 360 | 0 | -2 | 4 | 1 | 0 | -3 | -1 | 5 | -2 | -2 | 0 | 3 | 2 | -2 | 0 | 0 | -2 | -1 | 0 | -2 |
| 380 | 0 | -1 | 15 | -3 | 0 | -2 | -2 | 2 | -1 | -2 | 1 | 3 | 0 | -1 | -1 | 0 | -4 | 0 | 1 | -2 |
| 400 | 0 | -1 | -10 | 1 | 0 | -3 | 0 | 5 | 0 | -2 | 0 | 3 | 6 | -4 | 0 | 0 | 1 | -1 | 0 | -2 |
| 420 | -1 | -1 | 1 | 1 | 0 | -4 | -1 | 3 | 0 | -3 | -2 | 3 | 2 | -2 | 0 | 0 | -3 | 0 | 0 | -2 |
| 440 | 0 | -1 | 0 | 0 | 0 | -3 | -3 | 3 | 0 | -1 | -1 | 4 | 2 | -2 | 0 | 0 | -3 | 0 | 0 | -2 |
| 460 | 0 | 0 | 0 | 1 | 0 | -3 | 0 | 5 | 0 | -2 | 0 | 3 | 5 | -3 | 0 | 1 | -2 | -1 | 0 | -2 |
| 480 | 1 | -1 | -1 | 0 | 0 | -3 | -2 | 2 | 0 | -2 | -2 | 4 | 1 | -2 | -1 | 0 | -5 | 0 | 0 | -2 |
| 500 | 0 | -1 | 0 | 0 | -1 | -3 | -1 | 4 | -2 | -3 | -2 | 4 | 6 | -4 | 0 | 0 | 0 | -2 | 0 | -2 |
| 520 | -2 | -2 | -1 | 0 | 0 | -3 | -1 | 4 | -1 | -2 | 1 | 3 | 3 | -3 | 0 | 0 | -3 | 0 | 0 | -2 |

| | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 540 | 0 | -2 | 0 | 0 | 0 | -3 | -1 | 3 | 1 | -3 | 0 | 3 | 4 | -2 | 0 | 0 | 0 | -1 | 0 | -2 |
| 560 | 1 | 0 | -1 | 0 | 0 | -4 | -1 | 5 | 0 | -3 | -1 | 3 | 2 | -2 | -1 | 0 | 0 | -1 | 0 | -3 |
| 580 | 0 | -1 | 0 | 1 | 0 | -3 | -2 | 2 | 0 | 0 | 0 | 3 | 6 | -4 | -1 | 0 | -3 | 0 | 0 | -2 |
| 600 | 0 | -1 | -2 | 0 | 0 | -3 | -1 | 4 | 0 | -1 | 1 | 2 | 3 | -2 | 0 | 0 | 0 | -1 | 0 | -2 |
| 620 | -1 | -1 | -2 | -1 | 0 | -3 | -1 | 4 | 0 | -2 | -1 | 3 | 2 | -2 | 0 | 0 | -2 | 0 | 0 | -2 |
| 640 | -3 | 0 | 0 | 1 | 0 | -3 | -2 | 1 | -1 | -1 | -1 | 4 | 1 | -3 | 0 | 0 | -1 | 0 | 0 | -2 |
| 660 | 1 | -1 | -2 | 0 | 0 | -3 | -2 | 2 | 1 | -2 | 0 | 3 | 4 | -2 | 0 | 0 | -3 | -1 | 0 | -2 |
| 680 | 0 | -1 | -1 | 0 | 0 | -3 | -1 | 2 | 0 | -4 | 0 | 4 | 4 | -3 | 0 | 0 | 1 | -1 | 0 | -2 |
| 700 | 0 | -1 | -1 | 0 | 0 | -3 | -1 | 4 | -4 | -2 | 0 | 4 | 5 | -2 | 0 | 0 | -2 | 0 | 0 | -2 |
| 720 | -1 | -1 | -2 | 0 | 0 | -3 | -1 | 4 | -2 | -1 | 2 | 2 | 1 | -3 | 0 | 0 | 0 | -1 | 0 | -3 |
| 740 | -3 | -1 | 0 | 1 | 0 | -2 | -2 | 3 | 0 | -3 | 1 | 3 | 3 | -3 | -1 | 0 | -1 | -1 | 0 | -2 |
| 760 | 1 | -1 | -2 | 0 | 0 | -3 | -1 | 5 | 0 | -2 | 1 | 3 | 2 | -3 | -1 | 0 | -5 | -1 | 0 | -2 |
| 780 | 0 | -1 | -2 | 0 | 0 | -3 | -1 | 4 | -1 | -2 | 2 | 2 | 3 | -2 | 0 | 0 | -1 | -1 | 0 | -2 |
| 800 | -2 | -1 | -1 | 1 | 0 | -3 | -1 | 4 | 1 | -3 | 2 | 3 | 4 | -3 | 0 | 0 | 0 | -1 | 0 | -2 |
| 820 | 0 | -2 | 0 | 0 | 0 | -2 | -1 | 3 | 0 | -3 | 0 | 4 | 2 | -3 | -1 | 2 | -1 | 0 | 0 | -1 |
| 840 | 0 | -2 | 0 | 0 | 0 | -3 | -2 | 2 | 0 | -2 | 1 | 3 | 1 | -3 | -1 | 0 | -3 | 0 | 0 | -2 |
| 860 | 0 | 0 | -1 | 1 | 0 | -3 | -1 | 5 | 0 | -3 | 2 | 3 | 7 | -4 | -1 | 0 | 0 | -1 | 0 | -3 |
| 880 | -2 | -1 | -1 | 0 | 0 | -3 | -1 | 2 | -1 | -1 | 1 | 4 | 5 | -4 | 0 | 0 | -2 | -1 | 1 | -3 |
| 900 | 1 | 0 | -1 | 0 | 0 | -3 | -1 | 2 | 1 | -2 | 1 | 4 | 3 | -3 | -1 | 0 | 0 | -1 | 0 | -2 |
| 920 | 0 | -1 | -1 | 1 | 0 | -3 | -1 | 3 | -2 | -1 | 1 | 3 | 2 | -2 | 0 | 0 | -2 | 0 | -4 | -2 |
| 940 | 0 | -1 | -2 | 0 | 0 | -3 | -1 | 5 | 0 | -3 | 0 | 3 | 6 | -3 | 0 | 0 | -3 | 0 | 0 | -1 |
| 960 | -1 | 0 | -1 | 1 | 0 | -3 | -1 | 4 | 0 | -3 | -1 | 4 | 3 | -2 | -1 | 0 | -2 | -3 | 0 | -3 |
| 980 | -2 | -2 | -1 | 0 | 0 | -3 | -1 | 2 | 0 | 0 | -2 | 4 | 3 | -2 | 0 | 0 | 0 | -1 | 0 | -3 |
| 1000 | 0 | -2 | -1 | 1 | 0 | -3 | -1 | 4 | 0 | -1 | 0 | 3 | 2 | -2 | 0 | 0 | -2 | 0 | 0 | -2 |

Tabel C.2 Data Pengujian Statis Robot Terhadap Sudut *Roll* dalam Keadaan *Standby*

| Waktu (ms) | 15° | | 13° | | 10° | | 7° | | 5° | | -5° | | -7° | | -10° | | -13° | | -15° | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | *Roll* (°) | *Pitch* (°) | *Roll* (°) | *Pitch* (°) | *Roll* (°) | *Pitch* (°) | *Roll* (°) | *Pitch* (°) | *Roll* (°) | *Pitch* (°) | *Roll* (°) | *Pitch* (°) | *Roll* (°) | *Pitch* (°) | *Roll* (°) | *Pitch* (°) | *Roll* (°) | *Pitch* (°) | *Roll* (°) | *Pitch* (°) |
| 20 | -4 | -8 | -9 | -8 | -6 | -6 | -5 | -3 | -4 | -2 | 4 | 1 | 2 | 3 | 6 | 4 | 6 | 6 | 7 | 1 |
| 40 | -7 | -11 | -5 | -20 | 0 | 1 | 5 | 6 | 6 | -1 | 11 | 13 | 13 | 5 | 9 | 7 | 9 | 6 | 18 | 13 |
| 60 | 1 | -9 | 2 | -21 | 3 | -14 | 3 | -3 | 2 | -8 | 0 | -4 | 1 | 2 | 17 | 0 | 3 | 2 | 10 | 4 |
| 80 | 6 | -2 | 10 | -4 | 6 | 4 | 10 | 3 | 15 | 0 | 0 | 14 | 7 | 0 | 8 | 1 | 1 | 1 | 2 | -2 |
| 100 | 4 | -3 | -1 | 1 | 2 | -2 | -2 | -17 | -5 | -17 | -4 | -18 | -4 | -2 | 0 | -4 | 0 | 1 | -10 | -1 |
| 120 | -8 | -1 | 0 | 6 | 5 | -4 | 8 | -8 | 3 | 6 | -3 | -7 | -5 | -1 | -8 | 0 | -2 | -1 | 1 | -1 |
| 140 | -9 | -1 | -20 | 7 | -6 | 1 | -8 | -14 | -13 | 6 | -8 | -4 | 2 | -4 | -9 | 1 | 4 | -7 | -2 | -7 |
| 160 | -5 | -1 | -4 | 11 | -7 | 3 | -10 | 7 | -15 | 15 | -4 | 18 | -4 | 0 | -5 | 16 | 3 | 1 | -4 | 0 |
| 180 | -3 | -2 | -17 | 13 | -7 | 4 | -7 | 14 | -23 | 7 | 7 | 3 | -3 | -3 | -8 | 3 | 8 | -9 | 1 | 11 |
| 200 | -3 | -1 | 4 | -1 | -8 | 2 | -12 | 10 | -3 | 9 | -1 | 5 | -1 | 4 | 4 | 13 | -4 | 9 | -2 | 6 |
| 220 | -4 | -1 | 1 | 8 | -2 | 1 | 0 | 14 | -6 | -8 | 3 | 1 | -4 | 6 | -2 | -9 | 3 | 4 | -5 | 3 |
| 240 | -3 | -1 | 5 | -11 | -12 | 4 | -5 | 0 | 24 | -2 | 2 | 0 | 4 | 1 | 0 | -6 | -3 | 28 | 1 | -1 |
| 260 | -2 | -1 | 7 | 0 | -6 | 4 | -1 | -2 | 8 | -1 | -1 | 1 | -2 | 1 | 1 | -8 | 7 | 3 | 2 | 1 |
| 280 | -2 | -1 | 2 | 1 | -10 | 8 | 0 | -3 | 13 | -1 | 3 | 2 | 1 | 0 | -2 | -3 | 8 | -12 | 6 | -8 |
| 300 | -2 | -1 | 2 | 0 | 1 | 4 | 1 | -1 | -15 | -2 | 1 | 2 | 0 | 2 | -1 | -2 | 4 | -8 | 0 | 7 |
| 320 | -2 | -1 | 4 | -1 | -5 | 7 | 0 | -1 | -1 | 0 | 0 | 0 | 0 | 0 | -1 | -1 | 17 | -21 | 3 | -8 |
| 340 | -1 | -1 | 4 | -1 | 0 | -2 | -1 | 0 | -1 | 0 | 2 | 1 | 0 | 1 | -2 | -1 | 7 | -7 | 2 | 1 |
| 360 | -2 | 0 | 1 | 0 | 0 | 0 | 0 | -1 | -1 | 0 | 0 | 1 | 1 | 2 | -1 | -1 | 6 | -19 | 1 | 2 |
| 380 | -2 | 0 | 2 | -1 | 3 | -2 | 0 | -1 | -2 | 0 | 2 | 1 | 0 | 0 | -1 | -1 | -1 | -1 | 1 | 2 |
| 400 | -2 | -1 | 1 | 0 | 1 | -1 | 0 | -1 | -2 | 0 | 1 | 1 | 0 | 2 | -1 | -1 | 0 | -1 | 2 | 2 |
| 420 | -1 | -1 | 0 | 0 | 0 | -1 | 0 | 0 | -2 | 0 | 0 | 1 | 1 | 1 | -1 | -2 | -4 | 0 | 1 | 2 |
| 440 | -2 | -1 | 0 | 0 | 2 | -1 | 0 | -2 | -4 | 0 | 1 | 1 | 0 | 2 | -1 | -2 | -1 | 0 | 1 | 2 |
| 460 | -2 | -1 | 1 | 0 | 2 | 0 | 0 | -2 | -2 | 0 | 0 | 1 | -1 | 0 | -1 | -2 | -1 | -1 | 1 | 3 |
| 480 | -2 | 0 | 4 | 0 | 3 | -1 | 0 | 0 | -1 | 0 | 2 | 0 | 1 | 1 | -1 | -2 | -1 | 0 | 1 | 2 |
| 500 | -1 | 0 | 1 | -2 | 0 | -1 | 1 | 0 | -2 | 0 | 0 | 2 | 1 | 1 | -1 | -2 | 0 | -1 | 1 | 2 |
| 520 | -1 | 0 | 2 | 0 | 0 | -1 | 0 | -1 | -2 | 0 | 0 | 3 | -2 | 0 | -1 | -1 | -1 | 0 | 1 | 2 |
| 540 | -1 | 0 | 3 | 0 | 3 | 0 | 0 | -1 | -2 | 0 | 2 | 0 | 0 | 0 | -1 | -2 | -1 | 0 | 1 | 1 |

| | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 560 | -1 | 0 | 4 | -1 | 0 | 0 | 0 | -1 | -2 | 0 | 0 | 1 | 1 | 2 | -1 | -2 | 0 | -1 | 0 | 2 |
| 580 | -1 | 0 | 2 | -1 | 1 | -1 | -1 | 0 | -2 | 0 | 0 | 1 | 0 | 0 | -1 | -2 | -1 | 0 | 1 | 2 |
| 600 | -1 | -1 | 1 | -1 | 1 | 0 | 0 | -1 | -2 | 0 | 0 | 1 | 1 | 2 | -1 | -2 | -1 | 0 | 1 | 3 |
| 620 | -3 | 0 | 0 | 0 | 0 | -1 | 0 | -1 | -2 | 0 | 0 | 1 | 1 | 2 | -1 | -2 | -2 | 0 | 1 | 2 |
| 640 | -1 | 0 | 5 | -1 | 3 | 0 | -2 | -1 | -1 | 0 | 1 | 2 | 0 | 2 | -1 | -2 | 0 | 0 | 1 | 2 |
| 660 | -1 | 0 | 1 | 0 | 0 | 0 | 1 | -2 | -2 | 0 | 0 | 1 | 0 | 0 | -1 | -2 | 0 | 1 | 1 | 2 |
| 680 | -1 | 0 | 1 | 0 | 2 | -1 | -1 | -1 | -2 | 0 | 1 | 1 | 0 | 1 | -1 | -1 | -1 | 0 | 1 | 2 |
| 700 | -1 | 0 | 2 | 0 | 0 | -1 | 0 | -1 | -2 | 0 | 1 | 1 | 0 | 1 | -1 | -1 | -3 | 1 | 0 | 2 |
| 720 | -2 | -2 | 2 | 0 | 0 | -1 | 0 | -1 | -1 | 0 | 0 | 0 | -1 | 0 | -1 | -1 | 0 | 0 | 0 | 2 |
| 740 | -2 | -1 | 5 | -1 | 2 | -1 | -1 | -2 | 0 | 0 | 1 | 2 | 0 | 1 | -1 | -2 | -1 | -1 | 1 | 2 |
| 760 | -2 | 0 | 1 | 0 | 0 | 0 | -1 | -2 | -2 | 0 | -1 | 1 | 0 | 2 | -1 | -2 | -1 | 0 | 1 | 2 |
| 780 | -2 | 0 | 3 | -1 | 2 | -1 | 0 | -2 | -2 | 0 | 0 | 1 | 0 | 0 | -1 | -2 | 0 | 0 | 2 | 2 |
| 800 | -2 | 0 | 2 | -1 | 1 | 0 | 0 | -1 | -2 | 0 | 1 | 1 | 0 | 1 | -1 | -2 | 0 | -1 | 1 | 2 |
| 820 | -1 | 0 | 1 | 0 | 1 | -1 | 0 | -1 | -2 | 0 | 0 | 1 | 0 | 2 | -1 | -2 | -1 | 0 | 1 | 2 |
| 840 | -1 | 0 | 5 | -2 | 3 | -3 | 0 | -2 | -2 | 0 | 0 | 1 | 0 | 1 | -2 | -2 | -2 | -1 | 1 | 2 |
| 860 | -1 | -2 | 1 | 0 | 0 | 0 | 0 | -2 | -2 | 0 | 0 | 2 | 0 | 0 | -1 | -1 | -3 | 0 | 1 | 2 |
| 880 | -1 | -2 | 2 | -1 | 2 | -1 | 0 | -1 | -3 | 0 | 0 | 0 | 0 | 2 | -1 | -2 | -1 | -1 | 1 | 2 |
| 900 | -1 | -2 | 1 | 0 | 1 | 0 | 0 | -1 | -2 | -1 | 0 | 1 | 0 | 1 | -1 | -2 | -2 | 0 | 1 | 2 |
| 920 | -1 | 0 | 1 | 0 | 1 | 0 | 0 | -1 | -2 | 0 | -1 | 1 | 0 | 0 | -1 | -2 | 0 | 0 | 2 | 2 |
| 940 | -1 | 0 | 5 | -1 | 0 | -1 | 0 | -1 | -2 | 0 | 3 | 2 | 0 | 1 | -1 | -2 | 0 | 0 | 1 | 2 |
| 960 | -1 | 0 | 1 | 0 | 1 | -1 | 0 | -3 | -2 | 0 | 1 | 0 | 0 | 1 | -1 | -2 | -2 | 0 | 1 | 1 |
| 980 | -1 | -1 | 3 | -1 | 2 | -1 | -2 | -2 | -2 | 0 | 1 | 1 | 0 | 0 | -1 | -2 | -1 | 0 | 1 | 2 |
| 1000 | -1 | -1 | 1 | 0 | 1 | -1 | -1 | -1 | -2 | 0 | 1 | 1 | 0 | 3 | -1 | -2 | -1 | 0 | 0 | 2 |

Tabel C.3 Data Pengujian Dinamis Robot Terhadap Sudut *Roll* dan *Pitch* dalam Keadaan *Standby*

| Waktu (s) | Pengujian 1 | | Pengujian 2 | |
|---|---|---|---|---|
| | Roll (°) | Pitch (°) | Roll (°) | Pitch (°) |
| 0,2 | 1 | 0 | 0 | 0 |
| 0,4 | -1 | 0 | -1 | 7 |
| 0,6 | -2 | 1 | 0 | 5 |
| 0,8 | -1 | 0 | -1 | 7 |
| 1 | -1 | -2 | -1 | 4 |
| 1,2 | -1 | -3 | 0 | 5 |
| 1,4 | 0 | -3 | 2 | 3 |
| 1,6 | 0 | -4 | 0 | 5 |
| 1,8 | 0 | -3 | 0 | 6 |
| 2 | -1 | -3 | 0 | 7 |
| 2,2 | -2 | -4 | 0 | 7 |
| 2,4 | 0 | -1 | 0 | 4 |
| 2,6 | -1 | -1 | 0 | 2 |
| 2,8 | 0 | 0 | -2 | 2 |
| 3 | 3 | 0 | 0 | 4 |
| 3,2 | 1 | -1 | -3 | 4 |
| 3,4 | -3 | 5 | -3 | 3 |
| 3,6 | -3 | 5 | -1 | 3 |
| 3,8 | -1 | 5 | 2 | 0 |
| 4 | 0 | 4 | 2 | -5 |
| 4,2 | 2 | 4 | 1 | -8 |
| 4,4 | 2 | 3 | -2 | 7 |
| 4,6 | 0 | 4 | 0 | 1 |
| 4,8 | 2 | 3 | -1 | 6 |
| 5 | 2 | 4 | -3 | 6 |
| 5,2 | 0 | 6 | -2 | 4 |
| 5,4 | -1 | 3 | 0 | 5 |
| 5,6 | 2 | 1 | 1 | 5 |
| 5,8 | 2 | 0 | 3 | -1 |
| 6 | 2 | 0 | 6 | -9 |
| 6,2 | 3 | -1 | 6 | -10 |
| 6,4 | 4 | -1 | 8 | -11 |
| 6,6 | 3 | -1 | 4 | -10 |
| 6,8 | 1 | -2 | 1 | -5 |
| 7 | 2 | -4 | 3 | -6 |
| 7,2 | 2 | -4 | 0 | -3 |
| 7,4 | 1 | -4 | -3 | 0 |
| 7,6 | 1 | -5 | -6 | 11 |
| 7,8 | 1 | -6 | -2 | 11 |
| 8 | 2 | -5 | 2 | 9 |
| 8,2 | 1 | -6 | 4 | 9 |
| 8,4 | 1 | -6 | 1 | 9 |
| 8,6 | 1 | -4 | -2 | 6 |
| 8,8 | 1 | -5 | -4 | 6 |
| 9 | 2 | -6 | -10 | -6 |
| 9,2 | 3 | -7 | -12 | -15 |
| 9,4 | 1 | -5 | -11 | -15 |
| 9,6 | 2 | -4 | -10 | -17 |
| 9,8 | 1 | -4 | -10 | -18 |
| 10 | 1 | -4 | -8 | -15 |
| 10,2 | 2 | -4 | -9 | -6 |
| 10,4 | 1 | -3 | -4 | -7 |
| 10,6 | 0 | -3 | -2 | -12 |
| 10,8 | 1 | -3 | 7 | -15 |
| 11 | 1 | -3 | 8 | -5 |
| 11,2 | 2 | -2 | 7 | 8 |
| 11,4 | 2 | -3 | 2 | 15 |
| 11,6 | 1 | -2 | -5 | 13 |
| 11,8 | 6 | -4 | -6 | 16 |
| 12 | 4 | -7 | -1 | 11 |
| 12,2 | 5 | -7 | 10 | 19 |
| 12,4 | 4 | -7 | 14 | 17 |
| 12,6 | 1 | -10 | 14 | 17 |
| 12,8 | 0 | -2 | 12 | 18 |
| 13 | 1 | -4 | 13 | 17 |
| 13,2 | 1 | -2 | 13 | 18 |
| 13,4 | 0 | -3 | 13 | 18 |
| 13,6 | 0 | -1 | 10 | 16 |
| 13,8 | 0 | 0 | 8 | 15 |
| 14 | -3 | 3 | 13 | 5 |
| 14,2 | -4 | 2 | 14 | 2 |
| 14,4 | -3 | 1 | 14 | 1 |
| 14,6 | -3 | 1 | 13 | 2 |
| 14,8 | -2 | 7 | 12 | 3 |
| 15 | -2 | 6 | 10 | 1 |
| 15,2 | -3 | 6 | 10 | 2 |
| 15,4 | -2 | 6 | 5 | 7 |
| 15,6 | -8 | 0 | 5 | 5 |
| 15,8 | -7 | 1 | 6 | 2 |
| 16 | -9 | 0 | 6 | 2 |
| 16,2 | -9 | 0 | 7 | 2 |
| 16,4 | -4 | 0 | 8 | 0 |
| 16,6 | 0 | -5 | 7 | 0 |
| 16,8 | -3 | 1 | 4 | 1 |
| 17 | 0 | 5 | 2 | 4 |
| 17,2 | 0 | 1 | -4 | 10 |
| 17,4 | 1 | 0 | -8 | 12 |
| 17,6 | 2 | 6 | -16 | 15 |

| | | | | |
|---|---|---|---|---|
| 17,8 | 1 | 3 | -13 | 19 |
| 18 | 2 | 1 | -12 | 19 |
| 18,2 | 4 | 0 | -13 | 21 |
| 18,4 | 5 | 0 | -14 | 21 |
| 18,6 | 7 | -2 | -13 | 21 |
| 18,8 | 5 | 5 | -12 | 21 |
| 19 | 7 | 0 | -11 | 21 |
| 19,2 | 9 | 7 | -12 | 20 |
| 19,4 | 10 | 9 | -11 | 19 |
| 19,6 | 11 | 5 | -13 | 20 |
| 19,8 | 11 | 9 | -11 | 17 |
| 20 | 11 | 6 | -12 | 19 |
| 20,2 | 9 | 11 | -9 | 17 |
| 20,4 | 6 | 10 | -10 | 17 |
| 20,6 | 6 | 10 | -10 | 17 |
| 20,8 | 3 | 10 | -11 | 17 |
| 21 | 2 | 10 | -8 | 14 |
| 21,2 | 2 | 9 | -9 | 13 |
| 21,4 | 0 | 10 | -8 | 13 |
| 21,6 | 0 | 10 | -9 | 13 |
| 21,8 | -1 | 9 | -6 | 11 |
| 22 | -2 | 9 | -8 | 12 |
| 22,2 | -4 | 8 | -5 | 10 |
| 22,4 | -4 | 9 | -6 | 8 |
| 22,6 | -6 | 8 | -5 | 7 |
| 22,8 | -7 | 8 | -3 | 8 |
| 23 | -8 | 6 | -4 | 7 |
| 23,2 | -9 | 7 | -3 | 1 |
| 23,4 | -10 | 7 | | |
| 23,6 | -10 | 6 | | |
| 23,8 | -10 | 4 | | |
| 24 | -9 | 4 | | |
| 24,2 | -11 | 2 | | |
| 24,4 | -10 | 2 | | |
| 24,6 | -10 | 1 | | |
| 24,8 | -11 | 1 | | |
| 25 | -10 | 0 | | |
| 25,2 | -10 | 0 | | |
| 25,4 | -10 | -2 | | |
| 25,6 | -10 | -4 | | |
| 25,8 | -10 | -4 | | |
| 26 | -11 | -4 | | |

| | | | | |
|---|---|---|---|---|
| 26,2 | -11 | -5 | | |
| 26,4 | -11 | -6 | | |
| 26,6 | -12 | -8 | | |
| 26,8 | -10 | -11 | | |
| 27 | -11 | -7 | | |
| 27,2 | -12 | -8 | | |
| 27,4 | -10 | -9 | | |
| 27,6 | -12 | -8 | | |
| 27,8 | -10 | -9 | | |
| 28 | -11 | -9 | | |
| 28,2 | -10 | -10 | | |
| 28,4 | -11 | -10 | | |
| 28,6 | -10 | -10 | | |
| 28,8 | -10 | -10 | | |
| 29 | -11 | -11 | | |
| 29,2 | -11 | -11 | | |
| 29,4 | -11 | -11 | | |
| 29,6 | -11 | -11 | | |
| 29,8 | -11 | -11 | | |
| 30 | -10 | -11 | | |
| 30,2 | -11 | -11 | | |
| 30,4 | -11 | -11 | | |
| 30,6 | -10 | -11 | | |
| 30,8 | -11 | -11 | | |
| 31 | -11 | -11 | | |
| 31,2 | -11 | -11 | | |
| 32,86 | -10 | -10 | | |
| 33,06 | -11 | -10 | | |
| 33,26 | -7 | -13 | | |
| 33,46 | -4 | -15 | | |
| 33,66 | -3 | -12 | | |
| 33,86 | -2 | -12 | | |
| 34,06 | -5 | -7 | | |
| 34,26 | 0 | -10 | | |
| 34,46 | 0 | -9 | | |
| 34,66 | 1 | -10 | | |
| 34,86 | 2 | -8 | | |
| 35,06 | 0 | -6 | | |
| 35,26 | -4 | 0 | | |
| 35,46 | -3 | 1 | | |
| 35,66 | -2 | -1 | | |
| 35,86 | -3 | 1 | | |

Tabel C.4 Pengujian Statis Robot Menstabilkan *Body* Terhadap Sudut *Roll* Dalam Keadaan Berjalan

| Waktu (ms) | -15° | | -7° | | 7° | | 15° | |
|---|---|---|---|---|---|---|---|---|
| | Roll (°) | Pitch (°) | Roll (°) | Pitch (°) | Roll (°) | Pitch (°) | Roll (°) | Pitch (°) |
| 20 | 0 | 10 | -5 | 0 | 7 | 1 | 6 | -9 |
| 40 | 3 | 8 | -3 | 0 | 5 | -9 | 4 | -7 |
| 60 | 6 | 6 | -1 | 5 | 2 | -7 | 0 | -5 |
| 80 | 7 | 3 | 0 | 4 | -1 | -4 | -2 | -2 |
| 100 | 5 | 2 | 5 | -1 | -4 | -2 | -3 | -1 |
| 120 | 3 | 2 | 7 | -1 | -7 | -1 | -1 | 0 |
| 140 | 2 | 1 | 1 | -1 | -5 | -1 | 2 | 1 |
| 160 | 2 | 0 | -3 | 0 | -2 | -2 | 1 | -1 |
| 180 | 3 | 0 | -3 | -1 | 0 | -2 | 0 | -2 |
| 200 | 4 | 0 | -2 | -1 | 1 | -2 | -2 | -3 |
| 220 | 4 | 0 | -2 | -2 | 1 | -2 | -4 | -3 |
| 240 | 4 | 0 | -1 | -2 | 2 | -2 | -4 | -4 |
| 260 | 3 | -1 | 0 | -2 | 2 | -2 | -5 | -4 |
| 280 | 3 | -2 | 0 | -2 | 2 | -3 | -6 | -4 |
| 300 | 2 | -2 | 0 | -2 | 2 | -3 | -5 | -5 |
| 320 | 3 | 0 | 0 | -2 | 3 | -2 | -6 | -5 |
| 340 | 3 | 0 | 1 | -3 | 3 | -3 | -6 | -5 |
| 360 | 3 | 0 | 1 | -3 | 2 | -3 | -5 | -6 |
| 380 | 3 | 0 | 2 | -3 | 1 | -4 | -5 | -6 |
| 400 | 3 | 0 | 2 | -3 | 1 | -4 | -5 | -6 |
| 420 | 2 | 0 | 2 | -3 | 1 | -4 | -6 | -6 |
| 440 | 2 | 0 | 3 | -3 | 2 | -3 | -6 | -6 |
| 460 | 2 | 0 | 3 | -3 | 2 | -2 | -7 | -4 |
| 480 | 2 | -1 | 3 | -3 | 2 | -2 | -7 | -6 |
| 500 | 2 | -1 | 3 | -4 | 2 | -3 | -7 | -6 |
| 520 | 2 | 0 | 2 | -4 | 2 | -3 | -7 | -6 |
| 540 | 2 | 0 | 2 | -4 | 3 | -3 | -7 | -6 |
| 560 | 1 | -1 | 2 | -4 | 3 | -3 | -6 | -6 |
| 580 | 1 | -2 | 1 | -3 | 3 | -3 | -5 | -6 |
| 600 | 2 | -2 | 1 | -2 | 3 | -3 | -6 | -5 |
| 620 | 4 | -1 | 0 | -2 | 4 | -2 | -5 | -6 |
| 640 | 3 | 0 | 1 | -2 | 4 | -2 | -5 | -5 |
| 660 | 2 | -2 | 1 | -2 | 3 | -2 | -5 | -5 |
| 680 | 2 | -2 | 0 | -1 | 3 | -2 | -5 | -5 |
| 700 | 0 | -2 | 0 | -1 | 3 | -2 | -4 | -6 |
| 720 | 0 | -1 | 0 | -3 | 3 | -2 | -4 | -6 |
| 740 | 0 | 0 | 0 | -3 | 3 | -2 | -4 | -7 |
| 760 | 0 | 0 | 2 | -4 | 1 | -3 | -4 | -6 |
| 780 | -1 | 1 | 1 | -3 | 1 | -3 | -4 | -6 |
| 800 | 0 | 1 | 3 | -3 | 2 | -3 | -3 | -5 |
| 820 | 0 | 0 | 3 | -3 | 2 | -2 | -3 | -6 |
| 840 | 0 | 0 | 2 | -4 | 3 | -2 | -2 | -5 |
| 860 | -2 | 3 | 1 | -4 | 2 | -2 | -2 | -4 |
| 880 | 0 | 3 | 1 | -4 | 2 | -1 | -2 | -4 |
| 900 | 0 | 2 | 1 | -3 | 2 | -2 | -2 | -4 |
| 920 | 1 | 2 | 0 | -3 | 3 | -3 | -2 | -4 |
| 940 | 1 | 2 | 1 | -1 | 3 | -2 | -2 | -4 |
| 960 | 2 | 1 | 2 | -1 | 4 | -3 | -2 | -5 |
| 980 | 3 | 1 | 0 | -1 | 3 | -2 | -2 | -3 |
| 1000 | 3 | 0 | 0 | -1 | 2 | -2 | -1 | -2 |
| 1020 | 3 | 2 | 1 | -1 | 2 | -2 | 0 | -2 |
| 1040 | 2 | 1 | 1 | -3 | 2 | -2 | 0 | -2 |
| 1060 | 3 | 1 | 1 | -4 | 3 | -3 | -1 | -3 |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 1080 | 3 | 1 | 1 | -3 | 3 | -3 | -1 | -3 |
| 1100 | 1 | 0 | 1 | -4 | 4 | -4 | -1 | -4 |
| 1120 | 1 | 0 | 0 | -2 | 2 | -4 | 0 | -3 |
| 1140 | 2 | 0 | 1 | 0 | 2 | -5 | 0 | -3 |
| 1160 | 2 | 0 | 0 | 1 | 2 | -3 | 0 | -2 |
| 1180 | 1 | 1 | -1 | 0 | 3 | -1 | 0 | -2 |
| 1200 | 1 | 0 | -3 | 0 | 3 | -2 | 1 | -2 |
| 1220 | 0 | 0 | -2 | -1 | 4 | -3 | 1 | -2 |
| 1240 | 1 | 0 | 0 | -3 | 5 | -1 | 2 | -3 |
| 1260 | 1 | 0 | 0 | -3 | 3 | -1 | 2 | -3 |
| 1280 | 2 | 0 | 0 | 0 | 2 | -2 | 3 | -3 |
| 1300 | 3 | 0 | -1 | 0 | 3 | -2 | 3 | -2 |
| 1320 | 3 | 0 | -1 | -1 | 3 | -2 | 2 | -2 |
| 1340 | 3 | 0 | -1 | -2 | 2 | -2 | 2 | -1 |
| 1360 | 3 | 0 | -5 | -3 | 2 | -2 | -1 | -1 |
| 1380 | 3 | -1 | -3 | -4 | 2 | -1 | 0 | -2 |
| 1400 | 1 | 0 | -1 | -2 | 3 | -1 | 0 | 0 |
| 1420 | 2 | -1 | -2 | -2 | 3 | -1 | 1 | 0 |
| 1440 | 2 | 0 | -1 | 1 | 3 | -2 | 0 | -2 |
| 1460 | 2 | 0 | -2 | 0 | 4 | -2 | 1 | -1 |
| 1480 | 2 | -1 | -2 | 0 | 4 | -2 | 0 | -2 |
| 1500 | 3 | -1 | -2 | 0 | 3 | -1 | -1 | -1 |
| 1520 | 1 | -2 | 0 | 0 | 3 | -1 | -1 | 0 |
| 1540 | 1 | -3 | 0 | 0 | 4 | 0 | -1 | -1 |
| 1560 | 1 | -3 | 1 | -1 | 3 | -1 | -1 | -3 |
| 1580 | 1 | -2 | 0 | -1 | 5 | -1 | 0 | -3 |
| 1600 | 1 | -2 | 0 | 0 | 5 | -2 | 0 | -3 |
| 1620 | 1 | -2 | 0 | 0 | 5 | -1 | 0 | -5 |
| 1640 | 1 | -1 | 0 | 0 | 6 | -1 | 0 | -4 |
| 1660 | 1 | -1 | 0 | 0 | 5 | -1 | 0 | -3 |
| 1680 | 1 | -1 | 0 | 0 | 4 | -1 | 1 | -3 |
| 1700 | 1 | -1 | 0 | 0 | 4 | 1 | 0 | -2 |
| 1720 | 1 | 0 | 1 | 0 | 4 | 0 | 1 | -3 |
| 1740 | 3 | 0 | 0 | 0 | 5 | -1 | 1 | -3 |
| 1760 | 1 | -2 | 0 | 3 | 4 | -1 | 3 | -3 |
| 1780 | 1 | -1 | 0 | 1 | 3 | -2 | 1 | -3 |
| 1800 | 2 | -2 | 0 | 0 | 3 | -2 | 3 | -3 |
| 1820 | 1 | -1 | -1 | 0 | 3 | -2 | 3 | -4 |
| 1840 | 3 | -2 | -1 | -1 | 3 | -2 | 3 | -4 |
| 1860 | 2 | -2 | -2 | -1 | 4 | -1 | 3 | -3 |
| 1880 | 1 | -2 | -1 | -2 | 4 | -2 | 4 | -2 |
| 1900 | 2 | 0 | -1 | -1 | 5 | -3 | 3 | -1 |
| 1920 | 1 | 0 | -2 | -2 | 5 | -2 | 2 | -3 |
| 1940 | 1 | -1 | -3 | -2 | 4 | -2 | 0 | -3 |
| 1960 | 1 | -2 | -2 | -2 | 4 | -1 | 2 | -2 |
| 1980 | 1 | -4 | -2 | -3 | 5 | 0 | 1 | -1 |
| 2000 | 2 | -3 | -1 | -3 | 5 | 0 | 0 | -2 |
| 2020 | 0 | -2 | -1 | -4 | 5 | -1 | 0 | -3 |
| 2040 | 1 | -1 | 0 | -4 | 5 | -1 | -2 | -3 |
| 2060 | 4 | -3 | 0 | -3 | 5 | -1 | -1 | -4 |
| 2080 | 0 | 0 | 0 | -1 | 4 | -2 | -2 | -4 |
| 2100 | 0 | 0 | -1 | -1 | 2 | -1 | 0 | -4 |
| 2120 | 0 | 1 | -3 | -1 | 2 | -1 | 0 | -4 |
| 2140 | -3 | 2 | -3 | -3 | 3 | 0 | 1 | -3 |
| 2160 | -2 | 2 | -2 | -3 | 4 | 0 | 2 | -3 |
| 2180 | -2 | 0 | -2 | -2 | 4 | 0 | 2 | -3 |
| 2200 | -2 | -1 | -1 | -3 | 4 | 0 | 2 | -3 |
| 2220 | 0 | -3 | -2 | -2 | 5 | -1 | 2 | -2 |
| 2240 | 0 | -4 | -2 | 0 | 6 | -2 | 1 | -3 |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 2260 | -1 | -3 | -2 | 1 | 6 | -1 | 1 | -3 |
| 2280 | -3 | 0 | -2 | 0 | 6 | -1 | 2 | -4 |
| 2300 | 0 | -3 | -2 | -1 | 6 | -2 | 2 | -4 |
| 2320 | 0 | -4 | -1 | -1 | 5 | -1 | 2 | -3 |
| 2340 | 0 | -4 | -1 | -2 | 6 | 1 | 2 | -3 |
| 2360 | 1 | -4 | 0 | -2 | 5 | 0 | 1 | -2 |
| 2380 | 3 | -4 | -1 | -2 | 5 | 0 | 2 | -3 |
| 2400 | 3 | -3 | -1 | 0 | 5 | 1 | 2 | -2 |
| 2420 | 2 | -2 | -1 | -2 | 4 | 0 | 1 | -2 |
| 2440 | 2 | -2 | -1 | -4 | 5 | 1 | 1 | -1 |
| 2460 | 2 | -2 | -3 | -4 | 5 | 1 | 1 | -1 |
| 2480 | 2 | -2 | -2 | -3 | 5 | 1 | 3 | -1 |
| 2500 | 0 | -1 | -3 | -3 | 4 | 1 | 2 | -2 |

Tabel C.5 Pengujian Dinamis Robot Menstabilkan *Body* Terhadap Sudut *Roll* Dalam Keadaan Berjalan

| Waktu (ms) | -7° | | 7° | |
|---|---|---|---|---|
| | Roll (°) | Pitch (°) | Roll (°) | Pitch (°) |
| 20 | 1 | 2 | 7 | 3 |
| 40 | 0 | 2 | 3 | 0 |
| 60 | 0 | 2 | -2 | 0 |
| 80 | 0 | 3 | -6 | -1 |
| 100 | 0 | 2 | -4 | -1 |
| 120 | 0 | 3 | 0 | -1 |
| 140 | 0 | 4 | 0 | -1 |
| 160 | -1 | 4 | -1 | -1 |
| 180 | 0 | 4 | -2 | -1 |
| 200 | -2 | 5 | -1 | -2 |
| 220 | -1 | 4 | -1 | -2 |
| 240 | 0 | 4 | -1 | 2 |
| 260 | 0 | 4 | -1 | 0 |
| 280 | -1 | 4 | -1 | 0 |
| 300 | -3 | 5 | -2 | 1 |
| 320 | -4 | 6 | -1 | 0 |
| 340 | -3 | 5 | -2 | 0 |
| 360 | -2 | 4 | -3 | 0 |
| 380 | -2 | 3 | -3 | 0 |
| 400 | -2 | 2 | -3 | 0 |
| 420 | -1 | 2 | -4 | -1 |
| 440 | -3 | 2 | -4 | -2 |
| 460 | -4 | 2 | -4 | -2 |
| 480 | -2 | 2 | -4 | -3 |
| 500 | -2 | 1 | -5 | -2 |
| 520 | 1 | 0 | -6 | 0 |
| 540 | 0 | 0 | -6 | 0 |
| 560 | 0 | 0 | -6 | 0 |
| 580 | 0 | 0 | -6 | 0 |
| 600 | 1 | 2 | -5 | 0 |
| 620 | 0 | 2 | -5 | 0 |
| 640 | 0 | 2 | -4 | 0 |
| 660 | -2 | -3 | -5 | 0 |
| 680 | 0 | -1 | -5 | 0 |
| 700 | 0 | -1 | -5 | 0 |
| 720 | 0 | -2 | -4 | 0 |
| 740 | -1 | -2 | -3 | 0 |
| 760 | -2 | 0 | -5 | 0 |
| 780 | -3 | 4 | -4 | 1 |
| 800 | -2 | 3 | -3 | 0 |
| 820 | -1 | 3 | -3 | 1 |
| 840 | 4 | -2 | 2 | 0 |
| 860 | 3 | -1 | 0 | 0 |
| 880 | 1 | 0 | 0 | 0 |
| 900 | -1 | 2 | -2 | 1 |
| 920 | 0 | 5 | -4 | 1 |
| 940 | 0 | 4 | -4 | 1 |
| 960 | 0 | 3 | -5 | 2 |
| 980 | -4 | -3 | -5 | 3 |
| 1000 | -5 | -5 | -7 | 2 |
| 1020 | -6 | -6 | -7 | 1 |
| 1040 | -7 | -6 | -6 | 0 |
| 1060 | -7 | -6 | -3 | 1 |
| 1080 | -4 | -3 | -1 | 2 |
| 1100 | -5 | -2 | -1 | 1 |
| 1120 | -6 | -3 | -1 | 1 |
| 1140 | -4 | -4 | -1 | 1 |
| 1160 | -7 | -3 | -2 | 1 |
| 1180 | -5 | -6 | -3 | 1 |
| 1200 | -3 | -5 | -2 | 0 |
| 1220 | -3 | -5 | -2 | 1 |
| 1240 | -1 | -1 | -3 | 0 |
| 1260 | -1 | -1 | -1 | 0 |
| 1280 | -1 | -1 | 0 | 1 |
| 1300 | -2 | 0 | 0 | 3 |
| 1320 | -5 | -3 | 0 | 1 |
| 1340 | -2 | 3 | 1 | 1 |
| 1360 | -2 | 1 | 1 | 2 |
| 1380 | -3 | 0 | 0 | 2 |
| 1400 | -3 | 1 | 0 | 0 |
| 1420 | -7 | 1 | 0 | 1 |
| 1440 | -7 | 0 | 1 | 2 |
| 1460 | -6 | 0 | 2 | 3 |
| 1480 | -6 | -2 | -2 | 4 |
| 1500 | -8 | -5 | 0 | 4 |
| 1520 | -7 | -2 | -1 | 5 |
| 1540 | -6 | -2 | 0 | 9 |
| 1560 | -6 | -2 | 3 | 8 |
| 1580 | 2 | -3 | 6 | 4 |
| 1600 | 1 | -2 | 5 | 4 |
| 1620 | 0 | -2 | 7 | 1 |
| 1640 | -2 | -1 | 8 | 1 |
| 1660 | -3 | 0 | 6 | 1 |
| 1680 | -4 | 0 | 7 | -3 |
| 1700 | -5 | 0 | 9 | -4 |
| 1720 | -3 | -1 | 9 | -3 |
| 1740 | -14 | -8 | 16 | -5 |
| 1760 | -9 | -12 | 12 | -3 |
| 1780 | -5 | -8 | 5 | -3 |
| 1800 | -1 | -5 | 0 | -3 |
| 1820 | 4 | -5 | -2 | -1 |
| 1840 | 7 | -2 | -6 | 0 |
| 1860 | 8 | 0 | -7 | 0 |
| 1880 | 11 | 0 | -7 | 0 |
| 1900 | 12 | 1 | -8 | 0 |
| 1920 | 10 | 3 | -8 | 0 |
| 1940 | 5 | 2 | -9 | 1 |
| 1960 | 3 | 2 | -7 | 0 |
| 1980 | 2 | 3 | -6 | 0 |
| 2000 | 1 | 4 | -7 | 0 |
| 2020 | 0 | 1 | -7 | 1 |
| 2040 | 0 | 3 | -4 | 0 |
| 2060 | -1 | 4 | -4 | 0 |
| 2080 | 0 | 4 | -5 | 0 |
| 2100 | -2 | 5 | -5 | 0 |
| 2120 | -1 | 6 | -3 | 0 |
| 2140 | 0 | 2 | -3 | 0 |
| 2160 | 0 | 3 | -2 | 1 |
| 2180 | 2 | 4 | -3 | 0 |
| 2200 | 2 | 4 | -2 | 0 |

Tabel C.6 Pengujian Statis Robot Tanpa Kalman Filter dalam Menstabilkan *Body* Terhadap Sudut *Roll* Dalam Keadaan Berjalan

| Waktu (ms) | -15° | | -7° | | 7° | | 15° | |
|---|---|---|---|---|---|---|---|---|
| | Roll (°) | Pitch (°) | Roll (°) | Pitch (°) | Roll (°) | Pitch (°) | Roll (°) | Pitch (°) |
| 20 | 0 | 0 | 2 | 4 | 5 | -2 | 2 | -2 |
| 40 | 8 | -6 | 2 | 2 | 2 | -2 | 2 | -3 |
| 60 | 0 | 0 | 2 | 4 | 1 | 0 | 4 | -2 |
| 80 | 3 | -2 | 0 | 4 | 4 | -2 | 3 | -2 |
| 100 | 2 | -4 | -2 | -2 | 3 | -1 | 2 | -3 |
| 120 | 7 | 2 | 1 | 12 | 4 | -2 | -1 | -3 |
| 140 | 1 | 0 | 2 | -2 | 2 | -1 | -12 | 0 |
| 160 | 3 | 0 | 4 | 4 | -6 | 3 | 0 | -7 |
| 180 | 1 | -2 | 4 | 7 | 2 | -1 | 0 | 0 |
| 200 | 2 | -2 | 4 | 7 | 1 | -1 | 1 | 2 |
| 220 | 2 | -2 | 2 | 4 | 4 | -1 | 0 | -2 |
| 240 | 3 | -1 | 2 | 4 | 6 | -5 | 2 | -7 |
| 260 | 1 | 0 | 3 | 5 | 3 | 0 | 5 | -6 |
| 280 | 0 | -3 | -12 | 8 | 2 | -2 | -7 | 0 |
| 300 | 1 | -1 | -4 | 5 | 2 | 1 | 1 | -5 |
| 320 | 0 | 2 | 13 | 3 | -7 | 2 | 0 | 0 |
| 340 | -2 | 4 | 10 | 12 | 2 | 0 | 0 | -5 |
| 360 | 6 | -1 | 7 | 8 | 4 | 1 | 2 | 0 |
| 380 | 2 | 1 | 6 | 3 | 6 | 0 | 2 | -3 |
| 400 | 1 | 3 | 3 | 5 | 1 | -5 | 3 | -5 |
| 420 | 2 | -2 | 4 | 5 | 6 | -2 | 8 | -1 |
| 440 | 3 | 0 | -4 | 0 | 5 | -1 | 1 | -4 |
| 460 | 1 | -1 | 3 | -2 | 3 | -2 | -1 | -2 |
| 480 | 6 | 1 | 3 | 4 | -10 | -8 | 1 | -4 |
| 500 | 1 | 0 | 6 | 0 | 3 | -3 | 2 | -4 |
| 520 | 3 | -6 | 7 | 2 | 7 | 0 | 1 | -6 |
| 540 | 3 | 0 | 2 | 3 | 4 | 0 | 0 | -3 |
| 560 | 3 | -2 | 0 | 4 | 6 | -2 | 1 | -3 |
| 580 | 2 | -2 | 5 | 3 | 4 | 0 | 2 | -3 |
| 600 | 3 | -1 | 1 | 8 | 7 | -5 | -8 | 1 |
| 620 | 2 | -2 | 6 | 3 | 3 | -5 | 3 | -5 |
| 640 | 2 | -1 | 7 | 3 | 3 | -7 | 2 | 0 |
| 660 | 1 | -1 | 3 | 3 | 6 | -5 | 0 | -5 |
| 680 | 6 | -3 | 8 | 8 | 4 | -2 | 0 | -5 |
| 700 | 3 | -2 | 4 | 3 | 6 | -5 | 2 | -5 |
| 720 | 3 | -2 | 4 | 5 | 2 | -4 | 4 | -3 |
| 740 | 3 | -2 | 4 | 5 | 5 | -2 | 0 | -2 |
| 760 | 2 | -2 | -8 | -1 | 1 | -5 | -3 | 1 |
| 780 | 4 | -1 | 9 | 1 | 2 | -2 | 0 | -5 |
| 800 | 4 | -2 | 5 | 0 | -7 | -6 | 1 | -5 |
| 820 | 2 | -2 | 4 | 4 | 0 | -1 | 2 | -3 |
| 840 | 8 | -4 | 0 | -2 | 2 | -3 | 2 | -5 |
| 860 | 3 | -4 | 3 | 3 | 6 | -1 | 1 | -4 |
| 880 | 4 | -2 | 0 | 7 | 2 | -1 | 2 | -2 |
| 900 | 3 | -2 | 3 | 8 | 3 | -1 | 3 | -2 |
| 920 | 5 | 4 | 6 | 1 | 4 | -7 | -5 | 0 |
| 940 | 2 | -2 | 5 | 7 | 3 | -2 | 3 | -7 |
| 960 | 2 | -3 | 9 | 9 | -5 | 9 | 0 | 2 |
| 980 | 2 | -2 | 4 | 3 | 1 | -6 | 0 | -2 |
| 1000 | 0 | -1 | 9 | 2 | -1 | -10 | 1 | -6 |
| 1020 | 4 | -5 | 2 | 4 | 2 | -4 | 6 | -2 |
| 1040 | 7 | -1 | 3 | 4 | 3 | -4 | 1 | -8 |
| 1060 | 4 | -2 | 4 | 3 | 3 | -7 | 0 | -3 |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 1080 | 3 | 2 | 0 | 6 | 2 | -7 | -10 | 2 |
| 1100 | 2 | 0 | 6 | 3 | 8 | -2 | 1 | -4 |
| 1120 | 1 | -1 | 7 | 1 | -3 | -2 | 1 | -3 |
| 1140 | 2 | -1 | 4 | 2 | 0 | -3 | 2 | -4 |
| 1160 | 1 | -2 | 6 | 10 | 8 | 2 | 2 | -4 |
| 1180 | 2 | -3 | 2 | 3 | 0 | -7 | 4 | -4 |
| 1200 | 3 | 0 | 4 | 4 | 2 | -1 | 2 | -4 |
| 1220 | 2 | -3 | 3 | 3 | 3 | -3 | 5 | -4 |
| 1240 | 8 | -2 | -8 | -3 | 4 | -2 | 0 | 3 |
| 1260 | 3 | -2 | 5 | 4 | 3 | -2 | 3 | -4 |
| 1280 | 3 | 0 | 8 | 0 | -5 | -2 | 1 | -5 |
| 1300 | 3 | -3 | 4 | 4 | 3 | -3 | 3 | -5 |
| 1320 | 4 | -1 | 7 | 8 | 4 | -4 | 1 | -8 |
| 1340 | 3 | -3 | 4 | 2 | 7 | -2 | 2 | -5 |
| 1360 | 2 | -3 | 5 | 5 | 2 | -4 | 1 | -6 |
| 1380 | 4 | -2 | 5 | 2 | 0 | -1 | 5 | -3 |
| 1400 | 4 | -3 | -8 | -2 | 2 | -3 | 4 | -3 |
| 1420 | 4 | -1 | 6 | 2 | 3 | -3 | 3 | -4 |
| 1440 | 1 | -4 | 8 | 1 | -7 | -5 | 1 | -3 |
| 1460 | 2 | -3 | 8 | -2 | 3 | 7 | 1 | -3 |
| 1480 | 5 | 0 | 2 | 8 | 3 | -3 | 6 | -8 |
| 1500 | 3 | -3 | 5 | 3 | 4 | -3 | 3 | -3 |
| 1520 | 1 | 0 | 1 | 6 | 1 | -5 | 4 | -5 |
| 1540 | 4 | 4 | 2 | 8 | 6 | -5 | 8 | -9 |
| 1560 | 3 | -4 | -1 | 0 | 3 | -4 | -9 | 0 |
| 1580 | 3 | -2 | 5 | 3 | 3 | -3 | 4 | -2 |
| 1600 | 2 | 0 | 8 | -1 | -10 | -9 | 2 | -6 |
| 1620 | 2 | 0 | 7 | 3 | 4 | -1 | 3 | -1 |
| 1640 | 8 | -4 | 9 | 5 | 9 | 0 | 3 | 2 |
| 1660 | 3 | -2 | 5 | 2 | 6 | -1 | 0 | -2 |
| 1680 | 2 | -1 | 5 | 1 | 7 | -2 | 3 | -8 |
| 1700 | 4 | -2 | 5 | 3 | 1 | -4 | 5 | -7 |
| 1720 | 9 | 2 | -8 | 3 | 4 | 8 | 0 | -3 |
| 1740 | 1 | -2 | 6 | -2 | 4 | -3 | 4 | -2 |
| 1760 | 2 | -1 | 8 | 5 | -9 | 0 | 0 | -8 |
| 1780 | 0 | 0 | 7 | 4 | 4 | -4 | 2 | -2 |
| 1800 | 8 | -2 | 9 | 0 | 7 | -2 | -4 | 2 |
| 1820 | 2 | -1 | 2 | 4 | 9 | -7 | 1 | -3 |
| 1840 | 5 | -1 | 5 | 2 | 5 | -4 | 4 | -3 |
| 1860 | 1 | 3 | 3 | -7 | 4 | -4 | 1 | -1 |
| 1880 | 4 | 0 | -2 | 3 | -1 | -7 | -9 | -3 |
| 1900 | -1 | 1 | 5 | 4 | 3 | -1 | 2 | -2 |
| 1920 | 0 | 0 | 4 | 4 | -8 | -4 | 4 | -4 |
| 1940 | 2 | -1 | 4 | 1 | 5 | -4 | 4 | -4 |
| 1960 | 6 | -3 | 8 | 6 | 10 | -7 | 4 | 0 |
| 1980 | 3 | -1 | 5 | 0 | 5 | -5 | 6 | -4 |
| 2000 | 3 | -2 | 6 | 1 | 2 | -5 | 6 | -5 |
| 2020 | 2 | -1 | 6 | 2 | 3 | -3 | 3 | -2 |
| 2040 | 0 | 0 | -7 | -1 | -1 | -3 | -7 | 0 |
| 2060 | 1 | 0 | 6 | 1 | 3 | -6 | 2 | -1 |
| 2080 | 5 | 0 | 8 | 1 | -6 | 3 | 1 | -4 |
| 2100 | 3 | 0 | 5 | 1 | 7 | 0 | 3 | -4 |
| 2120 | 6 | -4 | 5 | 1 | 0 | -9 | 3 | 4 |
| 2140 | 0 | -2 | 6 | 2 | 0 | 2 | 2 | -2 |
| 2160 | 0 | 0 | 6 | 2 | 4 | -3 | 1 | -6 |
| 2180 | 2 | -2 | 4 | 1 | 2 | 0 | 4 | -6 |
| 2200 | 0 | 3 | -7 | 3 | 0 | 0 | -9 | 1 |
| 2220 | 0 | -1 | 6 | 2 | 4 | -2 | 7 | -7 |
| 2240 | 0 | -1 | 4 | 4 | 0 | 2 | 0 | -6 |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 2260 | 3 | 0 | 3 | -1 | 5 | 0 | 1 | -3 |
| 2280 | 8 | -3 | 5 | 1 | 4 | -2 | 3 | -6 |
| 2300 | 3 | -2 | 5 | 1 | 4 | -2 | 2 | -5 |
| 2320 | 4 | -1 | 2 | 2 | 4 | 4 | -2 | -3 |
| 2340 | 3 | -3 | 4 | 4 | 2 | -3 | 3 | -5 |
| 2360 | 3 | -1 | -4 | 0 | 2 | -3 | 3 | -6 |
| 2380 | 1 | -2 | 6 | 4 | 4 | -1 | 2 | -5 |
| 2400 | 0 | 1 | 4 | 2 | -8 | -5 | 3 | -3 |
| 2420 | 4 | -1 | 7 | 5 | 2 | -2 | 5 | -4 |
| 2440 | 4 | -3 | 6 | 0 | 7 | -2 | 4 | -6 |
| 2460 | 4 | -3 | 3 | 1 | 5 | -2 | 5 | -4 |
| 2480 | 6 | -4 | 0 | 4 | 4 | 0 | 6 | 0 |
| 2500 | 4 | -3 | 4 | 4 | 2 | -1 | 4 | -3 |

Tabel C.7 Pengujian Dinamis Robot Tanpa Kalman Filter dalam Menstabilkan *Body* Terhadap Sudut *Roll* Dalam Keadaan Berjalan

| Waktu (ms) | -7° | | 7° | |
|---|---|---|---|---|
| | Roll (°) | Pitch (°) | Roll (°) | Pitch (°) |
| 20 | 6 | -3 | -8 | -7 |
| 40 | 8 | 0 | -7 | -8 |
| 60 | 4 | 2 | -5 | -7 |
| 80 | 4 | 5 | -6 | -6 |
| 100 | 3 | 9 | -7 | -4 |
| 120 | 3 | 12 | -6 | 0 |
| 140 | 3 | 6 | -6 | -4 |
| 160 | -2 | 7 | -5 | -4 |
| 180 | 1 | 7 | -6 | -4 |
| 200 | 1 | 10 | -6 | -6 |
| 220 | 2 | 11 | -4 | -5 |
| 240 | 2 | 12 | -2 | -5 |
| 260 | 1 | 8 | -1 | -5 |
| 280 | 1 | 7 | 0 | -7 |
| 300 | 1 | 8 | -4 | -9 |
| 320 | 1 | 8 | -3 | -7 |
| 340 | 0 | 8 | -3 | -7 |
| 360 | 1 | 8 | -2 | -8 |
| 380 | 5 | 8 | -4 | -7 |
| 400 | 5 | 9 | -4 | -6 |
| 420 | 3 | 4 | -4 | -4 |
| 440 | 4 | 4 | -5 | -3 |
| 460 | 7 | 3 | -5 | -2 |
| 480 | 9 | 5 | -5 | -3 |
| 500 | 8 | 5 | -6 | -3 |
| 520 | 7 | 7 | -7 | -2 |
| 540 | 6 | 8 | -8 | -1 |
| 560 | 4 | 7 | -5 | -2 |
| 580 | 2 | 6 | -6 | -1 |
| 600 | 3 | 7 | -5 | 0 |
| 620 | 4 | 8 | -3 | -3 |
| 640 | 5 | 8 | -3 | -4 |
| 660 | 7 | 7 | -4 | -4 |
| 680 | 8 | 8 | -3 | -5 |
| 700 | 7 | 9 | -5 | -7 |
| 720 | 6 | 11 | -4 | -6 |
| 740 | 3 | 4 | -3 | -8 |
| 760 | 1 | 4 | -3 | -8 |
| 780 | -1 | 5 | -4 | -8 |
| 800 | -1 | 3 | -1 | -5 |
| 820 | 0 | 2 | 0 | -4 |
| 840 | 0 | 3 | 0 | -2 |
| 860 | -2 | 5 | 0 | -1 |
| 880 | 0 | 8 | -1 | 0 |
| 900 | 1 | -2 | 0 | 2 |
| 920 | 1 | -3 | 0 | 5 |
| 940 | 0 | -1 | -1 | -1 |
| 960 | 0 | 0 | 0 | -1 |
| 980 | 0 | 2 | 1 | -2 |
| 1000 | 0 | 4 | 0 | -3 |
| 1020 | 0 | 9 | 0 | -2 |
| 1040 | -1 | 12 | 0 | -1 |
| 1060 | 0 | 0 | 0 | -2 |
| 1080 | 0 | 0 | 0 | -4 |
| 1100 | -2 | 1 | -1 | -4 |
| 1120 | -2 | 2 | -2 | -4 |
| 1140 | 0 | 3 | -1 | -4 |
| 1160 | 0 | 4 | 0 | -1 |
| 1180 | 0 | 7 | 1 | 0 |
| 1200 | 1 | 9 | 0 | 2 |
| 1220 | 0 | 2 | 0 | 6 |
| 1240 | -1 | 1 | 0 | -2 |
| 1260 | -1 | 2 | 0 | -2 |
| 1280 | 0 | 2 | -1 | -2 |
| 1300 | -1 | 2 | -1 | -2 |
| 1320 | 0 | 3 | 0 | -2 |
| 1340 | 0 | 4 | 0 | -2 |
| 1360 | 0 | 5 | 0 | 0 |
| 1380 | 0 | 0 | 0 | 1 |
| 1400 | 2 | 0 | 1 | -4 |
| 1420 | 2 | 0 | 0 | -3 |
| 1440 | 5 | 0 | 1 | -1 |
| 1460 | 2 | 3 | 1 | 0 |
| 1480 | 0 | 2 | 0 | 2 |
| 1500 | 0 | 2 | -1 | -7 |
| 1520 | 0 | 2 | -1 | -5 |
| 1540 | 0 | -2 | -1 | -5 |
| 1560 | 0 | -3 | -1 | -3 |
| 1580 | 0 | -3 | -1 | -1 |
| 1600 | 1 | -3 | 0 | 2 |
| 1620 | 1 | 0 | -1 | 4 |
| 1640 | 0 | 1 | -3 | 7 |
| 1660 | 0 | 3 | -2 | -2 |
| 1680 | 0 | 6 | -3 | -2 |
| 1700 | 0 | -1 | -3 | 0 |
| 1720 | 0 | -1 | -2 | -1 |
| 1740 | 0 | -1 | 0 | 0 |
| 1760 | 2 | -2 | 2 | 0 |
| 1780 | -2 | 0 | 3 | -1 |
| 1800 | 0 | 0 | 5 | -2 |
| 1820 | 2 | 3 | 5 | -3 |
| 1840 | 2 | 5 | 7 | -4 |
| 1860 | -3 | -10 | 9 | -4 |
| 1880 | -6 | -12 | 12 | -4 |
| 1900 | -8 | -14 | 5 | -3 |
| 1920 | -12 | -11 | 4 | -2 |
| 1940 | -4 | -6 | 2 | -1 |
| 1960 | 1 | 1 | 2 | 0 |
| 1980 | 4 | 11 | 1 | -1 |
| 2000 | 6 | 13 | 3 | -2 |
| 2020 | 9 | -1 | 4 | -2 |
| 2040 | 2 | -1 | 1 | -2 |
| 2060 | 1 | 0 | 1 | -3 |
| 2080 | 6 | -1 | 2 | -3 |
| 2100 | 5 | 0 | 1 | -3 |
| 2120 | -1 | 1 | 0 | -3 |
| 2140 | -7 | 5 | 0 | -3 |
| 2160 | -4 | 5 | 1 | -3 |
| 2180 | 1 | 2 | 1 | -3 |
| 2200 | 0 | 3 | 1 | -3 |

## D. Program Arduino

```
#include <Wire.h>
#include <Kalman.h>
#define RESTRICT_PITCH
Kalman kalmanX;
Kalman kalmanY;
double accX, accY, accZ;
double gyroX, gyroY, gyroZ;
int16_t tempRaw;
double gyroXangle, gyroYangle;
double compAngleX, compAngleY;
int kalAngleX, kalAngleY;

uint32_t timer;
uint8_t i2cData[14];

#include <LiquidCrystal.h>
LiquidCrystal lcd(53,51,49,47,43,41);

#include<SPI.h>
#include<SD.h>
const int chipSelect = 39;

#include<Servo.h>
Servo coxa1, coxa2, coxa3, coxa4, femur1, femur2, femur3, femur4, tibia1,
tibia2, tibia3, tibia4;
float coxa=2;
float femur=8;
float tibia=10.2;
float r=11.0;
float M1, M2, M3, M4;
float tinggi1, tinggi2, tinggi3, tinggi4;
int c1=82, c2=88-10, c3=90+10, c4=95;
int f1=178, f2=172, f3=174, f4=178;
int tib1=10, tib2=10, tib3=7, tib4=7;

double t1=0, t2=0, t3=0, t4=0;
double t[4]={t1, t2, t3, t4};
double L1=0,L2=0,L3=0,L4=0;
double L[4]={L1,L2,L3,L4};
double y1=0, y2=0, y3=0, y4=0;
double y[4]={y1,y2,y3,y4};
double Al1=0, Al2=0, Al3=0, Al4=0;
double Al[4]={Al1,Al2,Al3,Al4};
double Bl1=0,Bl2=0,Bl3=0,Bl4=0;
double Bl[4]={Bl1,Bl2,Bl3,Bl4};
double Alf1=0, Alf2=0, Alf3=0, Alf4=0;
double Alf[4]={Alf1, Alf2, Alf3, Alf4};
double C1=0, C2=0, C3=0, C4=0;
double B[4]={C1, C2, C3, C4};

double g1=0, g1a=0;
double gg=0, gga=0;
double x1=0, x2=0, x2a=0, x3=0, x3a=0, x4=0;
double z1=0, z2=0, z3=0, z3a=0, z4=0, z4a=0;

int AccX=0;
int AccY=0;
int baca=0;
int bici=0;
int boco=0;
```

```
int bucu=0;

float lasterrorT0, lasterrorT1, lasterrorT2, lasterrorT3;
float errorT0, errorT1, errorT2, errorT3;
float It0, It1, It2, It3;
float KP=1.8, KI=0.05, KD=0.0125;

String dataString;
File logFile;

void setup() {
  Serial.begin(115200);
  pinMode(22,INPUT);
  pinMode(24,INPUT);
  pinMode(26,INPUT);
  pinMode(28,INPUT);
  pinMode(30,INPUT);
  pinMode(32,INPUT);
  pinMode(14,OUTPUT);
  pinMode(15,OUTPUT);
  pinMode(16,OUTPUT);
  pinMode(17,OUTPUT);
  Wire.begin();
  #if ARDUINO >= 157
  Wire.setClock(400000UL);
#else
  TWBR = ((F_CPU / 400000UL) - 16) / 2;
#endif

  i2cData[0] = 7;
  i2cData[1] = 0x00;
  i2cData[2] = 0x00;
  i2cData[3] = 0x00;
  while (i2cWrite(0x19, i2cData, 4, false));
  while (i2cWrite(0x6B, 0x01, true));
  while (i2cRead(0x75, i2cData, 1));
  if (i2cData[0] != 0x68) {
    Serial.print(F("Error reading sensor"));
    while (1);
  }

  delay(100);

  while (i2cRead(0x3B, i2cData, 6));
  accX = (int16_t)((i2cData[0] << 8) | i2cData[1]);
  accY = (int16_t)((i2cData[2] << 8) | i2cData[3]);
  accZ = (int16_t)((i2cData[4] << 8) | i2cData[5]);

#ifdef RESTRICT_PITCH
  double roll  = atan2(accY, accZ) * RAD_TO_DEG;
  double pitch = atan(-accX / sqrt(accY * accY + accZ * accZ)) *
RAD_TO_DEG;
#else
  double roll  = atan(accY / sqrt(accX * accX + accZ * accZ)) *
RAD_TO_DEG;
  double pitch = atan2(-accX, accZ) * RAD_TO_DEG;
#endif

  kalmanX.setAngle(roll);
  kalmanY.setAngle(pitch);
  gyroXangle = roll;
  gyroYangle = pitch;
  compAngleX = roll;
```

```
  compAngleY = pitch;

  timer = micros();
  coxa1.attach(13);   femur1.attach(9);    tibia1.attach(5);
  coxa2.attach(12);   femur2.attach(8);    tibia2.attach(4);
  coxa3.attach(11);   femur3.attach(7);    tibia3.attach(3);
  coxa4.attach(10);   femur4.attach(6);    tibia4.attach(44);
  lcd.begin(16, 2);
  lcd.print("Initializing...");
  delay(1000);
  lcd.clear();
  pinMode(39, OUTPUT);
  digitalWrite(39, HIGH);
  if (!SD.begin(chipSelect))
  {
    lcd.print("Card Failed or");
    lcd.setCursor(0, 1);
    lcd.print("Not present");
    delay(1000);
    return;
  }
  lcd.clear();
  File logFile = SD.open("dataku.txt", FILE_WRITE);
  if (logFile)
  {
    String header = "X \t Y \t errorT0 \t errorT1 \t errorT2 \t errorT3
\t t0 \t t1 \t t2 \t t3 \t Alf0 \t Alf1 \t Alf2 \t Alf3 \t B0 \t B1 \t B2
\t B3";
    logFile.println(header);
    logFile.close();
  }
}

void acc(){
  while (i2cRead(0x3B, i2cData, 14));
  accX = (int16_t)((i2cData[0] << 8) | i2cData[1]);
  accY = (int16_t)((i2cData[2] << 8) | i2cData[3]);
  accZ = (int16_t)((i2cData[4] << 8) | i2cData[5]);
  tempRaw = (int16_t)((i2cData[6] << 8) | i2cData[7]);
  gyroX = (int16_t)((i2cData[8] << 8) | i2cData[9]);
  gyroY = (int16_t)((i2cData[10] << 8) | i2cData[11]);
  gyroZ = (int16_t)((i2cData[12] << 8) | i2cData[13]);;

  double dt = (double)(micros() - timer) / 1000000;
  timer = micros();

#ifdef RESTRICT_PITCH
  double roll  = atan2(accY, accZ) * RAD_TO_DEG;
  double pitch = atan(-accX / sqrt(accY * accY + accZ * accZ)) *
RAD_TO_DEG;
#else // Eq. 28 and 29
  double roll  = atan(accY / sqrt(accX * accX + accZ * accZ)) *
RAD_TO_DEG;
  double pitch = atan2(-accX, accZ) * RAD_TO_DEG;
#endif

  double gyroXrate = gyroX / 131.0; // Convert to deg/s
  double gyroYrate = gyroY / 131.0;

#ifdef RESTRICT_PITCH
  // This fixes the transition problem when the accelerometer angle jumps
between -180 and 180 degrees
  if ((roll < -90 && kalAngleX > 90) || (roll > 90 && kalAngleX < -90)) {
```

```
    kalmanX.setAngle(roll);
    compAngleX = roll;
    kalAngleX = roll;
    gyroXangle = roll;
  } else
    kalAngleX = kalmanX.getAngle(roll, gyroXrate, dt);

  if (abs(kalAngleX) > 90)
    gyroYrate = -gyroYrate;
  kalAngleY = kalmanY.getAngle(pitch, gyroYrate, dt);
#else
  if ((pitch < -90 && kalAngleY > 90) || (pitch > 90 && kalAngleY < -90))
{
    kalmanY.setAngle(pitch);
    compAngleY = pitch;
    kalAngleY = pitch;
    gyroYangle = pitch;
  } else
    kalAngleY = kalmanY.getAngle(pitch, gyroYrate, dt);

  if (abs(kalAngleY) > 90)
    gyroXrate = -gyroXrate;
  kalAngleX = kalmanX.getAngle(roll, gyroXrate, dt);
#endif

  gyroXangle += gyroXrate * dt;
  gyroYangle += gyroYrate * dt;

  compAngleX = 0.93 * (compAngleX + gyroXrate * dt) + 0.07 * roll;
  compAngleY = 0.93 * (compAngleY + gyroYrate * dt) + 0.07 * pitch;

  if (gyroXangle < -180 || gyroXangle > 180)
    gyroXangle = kalAngleX;
  if (gyroYangle < -180 || gyroYangle > 180)
    gyroYangle = kalAngleY;

  Serial.print(roll); Serial.print("\t");
  Serial.print(kalAngleX); Serial.print("\t");

  Serial.print("\t");

  Serial.print(pitch); Serial.print("\t");
  Serial.print(kalAngleY); Serial.print("\t");
  lcd.setCursor(0,0);
  lcd.print(kalAngleX);
  lcd.setCursor(4,0);
  lcd.print(kalAngleY);
  AccX=baca+bici;
  if(kalAngleX>0){
    if(kalAngleX>baca){
      baca=kalAngleX;
      }
    }
  if(kalAngleX<0){
    if(kalAngleX<bici){
      bici=kalAngleX;
      }
    }
  AccY=boco+bucu;
  if(kalAngleY>0){
    if(kalAngleY>boco){
      boco=kalAngleY;
      }
```

```
    }
  if(kalAngleY<0){
    if(kalAngleY<bucu){
      bucu=kalAngleY;
      }
    }
}

void loop() {
  digitalWrite(14,HIGH);
  digitalWrite(15,LOW);
  standby();
  dataString = String(kalAngleX) + "\t" + String(kalAngleY) + "\t" +
String(errorT0)+ "\t" + String(errorT1) + "\t" + String(errorT2) + "\t" +
String(errorT3) + "\t" + String(t[0]) + "\t" + String(t[1]) + "\t" +
String(t[2]) + "\t" + String(t[3]) + "\t" + String(Alf[0]) + "\t" +
String(Alf[1]) + "\t" + String(Alf[2]) + "\t" + String(Alf[3]) + "\t" +
String(B[0]) + "\t" + String(B[1]) + "\t" + String(B[2]) + "\t" +
String(B[3]);
  logFile = SD.open("dataku.txt", FILE_WRITE);
  if (logFile)
  {
    logFile.println(dataString);
    logFile.close();
  }
  if(digitalRead(22)==HIGH){
    for(;;){
      digitalWrite(14,LOW);
      digitalWrite(15,HIGH);
      if(digitalRead(24)==HIGH){
        break;
        }
      langkah1();
      dataString = String(kalAngleX) + "\t" + String(kalAngleY) + "\t" +
String(errorT0)+ "\t" + String(errorT1) + "\t" + String(errorT2) + "\t" +
String(errorT3) + "\t" + String(t[0]) + "\t" + String(t[1]) + "\t" +
String(t[2]) + "\t" + String(t[3]) + "\t" + String(Alf[0]) + "\t" +
String(Alf[1]) + "\t" + String(Alf[2]) + "\t" + String(Alf[3]) + "\t" +
String(B[0]) + "\t" + String(B[1]) + "\t" + String(B[2]) + "\t" +
String(B[3]);
      logFile = SD.open("dataku.txt", FILE_WRITE);
      if (logFile)
      {
        logFile.println(dataString);
        logFile.close();
      }
      delay(100);
      if(digitalRead(24)==HIGH){
        break;
        }
      langkah2();
      dataString = String(kalAngleX) + "\t" + String(kalAngleY) + "\t" +
String(errorT0)+ "\t" + String(errorT1) + "\t" + String(errorT2) + "\t" +
String(errorT3) + "\t" + String(t[0]) + "\t" + String(t[1]) + "\t" +
String(t[2]) + "\t" + String(t[3]) + "\t" + String(Alf[0]) + "\t" +
String(Alf[1]) + "\t" + String(Alf[2]) + "\t" + String(Alf[3]) + "\t" +
String(B[0]) + "\t" + String(B[1]) + "\t" + String(B[2]) + "\t" +
String(B[3]);
      logFile = SD.open("dataku.txt", FILE_WRITE);
      if (logFile)
      {
        logFile.println(dataString);
        logFile.close();
```

```
      }
      delay(100);
      if(digitalRead(24)==HIGH){
        break;
        }
      langkah3();
      dataString = String(kalAngleX) + "\t" + String(kalAngleY) + "\t" +
String(errorT0)+ "\t" + String(errorT1) + "\t" + String(errorT2) + "\t" +
String(errorT3) + "\t" + String(t[0]) + "\t" + String(t[1]) + "\t" +
String(t[2]) + "\t" + String(t[3]) + "\t" + String(Alf[0]) + "\t" +
String(Alf[1]) + "\t" + String(Alf[2]) + "\t" + String(Alf[3]) + "\t" +
String(B[0]) + "\t" + String(B[1]) + "\t" + String(B[2]) + "\t" +
String(B[3]);
      logFile = SD.open("dataku.txt", FILE_WRITE);
      if (logFile)
      {
        logFile.println(dataString);
        logFile.close();
      }
      delay(100);
      if(digitalRead(24)==HIGH){
        break;
        }
      langkah4();
      dataString = String(kalAngleX) + "\t" + String(kalAngleY) + "\t" +
String(errorT0)+ "\t" + String(errorT1) + "\t" + String(errorT2) + "\t" +
String(errorT3) + "\t" + String(t[0]) + "\t" + String(t[1]) + "\t" +
String(t[2]) + "\t" + String(t[3]) + "\t" + String(Alf[0]) + "\t" +
String(Alf[1]) + "\t" + String(Alf[2]) + "\t" + String(Alf[3]) + "\t" +
String(B[0]) + "\t" + String(B[1]) + "\t" + String(B[2]) + "\t" +
String(B[3]);
      logFile = SD.open("dataku.txt", FILE_WRITE);
      if (logFile)
      {
        logFile.println(dataString);
        logFile.close();
      }
      delay(100);
      if(digitalRead(24)==HIGH){
        break;
        }
      langkah5();
      dataString = String(kalAngleX) + "\t" + String(kalAngleY) + "\t" +
String(errorT0)+ "\t" + String(errorT1) + "\t" + String(errorT2) + "\t" +
String(errorT3) + "\t" + String(t[0]) + "\t" + String(t[1]) + "\t" +
String(t[2]) + "\t" + String(t[3]) + "\t" + String(Alf[0]) + "\t" +
String(Alf[1]) + "\t" + String(Alf[2]) + "\t" + String(Alf[3]) + "\t" +
String(B[0]) + "\t" + String(B[1]) + "\t" + String(B[2]) + "\t" +
String(B[3]);
      logFile = SD.open("dataku.txt", FILE_WRITE);
      if (logFile)
      {
        logFile.println(dataString);
        logFile.close();
      }
      delay(100);
      if(digitalRead(24)==HIGH){
        break;
        }
      langkah6();
      dataString = String(kalAngleX) + "\t" + String(kalAngleY) + "\t" +
String(errorT0)+ "\t" + String(errorT1) + "\t" + String(errorT2) + "\t" +
String(errorT3) + "\t" + String(t[0]) + "\t" + String(t[1]) + "\t" +
```

```
String(t[2]) + "\t" + String(t[3]) + "\t" + String(Alf[0]) + "\t" +
String(Alf[1]) + "\t" + String(Alf[2]) + "\t" + String(Alf[3]) + "\t" +
String(B[0]) + "\t" + String(B[1]) + "\t" + String(B[2]) + "\t" +
String(B[3]);
      logFile = SD.open("dataku.txt", FILE_WRITE);
      if (logFile)
      {
        logFile.println(dataString);
        logFile.close();
      }
      delay(100);
      if(digitalRead(24)==HIGH){
        break;
        }
      langkah7();
      dataString = String(kalAngleX) + "\t" + String(kalAngleY) + "\t" +
String(errorT0)+ "\t" + String(errorT1) + "\t" + String(errorT2) + "\t" +
String(errorT3) + "\t" + String(t[0]) + "\t" + String(t[1]) + "\t" +
String(t[2]) + "\t" + String(t[3]) + "\t" + String(Alf[0]) + "\t" +
String(Alf[1]) + "\t" + String(Alf[2]) + "\t" + String(Alf[3]) + "\t" +
String(B[0]) + "\t" + String(B[1]) + "\t" + String(B[2]) + "\t" +
String(B[3]);
      logFile = SD.open("dataku.txt", FILE_WRITE);
      if (logFile)
      {
        logFile.println(dataString);
        logFile.close();
      }
      delay(100);
      if(digitalRead(24)==HIGH){
        break;
        }
      langkah8();
      dataString = String(kalAngleX) + "\t" + String(kalAngleY) + "\t" +
String(errorT0)+ "\t" + String(errorT1) + "\t" + String(errorT2) + "\t" +
String(errorT3) + "\t" + String(t[0]) + "\t" + String(t[1]) + "\t" +
String(t[2]) + "\t" + String(t[3]) + "\t" + String(Alf[0]) + "\t" +
String(Alf[1]) + "\t" + String(Alf[2]) + "\t" + String(Alf[3]) + "\t" +
String(B[0]) + "\t" + String(B[1]) + "\t" + String(B[2]) + "\t" +
String(B[3]);
      logFile = SD.open("dataku.txt", FILE_WRITE);
      if (logFile)
      {
        logFile.println(dataString);
        logFile.close();
      }
      delay(100);
      if(digitalRead(24)==HIGH){
        break;
        }
      }
    }
}

void IK_body(){
  acc();
  t[2] =r*sin(AccY*3.1412/180);
  t[1] =r*sin(AccX*(-1)*3.1412/180);
  t[0] =r*sin(AccY*(-1)*3.1412/180);
  t[3] =r*sin(AccX*3.1412/180);
  lasterrorT0=errorT0;
  lasterrorT1=errorT1;
  lasterrorT2=errorT2;
```

```
  lasterrorT3=errorT3;
  errorT0=0-t[0];
  errorT1=0-t[1];
  errorT2=0-t[2];
  errorT3=0-t[3];
  It0+=errorT0;
  It1+=errorT1;
  It2+=errorT2;
  It3+=errorT3;
  t[0]=(KP*errorT0)+(KI*It0)+(KD*(errorT0-lasterrorT0));
  t[1]=(KP*errorT1)+(KI*It1)+(KD*(errorT1-lasterrorT1));
  t[2]=(KP*errorT2)+(KI*It2)+(KD*(errorT2-lasterrorT2));
  t[3]=(KP*errorT3)+(KI*It3)+(KD*(errorT3-lasterrorT3));
  lcd.setCursor(0,1);
  lcd.print(errorT0);
  lcd.setCursor(4,1);
  lcd.print(errorT1);
  lcd.setCursor(8,1);
  lcd.print(errorT2);
  lcd.setCursor(12,1);
  lcd.print(errorT3);
  Serial.print(errorT0); Serial.print("\t");
  Serial.print(errorT1); Serial.print("\t");
  Serial.print(errorT2); Serial.print("\t");
  Serial.print(errorT3); Serial.print("\t");
  Serial.print(t[0]); Serial.print("\t");
  Serial.print(t[1]); Serial.print("\t");
  Serial.print(t[2]); Serial.print("\t");
  Serial.print(t[3]); Serial.print("\t");
}

void standby(){
  M1=5.0;         M2=5.0;         M3=5.0;         M4=5.0;
  tinggi1=6.6;    tinggi2=7.3;    tinggi3=7.3;    tinggi4=6.6;
  IK_body();
  IK_kaki();
  proteksi();
  coxa1.write(c1);     femur1.write(f1-Alf[0]);
tibia1.write(tib1+B[0]);
  coxa2.write(c2);     femur2.write(f2-Alf[1]);
tibia2.write(tib2+B[1]);
  coxa3.write(c3);     femur3.write(f3-Alf[2]);
tibia3.write(tib3+B[2]);
  coxa4.write(c4);     femur4.write(f4-Alf[3]);
tibia4.write(tib4+B[3]);
  delay(50);
  }

void IK_kaki(){
  float M[4]={M1, M2, M3, M4};
  float tinggi[4]={tinggi1, tinggi2, tinggi3, tinggi4};
  for(int a=0;a<4;a++){
    y[a]=tinggi[a]+t[a];
    L[a]=sqrt(pow(y[a],2) + pow((M[a]-coxa),2));
    Al[a]=acos(y[a]/L[a]);
    Al[a]=Al[a]*57296/1000;
    }
  double tib=pow(tibia,2);
  double fem=pow(femur,2);
  double Li1=pow(L[0],2);
  double Li2=pow(L[1],2);
  double Li3=pow(L[2],2);
  double Li4=pow(L[3],2);
```

```
  double LL[4]={Li1,Li2,Li3,Li4};
  for(int a=0;a<4;a++){
    Bl[a]=acos((tib-fem-LL[a])/(-2*femur*L[a]));
    Bl[a]=Bl[a]*57296/1000;
    Alf[a]=Al[a]+Bl[a];
    B[a]=acos((LL[a]-tib-fem)/(-2*tibia*femur));
    B[a]=B[a]*57296/1000;
    }
  Serial.print(Alf[0]);   Serial.print("\t");
  Serial.print(Alf[1]);   Serial.print("\t");
  Serial.print(Alf[2]);   Serial.print("\t");
  Serial.print(Alf[3]);   Serial.print("\t");

  Serial.print(B[0]);   Serial.print("\t");
  Serial.print(B[1]);   Serial.print("\t");
  Serial.print(B[2]);   Serial.print("\t");
  Serial.println(B[3]);
}

void proteksi(){
  if(Alf[0]>=130){
    Alf[0]=130;
    }
  if(Alf[0]<=40){
    Alf[0]=40;
    }
  if(Alf[1]>=130){
    Alf[1]=130;
    }
  if(Alf[1]<=40){
    Alf[1]=40;
    }
  if(Alf[2]>=130){
    Alf[2]=130;
    }
  if(Alf[2]<=40){
    Alf[2]=40;
    }
  if(Alf[3]>=130){
    Alf[3]=130;
    }
  if(Alf[3]<=40){
    Alf[3]=40;
    }
  if(B[0]>=100){
    B[0]=100;
    }
  if(B[0]<=30){
    B[0]=30;
    }
  if(B[1]>=100){
    B[1]=100;
    }
  if(B[1]<=30){
    B[1]=30;
    }
  if(B[2]>=100){
    B[2]=100;
    }
  if(B[2]<=30){
    B[2]=30;
    }
  if(B[3]>=100){
```

```
    B[3]=100;
    }
  if(B[3]<=30){
    B[3]=30;
    }
  }

void langkah(){
  langkah1();
  delay(300);
  langkah2();
  delay(300);
  langkah3();
  delay(300);
  langkah4();
  delay(300);
  langkah5();
  delay(300);
  langkah6();
  delay(300);
  langkah7();
  delay(300);
  langkah8();
  delay(300);
}

void langkah1(){
  x1=sin(45*0.0174532925199433)*5;
  x2=x1+1.0;
  x2a=x1-1.0;
  x3=x1+2.0;
  x3a=x1-2.0;
  z1=sqrt(pow(6,2)-pow(x1,2));
  z2=6.0;
  z3=sqrt(pow(z1,2)+pow(x2,2));
  z3a=sqrt(pow(z1,2)+pow(x2a,2));
  z4=sqrt(pow(z1,2)+pow(x3,2));
  z4a=sqrt(pow(z1,2)+pow(x3a,2));
  g1 = atan(x2/z1);
  g1=(g1*57296/1000)-45;
  g1a= atan(x2a/z1);
  g1a=45-(g1a*57296/1000);
  gg = atan(x3/z1);
  gg=(gg*57296/1000)-45;
  gga= atan(x3a/z1);
  gga=45-(gga*57296/1000);
  Serial.print(g1);
  Serial.print("  ");
  Serial.print(gg);
  Serial.print("  ");
  Serial.print(g1a);
  Serial.print("  ");
  Serial.print(gga);
  Serial.print("  ");
  M1=z3;          M2=5.0;          M3=z4;          M4=5.0;
  tinggi1=6.6;    tinggi2=7.3;    tinggi3=8.3;    tinggi4=6.6;
  IK_body();
  IK_kaki();
  proteksi();
  coxa1.write(c1-g1);  femur1.write(f1-Alf[0]);
tibia1.write(tib1+B[0]);
  coxa2.write(c2);     femur2.write(f2-Alf[1]);
tibia2.write(tib2+B[1]);
```

```
  coxa3.write(c3-gg);  femur3.write(f3-Alf[2]);
tibia3.write(tib3+B[2]);
  coxa4.write(c4);     femur4.write(f4-Alf[3]);
tibia4.write(tib4+B[3]);
  delay(200);
  }
void langkah2(){
  x1=sin(45*0.0174532925199433)*5;
  x2=x1+1.0;
  x2a=x1-1.0;
  x3=x1+2.0;
  x3a=x1-2.0;
  z1=sqrt(pow(6,2)-pow(x1,2));
  z2=6.0;
  z3=sqrt(pow(z1,2)+pow(x2,2));
  z3a=sqrt(pow(z1,2)+pow(x2a,2));
  z4=sqrt(pow(z1,2)+pow(x3,2));
  z4a=sqrt(pow(z1,2)+pow(x3a,2));
  g1 = atan(x2/z1);
  g1=(g1*57296/1000)-45;
  g1a= atan(x2a/z1);
  g1a=45-(g1a*57296/1000);
  gg = atan(x3/z1);
  gg=(gg*57296/1000)-45;
  gga= atan(x3a/z1);
  gga=45-(gga*57296/1000);
  Serial.print(g1);
  Serial.print("  ");
  Serial.print(gg);
  Serial.print("  ");
  Serial.print(g1a);
  Serial.print("  ");
  Serial.print(gga);
  Serial.print("  ");
  M1=z4;         M2=5.0;         M3=z4;         M4=5.0;
  tinggi1=7.6;    tinggi2=7.3;    tinggi3=8.1;    tinggi4=6.6;
  IK_body();
  IK_kaki();
  proteksi();
  coxa1.write(c1-gg);  femur1.write(f1-Alf[0]);
tibia1.write(tib1+B[0]);
  coxa2.write(c2);     femur2.write(f2-Alf[1]);
tibia2.write(tib2+B[1]);
  coxa3.write(c3-gg);  femur3.write(f3-Alf[2]);
tibia3.write(tib3+B[2]);
  coxa4.write(c4);     femur4.write(f4-Alf[3]);
tibia4.write(tib4+B[3]);
  delay(200);
  }
void langkah3(){
  x1=sin(45*0.0174532925199433)*5;
  x2=x1+1.0;
  x2a=x1-1.0;
  x3=x1+2.0;
  x3a=x1-2.0;
  z1=sqrt(pow(6,2)-pow(x1,2));
  z2=6.0;
  z3=sqrt(pow(z1,2)+pow(x2,2));
  z3a=sqrt(pow(z1,2)+pow(x2a,2));
  z4=sqrt(pow(z1,2)+pow(x3,2));
  z4a=sqrt(pow(z1,2)+pow(x3a,2));
  g1 = atan(x2/z1);
  g1=(g1*57296/1000)-45;
```

```
  g1a= atan(x2a/z1);
  g1a=45-(g1a*57296/1000);
  gg = atan(x3/z1);
  gg=(gg*57296/1000)-45;
  gga= atan(x3a/z1);
  gga=45-(gga*57296/1000);
  Serial.print(g1);
  Serial.print("  ");
  Serial.print(gg);
  Serial.print("  ");
  Serial.print(g1a);
  Serial.print("  ");
  Serial.print(gga);
  Serial.print("  ");
  M1=z4;          M2=5.0;          M3=z3a;          M4=5.0;
  tinggi1=7.4;    tinggi2=7.3;     tinggi3=7.1;     tinggi4=6.6;
  IK_body();
  IK_kaki();
  proteksi();
  coxa1.write(c1-gg);  femur1.write(f1-Alf[0]);
tibia1.write(tib1+B[0]);
  coxa2.write(c2);     femur2.write(f2-Alf[1]);
tibia2.write(tib2+B[1]);
  coxa3.write(c3+g1a); femur3.write(f3-Alf[2]);
tibia3.write(tib3+B[2]);
  coxa4.write(c4);     femur4.write(f4-Alf[3]);
tibia4.write(tib4+B[3]);
  delay(200);
  }
void langkah4(){
  x1=sin(45*0.0174532925199433)*5;
  x2=x1+1.0;
  x2a=x1-1.0;
  x3=x1+2.0;
  x3a=x1-2.0;
  z1=sqrt(pow(6,2)-pow(x1,2));
  z2=6.0;
  z3=sqrt(pow(z1,2)+pow(x2,2));
  z3a=sqrt(pow(z1,2)+pow(x2a,2));
  z4=sqrt(pow(z1,2)+pow(x3,2));
  z4a=sqrt(pow(z1,2)+pow(x3a,2));
  g1 = atan(x2/z1);
  g1=(g1*57296/1000)-45;
  g1a= atan(x2a/z1);
  g1a=45-(g1a*57296/1000);
  gg = atan(x3/z1);
  gg=(gg*57296/1000)-45;
  gga= atan(x3a/z1);
  gga=45-(gga*57296/1000);
  Serial.print(g1);
  Serial.print("  ");
  Serial.print(gg);
  Serial.print("  ");
  Serial.print(g1a);
  Serial.print("  ");
  Serial.print(gga);
  Serial.print("  ");
  M1=z4;          M2=5.0;          M3=z4a;          M4=5.0;
  tinggi1=7.4;    tinggi2=7.3;     tinggi3=7.3;     tinggi4=6.6;
  IK_body();
  IK_kaki();
  proteksi();
```

```
  coxa1.write(c1-gg);  femur1.write(f1-Alf[0]);
tibia1.write(tib1+B[0]);
  coxa2.write(c2);     femur2.write(f2-Alf[1]);
tibia2.write(tib2+B[1]);
  coxa3.write(c3+gga); femur3.write(f3-Alf[2]);
tibia3.write(tib3+B[2]);
  coxa4.write(c4);     femur4.write(f4-Alf[3]);
tibia4.write(tib4+B[3]);
  delay(200);
  }
void langkah5(){
  x1=sin(45*0.0174532925199433)*5;
  x2=x1+1.0;
  x2a=x1-1.0;
  x3=x1+2.0;
  x3a=x1-2.0;
  z1=sqrt(pow(6,2)-pow(x1,2));
  z2=6.0;
  z3=sqrt(pow(z1,2)+pow(x2,2));
  z3a=sqrt(pow(z1,2)+pow(x2a,2));
  z4=sqrt(pow(z1,2)+pow(x3,2));
  z4a=sqrt(pow(z1,2)+pow(x3a,2));
  g1 = atan(x2/z1);
  g1=(g1*57296/1000)-45;
  g1a= atan(x2a/z1);
  g1a=45-(g1a*57296/1000);
  gg = atan(x3/z1);
  gg=(gg*57296/1000)-45;
  gga= atan(x3a/z1);
  gga=45-(gga*57296/1000);
  Serial.print(g1);
  Serial.print("  ");
  Serial.print(gg);
  Serial.print("  ");
  Serial.print(g1a);
  Serial.print("  ");
  Serial.print(gga);
  Serial.print("  ");
  M1=5.0;         M2=z4;         M3=5.0;         M4=z3;
  tinggi1=6.6;    tinggi2=8.1;    tinggi3=7.3;    tinggi4=6.0;
  IK_body();
  IK_kaki();
  proteksi();
  coxa1.write(c1);      femur1.write(f1-Alf[0]);
tibia1.write(tib1+B[0]);
  coxa2.write(c2+gg);   femur2.write(f2-Alf[1]);
tibia2.write(tib2+B[1]);
  coxa3.write(c3);      femur3.write(f3-Alf[2]);
tibia3.write(tib3+B[2]);
  coxa4.write(c4+g1);   femur4.write(f4-Alf[3]);
tibia4.write(tib4+B[3]);
  delay(200);
  }
void langkah6(){
  x1=sin(45*0.0174532925199433)*5;
  x2=x1+1.0;
  x2a=x1-1.0;
  x3=x1+2.0;
  x3a=x1-2.0;
  z1=sqrt(pow(6,2)-pow(x1,2));
  z2=6.0;
  z3=sqrt(pow(z1,2)+pow(x2,2));
  z3a=sqrt(pow(z1,2)+pow(x2a,2));
```

```
  z4=sqrt(pow(z1,2)+pow(x3,2));
  z4a=sqrt(pow(z1,2)+pow(x3a,2));
  g1 = atan(x2/z1);
  g1=(g1*57296/1000)-45;
  g1a= atan(x2a/z1);
  g1a=45-(g1a*57296/1000);
  gg = atan(x3/z1);
  gg=(gg*57296/1000)-45;
  gga= atan(x3a/z1);
  gga=45-(gga*57296/1000);
  Serial.print(g1);
  Serial.print("  ");
  Serial.print(gg);
  Serial.print("  ");
  Serial.print(g1a);
  Serial.print("  ");
  Serial.print(gga);
  Serial.print("  ");
  M1=5.0;         M2=z4;         M3=5.0;         M4=z4;
  tinggi1=6.6;    tinggi2=8.1;    tinggi3=7.3;    tinggi4=7.4;
  IK_body();
  IK_kaki();
  proteksi();
  coxa1.write(c1);      femur1.write(f1-Alf[0]);
tibia1.write(tib1+B[0]);
  coxa2.write(c2+gg);   femur2.write(f2-Alf[1]);
tibia2.write(tib2+B[1]);
  coxa3.write(c3);      femur3.write(f3-Alf[2]);
tibia3.write(tib3+B[2]);
  coxa4.write(c4+gg);   femur4.write(f4-Alf[3]);
tibia4.write(tib4+B[3]);
  delay(200);
    }
void langkah7(){
  x1=sin(45*0.0174532925199433)*5;
  x2=x1+1.0;
  x2a=x1-1.0;
  x3=x1+2.0;
  x3a=x1-2.0;
  z1=sqrt(pow(6,2)-pow(x1,2));
  z2=6.0;
  z3=sqrt(pow(z1,2)+pow(x2,2));
  z3a=sqrt(pow(z1,2)+pow(x2a,2));
  z4=sqrt(pow(z1,2)+pow(x3,2));
  z4a=sqrt(pow(z1,2)+pow(x3a,2));
  g1 = atan(x2/z1);
  g1=(g1*57296/1000)-45;
  g1a= atan(x2a/z1);
  g1a=45-(g1a*57296/1000);
  gg = atan(x3/z1);
  gg=(gg*57296/1000)-45;
  gga= atan(x3a/z1);
  gga=45-(gga*57296/1000);
  Serial.print(g1);
  Serial.print("  ");
  Serial.print(gg);
  Serial.print("  ");
  Serial.print(g1a);
  Serial.print("  ");
  Serial.print(gga);
  Serial.print("  ");
  M1=5.0;         M2=z3a;         M3=5.0;         M4=z4;
  tinggi1=6.6;    tinggi2=6.3;    tinggi3=7.3;    tinggi4=8.0;
```

```
  IK_body();
  IK_kaki();
  proteksi();
  coxa1.write(c1);      femur1.write(f1-Alf[0]);
tibia1.write(tib1+B[0]);
  coxa2.write(c2-g1a);  femur2.write(f2-Alf[1]);
tibia2.write(tib2+B[1]);
  coxa3.write(c3);      femur3.write(f3-Alf[2]);
tibia3.write(tib3+B[2]);
  coxa4.write(c4+gg);   femur4.write(f4-Alf[3]);
tibia4.write(tib4+B[3]);
  delay(200);
  }
void langkah8(){
  x1=sin(45*0.0174532925199433)*5;
  x2=x1+1.0;
  x2a=x1-1.0;
  x3=x1+2.0;
  x3a=x1-2.0;
  z1=sqrt(pow(6,2)-pow(x1,2));
  z2=6.0;
  z3=sqrt(pow(z1,2)+pow(x2,2));
  z3a=sqrt(pow(z1,2)+pow(x2a,2));
  z4=sqrt(pow(z1,2)+pow(x3,2));
  z4a=sqrt(pow(z1,2)+pow(x3a,2));
  g1 = atan(x2/z1);
  g1=(g1*57296/1000)-45;
  g1a= atan(x2a/z1);
  g1a=45-(g1a*57296/1000);
  gg = atan(x3/z1);
  gg=(gg*57296/1000)-45;
  gga= atan(x3a/z1);
  gga=45-(gga*57296/1000);
  Serial.print(g1);
  Serial.print("  ");
  Serial.print(gg);
  Serial.print("  ");
  Serial.print(g1a);
  Serial.print("  ");
  Serial.print(gga);
  Serial.print("  ");
  M1=5.0;         M2=z4a;         M3=5.0;         M4=z4;
  tinggi1=6.6;    tinggi2=6.9;    tinggi3=7.3;    tinggi4=7.4;
  IK_body();
  IK_kaki();
  proteksi();
  coxa1.write(c1);      femur1.write(f1-Alf[0]);
tibia1.write(tib1+B[0]);
  coxa2.write(c2-gga);  femur2.write(f2-Alf[1]);
tibia2.write(tib2+B[1]);
  coxa3.write(c3);      femur3.write(f3-Alf[2]);
tibia3.write(tib3+B[2]);
  coxa4.write(c4+gg);   femur4.write(f4-Alf[3]);
tibia4.write(tib4+B[3]);
  delay(200);
  }
```