IMPLEMENTASI IMU (*INERTIAL MEASUREMENT UNIT*) DAN *FLEX SENSOR* PADA SISTEM PERGERAKAN *ARM MANIPULATOR*

SKRIPSI

TEKNIK ELEKTRO KONSENTRASI TEKNIK ELEKTRONIKA

Ditujukan untuk memenuhi persyaratan

memperoleh gelar Sarjana Teknik

FAUZI IZZUL HAQ

NIM. 145060307111029

UNIVERSITAS BRAWIJAYA

FAKULTAS TEKNIK

MALANG

2018

PENGANTAR

Alhamdulillahirobbil 'aalamin, Segala puji serta syukur penulis panjatkan kehadirat Allah *Subhanahu wa ta'ala*, atas semua rahmat dan karunia-Nya penulis bisa menyusun skripsi ini yang berjudul *"Implementasi IMU (Inertial Measurement Unit) dan Flex Sensor pada Sistem Pergerakan Arm Manipulator"*. Laporan skripsi ini dibuat sebagai syarat untuk memperoleh gelar Sarjana Teknik pada Jurusan Teknik Elektro Konsentrasi Teknik Elektronika Universitas Brawijaya. Penulis ingin menyampaikan rasa terima kasih yang sebesar-besarnya kepada pihak-pihak yang berkontribusi hingga skripsi dapat terselesaikan dengan baik, antara lain:

1. Ibu dan ayah yang selalu mendoakan yang terbaik untuk saya dan menjadi motivasi utama saya dalam pengerjaan skripsi ini.
2. Saudara-saudara saya di Balikpapan yang selalu men-*support* saya untuk bisa mempercepat pengerjaan skripsi ini.
3. Bapak Ir. Hadi Suyono, S.T.,Ph.D.,IPM. Selaku ketua Jurusan Teknik Elektro Fakultas Teknik Universitas Brawijaya.
4. Ibu Ir. Nurussa'adah, M.T. selaku Sekretaris Jurusan Teknik Elektro Fakultas Teknik Universitas Brawijaya.
5. Bapak Ali Mustofa, S.T., M.T. selaku Ketua Program Studi Sarjana Jurusan Teknik Elektro Fakultas Teknik Universitas Brawijaya.
6. Bapak Raden Arief Setiawan, S.T., M.T. selaku Ketua Kelompok Dosen Konsentrasi Teknik Elektronika Jurusan Teknik Elektro Fakultas Teknik Universitas Brawijaya.
7. Bapak Dr. Ir. Ponco Siwindarto, M.Eng.Sc selaku Dosen Pembimbing I atas segala bimbingan, pengarahan, kritik, dan saran yang telah diberikan selama proses pengerjaan skripsi.
8. Segenap dosen pengajar dan staff administrasi Jurusan Teknik Elektro Fakultas Teknik Universitas Brawijaya.
9. Keluarga besar Tim Robotika TEUB terkhusus sub divisi KRSTI atas segala pengalaman, kebersamaan, ilmu, dan bantuan selama ini.
10. Keluarga besar Al-Hadiid FTUB atas segala pengalaman, motivasi, dan kebersamaan selama perkuliahan.

11. Teman-teman Tim Robotika TEUB angkatan 2014 atas segala pengalaman, kebersamaan, ilmu dan bantuan selama 3 tahun menjadi anggota tim robotika dan aslab.
12. Teman-teman kontrakan "NATO" (Baim, Yudis, Eko, Bowo, Fatih, Aditya, Aji, Adit, Rahman, Restu) atas segala pengalaman dan kebersamaan selama ini.
13. Teman-teman "Rea-Reo" (Okto, Topan, Bimo, Panser, Ivan, Rakhmat, Daka, Jahdan, Helmi, Bobi, Isnan, Denis, Adi, Gammal, Satria, Indra, Cinta, Revo, Solo) atas segala pengalaman dan kebersamaan selama ini.
14. Teman-teman se-angkatan Dioda'14 atas segala pengalaman dan kebersamaan selama ini.
15. Seluruh teman-teman serta semua pihak yang tidak mungkin bagi penulis untuk mencamtumkan satu-persatu, terimakasih banyak atas bantuan dan dukungannya.

Pada akhirnya, penulis menyadari bahwa skripsi ini masih belum sempurna. Oleh karena itu penulis sangat mengharapkan kritik dan saran yang membangun. Penulis berharap semoga skripsi ini dapat bermanfaat bagi pengembangan ilmu pengetahuan dan teknologi serta bagi masyarakat.

Malang, September 2018

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR LAMPIRAN

LEMBAR PENGESAHAN

IMPLEMENTASI IMU (*INERTIAL MEASUREMENT UNIT*) DAN *FLEX SENSOR* PADA SISTEM PERGERAKAN *ARM MANIPULATOR*

SKRIPSI

TEKNIK ELEKTRO KONSENTRASI TEKNIK ELEKTRONIKA

Diajukan untuk memenuhi persyaratan
memperoleh gelar Sarjana Teknik

FAUZI IZZUL HAQ
NIM. 145060307111029

Skripsi ini telah direvisi dan disetujui oleh dosen pembimbing
Pada tanggal 6 November 2018

Mengetahui,

Ketua Jurusan Teknik Elektro	**Dosen Pembimbing**
Ir. Hadi Suyono, S.T., M.T., Ph.D. IPM.	**Dr. Ir. Ponco Siwindarto, M.Eng.Sc.**
NIP. 19730520 200801 1 013	**NIP. 19590304 198903 1 001**

JUDUL SKRIPSI :

IMPLEMENTASI IMU (*INERTIAL MEASUREMENT UNIT*) DAN *FLEX SENSOR* PADA SISTEM PERGERAKAN *ARM MANIPULATOR*

Nama Mahasiswa	: Fauzi Izzul Haq
NIM	: 145060307111029
Program Studi	: Teknik Elektro
Konsentrasi	: Teknik Elektronika

KOMISI PEMBIMBING :

Ketua : Dr. Ir. Ponco Siwindarto, M.Eng.Sc. .....

TIM DOSEN PENGUJI :

Dosen Penguji 1 : Eka Maulana, S.T., M.T., M.Eng. .....

Dosen Penguji 2 : Dr. Ir. M.Aswin, M.T. .....

Dosen Penguji 3 : Dr. Ing. Onny S., S.T., M.T., M.Sc. .....

Tanggal Ujian : 26 Oktober 2018

SK Penguji : No. 2418 Tahun 2018

PERNYATAAN ORISINALITAS SKRIPSI

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya dan berdasarkan hasil penelusuran berbagai karya ilmiah, gagasan dan masalah ilmiah yang diteliti dan diulas di dalam Naskah Skripsi ini adalah asli dari pemikiran saya. Tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu Perguruan Tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis dikutip dalam naskah ini dan disebutkan dalam sumber kutipan dan daftar pustaka.

Apabila ternyata di dalam naskah Skripsi ini dapat dibuktikan terdapat unsur-unsur jiplakan, saya bersedia Skripsi dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, pasal 25 ayat 2 dan pasal 70).

Malang, 26 Oktober 2018

Mahasiswa,

FAUZI IZZUL HAQ
NIM. 145060307111029

RINGKASAN

Fauzi Izzul Haq, Jurusan Teknik Elektro, Fakultas Teknik, Universitas Brawijaya, September 2018. *Implementasi IMU (Inertial Measurement Unit) dan Flex Sensor pada Sistem Pergerakan Arm Manipulator*, Dosen Pembimbing: Ponco Siwindarto.

Arm manipulator merupakan robot yang memiliki kemampuan bergerak seperti lengan manusia. Struktur dari *arm manipulator* terdiri atas lengan (*link*), sendi (*joint*), dan ujung (*end-effector*) yang saling terhubung. *End-effector* dapat berupa *gripper, wielding tourch, wielding gun, cutter*/pahat yang bisa dipakai untuk proses *milling* atau proses lainnya.

Arm manipulator umumnya dikendalikan dengan GUI (*Graphical User Interface*). *Arm manipulator* juga bisa dikendalikan menggunakan IMU (*Inertial Measurement Unit*) dan *flex sensor*. Pada prakteknya IMU sering terkendala dengan kalibrasi nilai sudut yang kurang akurat. Untuk itu filter kalman digunakan untuk mendapatkan nilai sudut atau data kemiringan yang lebih akurat dengan masukan dari akselerometer dan giroskop dari IMU. Filter kalman bekerja dengan memperkirakan hasil berikutnya berdasarkan data-data yang sudah ada sebelumnya.

Pada penelitian ini *arm manipulator* dapat dikendalikan menggunakan IMU dan *flex sensor* berdasarkan gestur pergelangan tangan dan jumlah DOF (*Degree of Freedom*) dari *arm manipulator* ini berjumlah 5 DOF. Pada pengujian sudut *pitch, roll,* dan *yaw* dari IMU yang sudah difilter kalman terhadap sudut servo yang dihasilkan, presentase *error* rata-rata masing-masing sebesar 0.73%, 0.67%, dan 2.96%. Pada pengujian resistansi *flex sensor* terhadap sudut servo yang dihasilkan, semakin melengkung *flex sensor* maka semakin besar nilai resistansi *flex sensor* dan semakin besar pula sudut servo yang dihasilkan. Pada pengujian keseluruhan sistem, *arm manipulator* berhasil memindahkan objek ke wadah yang dituju sebanyak 6 kali masing-masing berada pada koordinat (7.5,28.5), (15,25.5), (21,21), (26,15), (29,10), dan (30,0) dimana *arm manipulator* berada pada koordinat (0,0) dan objek berada pada koordinat (0,30).

Kata kunci – *Arm manipulator,* IMU, filter kalman, *flex sensor,* DOF, servo

SUMARRY

Fauzi Izzul Haq. Department of Electrical Engineering, Faculty of Engineering Brawijaya University, September 2018. *Implementation of IMU (Inertial Measurement Unit) and Flex Sensor on Arm Manipulator Movement System. Academic Supervisor*: Ponco Siwindarto.

Arm manipulator is a robot that has the ability to move like a human arm. The structure of the arm manipulator consists of the arms (links), joints (joints), and end (end-effector) that are interconnected. End-effector can be a gripper, wielding tourch, wielding gun, cutter / tool that can be used for milling or other processes.

Arm manipulators are generally controlled by GUI (Graphical User Interface). Arm manipulator can also be controlled using IMU (Inertial Measurement Unit) and flex sensors. In practice IMU is often constrained by calibration of less accurate angle values. Therefore kalman filter is used to get more accurate data from accelerometer and gyroscope on IMU. The kalman filter works by estimating the next results based on data that already exists before.

In this research, arm manipulator controlled by IMU and flex sensor through hand wrist movements and the number of DOF from arm manipulator is 5 DOF. In testing the pitch, roll, and yaw angle of the kalman filtered IMU to the resulting servo angle, the average percentage of errors was 0.80%, 0.73%, and 2.96%, respectively. In testng the flex sensor resistance to the resulting servo angle, the more flexed the flex sensor, the greater the flex sensor resistance value and the greater the servo angle produced. In testing the entire system, arm manipulator succesfully move an object to the intended container 6 times each at coordinates (7.5,28.5), (15,25.5), (21,21), (26,15), (29,10), and (30,0) where the arm manipulator is at the coordinate (0,0) and the object is at coordinate (0,30).

Keywords - Arm manipulator, IMU, kalman filter, flex sensor, DOF, servo

BAB I

PENDAHULUAN

1.1 Latar Belakang

Teknologi dibidang robotika telah berkembang pesat di zaman modern ini. Fungsi dari robot sendiri adalah untuk memudahkan dari tugas manusia. Robot dituntut untuk bergerak sesuai yang manusia inginkan. Kemajuan teknologi yang begitu pesat membuat proses industri maupun manufaktur menjadi lebih efektif dan efisien. Revolusi industri 4.0 berdampak pada teknologi yang semakin canggih, termasuk di bidang robotika itu sendiri.

Arm manipulator atau robot lengan memiliki kemampuan bergerak seperti lengan manusia dan banyak digunakan oleh masyarakat industri. Struktur dari *arm manipulator* terdiri atas lengan (*link*), sendi (*joint*), dan ujung (*end-effector*) yang saling terhubung. Servo digunakan sebagai aktuator untuk menggerakkan robot. Pada sistem pergerakan robot dikenal istilah DOF (*Degree of Freedom*) atau derajat kebebasan yaitu setiap titik sumbu gerakan mekanik pada robot. Semakin banyak DOF dari robot tersebut maka akan semakin kompleks pergerakan robot tersebut.

Arm manipulator umumnya dikendalikan dengan cara manual. Robot lengan juga bisa dikendalikan menggunakan IMU (*Inertial Measurement Unit*) dan *flex sensor*. IMU bekerja dengan mengukur posisi relatif, kecepatan, dan akselerasi gerakan objek menggunakan sistem pengukuran akselerometer dan giroskop sedangkan *flex sensor* berfungsi untuk mendeteksi kelengkungan. Pada sebuah penelitian sistem kendali *arm manipulator* menggunakan IMU dan *flex sensor* sebelumnya yang dilakukan oleh Arief Saifuddin pada tahun 2017, masih terdapat *error* sudut yang relatif besar antara sudut yang didapat dari IMU dengan sudut servo yang dihasilkan pada saat pergerakan servo.

Filter kalman digunakan untuk mendapatkan data kemiringan yang lebih akurat dengan masukan dari akselerometer dan giroskop yang diperoleh dari IMU. Filter kalman bekerja dengan memperkirakan hasil berikutnya berdasarkan data-data yang sudah ada sebelumnya.

Berdasarkan masalah tersebut, penulis mengajukan skripsi ini untuk merancang sistem kendali *arm manipulator* menggunakan *flex sensor* dan IMU yang difilter menggunakan algoritma filter kalman berdasarkan gestur pergelangan tangan manusia.

1.2 Rumusan Masalah

Berdasarkan masalah yang telah dijelaskan pada latar belakang, maka dapat disusun rumusan masalah sebagai berikut :

1. Bagaimana merancang sistem kendali pergerakan robot *arm manipulator* menggunakan IMU dan *flex sensor*?
2. Bagaimana menerapkan algoritma filter kalman untuk menghilangkan *noise* pada IMU?
3. Bagaimana mengendalikan robot *arm manipulator* untuk memindahkan suatu objek ke wadah yang dituju?

1.3 Batasan Masalah

Dalam perancangan skripsi ini permasalahan dibatasi oleh hal-hal berikut :

1. Pembahasan pada penelitian ini lebih kepada kinerja sensor yang dipakai yaitu IMU dan *flex sensor* terhadap sudut servo yang dihasilkan dan tidak membahas keseluruhan mekanika *arm manipulator* secara mendalam.
2. Pengujian keluaran filter kalman ketika pergerakan hanya dilakukan pada keluaran akselerometer.
3. Pengujian pemindahan barang hanya dilakukan pada bidang datar.

1.4 Tujuan

Tujuan dari penyusunan skripsi ini adalah untuk merancang robot *arm manipulator* yang dapat dikendalikan berdasarkan gestur pergelangan tangan manusia menggunakan IMU dan *flex sensor*.

1.5 Manfaat

Manfaat yang diperoleh dari penelitian ini adalah diharapkan dapat memudahkan dalam mengimplementasikan pengendalian robot menggunakan IMU dan metode kinematika. Tidak hanya pada robot lengan saja, namun untuk pergerakan robot-robot lainnya juga.

BAB II

TINJAUAN PUSTAKA

2.1 *Arm Manipulator*

Arm manipulator atau robot lengan merupakan robot yang memiliki kemampuan bergerak seperti lengan manusia dan banyak digunakan pada industri. Struktur dari *arm manipulator* terdiri atas lengan (*link*), sendi (*joint*), dan ujung (*end-effector*) yang saling terhubung. *End-effector* dapat berupa *gripper, wielding tourch, wielding gun, cutter*/pahat yang bisa dipakai untuk proses *milling* atau proses lainnya. Robot lengan memiliki *link* dan *joint* dengan lengan kaku yang terhubung secara seri serta memiliki pergerakan memutar (rotasi) dan memanjang/memendek (translasi/prismatik). Robot lengan ini banyak diimplementasikan di dalam industri, diantaranya digunakan dalam proses otomasi dalam produksi karena memiliki keakuratan yang tinggi dalam menjalankan tugasnya.

Gambar 2.1 *Arm Manipulator*
Sumber : mcwhizzone

Robot lengan ini memiliki sistem pergerakan manual menggunakan model matematis yang kompleks. Diantara model matematis yang digunakan adalah dengan memakai metode kinematika maju (*forward kinematic*) dan kinematika balik (*inverse kinematic*). Untuk mengetahui berapa sudut dari tiap sendi robot lengan maka kita harus men-*set* terlebih dahulu berapa koordinat (x,y,z) yang ingin dituju. Sedangkan untuk mengetahui berapa koordinat (x,y,z) yang ingin dicapai maka kita harus men-*set* berapa sudut dari masing-masing sendi terlebih dahulu. Secara geometris jenis robot lengan terdiri dari *Anthropomorphic*,

Cartesian, Silindris, Kutup, dan SCARA (*Selective Compliant Assembly Robot Arm)* dengan sumbu dan sistem pergerakan yang bervariasi.

Pada sistem pergerakan robot dikenal istilah DOF (*Degree of Freedom*) atau derajat kebebasan yaitu setiap titik sumbu gerakan mekanik pada robot. Semakin banyak DOF dari robot tersebut maka akan semakin kompleks pergerakan robot tersebut. Pada robot lengan jumlah DOF bisa bervariasi mulai dari tiga hingga enam tergantung dengan kebutuhan.

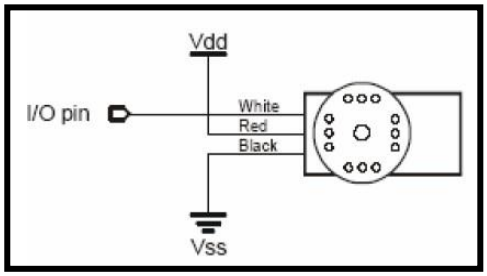
2.2 Motor Servo

Motor servo merupakan sebuah motor dengan sistem *closed feedback* di mana posisi dari motor akan diinformasikan kembali ke rangkaian kontrol yang ada di dalam motor servo. Motor servo ini terdiri dari sebuah motor DC, beberapa *gear*, sebuah potensiometer, sebuah *output shaft* dan sebuah rangkaian kontrol elektronik.

Gambar 2.2 Motor servo
Sumber : *Tetrixrobotics*

Motor servo dikemas dalam bentuk kotak segi empat dengan sebuah *output shaft* motor dan konektor dengan tiga kabel yaitu *power, control,* dan *ground*. *Gear* motor servo ada yang terbuat dari plastic, metal, atau titanium. Di dalam motor servo terdapat potensiometer yang digunakan untuk sebagai sensor posisi.

Potensiometer tersebut dihubungkan dengan *output shaft* untuk mengetahui posisi aktual *shaft*. Ketika motor DC berputar, maka *output shaft* juga berputar dan sekaligus memutar potensiomter. Rangkaian kontrol kemudian dapat membaca kondisi potensiometer tersebut untuk mengetahui posisi aktual *shaft.* Jika posisinya sesuai dengan yang diinginkan maka motor DC akan berhenti (Heri Andrianto: 2015).



Gambar 2.3 Konfigurasi pin motor servo
Sumber : Parallax, Inc.

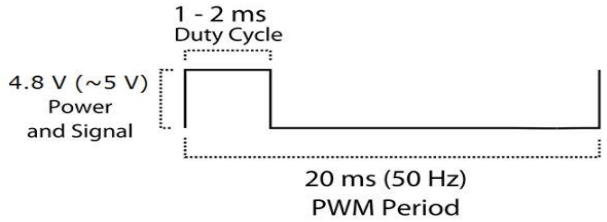
Sudut operasi motor servo (*operating angle*) bervariasi tergantung jenis motor servo. Ada dua jenis motor servo yaitu :

- Motor Servo *Standard*

  Yaitu motor servo yang mampu bergerak CW (*Clockwise*) dan CCW (*Counter Clockwise*) dengan sudut operasi tertentu, misalnya 60^o, 90^o, atau 180^o.
- Motor Servo *Continuous*

  Yaitu motor servo yang mampu bergerak CW dan CCW tanpa batasan sudut operasi (berputar secara kontinyu).

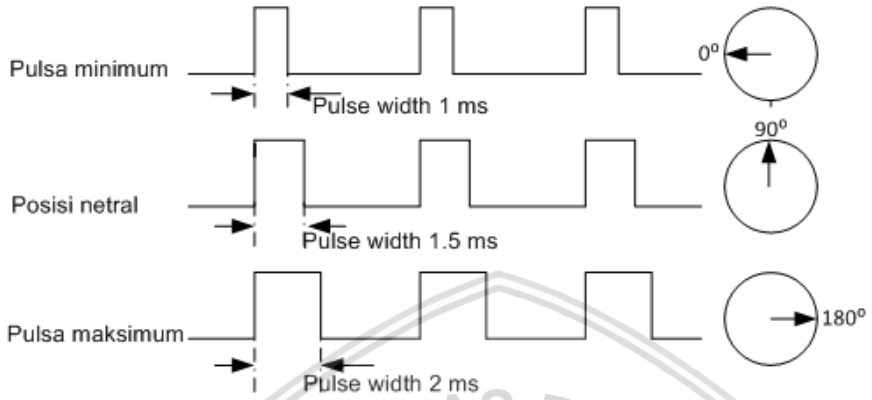
Motor servo menggunakan tegangan *supply* mulai dari 5 – 8 Volt. Motor servo dikendalikan dengan cara mengirimkan sebuah pulsa yang lebar pulsanya bervariasi. Pulsa tersebut dimasukkan melalui kabel kontrol motor servo. Sudut atau posisi *shaft motor servo* akan diturunkan dari lebar pulsa. Lebar pulsa berkisar antara 1 mS sampai 2 mS dengan priode pulsa sebesar 20 ms.



Gambar 2.4 Sinyal untuk mengendalikan motor servo
Sumber : Hendra Soewarno

Lebar pulsa akan mengakibatkan perubahan posisi pada servo. Misalnya sebuah pulsa 1.5 ms akan memutar motor pada posisi 90^o (posisi netral). Agar posisi

servo tetap pada posisi ini maka pulsa harus terus diberikan pada servo. Jadi meskipun ada gaya yang melawan, servo akan tetap bertahan pada posisinya. Gaya maksimum servo tergantung dari rentang torsi servo.



Gambar 2.5 Lebar pulsa dan posisi servo
Sumber : relifline

Ketika sebuah pulsa yang dikirim ke servo kurang dari 1.5 ms, servo akan berputar *counter clockwise* menuju ke posisi tertentu dari posisi netral. Jika pulsa yang dikirim lebih dari 1.5 ms, servo akan berputar *clockwise* menuju ke posisi tertentu dari posisi netral. Setiap servo memiliki spesifikasi lebar pulsa minimum dan maksimum sendiri-sendiri, tergantung jenis dan merek servo. Umumnya antara 1 ms sampai 2 ms. Parameter lain yang berbeda antara servo satu dengan servo lainnya adalah kecepatan servo untuk berubah dari posisi satu ke posisi lainnya (*operating speed*).

2.3 Arduino

Arduino merupakan *platform open source* baik secara *hardware* dan *software.* Arduino terdiri dari mikrokontroler ATMega328 dengan menggunakan Kristal osilator 16 MHz. Catu daya yang dibutuhkan untuk mencatu sistem minimum arduino cukup dengan tegangan 5 VDC. Port Arduino seri ATmega terdiri dari 20 pin yang meliputai 14 pin I/O digital dengan 6 pin dapat berfungsi sebagai output PWM (*Pulse Width Modulation*) dan 6 pin sebagai I/O analog. Kelebihan Arduino adalah tidak membutuhkan *flash programmer external* karena

di dalam chip mikrokontroler arduino telah diisi dengan *bootloader* yang membuat proses *upload* menjadi lebih sederhana.

Untuk koneksi terhadap komputer dapat menggunakan RS232 *to* TTL *converter* atau menggunakan *chip* USB ke *serial converter*. Arduino Uno dilengkapi dengan SRAM (*Static Random-Access Memory*) berukuran 2KB untuk menyimpan data, *flash memory* berukuran 32KB, dan *Erasable Programmable Read-Only Memory* (EEPROM) untuk menyimpan program.

Gambar 2.6 Arduino Uno
Sumber : *datasheet*

Tabel 2.1 Spesifikasi Arduino Uno

Mikrokontroler	ATmega328
Tegangan pengoperasian	5V
Tegangan input yang disarankan	7-12V
Batas tegangan input	6-20V
Jumlah pin I/O digital	14 (6 di antaranya menyediakan keluaran PWM)
Jumlah pin input analog	6
Arus DC tiap pin I/O	40 mA
Arus DC untuk pin 3.3V	50 mA
Memori Flash	32 KB (ATmega328), sekitar 0.5 KB digunakan oleh bootloader
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Clock Speed	16 MHz

Setiap empat belas pin digital pada Arduino Uno dapat digunakan sebagai *input* dan *output*, menggunakan fungsi *pinMode*(), *digitalWrite*(), dan

digitalRead(). Fungsi-fungsi tersebut beroperasi di tegangan 5 Volt. Setiap pin dapat memberikan atau menerima suatu arus maksimum 40 mA dan mempunyai sebuah resistor pull-up (terputus secara *default*) 20 – 50 kΩ. Selain itu beberapa pin mempunyai fungsi–fungsi spesial :

- Serial : 0 (RX) dan 1 (TX). Digunakan untuk menerima (RX) dan memancarkan (TX) serial data TTL (Transistor-Transistor Logic). Kedua pin ini dihubungkan ke pin–pin yang sesuai dari *chip* Serial Atmega8U2 USB – TTL.
- *External Interrupts* : 2 dan 3. Pin–pin ini dapat dikonfigurasikan untuk dipicu sebuah *interrupt* pada sebuah nilai rendah, suatu kenaikan atau penurunan yang besar atau suatu perubahan nilai. Lihat fungsi *attachInterrupt*() untuk lebih jelasnya.
- PWM : 3, 5, 6, 9, 10, dan 11. Memberikan 8-bit PWM output dengan fungsi analogWrite().
- SPI : 10(SS), 11(MOSI), 12(MISO), 13 (SCK). Pin-pin ini mensupport komunikasi SPI menggunakan SPI *library*.
- LED :13. Ada sebuah LED yang terpasang, terhubung ke pin digital 13. Ketika pin bernilai HIGH LED menyala ketika pin bernilai LOW LED mati,

Arduino UNO mempunyai 6 input analog, diberi label A0 sampai A5, setiapnya memberikan 10bit resolusi (contohnya 1024 nilai yang berbeda). Secara *default*, 6 input analog tersebut mengukur dari *ground* sampai tegangan 5 Volt, dengan itu mungk in untuk mengganti batas atas dari *range*-nya dengan menggunakan pin AREF dan fungsi analogReference(). Di sisi lain, beberapa pin mempunyai fungsi spesial:

- TWI: pin A4 atau SDA dan pin A5 atau SCL. Mensupport komunikasi TWI dengan menggunakan *Wire library*.

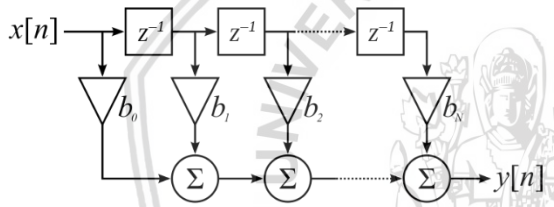
Ada sepasang pin lainnya pada *board* :

- AREF. Referensi tegangan untuk input analog. Digunakan dengan *analogReference*().
- Reset Membawa saluran ini *LOW* untuk me-*reset* mikrokontroler. Secara khusus, digunakan untuk menambahkan sebuah tombol *reset* untuk melindungi yang mem-*block* sesuatu pada *board*.

2.4 Filter Kalman

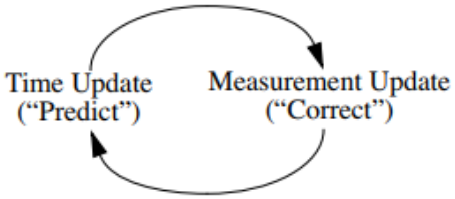
Filter kalman pada dasarnya adalah seperangkat persamaan matematis yang menerapkan tipe estimasi korektor-prediktor yang optimal dalam artian meminimalisir perkiraan kesalahan dari kovarian bila kondisi yang diduga terpenuhi. Teknik filter ini dinamakan berdasarkan nama penemunya, Rudolf E. Kalman yang sangat berguna terutama dalam menghilangkan *noise* dalam lingkungan dan navigasi.

Filter kalman adalah jenis dari filter digital. Pada prinsipnya filter digital sama dengan filter analog. Tetapi filter digital pada *input* dan *output* nya adalah sinyal digital dan bukan sinyal analog. Filter digital juga tidak memakai komponen R,L,C melainkan terdiri dari penjumlah (*adder*), pengali (*multiplier*), dan elemen tunda (*delay element*) atau gabungannya.



Gambar 2.7 Filter digital
Sumber : *wikibooks*

Filter kalman merupakan algoritma matematis untuk mengestimasi *state* dari proses dengan tujuan meminimalkan *noise*. Filter ini mendukung estimasi *state* sebelumnya, saat ini dan berikutnya. Filter kalman akan mengestimasi proses dengan menggunakan bentuk pengendali *feedback*. Filter mengestimasi *state* proses pada beberapa waktu dan kemudian mendapatkan umpan balik dalam bentuk pengukuran *noise*. Persamaan filter kalman dibagi menjadi dua kelompok, persamaan *time update* dan persamaan *measurement update*. *Time update* bisa juga disebut proses prediksi yaitu menggunakan estimasi *state* dari satu waktu sebelumnya untuk mendapatkan sebuah estimasi *state* untuk saat ini. Sedangkan *measurement update* disebut juga proses koreksi yaitu informasi pengukuran saat ini digunakan untuk memperbaiki prediksi dengan ekspektasi akan didapatkan *state* estimasi yang lebih akurat.



Gambar 2.8 Siklus algoritma filter kalman
Sumber : *An Introduction to Kalman Filter* hal.5

Berikut ini adalah persamaan prediksi-koreksi pada filter kalman:

***Time Update* (Prediksi)**

Predicted state $\hat{x}_k^- = A\hat{x}_{k-1} + Bu_{k-1}$ (2-1)

Predicted error covariance $P_k^- = P_{k-1}A^T + Q$ (2-2)

***Measurement Update* (Koreksi)**

Optimal Kalman Gain $K_k = P_{\bar{k}}H^T(HP_k^-H^T + R)^{-1}$ (2-3)

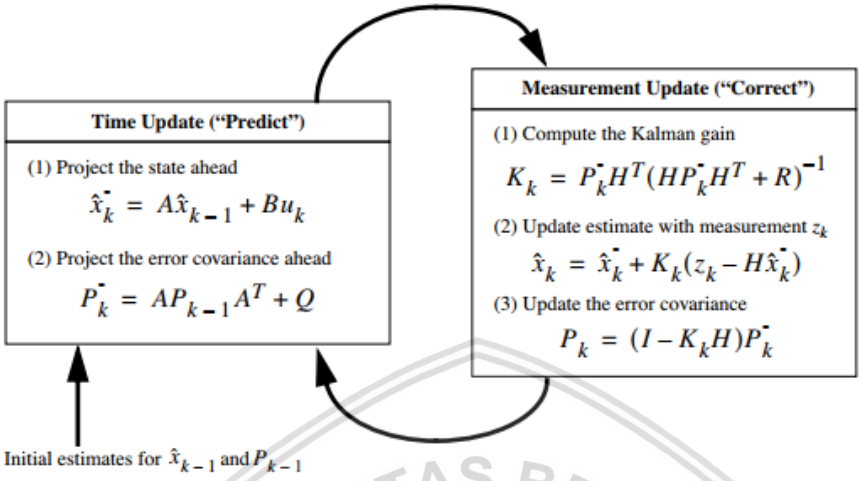
Update state estimate with z_k $\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k)$ (2-4)

Update error covariance $P_k = (I - K_kH)P_k^-$ (2-5)

Dalam pengukuran koreksi harus ditentukan kalman gain seperti pada Persamaan (2-3). Setelah itu tentukan berapa estimasi *state* yang baru dengan adanya selisih antara *observation* (z_k) yang diukur dengan sensor dengan nilai *predicted sensor* ($H\hat{x}_k$) seperti pada Persamaan (2-4). Kemudian tentukan berapa *error covariance* yang baru seperti pada Persamaan (2-5). Selisih antara z_k dan $H\hat{x}_k$ pada Persamaan (2-4) disebut dengan *innovation*. Jika nilai *innovation* sama dengan nol maka menunjukkan bahwa hasil estimasi sama dengan hasil pengukuran.

Matriks A $(n \times n)$ pada Persamaan (2-1) menghubungkan *state* pada waktu sebelumnya (k-1) ke *state* waktu sekarang (k), dengan tidak adanya *driving function* atau proses *noise*. Perlu diperhatikan bahwa dalam prakteknya matriks A bisa berubah seiring waktu, tetapi kita asumsikan bahwa matriks A adalah konstan. Matriks B $(n \times l)$ menghubungkan *input* kontrol opsional ke *state* x. matriks H $(m \times n)$ pada perhitungan koreksi menghubungkan *state* ke perhitungan z_k. pada prakteknya matriks H bisa berubah seiring waktu atau

perhitungan, tetapi kita asumsikan bahwa matriks H adalah konstan (Welch dan Bishop : 2006).



Gambar 2.9 Perhitungan filter kalman
Sumber : *An Introduction to Kalman Filter* hal.6

2.5 IMU (*Inertial Measurement Unit*)

IMU atau *Inertial Measurement Unit* merupakan sebuah sensor yang digunakan untu mengukur percepatan linear (x,y,z) dan kecepatan sudut (*yaw, pitch, roll*). Prinsip dasar dari sistem navigasi inersia adalah dengan mengintegralkan percepatan linear dan kecepatan putar yang didapatkan dari IMU, sehingga didapatkan data posisi dan data kemiringan. Posisi dan kemiringan yang didapatkan dari IMU ini relatif terhdadap kondisi awal (posisi dan kemiringan awal) (Nando Kusmanto: 2009).

IMU pada umumnya terdiri dari sensor akselerometer dan sensor giroskop. Untuk mendapatkan data posisi diperlukan data percepatan linear yang didapat dari akselerometer, yang didapat pada perhitungan berikut.

$$v = \int a \, dt \tag{2-6}$$

$$s = \int v \, dt \tag{2-7}$$

Dengan a adalah percepatan linear, v adalah kecepatan linear, dan s adalah jarak atau posisi. Sedangakan untuk mendapatkan sudut kemiringan maka diperlukan data kecepatan sudut yang diperoleh dari giroskop, berdasarkan perhitungan

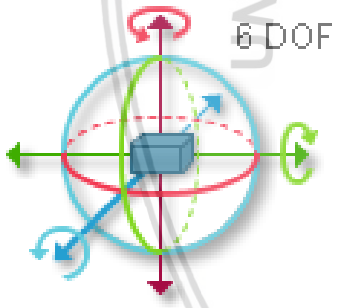
berikut.

$$\theta = \int \omega \, dt \tag{2-8}$$

Dengan ω adalah kecepatan sudut dan θ adalah sudut.

Umumnya IMU mencakup 6 aksis atau arah gerak yaitu 3 aksis dari akselerometer (x,y,z) dan 3 aksis dari giroskop (*yaw, pitch, roll*). Untuk sudut *yaw, pitch,* dan *roll* bisa didapat dari giroskop dan akselerometer. tetapi pada praktiknya giroskop memiliki kekurangan yaitu kurang akurat untuk mendapatkan sudut *yaw, pitch,* dan *roll.* Untuk itu akselerometer lebih bisa diandalkan untuk mendapatkan sudut *pitch* dan *roll.*

Namun akselerometer tidak bisa mendapatkan sudut *yaw* karena selama kalibrasi kebanyakan sumbu z sejajar dengan poros gravitasi bumi. Pada situasi ini setiap putaran disekitar sumbu z akan sedikit sehingga tidak berpengaruh pada keluran akselerometer dan *noise* pada keluaran akselerometer akan jauh lebih besar.

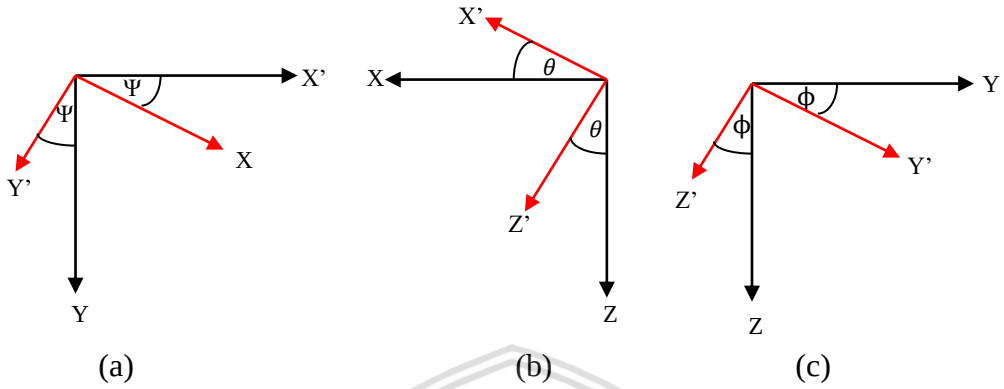


Gambar 2.10 Enam aksis IMU
Sumber : *signalquest*

2.5.1 Akselerometer

Akselerometer merupakan suatu sensor yang berfungsi untuk mengukut percepatan, mendeteksi dan mengukur getaran, mengukur percepatan gravitasi bumi, dan juga dapat digunakan untuk mendeteksi perubahan posisi pada suatu perangkat sekaligus menghitung nilai perubahannya. Percepatan sendiri dapat diukur dalam satuan SI, seperti meter per detik kuadrat (m/s^2) atau untuk percepatan gravitasi bumi diukur dalam satuan *g-force* (g) di mana 1g = $9.8 m/s^2$ (Rahmat Hidayat: 2014).

Sensor akselerometer digunakan untuk menghitung sudut *pitch* dan *roll* pada benda. Pembentukan sudut *yaw, pitch,* dan *roll* ditunjukkan sebagai berikut.



(a) (b) (c)

Gambar 2.11 (a) rotasi *yaw* (b) rotasi *pitch* (c) rotasi *roll*

Gambar 2.7 adalah sudut rotasi *yaw,pitch,roll* atau bisa disebut Sudut Euler yang diperlukan untuk berpindah dari satu sistem koordinat ke koordinat yang lain. Gambar 2.7(a) menunjukan gerakan rotasi dari *yaw* yaitu perputaran pada sumbu z atau sumbu vertikal. Gambar 2.7(b) menunjukan gerakan rotasi dari *pitch* yaitu perputaran pada sumbu y atau sumbu lateral. Gambar 2.7(c) menunjukan gerakan rotasi dari *roll* yaitu perputaran pada sumbu x atau sumbu longitudinal. Dari ketiga gerakan rotasi tersebut menghasilkan matriks masing-masing sebagai berikut.

$$R_1(\Psi) = \begin{bmatrix} \cos\Psi & \sin\Psi & 0 \\ -\sin\Psi & \cos\Psi & 0 \\ 0 & 0 & 1 \end{bmatrix} R_2(\theta) = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} R_3(\varphi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix} \quad (2\text{-}9)$$

Untuk mendapatkan koordinat benda ($\overrightarrow{V_B}$) maka ketiga matriks (2-8) dikalikan dengan koordinat bumi ($\overrightarrow{V_E}$) sebagai kerangka acuan (*inertial frame*). Koordinat benda mengalami perputaran dengan urutan rotasi *yaw-pitch-roll.* Dengan asumsi posisi awal dimana percepatan yang hanya dirasakan objek berasal dari tarikan gravitasi bumi, vektor $\overrightarrow{V_E}$ adalah (0,0,1). Maka perhitungan transformasi koordinatnya dapat dinyatakan sebagai berikut.

$$\overrightarrow{V_B} = R1(\Psi)\, R2(\theta)\, R_3(\phi)\, \overrightarrow{V_E}$$

$$= \begin{bmatrix} \cos\theta\cos\Psi & \cos\theta\sin\Psi & -\sin\theta \\ -\cos\phi\sin\Psi + \sin\phi\sin\theta\cos\Psi & \cos\phi\cos\Psi + \sin\phi\sin\theta\sin\Psi & \sin\phi\cos\theta \\ \sin\phi\sin\Psi + \cos\phi\sin\theta\cos\Psi & -\sin\phi\cos\Psi + \cos\phi\sin\theta\sin\Psi & \cos\phi\cos\theta \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} sin\,\theta \\ -cos\,\theta\,sin\,\phi \\ -cos\,\theta\,cos\,\phi \end{bmatrix} \quad (2\text{-}10)$$

Selanjutnya untuk mendapatkan percepatan pada sumbu x,y,z maka $\overrightarrow{V_B}$ harus dikalikan dengan g atau percepatan gravitasi.

$$\begin{bmatrix} f_x \\ f_y \\ f_z \end{bmatrix} = g \begin{bmatrix} \sin\theta \\ -\cos\theta\sin\phi \\ -\cos\theta\cos\phi \end{bmatrix} \quad (2\text{-}11)$$

Di mana f_x, f_y, f_z adalah percepatan pada sumbu x,y, dan z. θ adalah sudut *pitch*, ϕ adalah sumbu *roll*, dan g adalah besar nilai percepatan gravitasi (Arief Saifuddin: 2017). Sehingga didapatkan:

$$\frac{f_y}{f_z} = \frac{-g\,\cos\theta\sin\phi}{-g\,\cos\theta\cos\phi} \quad (2\text{-}12)$$

$$\frac{f_x}{\sqrt{{f_y}^2+{f_z}^2}} = \frac{-g\sin\theta}{\sqrt{g^2\cos^2\theta\,(\sin^2\phi+\cos^2\theta)}} \quad (2\text{-}13)$$

Dengan penyederhanaan persamaan di atas maka didapatkan:

$$\tan\phi = \frac{f_y}{f_z}, \tan\theta = \frac{-f_x}{\sqrt{{f_y}^2+{f_z}^2}} \quad (2\text{-}14)$$

Maka didapatkan nilai *pitch* (θ) dan *roll* (ϕ):

$$\phi = \tan^{-1}\left(\frac{f_y}{f_z}\right), \theta = \tan^{-1}\left(\frac{-f_x}{\sqrt{{f_y}^2+{f_z}^2}}\right) \quad (2\text{-}15)$$

2.5.2 Giroskop

Giroskop digunakan untuk mengukur orientasi berdasarkan prinsip momentum sudut. Sensor ini akan mengukur kecepatan sudut dari suatu rotasi yang satuannya adalah rad/s. Hasil pengukuran kecepatan sudut sebuah benda dengan menggunakan giroskop pada sumbu horizontal adalah sebagai berikut.

$$\theta(t) = \int \omega\, dt \quad (2\text{-}16)$$

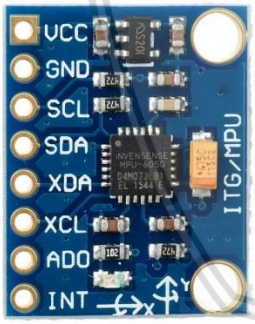
Dimana $\theta(t)$ adalah sudut atau kemiringan yang didapat dari pengintegralan ω atau kecepatan sudut yang didapat dari giroskop. *Yaw* (Ψ) tidak dapat

ditentukan oleh akselerometer karena tidak menggunakan titik referensi saat kalibrasi. Selama kalibrasi kebanyakan sumbu z sejajar dengan poros gravitasi bumi. Untuk mendapatkan nilai *yaw* maka harus menggunakan giroskop seperti pada Persamaan (2-16) dengan mengintegralkan kecepatan sudut pada sumbu z yang didapat dari giroskop atau dengan menggunakan kompas.

2.6 Sensor MPU6050

GY-521 MPU-6050 *Module* merupakan sebuah modul berinti MPU-6050 yang merupakan IMU 6 *axis Motion Processing Unit* dengan penambahan regulator tegangan dan beberapa komponen pelengkap lainnya yang membuat modul ini siap dipakai dengan tegangan *supply* sebesar 3-5 VDC. Modul ini memiliki *interface* I2C yang dapat disambungkan langsung ke MCU yang memiliki fasilitas I^2C.

Sensor MPU-6050 berisi sebuah MEMS akselerometer dan sebuah MEMS *Gyro* yang saling terintegrasi. Sensor ini sangat akurat dengan fasilitas *hardware* internal 16 bit ADC untuk setiap kanalnya. Sensor ini akan menangkap nilai kanal axis X, Y, dan Z bersamaan dalam satu waktu.



Gambar 2.12 MPU6050 *Module*
Sumber : htfelectronics

Berikut adalah spesifikasi dari Modul ini :

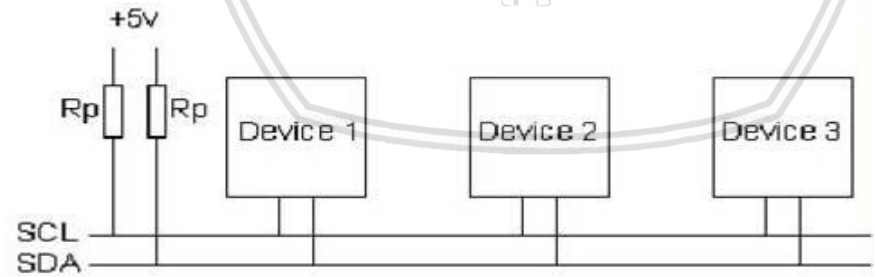
- Berbasis *Chip* MPU-6050
- *Supply* tegangan berkisar 3-5V
- *Gyroscope range* : + 250, 500, 1000, 2000 ° / s
- *Acceleration range* : ± 2 ± 4 ± 8 ± 16 g
- *Communication standard* : I^2C
- *Chip built-in* 16 bit AD *converter*, 16 bit data output

- Jarak antar *pin header* 2.54 mm
- Dimensi modul 20.3mm x 15.6mm

2.7 Komunikasi I²C

I²C atau *Inter-Integrated Circuit* merupakan protokol yang digunakan yang digunakan pada *multi-master serial computer bus* untuk saling berkomunikasi dengan perangkat *low-speed* lainnya. Jalur I²C hanya berupa 2 jalur yang disebut SDA (*Serial Data*) yang merupakan jalur untuk data dan SCL (*Serial Clock*)(Frans Surya: 2007).

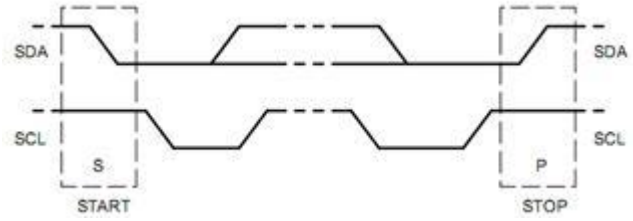
Jenis komunikasi yang dilakukan antar perangkat dengan menggunakan protokol I²C mempunyai sifat *serial synchronous half duplex bidirectional*, dimana data yang ditransmisikan dan diterima hanya melalui satu jalur SDA (bersifat *serial*), setiap penggunaan jalur data bergantian antar perangkat (bersifat *half duplex*) dan data dapat ditransmisikan dari dan ke sebuah perangkat (bersifat *bidirectional*). Sumber *clock* yang digunakan I²C hanya berasal dari satu perangkat master melalui jalur SCL (bersifat *synchronous*). Jalur dari SDA dan SCL ini terhubung dengan *pull-up resistor* yang besar resistansinya bisa berapa saja (1K, 1.8K, 4.7K, 10K, 47K atau nilai diantara range tersebut) (Frans Surya: 2007).



Gambar 2.13 Rangkaian I²C
Sumber : Wordpress

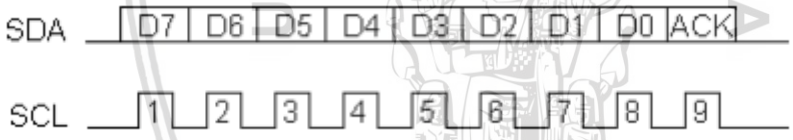
Jalur SDA dan SCL menjadi *open drain* dengan adanya *pull-up resistor* tersebut, yang maksudnya adalah perangkat hanya memberikan *output* 0 (*LOW*) untuk membuat jalur menjadi *LOW*, dan dengan memberikan *pull-up resistor* sudah membuatnya *HIGH*. Perangkat yang dihubungkan dengan sistem I²C dapat dioperasikan sebagai *master* atau *slave*. *Master* adalah perangkat yang memulai

transfer data pada I^{2}C *Bus* dengan membentuk sinyal *start*, mengakhiri transfer data dengan membentuk sinyal *stop*, dan membangkitkan sinyal *clock*. *Slave* adalah perangkat yang dialamati *master*.



Gambar 2.14 Kondisi sinyal *start* dan *stop* pada I^{2}C
Sumber : Wordpress

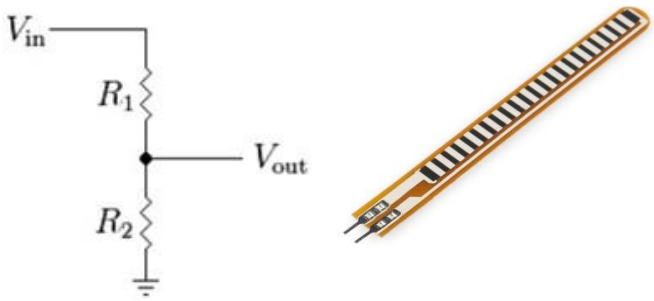
Sinyal dasar yang lain dalam I^{2}C adalah sinyal ACK atau *acknowledge*. Setelah transfer data oleh *master* berhasil diterima *slave*, *slave* akan menjawabnya dengan mengirim sinyal ACK, yaitu dengan membuat SDA menjadi "0" pada siklus *clock* ke-9. Ini menunjukkan bahwa *slave* telah menerima 8 bit data dari *master*.



Gambar 2.15 SDA dan SCL pada I^{2}C
Sumber : Comp-eng binus

2.8 *Flex Sensor*

Flex sensor merupakan suatu sensor yang berfungsi untuk mendeteksi kelengkungan dari suatu objek. Prinsip kerjanya sama dengan potensio. Untuk mengunakan *flex sensor* kita membutuhkan rangkaian pembagi tegangan yang digunakan untuk membaca nilai tegangan dari *flex sensor*. *Flex sensor* apabila semakin melegkung maka nilai resistansinya akan semakin bertambah.



(a) (b)

Gambar 2.16 (a) Rangkaian pembagi tegangan (b) *flex sensor*
Sumber : elektrokita dan *sparkfun*

$$V_{out} = \frac{R_2}{R_1+R_2} \times V_{in} \quad (2\text{-}17)$$

Flex sensor berfungsi sebagai R1 yang cara kerjanya sama dengan potensio, semakin melengkung *flex sensor* maka resistansinya akan semakin menambah. Nilai resistansi R2 bisa ditetapkan dengan nilai berapapun. Pada umumnya Vin bernilai 5V, setelah itu Vout bisa dihasilkan sesuai Persamaan (2-17).

2.9 ADC (*Analog to Digital Converter*)

ADC (*Analog to Digital Converter*) merupakan suatu perhitungan konversi yang mengubah sinyal atau tegangan analog menjadi sinyal digital. Resolusi ADC selalu dinyatakan sebagai jumlah dari bit-bit dalam kode keluaran digitalnya. ADC dengan resolusi n-bit memiliki 2^n kode digital yang mungkin dan berarti juga memiliki 2^n *step level*. Jika resolusi ADC semakin tinggi, maka semakin banyak kemungkinan nilai-nilai analog yang bisa disajikan. Misal ADC dengan resolusi 8 bit menghasilkan bilangan 0 sampai dengan 255 (256 bilangan dan 255 *step*)(Andriatno: 2015).

Arduino Uno memiliki resolusi ADC 10 bit maka menghasilkan bilangan 0 sampai 1023 (1024 bilangan dan 1023 *step*). Rumus perhitungan ADC ditunjukkan pada persamaan di bawah ini.

$$V_{ADC} = \frac{Nilai\ Digital}{1023} \times V_{in} \quad (2\text{-}18)$$

$$Nilai\ Digital = \frac{V_{ADC}}{V_{Ref}} \times 1023 \quad (2\text{-}19)$$

Dengan:

V_{ADC} : Tegangan konversi ADC (V)

V_{ref} : Tegangan yang digunakan untuk referensi ADC (V)

Nilai Digital : Nilai antara 0 sampai 1023 atau nilai yang dihasilkan sesuai resolusi ADC.

BAB III

METODOLOGI PENELITIAN

Untuk menyelesaikan rumusan masalah dan merealisasikan tujuan penelitian maka diperlukan langkah-langkah metode untuk menyelesaikan masalah tersebut. Adapun langkah-langkah yang perlu dilakukan untuk merealisasikan sistem yang dirancang adalah penentuan spesifikasi alat, perancangan dan pembuatan alat, pengujian alat, dan pengambilan kesimpulan.

3.1 Penentuan Spesifikasi Alat

Spesifikasi alat secara umum ditentukan terlebih dahulu sebagai acuan dalam perancangan alat. Spesifikasi alat yang digunakan adalah sebagai berikut:

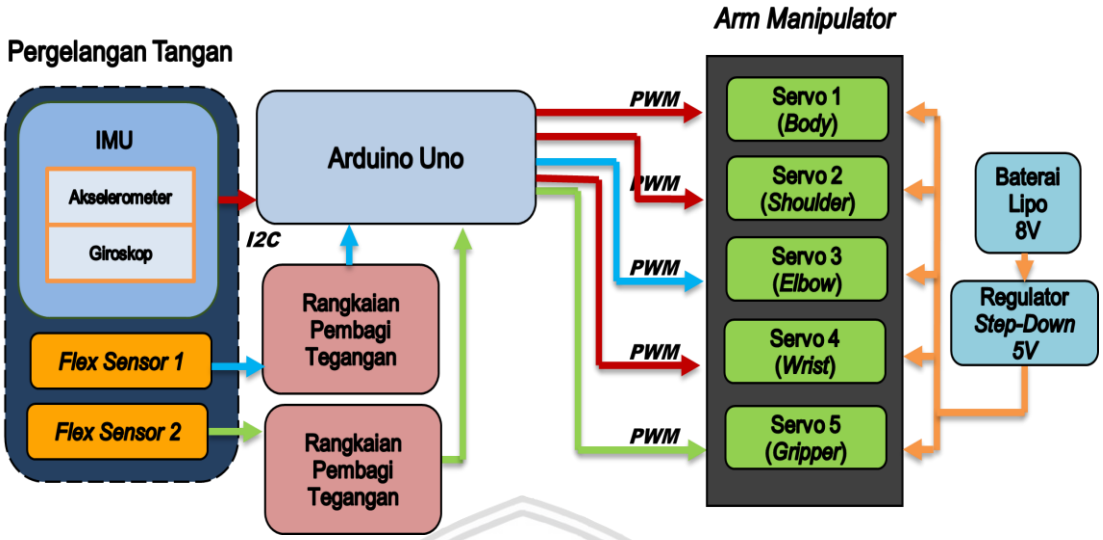
1. Mikrokontroler yang digunakan adalah Arduino Uno.
2. Jumlah DOF (*Degree of Freedom*) pada robot lengan berjumlah 5 DOF.
3. Servo yang digunakan pada lengan robot adalah Servo HS-422 dengan *range* torsi antara 3,3 – 4,1 kg/cm dan berat 45,5gr.
4. Catu daya yang digunakan adalah baterai lipo 8V yang dilewatkan ke regulator *switching* 5V.
5. IMU yang digunakan untuk menggerakkan *arm manipulator* adalah sensor MPU6050.
6. *Flex sensor* yang digunakan untuk menggerakkan *arm manipulator* adalah berjumlah 2.

3.2 Perancangan dan Pembuatan Alat

Perancangan dan pembuatan alat dalam penelitian ini terdiri dari perncangan blok diagram keseluruhan, perancangan perangkat keras (*hardware*) dan perancangan perangkat lunak (*software*).

3.2.1 Perancangan Diagram Blok Secara Keseluruhan

Diagram blok keseluruhan sistem dirancang untuk menjelaskan sistem secara garis besar dan diharapkan alat dapat bekerja sesuai dengan yang diinginkan.



Gambar 3.1 Diagram blok keseluruhan

Fungsi dari setiap blok di atas adalah sebagai berikut:

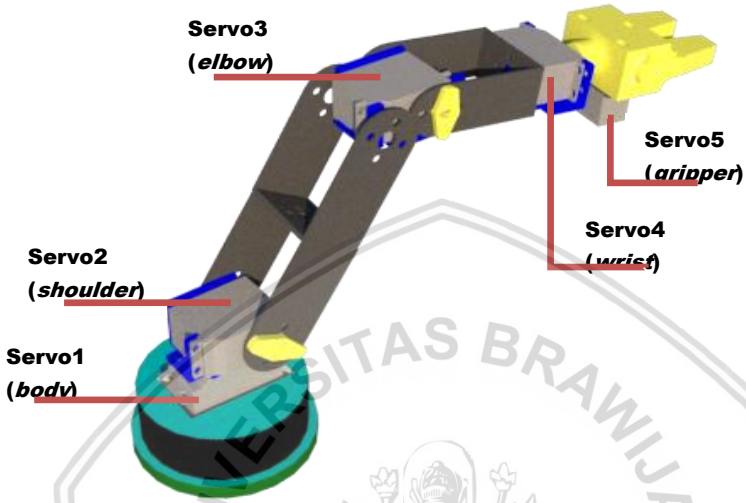
1. Sensor MPU6050 berfungsi sebagai masukan ke Arduino berupa nilai sudut yang didapat dari nilai *yaw*, *roll*, dan *pitch* yang akan menggerakkan servo *body*, *shoulder*, dan *wrist* *arm manipulator* sesuai gestur pergelangan tangan.
2. Sensor *flex* 1 berfungsi untuk menggerakkan servo *elbow* dan sensor *flex* 2 berfungsi untuk menggerakkan servo *gripper* (*end-effector*) sesuai gestur pergelangan tangan setelah diolah pada Arduino.
3. Filter kalman berfungsi sebagai filter digital untuk menghilangkan *noise* pada sensor MPU6050 dalam pengkalibrasian nilai sudut.
4. Mikrokontroler Arduino berfungsi sebagai pengolah data masukan yang akan dikontrol untuk menghasilkan keluaran yang diinginkan.
5. Servo 1-5: sebagai aktuator untuk menggerakkan robot lengan.
6. Baterai lipo 8V berfungsi sebagai catu daya kelima servo setelah dilewatkan pada regulator *step-down* 5V.

3.2.2 Perancangan Perangkat Keras (*Hardware*)

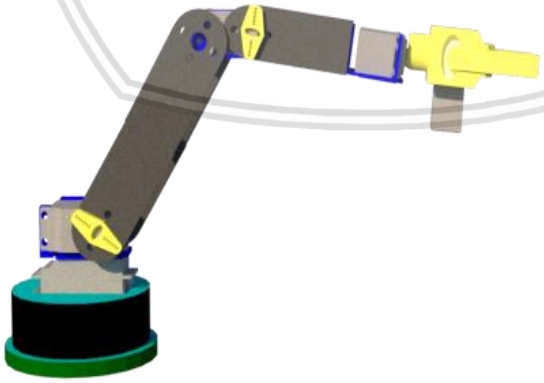
3.2.2.1 Perancangan Mekanik

Perancangan mekanik merupakan desain bentuk fisik dari *arm manipulator* dengan semua komponen yang sudah dijelaskan agar dalam

penelitian dapat lebih mudah digunakan. Pada perancangan mekanik *arm manipulator* memiliki DOF sebanyak 5 buah.



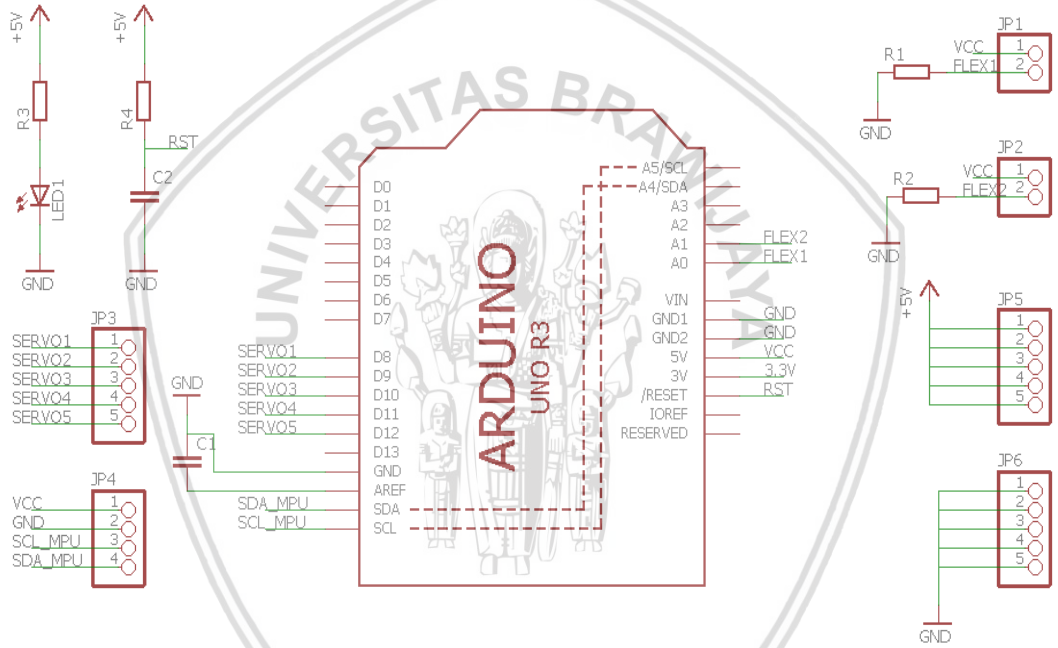
Gambar 3.2 Desain mekanik *arm manipulator* tampak keseluruhan



Gambar 3.3 Desain mekanik *arm manipulator* tampak samping

3.2.2.2 Perancangan Elektrik

Perancangan elektrik yaitu pembuatan rangkaian *shield* Arduino yang bertujuan untuk mempermudah koneksi kabel dari Arduino ke sensor MPU6050, *flex sensor*, dan servo. Awal mula harus menghubungkan pin I^2C dari sensor MPU6050 ke Arduino karena jenis dari komunikasi sensor MPU6050 adalah komunikasi I^2C. Kemudian menghubungkan pin VCC dan *ground* masing-masing komponen ke pin VCC dan *ground* catu daya. Catu daya yang dipakai adalah baterai lipo 8V yang dilewatkan ke regulator 5V. Selanjutnya menghubungkan masing-masing pin data dari setiap komponen ke pin-pin data dari Arduino.



Gambar 3.4 Konfigurasi pin Arduino ke masing-masing komponen

Konfigurasi pin Arduino dapat dilihat berikut ini:

Tabel 3.1 Konfigurasi Pin Arduino ke masing-masing komponen

PIN	Keterangan
A0	Dihubungkan ke V_{out} Rangk. Pembagi Tegangan 1
A1	Dihubungkan ke V_{out} Rangk. Pembagi Tegangan 2
D8	Dihubungkan ke pin data servo 1

D9	Dihubungkan ke pin data servo 2
D10	Dihubungkan ke pin data servo 3
D11	Dihubungkan ke pin data servo 4
D12	Dihubungkan ke pin data servo 5
SDA	Dihubungkan ke pin SDA MPU6050
SCL	Dihubungkan ke pin SCL MPU6050
+5V	Dihubungkan ke VCC
GND	Dihubungkan ke *ground*
RESET	Dihubungkan ke RST

3.2.3 Perancangan Perangkat Lunak (Software)

Perancangan perangkat lunak dilakukan dengan perancangan diagram alir dari algoritma filter kalman kemudian dilakukan perancangan diagram alir pergerakan *arm manipulator* sesuai gestur pergelangan tangan menggunakan sensor MPU6050.

3.2.3.1 Perancangan Algoritma Filter Kalman

Pada perancangan algoritma filter kalman perlu memperhatikan persamaan matematis dari filter kalman. Persamaan yang digunakan pada algoritma ini sudah disederhanakan terlebih dahulu untuk memudahkan dalam proses perhitungan. Dalam proses ini terbagi menjadi proses perhitungan *state* (prediksi) dan perhitungan kovarian (koreksi). Perhitungan prediksi meliputi perhitungan estimasi lama dan *error* kovarian lama. Sedangkan perhitungan koreksi meliputi perhitungan *kalman gain,* estimasi, dan *error* kovarian.

Perhitungan prediksi meliputi:

- Estimasi lama:

$$\hat{x}_{t-1} = \hat{x}_t \tag{3-1}$$

- Error kovarian lama:

$$P_{t-1} = P_t \tag{3-2}$$

Keterangan:

x_t = nilai estimasi keluaran filter saat waktu t

x_{t-1} = nilai estimasi keluaran filter saat waktu t-1

P_t = error kovarian saat waktu t

P_{t-1} = error kovarian saat waktu t-1

Perhitungan koreksi meliputi:

- Perhitungan *Kalman Gain*:

$$K_t = \frac{P_{t-1}}{P_{t-1}+R} \quad (3\text{-}3)$$

- Perhitungan estimasi:

$$\hat{x}_t = \hat{x}_{t-1} + K_t(z_t - \hat{x}_{t-1}) \quad (3\text{-}4)$$

- Perhitungan error kovarian:

$$P_t = (1 - K_t)P_{t-1} + Q \quad (3\text{-}5)$$

Keterangan:

K_t = nilai penguatan kalman saat waktu t

R = kovarian *noise* pengukuran ($R \geq 0$)

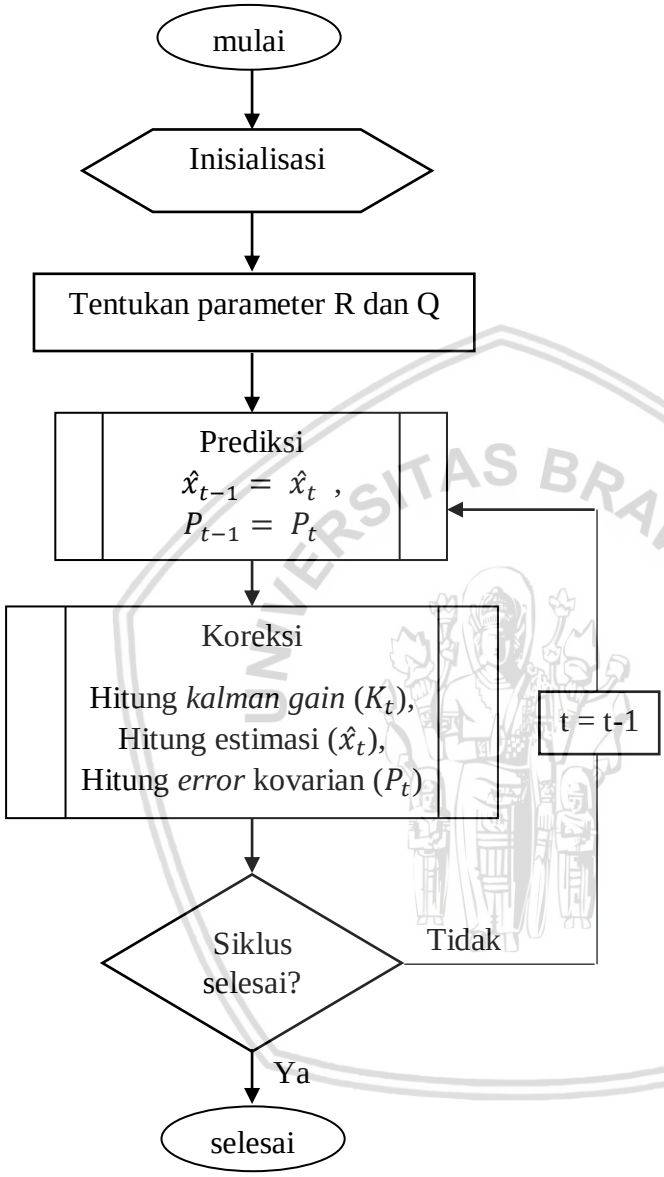
Z_t = nilai masukan filter kalman saat waktu t (didapat dari sensor)

Q = kovarian *noise* proses

Perhitungan prediksi menggunakan estimasi state satu waktu sebelumnya untuk mendapatkan estimasi state saat ini. Perhitungan koreksi menggunakan informasi pengukuran saat ini dan digunakan untuk memperbaiki prediksi, dengan harapan akan didapatkan estimasi yang lebih akurat. Perhitungan dari filter kalman pada perhitungan *state* dan perhitungan kovarian akan terus berulang. Nilai Q (kovarian *noise* proses) dan R (kovarian *nosie* pengukuran) bisa berubah seiring waktu atau perhitungan, namun kita asumsikan bahwa nilai Q dan R adalah konstan.

Ketika memulai filter kalman diperlukan *initial condition* dari P_{t-1} dan $\hat{x}_{t-1}$ atau bisa disebut P_0 dan $\hat{x}_0$. Penentuan berdasarkan sistem yang digunakan. $\hat{x}_0$ biasanya didapatkan dengan memperkirakan *state* pada keadaan awal, sedangkan P_0 sebaiknya tidak sama dengan nol, sebab jika $P_0 = 0$ akan

menyebabkan filter menginisialisasi dan selalu percaya bahwa $\hat{x}_k = \hat{x}_0$ (Welch dan Bishop: 2006). Berikut adalah diagram alir perhitungan filter kalman.



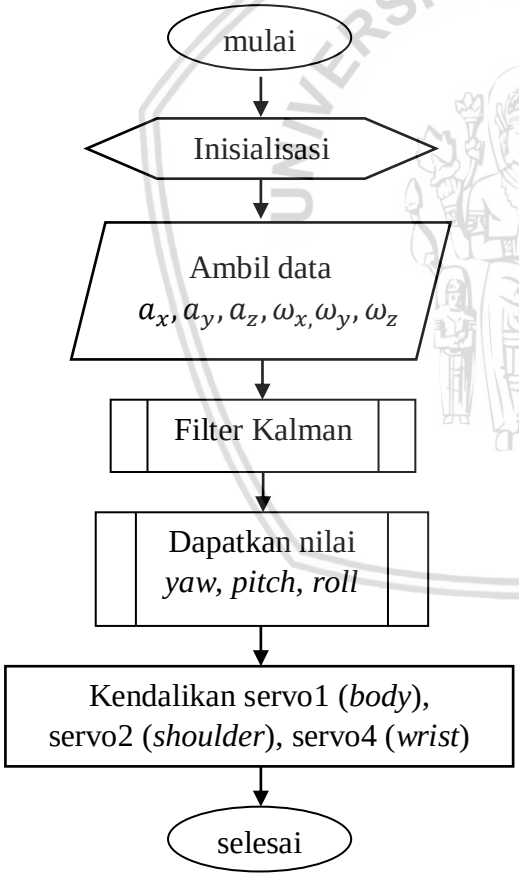
Gambar 3.5 Diagram alir proses perhitungan filter kalman

Pada gambar diagam alir di atas dapat dilihat bahwa perhitungan dari filter kalman pada perhitungan *state* dan perhitungan kovarian akan terus berulang. Proses dimulai dari inisialisasi perhitungan, kemudian menentukan prediksi bahwa estimasi dan *error* yang akan datang sama dengan estimasi dan *error* sekarang. Kemudian hitung *kalman gain* sesuai rumus dengan memasukan nilai R yang sudah di *set* di awal program. Lalu hitung estimasi sesuai rumus dengan

memasukan nilai z_k yang didapat dari IMU. Setelah itu hitung *error kovarian* sesuai rumus dengan memasukan nilai Q yang sudah di *set* di awal program. Dan perhitungan ini akan terus berulang selama program berjalan.

3.2.3.2 Perancangan Algoritma Pergerakan *Arm Manipulator*

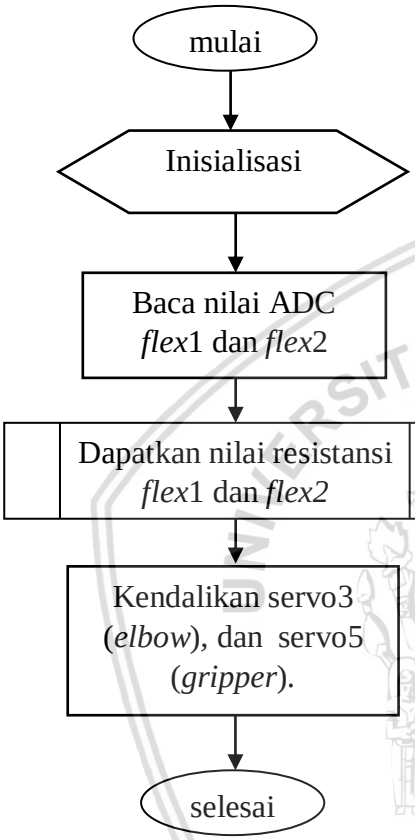
Pada perancangan algoritma untuk pergerakan *arm manipulator* yaitu dengan menggerakkan setiap servo pada *arm manipulator* sesuai sudut yang telah diatur oleh mikrokontroler dari sensor MPU6050 dan *flex sensor*. Setiap servo akan bergerak sesuai dengan sudut dari sensor. Sensor MPU6050 akan menggerakkan servo *body, shoulder,* dan *wrist. Flex sensor* 1 akan menggerakkan servo *elbow.* Dan *flex sensor* 2 akan menggerakkan servo *gripper.* Berikut adalah diagram alir dari sistem pergerakan *arm manipulator* yang dikendalikan dengan IMU.



Gambar 3.6 Diagram alir kendali *arm manipulator* dengan IMU

Pada diagram alir di atas setelah proses inisialisasi, maka mikrokontroler akan mengambil data percepatan linear (a_x, a_y, a_z) dari akselerometer dan kecepatan sudut ($\omega_{x,}\omega_y, \omega_z$) dari giroskop pada IMU dan akan difilter dengan filter kalman.

Setelah itu maka servo *body, shoulder,* dan *wrist* akan dikendalikan berdasarkan sudut *yaw, roll,* dan *pitch.* Untuk sistem pergerakan servo *elbow* dan *gripper* akan dikendalikan oleh *flex sensor*. Berikut adalah diagram alir dari sistem pergerakan *arm manipulator* menggunakan *flex sensor*:



Gambar 3.7 Diagram alir kendali *arm manipulator* dengan *flex sensor*

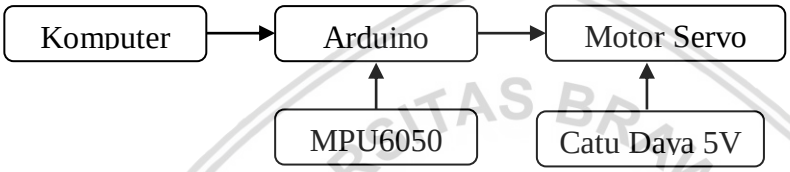
Pada diagram alir di atas setelah diinisialisasi dan pembacaan nilai ADC dan didapatkan nilai resistansi dari *flex sensor,* maka servo elbow dan *gripper* akan dikendalikan. *Flex sensor*1 akan mengendalikan servo3 (*elbow*), sedangkan *flex sensor*2 akan mengendalikan servo5 (*gripper*).

3.3 Pengujian Alat

Untuk mengetahui kinerja alat apakah sesuai dengan yang direncanakan maka dilakukan pengujian alat. Pengujian dilakukan pada masing-masing bagian dan kemudian secara keseluruhan sistem. Secara garis besar pengujian yang dilakukan adalah sebagai berikut:

3.3.1 Pengujian Filter Kalman

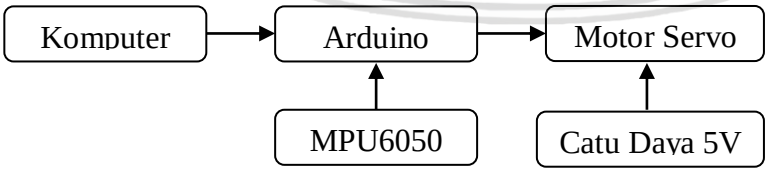
Pengujian ini bertujuan untuk mengetahui seberapa akurat filter kalman untuk menghilangkan *noise* pada IMU. Pengujian dilakukan dengan membandingkan keluaran *pitch, roll, yaw* tanpa filter kalman dan dengan filter kalman ketika diberi goncangan dan ketika dibandingkan dengan sudut servo yang dihasilkan oleh IMU melalui perhitungan *pitch, roll, yaw* yang dilakukan oleh mikrokontroler Arduino. Alat yang digunakan pada pengujian ini yaitu motor servo yang ditempelkan busur, sensor MPU6050, catu daya 5V, mikrokontroler Arduino, dan komputer.



Gambar 3.8 Diagram blok pengujian filter kalman

3.3.2 Pengujian IMU

Pengujian ini bertujuan untuk mengetahui apakah perhitungan *pitch, roll, yaw* melalui mikrokontroler Arduino sesuai dengan sudut yang dihasilkan motor servo. Pengujian ini dilakukan dengan menguji keluaran *pitch, roll, yaw* selama 5 kali pada setiap sudut kelipatan 15 dengan melihat sudut yang dihasilkan oleh motor servo. Alat yang digunakan pada pengujian ini yaitu motor servo yang ditempelkan busur, sensor MPU6050, catu daya 5V, mikrokontroler Arduino, dan komputer.

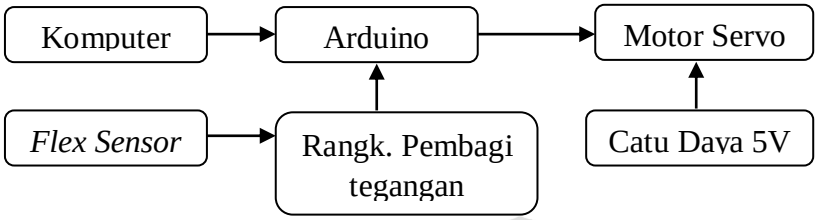


Gambar 3.9 Diagram blok pengujian IMU

3.3.3 Pengujian *Flex Sensor*

Pengujian ini bertujuan untuk mengetahui berapa resistansi *flex sensor* ketika diberi kelengkungan mengikuti gerakan jari tangan dan untuk mengetahui pengaruh resistansi *flex sensor* terhadap sudut servo yang dihasilkan. Pengujian ini dilakukan dengan menempelkan *flex sensor* pada jari tangan lalu dilihat berapa

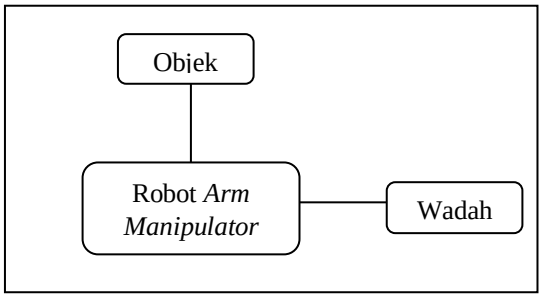
resistansi dari *flex sensor* pada *serial monitor* Arduino kemudian dapat dilihat berapa sudut servo yang dihasilkan motor servo. Alat yang digunakan pada pengujian ini yaitu motor servo yang ditempelkan busur, *flex sensor*, catu daya 5V, mikrokontroler Arduino, dan komputer.



Gambar 3.10 Diagram blok pengujian *flex sensor*

3.3.4 Pengujian Keseluruhan Sistem

Pengujian keseluruhan sistem bertujuan untuk mengetahui performa keseluruhan sistem apakah sesuai dengan target yang ingin dicapai. Penguijan ini dilakukan dengan melibatkan keseluruhan sistem *arm manipulator* untuk melakukan pemindahan suatu objek yaitu sebuah penghapus. *Arm manipulator* dituntut untuk mengambil objek tersebut kemudian diharuskan untuk memindahkannya ke suatu wadah. Pengujian dilakukan dengan menempatkan objek yang akan dipindah dengan jarak antara objek tersebut ke wadah yang akan dituju dimulai dengan jarak sebesar 5 cm sampai 30 cm dengan kenaikan bertahap sebesar 5 cm. Alat yang digunakan pada pengujian ini yaitu keseluruhan robot *arm manipulator*, catu daya 5V, mikrokontroler Arduino, sensor MPU6050, *flex sensor*, dan komputer.



Gambar 3.11 Skema pengujian keseluruhan sistem

BAB IV

HASIL DAN PEMBAHASAN

Setelah melakukan perhitungan dan perancangan langkah selanjutnya adalah pembahasan hasil pengujian yang bertujuan untuk menganalisis alat yang telah dirancang dan diimplementasikan telah bekerja sesuai dengan perancangan yang diharapkan. Pengujian dilakukan tiap-tiap blok dengan tujuan untuk mengamati apakah tiap blok sistem sudah sesuai dengan perancangan, kemudian dilanjutkan dengan pengujian secara keseluruhan sistem. Adapun pengujian yang perlu dilakukan sebagai berikut.

1. Pengujian Filter Kalman
    a) Pengujian Keluaran Filter Kalman Ketika Pergerakan
    b) Pengujian Keakuratan Filter Kalman
2. Pengujian IMU
    a) Pengujian Sudut *Pitch* Terhadap Sudut Servo yang Dihasilkan
    b) Pengujian Sudut *Roll* Terhadap Sudut Servo yang Dihasilkan
    c) Pengujian Sudut *Yaw* Terhadap Sudut Servo yang Dihasilkan
3. Pengujian *Flex Sensor*
    a) Pengujian Keluaran Resistansi *Flex Sensor* Sesuai Gestur Jari Tangan
    b) Pengujian Resistansi *Flex Sensor* Terhadap Sudut Servo yang Dihasilkan
4. Penguijan Keseluruhan Sistem

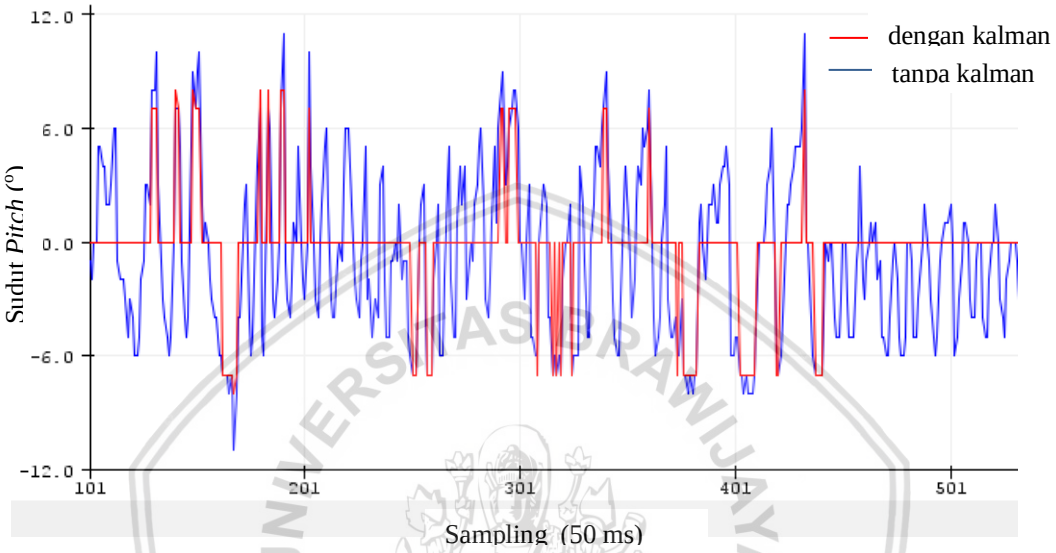
4.1 Pengujian Filter Kalman

4.1.1 Pengujian Keluaran Filter Kalman Ketika Pergerakan

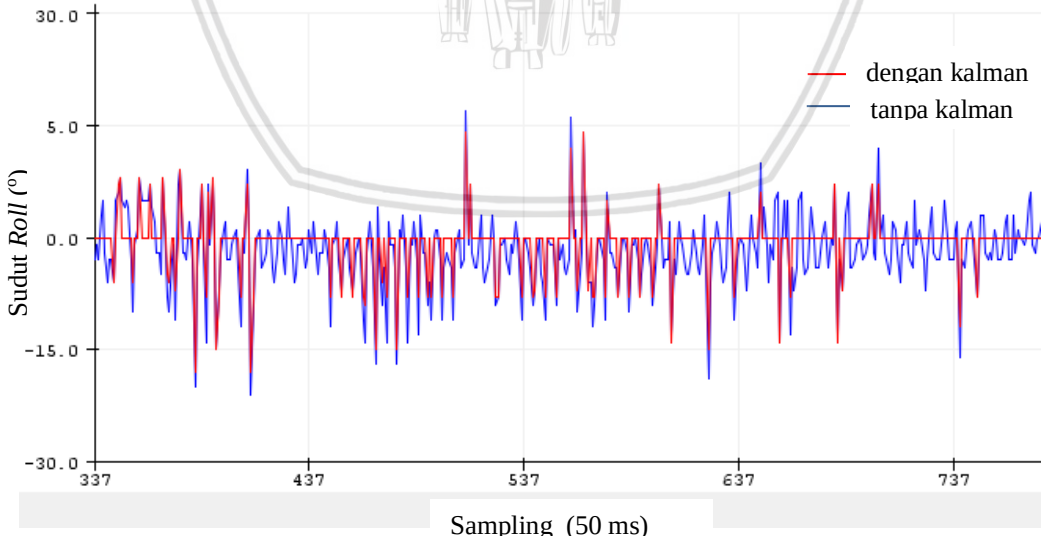
Pada pengujian ini dilakukan untuk mengetahui sudut yang dihasilkan dari keluaran akselerometer yaitu *pitch* dan *roll* yang difilter kalman apabila diberi gerakan berupa goncangan pada sudut kemiringan konstan yaitu 0°. Sedangkan untuk sudut yang dihasilkan dari keluaran giroskop yaitu *yaw* tidak diuji pada pengujian ini dikarenakan keluaran dari giroskop yang kurang akurat apabila diberi goncangan.

Pada pengujian ini parameter R (kovarian *noise* pengukuran) ditetapkan bernilai 0,5 dan parameter Q (kovarian *noise* proses) ditetapkan bernilai 100 dengan periode *sampling* 50 ms. Gambar 4.1 menunjukkan grafik sudut *pitch*

tanpa filter kalman dengan sudut *pitch* ketika memakai filter kalman hasil dari keluaran akselerometer apabila diberi goncangan pada sumbu x. Gambar 4.2 menunjukkan grafik sudut *roll* tanpa filter kalman dengan sudut *roll* ketika memakai filter kalman hasil dari keluaran akselerometer apabila diberi goncangan pada sumbu y.



Gambar 4.1 Grafik keluaran *pitch* filter kalman ketika diberi goncangan pada sumbu x



Gambar 4.2 Grafik keluaran *roll* filter kalman ketika diberi goncangan pada sumbu y

Dari kedua grafik di atas dapat dilihat bahwa filter kalman (kurva merah) dapat meredam pengaruh goncangan pada sudut *pitch* dan *roll* pada kemiringan 0^o, sedangkan jika tidak memakai filter kalman (kurva biru) maka amplitudo kurva menjadi lebih besar dikarenakan akselerometer juga memperhitungkan percepatan linear.

4.1.2 Pengujian Keakuratan Filter Kalman

Penguijan ini bertujuan untuk mengetahui keakuratan dari filter kalman dengan membandingkan antara *output* yaitu sudut *pitch, roll,* dan *yaw* tanpa filter kalman dan dengan filter kalman. Pengujian ini dilakukan masing-masing 1 kali dimulai dari sudut 15^o sampai 180^o (kelipatan 15) dengan nilai parameter R sebesar 0.5 dan nilai parameter Q sebesar 100 dengan periode sampling 50 ms. Pengujian dilakukan dengan menggerakkan sensor MPU6050 pada sumbu x untuk mendapatkan sudut *pitch,* sumbu y untuk mendapatkan sudut *roll,* dan sumbu z untuk mendapatkan sudut *yaw.*

Kemudian dilihat berapa sudut servo yang dihasilkan (sudut servo aktual) lalu dilihat berapa nilai sudut hasil perhitungan pada mikrokontroler Arduino melalui *serial monitor* pada saat *steady state*. Pengujian dilakukan terpisah antara sudut *pitch, roll,* dan *yaw* yang kemudian membandingkannya dengan sudut servo aktual hasil dari gerakan *pitch, roll,* dan *yaw.*

Perbedaan keluaran sudut *pitch, roll,* dan *yaw* tanpa filter kalman dan dengan filter kalman dapat dilihat pada tabel 4.1 dan tabel 4.2 di bawah ini. Terbukti bahwa presentase *error* pada sudut *pitch, roll,* dan *yaw* tanpa filter kalman lebih besar dibanding dengan memakai filter kalman. Presentase *error* sudut *pitch, roll,* dan *yaw* tanpa filter kalman masing-masing bernilai 4.17%, 5.51 %, dan 3.08%. Presentase *error* sudut *pitch, roll,* dan *yaw* tanpa filter kalman masing-masing bernilai 1.31%, 1.30%, dan 2.17%. Perhitungan presentase *error* didapat pada persamaan berikut ini.

$$\%\ error = \left|\frac{\text{sudut servo aktual} - \text{sudut } pitch}{\text{sudut servo aktual}} \times 100\%\right| \quad (4\text{-}1)$$

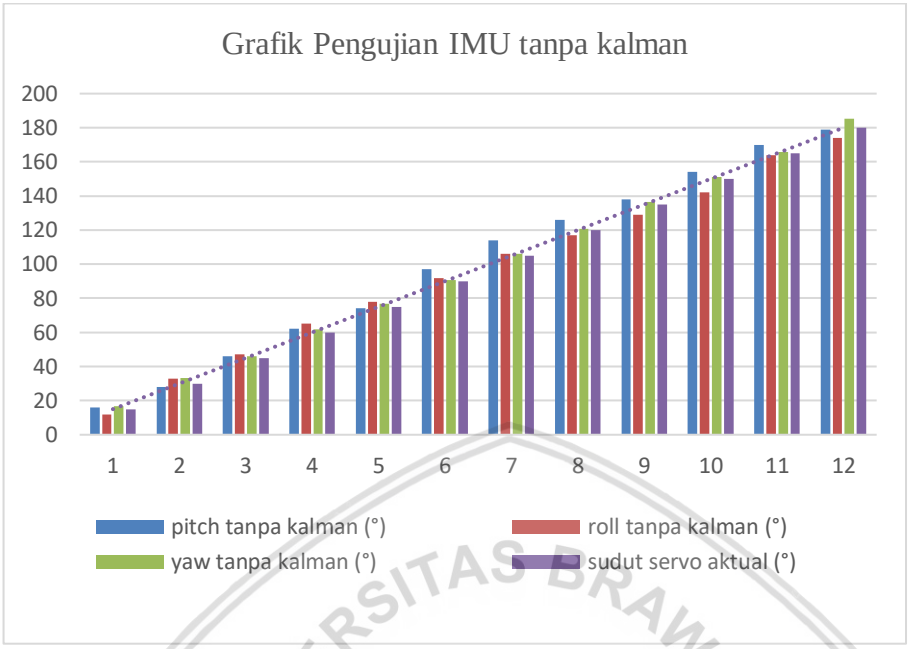
Tabel 4.1 Perbandingan antara nilai sudut *pitch, roll,* dan *yaw* tanpa filter kalman dengan sudut servo aktual

Sudut servo aktual (°)	***Pitch***		***Roll***		***Yaw***	
	tanpa kalman (°)	***error* (%)**	**tanpa kalman (°)**	***error* (%)**	**tanpa kalman (°)**	***error* (%)**
15	16	6.67	12	20.00	16.73	11.53
30	28	6.67	33	10.00	33.15	10.50
45	46	2.22	47	4.44	45.90	2.00
60	62	3.33	65	8.33	61.66	2.77
75	74	11.33	78	4.00	76.98	2.64
90	97	7.78	92	2.22	90.75	0.83
105	114	8.57	106	0.95	106.02	0.97
120	126	5.00	117	2.50	120.68	0.57
135	138	2.22	129	4.44	136.37	1.01
150	154	2.67	142	5.33	151.03	0.69
165	170	3.03	164	0.61	165.90	0.55
180	179	0.56	174	3.33	185.22	2.90
***error* rata-rata**		4.17		5.51		3.08

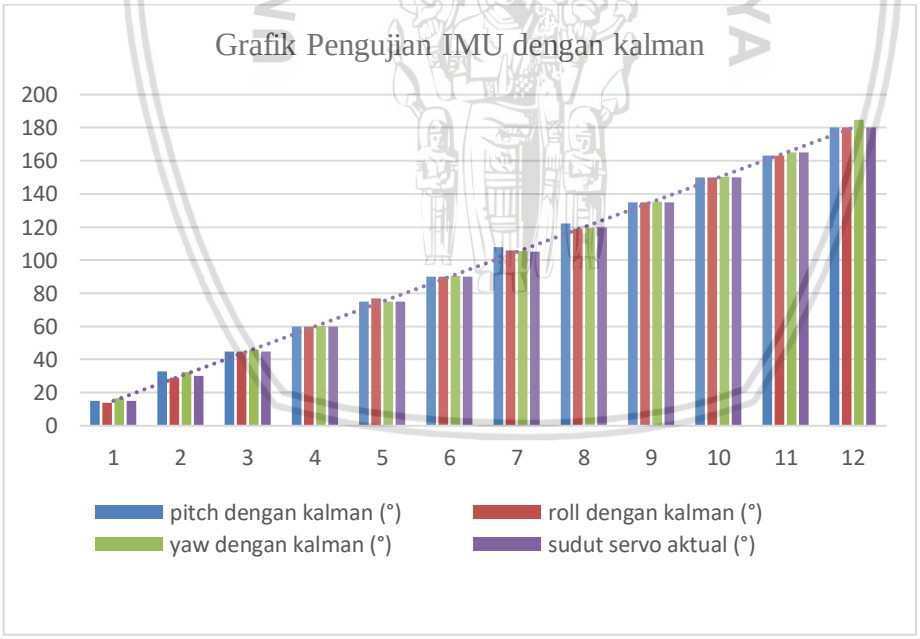
Tabel 4.2 Perbandingan antara nilai sudut *pitch*, *roll*, dan *yaw* dengan filter kalman dengan sudut servo aktual

Sudut servo aktual(°)	***Pitch***		***Roll***		***Yaw***	
	kalman (°)	***error* (%)**	**kalman (°)**	***error* (%)**	**kalman (°)**	***error* (%)**
15	15	0.00	14	6.67	16.42	9.47
30	33	10.00	29	3.33	32.46	8.20
45	45	0.00	45	0.00	46.18	2.62
60	60	0.00	60	0.00	60.39	0.65
75	75	0.00	77	2.67	74.85	0.20
90	90	0.00	90	0.00	90.41	0.46
105	108	2.86	106	0.95	105.72	0.69
120	122	1.67	119	0.83	119.56	0.37
135	135	0.00	135	0.00	135.36	0.27
150	150	0.00	150	0.00	150.43	0.29
165	163	1.21	163	1.21	165.25	0.15
180	180	0.00	180	0.00	184.73	2.63
***error* rata-rata**		1.31		1.30		2.17

Berikut adalah grafik dari kedua tabel diatas.



Gambar 4.3 Grafik sudut IMU tanpa filter kalman terhadap sudut servo aktual yang dihasilkan



Gambar 4.4 Grafik sudut IMU dengan filter kalman terhadap sudut servo aktual yang dihasilkan

Dari kedua gambar grafik diatas dapat dilihat bahwa grafik sudut IMU dengan filter kalman lebih linier terhadap sudut servo aktual dibanding dengan sudut IMU tanpa filter kalman.

4.2 Pengujian IMU

4.2.1 Pengujian Sudut *Pitch* Terhadap Sudut Servo yang Dihasilkan

Pengujian ini bertujuan untuk mengetahui apakah sudut *pitch* yang didapat dari IMU setelah difilter kalman sesuai dengan sudut servo yang dihasilkan atau sudut servo aktual. Pengujian ini dilakukan sebanyak 5 kali untuk masing-masing sudut dimulai dari 15^o sampai 180^o (kelipatan 15). Kemudian dari pengujian masing-masing sebanyak 5 kali tadi dirata-ratakan dan dihitung berapa presentase *error* nya. Pengujian ini terbukti bahwa nilai sudut *pitch* yang didapat dari keluaran akselerometer setelah difilter kalman sangat mendekati nilai sudut servo yang dihasilkan dengan presentase *error* sebesar 0,73%.

Gambar 4.5 Servo dipasangi busur yang dibandingkan dengan nilai sudut dari IMU

Tabel 4.3 Perbandingan antara nilai sudut *pitch* dengan sudut servo aktual

Sudut servo aktual (°)	**Pengujian *pitch* (°) ke-**					**rata-rata (°)**	***error* (%)**
	1	**2**	**3**	**4**	**5**		
15	15	14	15	15	15	14.8	1.33
30	29	29	29	29	29	29.0	3.33
45	45	45	45	45	45	45.0	0.00
60	60	60	60	60	60	60.0	0.00
75	75	75	75	75	75	75.0	0.00
90	90	90	90	90	90	90.0	0.00
105	108	106	108	106	108	107.2	2.10
120	123	119	119	119	122	120.4	0.33
135	134	134	134	134	134	134.0	0.74
150	150	150	150	150	150	150.0	0.00
165	163	163	164	163	164	163.4	0.97
180	180	180	180	180	180	180.0	0.00
						***error* rata-rata**	0.73

4.2.2 Pengujian Sudut *Roll* Terhadap Sudut Servo yang Dihasilkan

Pengujian ini bertujuan untuk mengetahui apakah sudut *roll* yang didapat dari IMU setelah difilter kalman sesuai dengan sudut servo yang dihasilkan atau sudut servo aktual. Pengujian ini dilakukan sebanyak 5 kali untuk masing-masing sudut dimulai dari 15^o sampai 180^o (kelipatan 15). Kemudian dari pengujian masing-masing sebanyak 5 kali tadi dirata-ratakan dan dihitung berapa presentase *error* nya. Pengujian ini terbukti bahwa nilai sudut *roll* yang didapat dari keluaran akselerometer setelah difilter kalman sangat mendekati nilai sudut servo yang dihasilkan dengan presentase *error* sebesar 0.67%.

Tabel 4.4 Perbandingan antara nilai sudut *roll* dengan sudut servo aktual

Sudut servo aktual (°)	**Pengujian *roll* (°) ke-**					**rata-rata (°)**	***error* (%)**
	1	**2**	**3**	**4**	**5**		
15	15	14	15	15	15	14.8	1.33
30	29	29	29	29	29	29.0	3.33
45	45	45	45	45	45	45.0	0.00
60	60	60	60	60	60	60.0	0.00
75	75	75	75	75	75	75.0	0.00
90	90	90	90	90	90	90.0	0.00
105	108	106	108	106	108	107.2	2.10
120	123	119	119	119	122	120.4	0.33
135	135	135	135	135	135	135.0	0.00
150	150	150	150	150	150	150.0	0.00
165	163	163	164	163	164	163.4	0.97
180	180	180	180	180	180	180.0	0.00
						***error* rata-rata**	0.67

4.2.3 Pengujian Sudut *Yaw*Terhadap Sudut Servo yang Dihasilkan

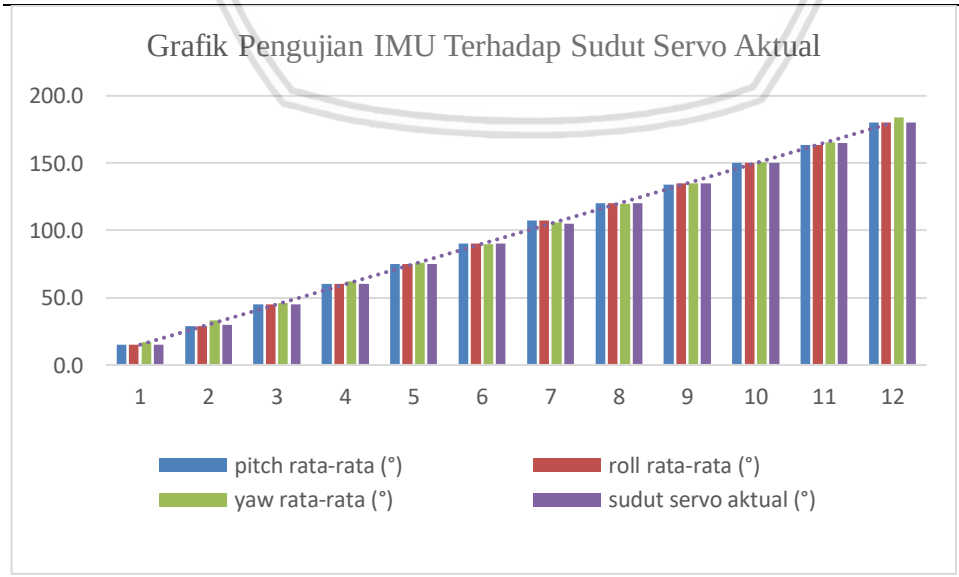
Pengujian ini bertujuan untuk mengetahui apakah sudut *yaw* yang didapat dari IMU setelah difilter kalman sesuai dengan sudut servo yang dihasilkan atau sudut servo aktual. Pengujian ini dilakukan sebanyak 5 kali untuk masing-masing sudut dimulai dari 15^o sampai 180^o (kelipatan 15). Kemudian dari pengujian masing-masing sebanyak 5 kali tadi dirata-ratakan dan dihitung berapa presentase *error* nya. Pengujian ini terbukti bahwa nilai sudut *yaw* yang didapat dari

keluaran giroskop setelah difilter kalman mendekati nilai sudut servo yang dihasilkan dengan presentase *error* sebesar 2.96%.

Hasil pengujian *yaw* ini masih memiliki banyak kekurangan. Berbeda dengan *pitch* dan *roll* dengan filter kalman, *yaw* dengan filter kalman masih memiliki *error* yang cukup tinggi dan tingkat kestabilan data juga kurang baik. Hal ini dikarenakan data mentah dari giroskop yang tidak linear yang menyebabkan filter kalman tidak mampu bekerja optimal untuk mendapatkan data yang akurat dan stabil.

Tabel 4.5 Perbandingan antara nilai sudut *yaw* dengan sudut servo aktual

Sudut servo aktual (°)	**Pengujian *yaw* (°) ke-**					**rata-rata (°)**	***error* (%)**
	1	**2**	**3**	**4**	**5**		
15	17.03	16.17	17.15	17.40	17.27	17.00	13.36
30	33.81	32.71	32.95	33.44	33.07	33.20	10.65
45	45.20	45.08	45.57	47.16	46.43	45.89	1.97
60	61.74	62.35	63.94	62.35	61.49	62.37	3.96
75	76.79	74.97	75.22	75.46	76.20	75.73	0.97
90	89.43	89.79	89.67	88.93	91.63	89.89	0.12
105	104.98	105.96	106.45	107.31	105.72	106.08	1.03
120	119.68	119.32	118.70	120.42	120.91	119.81	0.16
135	135.61	135.36	134.26	134.14	134.51	134.78	0.17
150	150.92	150.31	150.55	150.31	151.66	150.75	0.50
165	164.64	165.50	165.38	164.64	167.46	165.52	0.32
180	182.52	182.28	184.73	185.47	185.34	184.07	2.26
					***error* rata-rata**		2.96



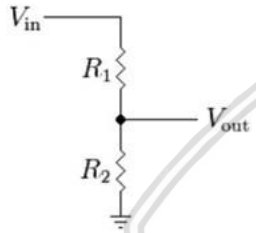
Gambar 4.6 Grafik sudut *pitch*, *roll*, dan *yaw* terhadap sudut servo aktual yang dihasilkan

4.3 Pengujian *Flex Sensor*

4.3.1 Pengujian Keluaran Nilai ADC *Flex Sensor* Sesuai Gestur Jari Tangan

Pengujian dilakukan dengan menempelkan *flex sensor* pada jari tangan, kemudian digerakan sesuai gestur jari tangan lalu dihitung secara manual berapa tegangan keluaran *flex sensor* yang didapat dari nilai resistansi *flex sensor*, lalu dihitung berapa nilai ADC dari *flex sensor* itu sendiri.

Pengujian dilakukan dengan menjadikan *flex sensor* sebagai R_1 pada rangkaian pembagi tegangan. Nilai R_2 ditetapkan sebesar 10kΩ, sedangkan output rangkaian pembagi tegangan dihubungkan ke pin analog A0 pada Arduino.



Gambar 4.7 Rangkaian pembagi tegangan

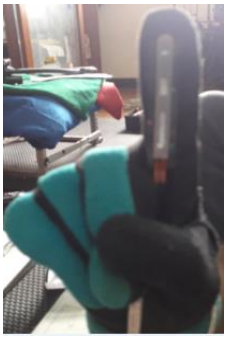
Tegangan *flex sensor* atau V_{out} didapat melalui rumus pembagi tegangan:

$$V_{flex} = \frac{R_2}{R_1 + R_2} \times V_{in} \qquad (3\text{-}2)$$

Nilai V_{in} adalah nilai tegangan catu daya yaitu 5V. Untuk mendapatkan nilai ADC *flex sensor* menggunakan rumus:

$$ADC_{flex} = \frac{V_{flex}}{V_{ref}} \times ADC_{ref} \qquad (3\text{-}3)$$

Arduino Uno memiliki resolusi 10 bit maka nilai ADC_{ref} adalah 1023. Sensor pada keadaan jari lurus ke atas dianalisa sebagai berikut:

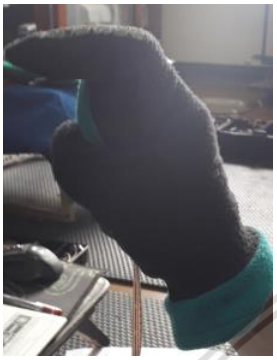


Gambar 4.8 Posisi jari lurus terhadap *flex sensor*

$$V_{flex} = \frac{10k}{145k + 10k} \times 5V = 0.322V$$

$$ADC_{flex} = \frac{0{,}322V}{5V} \times 1023 = 65.88$$

Sensor pada keadaan jari melengkung membentuk sudut 90^o dianalisa sebagai berikut:



Gambar 4.9 Posisi jari membentuk 90^o terhadap *flex sensor*

$$V_{flex} = \frac{10k}{266k + 10k} \times 5V = 0.181V$$

$$ADC_{flex} = \frac{0{,}181V}{5V} \times 1023 = 37.06$$

Sensor pada keadaan jari menggenggam dianalisa sebagai berikut:

Gambar 4.10 Posisi jari menggenggam terhadap *flex sensor*

$$V_{flex} = \frac{10k}{455k + 10k} \times 5V = 0.107V$$

$$ADC_{flex} = \frac{0{,}107V}{5V} \times 1023 = 22$$

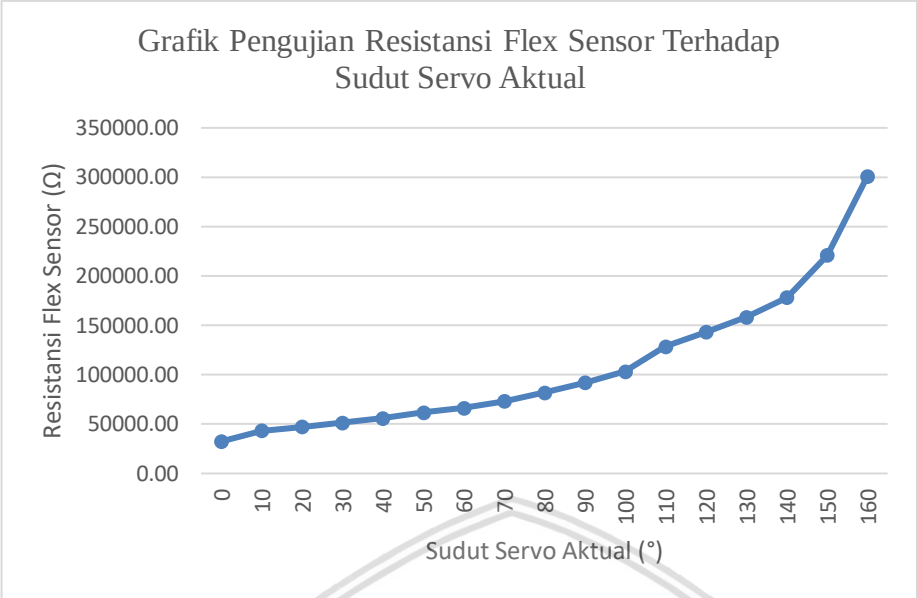
4.3.2 Pengujian Resistansi *Flex Sensor* Terhadap Sudut Servo yang Dihasilkan

Pengujian ini dilakukan untuk mengetahui hubungan antara resistansi dari *flex sensor* terhadap sudut servo yang dihasilkan atau sudut servo aktual. Pengujian dilakukan dengan menemplkan *flex sensor* pada jari telunjuk yang kemudian digerakkan mengikuti gerakan jari telunjuk.

Semakin besar lengkungan *flex sensor* maka akan semakin besar pula resistansinya dan akan semakin buesar pula sudut servo aktual hasil dari perubahan resistansi *flex sensor*. Pengujian dilakukan sebanyak 5 kali untuk mendapatkan nilai resistansi *flex sensor* pada masing-masing sudut servo aktual dimulai dari 0^o sampai 160^o. Kemudian dari 5 kali pengujian masing-masing sudut servo tadi dirata-ratakan nilai resistansi *flex sensor*.

Tabel 4.6 Pengujian resistansi *flex sensor* terhadap sudut servo aktual

Sudut servo aktual (°)	**Pengujian resistansi flex sensor (Ω) ke-**					**rata-rata (Ω)**	***hold* (Ω)**
	1	**2**	**3**	**4**	**5**		
0	33164.56	33272.73	32625.00	31250.00	32098.76	32482.21	3.2K
10	45597.83	41928.93	42461.54	43281.25	43005.18	43254.95	4.3K
20	46028.79	47471.91	48125.00	47796.61	46519.34	47188.33	4.7K
30	50532.25	52378.05	52000.00	50532.55	50842.56	51257.08	5.1K
40	56428.57	55159.24	56428.27	55159.24	56000.00	55835.06	5.6K
50	60551.73	62042.26	63597.13	62042.26	61041.67	61855.01	6.2K
60	65777.77	64921.88	66917.30	68091.60	66343.29	66410.37	6.6K
70	71840.00	72500.00	73170.73	72500.00	73852.45	73005.80	7.3K
80	82999.99	82162.16	81339.38	82162.16	80530.97	81838.93	8.2K
90	90299.12	92300.00	93333.34	92300.00	91287.13	91903.92	9.2K
100	98829.78	102147.18	103666.66	110352.35	101195.66	103238.33	10.3K
110	138200.86	126400.00	132083.34	124605.27	121153.84	128488.66	12.8K
120	147384.63	142886.86	140441.17	140441.17	145000.00	143230.77	14.3K
130	157704.91	160500.00	155000.00	160500.00	157704.91	158281.96	15.8K
140	179444.45	172678.56	175999.98	183018.88	179444.45	178117.26	17.8K
150	203125.00	239512.19	227906.98	222500.00	212391.31	221087.10	22.1K
160	331000.00	290882.34	282285.72	290882.34	309687.50	300947.58	30.0K



Gambar 4.11 Grafik resistansi *flex sensor* terhadap sudut servo aktual yang dihasilkan

Dari gambar grafik diatas bisa disimpulkan bahwa semakin melengkung *flex sensor* maka akan semakin besar nilai resistansinya dan akan semakin besar pula sudut servo yang dihasilkan dari nilai resistansi tersebut.

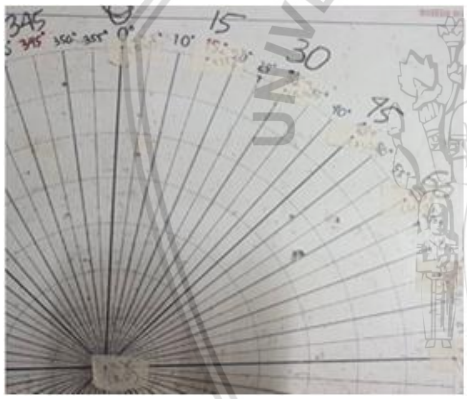
4.4 Pengujian Keseluruhan Sistem

Pengujian ini dilakukan dengan melibatkan keseluruhan sistem yaitu robot *arm manipulator*, IMU dan *flex sensor* yang ditempelkan pada sarung tangan, mikrokontroler Arduino, baterai lipo 8V, regulator *step-down* 5V, dan lapangan pengujian.

Pengujian ini dilakukan dengan menggerakkan robot *arm manipulator* menggunakan IMU dan *flex sensor* sesuai gestur pergelangan tangan. *Arm manipulator* digerakkan dengan tujuan untuk mengambil suatu objek lalu *arm manipulator* digerakkan untuk memindahkan objek tersebut ke wadah yang dituju. Objek yang dipindahkan berupa gulungan plastik dengan tinggi 11cm dan lebar 3cm sebagai berikut:

Gambar 4.12 Objek untuk dipindahkan

Pengujian dilakukan sebanyak 6 kali di atas lapangan pengujian yang telah terdapat gambar sumbu X, sumbu Y, dan titik koordinatnya. *Arm manipulator* ditempatkan pada koordinat XY (0,0), lalu objek ditempatkan pada koordinat (0,30) agar tegak lurus dengan robot *arm manipulator.* Sedangkan wadah yang dituju berada pada titik koordinat yang berbeda-beda pada kuadran I yang telah ditentukan titik koordinatnya.



Gambar 4.13 Lapangan tempat pengujian

Gambar 4.14 Robot *arm manipulator* dikendalikan dengan IMU dan flex sensor berdasarkan gestur pergelangan tangan

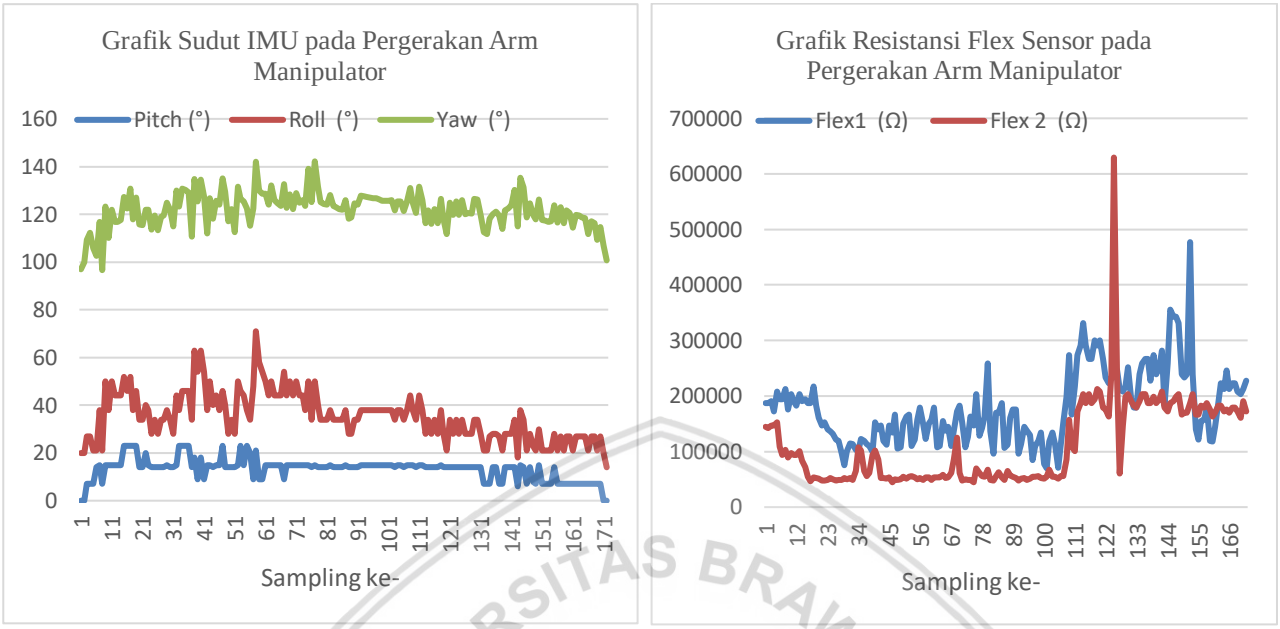
Tabel 4.7 Pengujian keseluruhan sistem (pemindahan objek)

No.	Posisi robot (cm)		Posisi objek (cm)		Posisi wadah (cm)		Keterangan	Waktu Pengujian	Respon waktu Rata-rata
	X	Y	X	Y	X	Y			
1	0	0	0	30	7.5	28.5	Berhasil	23.84 detik	3.09 ms
2					15	25.5	Berhasil	24.26 detik	3.06 ms
3					21	21	Berhasil	25.98 detik	3.06 ms
4					26	15	Berhasil	23.75 detik	3.07 ms
5					29	10	Berhasil	25.55 detik	3.07 ms
6					30	0	Berhasil	39.74 detik	3.08 ms

Dari tabel diatas dapat dilihat bahwa robot *arm manipulator* berhasil memindahkan objek ke wadah yang dituju pada koordinat wadah yang berbeda-beda dengan waktu pengujian yang bervariasi. Waktu pengujian yaitu waktu dari mulai robot bergerak sampai robot meletakkan objek ke wadah.

Sedangkan respon waktu adalah selisih waktu dari sensor mulai bekerja sampai aktuator mulai bergerak. Respon waktu dihitung setiap kali *sampling* dengan periode setiap *sampling* adalah 100 ms. Lalu respon waktu tadi dirata-ratakan setiap kali pengujian seperti dilihat pada tabel diatas. Berikut akan ditampilkan grafik nilai sudut *pitch*, *roll*, dan *yaw* dari IMU dan nilai resistansi *flex sensor* setiap kali *sampling* pada setiap kali pengujian pemindahan objek. Untuk data keseluruhan bisa dilihat di lampiran.

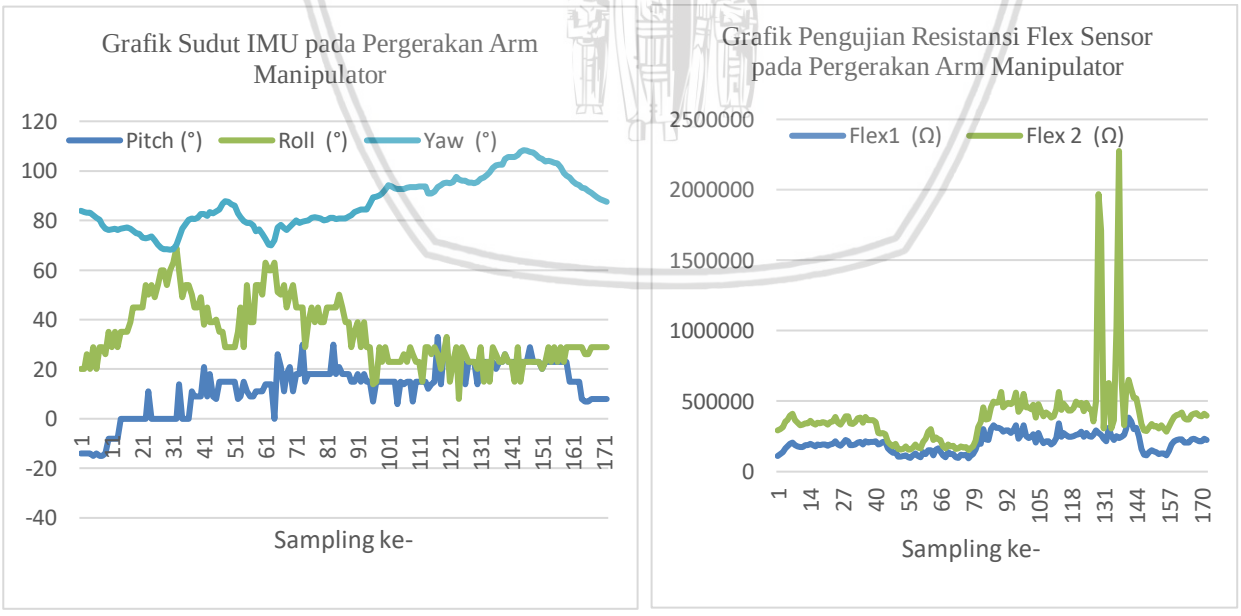
- **Koordinat robot (0,0), koordinat objek (0,30), koordinat wadah (7.5,22.5)**



(a) (b)

Gambar 4.15 (a) Grafik sudut IMU pada pengujian keseluruhan pertama (b) Grafik resistansi *flex sensor* pada pengujian keseluruhan pertama

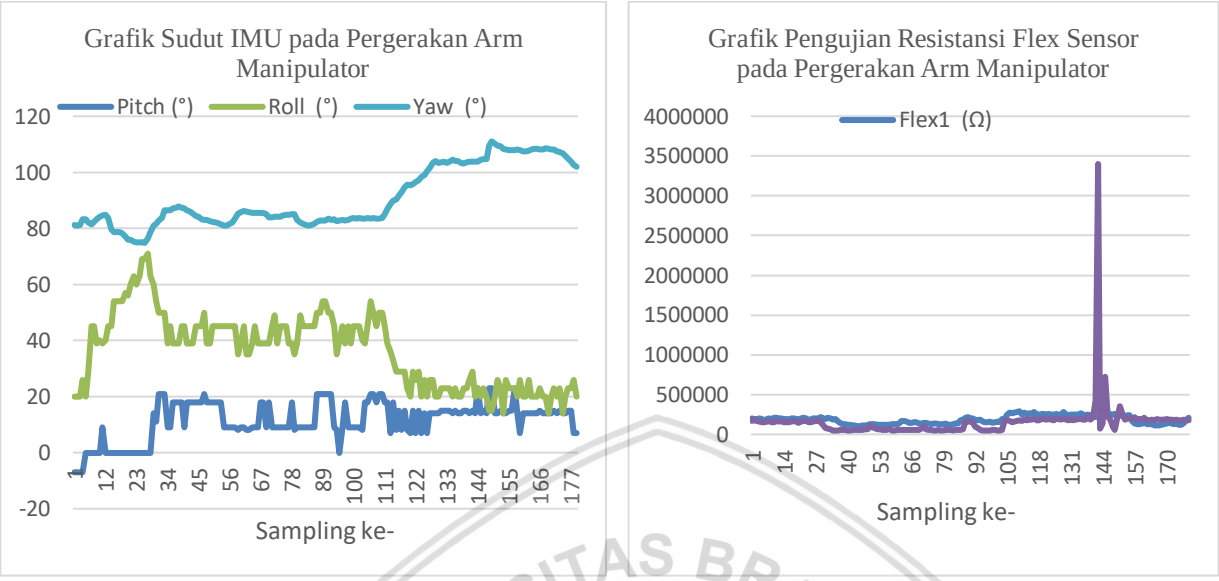
- **Koordinat robot (0,0), koordinat objek (0,30), koordinat wadah (15,25.5)**



(a) (b)

Gambar 4.16 (a) Grafik sudut IMU pada pengujian keseluruhan kedua (b) Grafik resistansi *flex sensor* pada pengujian keseluruhan kedua

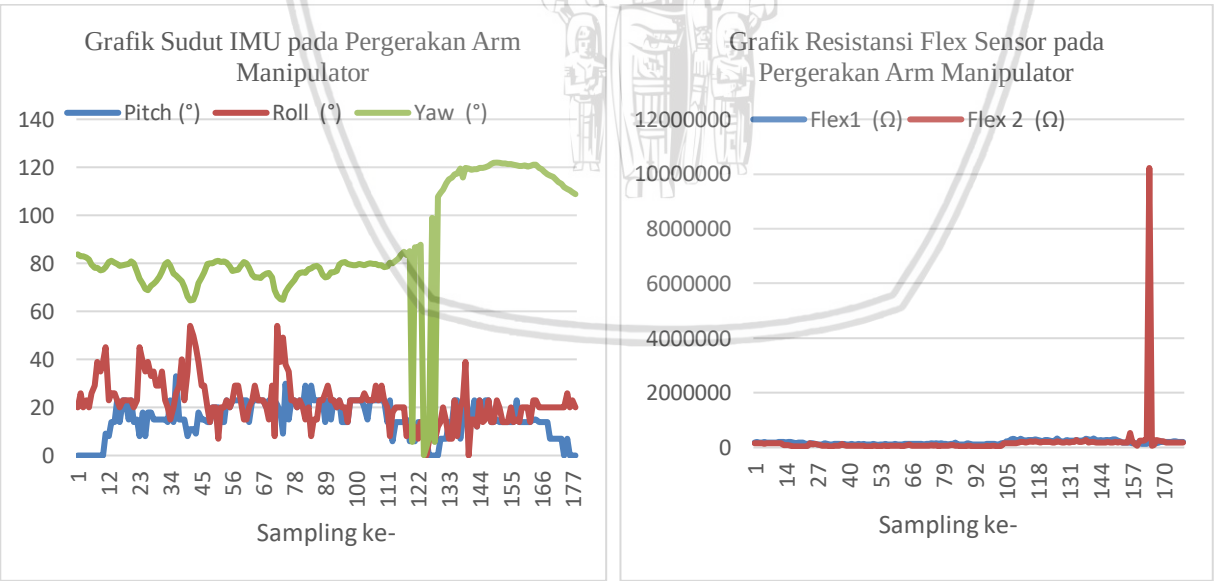
- **Koordinat robot (0,0), koordinat objek (0,30), koordinat wadah (21,21)**



(a) (b)

Gambar 4.17 (a) Grafik sudut IMU pada pengujian keseluruhan ketiga (b) Grafik resistansi *flex sensor* pada pengujian keseluruhan ketiga

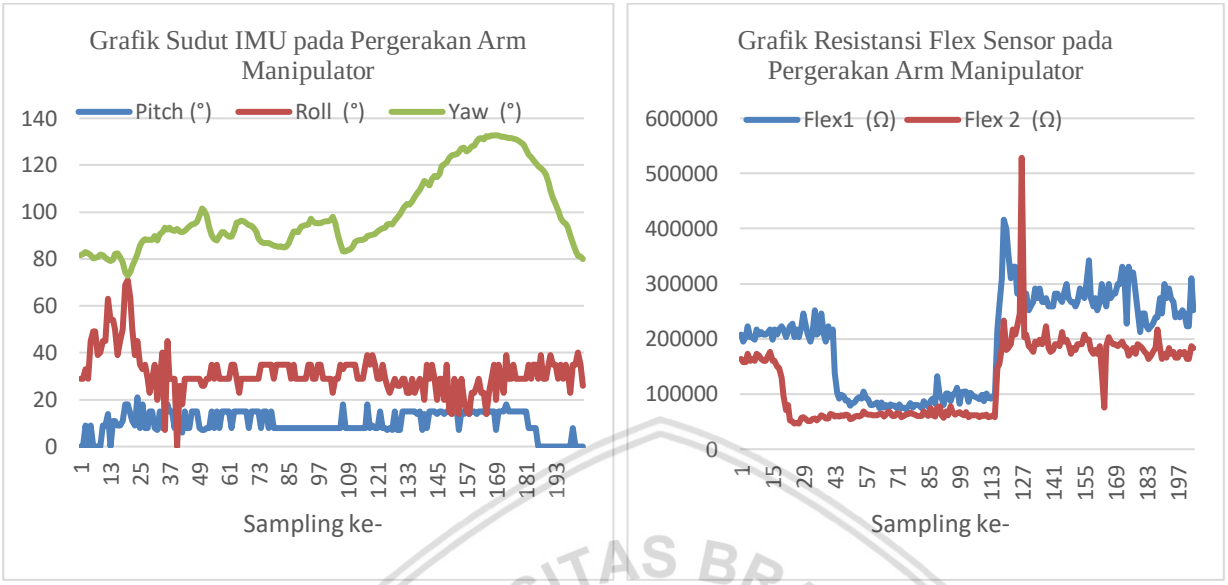
- **Koordinat robot (0,0), koordinat objek (0,30), koordinat wadah (26,15)**



(a) (b)

Gambar 4.18 (a) Grafik sudut IMU pada pengujian keseluruhan keempat (b) Grafik resistansi *flex sensor* pada pengujian keseluruhan keempat

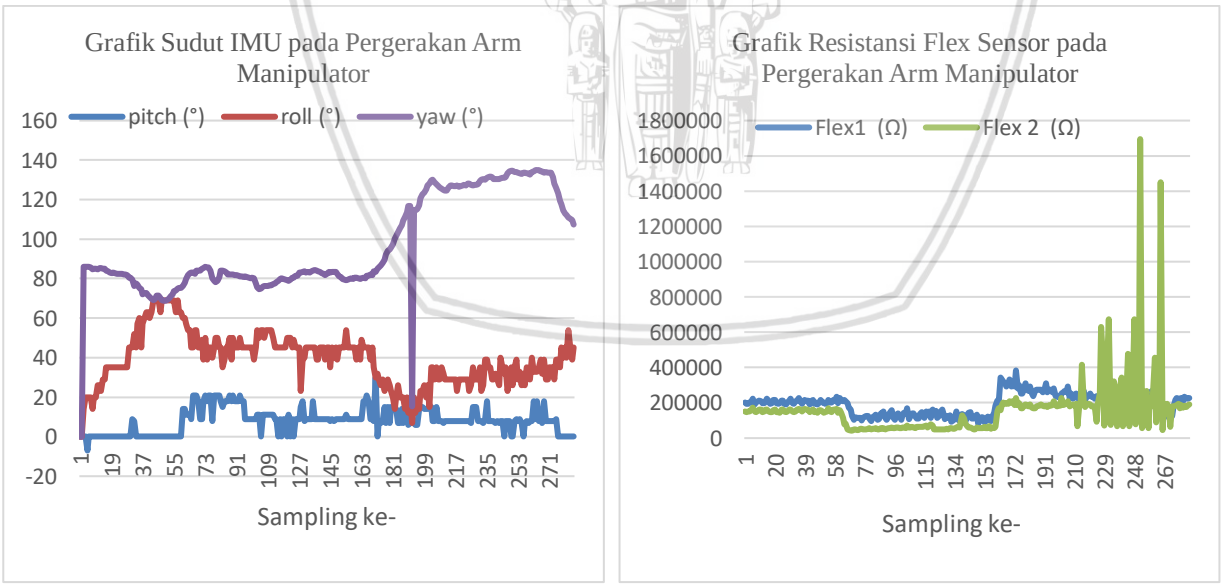
- **Koordinat robot (0,0), koordinat objek (0,30), koordinat wadah (29,10)**



(a) (b)

Gambar 4.19 (a) Grafik sudut IMU pada pengujian keseluruhan keempat (b) Grafik resistansi *flex sensor* pada pengujian keseluruhan kelima

- **Koordinat robot (0,0), koordinat objek (0,30), koordinat wadah (30,0)**



(a) (b)

Gambar 4.20 (a) Grafik sudut IMU pada pengujian keseluruhan keempat (b) Grafik resistansi *flex sensor* pada pengujian keseluruhan keenam

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil pengujian dan analisis setiap bagian serta keseluruhan sistem yang telah dilakukan, maka dapat diambil kesimpulan sebagai berikut:

1. Pada pengujian IMU, sudut *pitch, roll,* dan *yaw* yang didapat dari IMU mampu menggerakkan motor servo dengan presentase *error* rata-rata dari lima kali pengujian setiap sudut servo kelipatan 15 masing-masing 0.73%, 0.67%, dan 2.96%.
2. Nilai resistansi *flex sensor* akan semakin besar jika semakin melengkung mengikuti jari tangan. Berdasarkan pengujian nilai resistansi *flex sensor* terhadap sudut servo yang dihasilkan minimal adalah 3.2kΩ menghasilkan sudut servo 0^o dan maksimal adalah 30kΩ menghasilkan sudut servo 160^o.
3. Filter kalman terbukti mampu menambah keakuratan nilai sudut pada IMU dimana presentase *error* rata-rata sudut *pitch, roll,* dan *yaw* ketika difilter kalman masing-masing 1.31%, 1.30%, dan 2.17%. Sedangkan presentase *error* rata-rata sudut *pitch, roll,* dan *yaw* tanpa filter kalman masing-masing 4.17%, 5.51%, dan 3.08%.
4. Robot *arm manipulator* mampu memindahkan objek ke wadah yang dituju yang dikendalikan dengan IMU dan *flex sensor* berdasarkan gestur pergelangan tangan selama 6 kali pengujian dimana *arm manipulator* diletakkan pada koordinat (0,0), objek diletakkan pada koordinat (0,30), dan wadah yang dituju masing-masing ditempatkan pada koordinat (7.5,28.5), (15,25.5), (21,21), (26,15), (29,10), dan (30,0).

5.2 Saran

Terdapat beberapa saran dalam penelitian ini untuk perbaikan kedepannya antara lain:

1. Dalam penelitian ini sudut *yaw* didapat dari giroskop IMU yang mana dalam praktek nya giroskop kurang akurat pada saat kalibrasi. Oleh karena itu sebaiknya memakai IMU yang memiliki sensor magnetometer atau kompas agar didapat sudut *yaw* yang lebih akurat.

2. Aktuator yang digunakan pada robot *arm manipulator* sebaiknya adalah servo yang memiliki torsi yang lebih besar khususnya untuk servo bagian *body* dan *shoulder* yang dituntut mengangkat beban yang lebih berat.

DAFTAR PUSTAKA

Andrianto, Heri. (2015). *Pemrograman Mikrokontroler AVR Atmega16 Menggunakan Bahasa C (CodeVisionAVR).* Bandung: Informatika.

Hasan. (2018). *Pembaca Kelajuan Kendaraan Menggunakan IMU.* Malang: Universitas Brawijaya.

Kusmanto, Nando. (2009). *Rancang Bangun Sistem Navigasi Inersia Dengan Kalman Filter Pada Mikrokontroler AVR.* Depok: Universitas Indonesia.

Mubarok, Asep. (2011). *Pendeteksi Rotasi Menggunakan Gyroscope Berbasis Mikrokontroler ATmega8535*. Semarang: Universitas Diponegoro.

Saifuddin, Arief. (2017). *Perancangan Sistem Kendali Pergerakan Arm Manipulator Berbasis Sensor Inertial Measurement Unit (IMU) dan Sensor Flex*. Semarang: Universitas Diponegoro.

Sharma, Ashish, dkk. (2014). *Design and Implementation of Anthropomorpic Robotic Arm*. Department of Electronics and Telecomunication: St. Francis of Technology Mumbai, India.

Surya, Frans. 2018. *I^2C Protokol*. http://comp-eng.binus.ac.id/files/2014/05/Artikel-I2C-Protokol.pdf/. (diakses tanggal 9 September 2018).

Welch, Greg & Bishop, Gary. (2006). *An Introduction to the Kalman Filter*. Chappel Hill: Department of Computer Science University of North Carolina.