Institut
Teknologi
Sepuluh Nopember

TUGAS AKHIR – TE 141599

SISTEM STABILISASI NAMPAN MENGGUNAKAN IMU SENSOR DAN ARDUINO NANO

Abu Hatim Kurniawan
NRP 07111645000072

Dosen Pembimbing
Dr. Muhammad Rivai S.T., M.T.

DEPARTEMEN TEKNIK ELEKTRO
Fakultas Teknologi Elektro
Institut Teknologi Sepuluh Nopember
Surabaya 2018

-----Halaman ini sengaja dikosongkan-----

TUGAS AKHIR – TE 141599

SISTEM STABILISASI NAMPAN MENGGUNAKAN IMU SENSOR DAN ARDUINO NANO

Abu Hatim Kurniawan
NRP 07111645000072

Dosen Pembimbing
Dr. Muhammad Rivai, S.T., M.T.

DEPARTEMEN TEKNIK ELEKTRO
Fakultas Teknologi Elektro
Institut Teknologi Sepuluh Nopember
Surabaya 2018

FINAL PROJECT – TE 145561

TRAY STABILIZATION SYSTEM USING IMU SENSOR AND ARDUINO NANO

Abu Hatim Kurniawan
NRP 07111645000072

Advisor Lecturer:
Dr. Muhammad Rivai, S.T., M.T.

DEPARTEMENT OF ELECTRICAL ENGINEERING
Faculty of Industrial Technology
Institut Teknologi Sepuluh Nopember
Surabaya 2018

PERNYATAAN KEASLIAN TUGAS AKHIR

Dengan ini saya menyatakan bahwa isi sebagian maupun keseluruhan Tugas Akhir saya dengan judul "**SISTEM STABILISASI NAMPAN MENGGUNAKAN IMU SENSOR DAN ARDUINO NANO**" adalah benar-benar hasil karya intelektual mandiri, diselesaikan tanpa menggunakan bahan-bahan yang tidak diijinkan dan bukan merupakan karya pihak lain yang saya akui sebagai karya sendiri.

Semua referensi yang dikutip maupun dirujuk telah ditulis secara lengkap pada daftar pustaka.

Apabila ternyata pernyataan ini tidak benar, saya bersedia menerima sanksi sesuai peraturan yang berlaku.

Surabaya, Juli 2018

Abu Hatim Kurniawan
NRP 07111645000072

-----Halaman ini sengaja dikosongkan-----

SISTEM STABILISASI NAMPAN MENGGUNAKAN IMU SENSOR DAN ARDUINO NANO

TUGAS AKHIR

Diajukan Guna Memenuhi Sebagian Persyaratan
Untuk Memperoleh Gelar Sarjana Teknik
Pada
Bidang Studi Elektronika
Departemen Teknik Elektro
Institut Teknologi Sepuluh Nopember

Menyetujui:

Dosen Pembimbing I

Dr. Muhammad Rivai, S.T., M.T.
NIP : 196904261994031003

SURABAYA
JULI, 2018

-----Halaman ini sengaja dikosongkan-----

Sistem Stabilisasi Nampan Menggunakan IMU Sensor Dan Arduino Nano

Nama : Abu Hatim Kurniawan
Pembimbing : Dr. Muhammad Rivai, ST. MT..

ABSTRAK

Penderita penyakit parkinson kerap kali mengalami kesulitan dalam membawa sesuatu barang. Hal ini dikarenakan berkurangnya kemampuan syaraf motorik sehingga mengakibatkan beberapa bagian tubuh bergetar terutama tangan. Pada saat ini belum terdapat suatu media yang digunakan untuk membawa makanan atau barang yang stabil terhadap goncangan. Pada penelitian ini diusulkan membuat suatu nampan yang seimbang dengan menggunakan *Inertial Measurement Unit* (IMU) Sensor MPU6050. Sensor tersebut mampu mendeteki perubahan sudut atau posisi pada 3 dimensi. Sistem ini menggunakan mikrokontroler Arduino Nano sebagai pemroses sinyal yang diberikan oleh sensor. Mikrokontroler ini mempunyai pin *input/output* baik digital maupun *analog* dan *Analog To Digital Convertion* (ADC) pada mikrokontroler tersebut mampu untuk mengolah *output* sensor. Bentuk fisik Arduino Nano mempunyai ukuran yang kecil sehingga *portable* dan tidak terlalu berat dalam stabilisator tersebut. Nilai galat yang merupakan selisih antara *setting point* dan keluaran sensor tersebut kemudian akan digunakan sebagai sinyal masukan kontroler *Proportional Integrator Derivative* (PID). Motor servo digunakan sebagai aktuator yang akan bergerak sesuai dengan besarnya galat, sehingga akan menghasilkan kestabilan gerakan nampan. Hasil pengujian sistem *success rate* stabilisator ketika sistem tanpa beban adalah sebesar 100% untuk keadaan diam dan 60% ketika keadaan berjalan. Sedangkan pada keadaan dengan beban didapatkan success rate sebesar 70% pada keadaan diam dan 60% saat keadaan berjalan. Hasil penelitian ini diharapkan mengurangi resiko jatuh atau tumpahnya barang atau makanan terutama pada saat dibawa oleh penderita parkinson.

Kata Kunci : IMU Sensor, Kontroler *Proportional, Stabilisator*

-----Halaman ini sengaja dikosongkan-----

Tray Stabilization System Using IMU Sensor And Arduino Nano

Name : Abu Hatim Kurniawan
Advisor : Dr. Muhammad Rivai, ST. MT.

ABSTRACT

People with Parkinson's disease often have difficulty in carrying something. This is due to the reduced ability of motor neurons that resulted in some parts of the body vibrating, especially the hands. At present there is no medium used to carry food or goods that are stable to shocks. In this research it is proposed to create a balanced tray using the MPU6050 Inertial Measurement Unit (IMU) Sensor. Sensors are able to mendeteki position changes in 3 dimensions. The system uses the Arduino Nano microcontroller as a signal processor provided by the sensor. This microcontroller has input / output pin both digital and analog and Analog To Digital Convertion (ADC) in microcontroller is able to process sensor output. The Arduino Nano's physical shape has a small size that is portable and not too heavy in the stabilizer. The error value which is the difference between the point setting and the sensor output will then be used as input signal of the proportional integrator derivative (PID) controller. Servo motors are used as actuators that will move in accordance with the magnitude of the error, so that will result in stability of the tray motion. The result of system testing of success rate stabilizer when system without load is 100% for rest and 60% when the state runs. While in the state with the load obtained success rate of 70% at rest and 60% when the state runs. The results of this study are expected to reduce the risk of falling or spillage of goods or foods, especially when brought by people with Parkinson's.

Keywords : Controller Proportional, IMU Sensor, Stabilisator

-----Halaman ini sengaja dikosongkan-----

KATA PENGANTAR

Puji syukur penulis panjatkan kehadirat Allah SWT yang selalu memberikan rahmat dan hidayah-Nya sehingga penelitian ini dapat terselesaikan dengan baik. Penelitian ini disusun untuk memenuhi sebagian persyaratan guna menyelesaikan pendidikan Strata-1 pada Bidang Studi Elektronika, Departemen Teknik Elektro, Fakultas Teknologi Elektro, Institut Teknologi Sepuluh Nopember Surabaya dengan judul:

SISTEM STABILISASI NAMPAN MENGGUNAKAN IMU SENSOR DAN ARDUINO NANO

Penulis menyadari bahwa dalam pelaksanaan dan penyelesaian penelitian ini banyak mengalami kendala, namun berkat bantuan, bimbingan, dan kerjasama dari berbagai pihak semua kendala tersebut dapat diatasi. Oleh karena itu pada kesempatan ini penulis ingin menyampaikan banyak terimakasih, rasa hormat dan penghargaan setingi-tingginya kepada:

1. Bapak Dr. Muhammad Rivai, S.T., M.T. selaku dosen pembimbing mata kuliah Tugas Akhir.
2. Bapak Dr., Totok Mujiono, Ir., M.IKom ,Dr.Eng., Astria Nur Irfansyah, ST.,M.Eng, Dr., dan Muhammad Attamimi, B.Eng, M.Eng, PhD. selaku dosen penguji sidang Tugas Akhir.
3. Rekan-rekan Lintas Jalur angkatan 2016 atas momen kekeluargaan dan kerja sama yang luar biasa.

Penulis menyadari bahwa pada penyusunan laporan penelitian ini masih terdapat beberapa kekurangan dikarenakan keterbatasan kemampuan penulis, walaupun demikian penulis berharap penelitian ini dapat bermanfaat bagi khalayak dan pihak-pihak yang membutuhkan.

Surabaya, 25 Juni 2018

Penulis

-----Halaman ini sengaja dikosongkan-----

DAFTAR ISI

HALAMAN

DAFTAR GAMBAR

HALAMAN

DAFTAR TABEL

HALAMAN

-----Halaman ini sengaja dikosongkan-----

BAB I
PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi belakangan ini berkembang sangat cepat, banyak sekali terobosan-terobosan yang mendukung kehidupan manusia. Namum perkembangan tersebut belum sampai menjangkau perkembangan teknologi untuk penderita penyakit parkinson. Para penderita penyakit parkinson kerap kali mengalami kesulitan dalam menjaliani kehidupan sehari-hari, terutama pada saat mereka hendak membawa makanan yang akan mereka makan. Pada saat ini mereka hanya membawa makanan pada nampan yang membuat mereka mengalami kesulitan karena makanan kerap kali tumpah yang diakibatkan oleh bergetarnya tangan mereka. Hal ini dikarenakan pada penderita sindrom parkinson mengalami melemahnya kemampuan syaraf motorik yang mengakibatkan mereka kesulitan dalam mengontrol anggota tubuh [1] [2].

Penggunaan suatu sistem stabilisasi pada nampan akan memudahkan penderita parkinson dalam membawa makanan ataupun barang tanpa harus menjatuhkannya. Stabilisator ini menggunakan *Inertial Measurement Unit* (IMU) Sensor yang mampu mendeteksi perubahan posisi suatu benda. Pennggunaan metode kontrol *proportional integrator derivative* (PID) menjadikan stabilisator makin cepat dalam mencapai keadaan stabil.

Dalam menentukan nilai PID bergantung pada jenis plant yang digunakan, dimana plant merupakan perangkat keras yang dikontrol. Misal pada motor servo, perbedaan jenis motor servo juga mempengaruhi nilai PID untuk mendapatkan hasil akurasi yang terbaik dan meminimalkan galat. Salah satu kekuatan PID adalah untuk jenis plant sederhana ada korelasi langsung antara respon plant, penggunaan dan penyesuaian dari tiga istilah kontroler [3]. Salah satu cara yang sering digunakan untuk mendapatkan nilai PID terbaik adalah menggunakan metode *zieger Nichols* serta *trial and error.*

Dengan mengubah-ubah nilai PID pada *coding* dan dicoba pada hardware hingga mendapatkan hasil yang terbaik. Dari berbagai riset mengenai PID ini yang mempengaruhi nilai PID adalah jenis *plant.* Selain jenis *plan*t yang digunakan, mikrokontroler juga dapat

mempengaruhi nilai PID dengan mengesampingkan berbagai macam kebutuhan sistem terhadap mikrokontroler. Tiap mikrokontroler memiliki kecepatan proses, cara memproses *coding* dan juga pin-pin yang berbeda.

Pada penelitian ini akan dirancang dan diimplementasikan suatu sistem stabilisasi nampan dengan menggukanak arduino nano dan IMU Sensor. Untuk mempercepat dalam mencapai suatu kestabilan digunakan kontrol *Proportional-Integral-Derivative* (PID) yang dikontrol secara *close loop*. Perubahan sudut dari nampan akan terbaca oleh *IMU Sensor*. Dimana, nilai keluaran sensor akan dibandingkan dengan nilai *setpoint* dari sistem.

1.2 Permasalahan

Permasalahan pada penelitian ini adalah sebagai berikut:

1. Jenis sensor yang digunakan mendeteksi guncangan
2. Bagaimana menjaga kestabilan pada nampan meskipun terjadi guncangan
3. Proses pengkoreksian pada sinyal galat terjadi secara langsung
4. Pengimplementasian sistem kontrol dalam ukuran yang kecil

1.3 Batasan Masalah

Batasan masalah pada penelitian ini adalah sebagai berikut:

1. Berat maksimum beban tidak melebihi 0.5 Kilogram
2. Sensor menggunakan 2 axis (*yaw, roll*)
3. Subjek pengujian dikhususkan untuk kondisi penderita parkinson

1.4 Tujuan

Tujuan dari pembuatan penelitian ini adalah sebagai berikut:

1. Penggunaan IMU Sensor MPU6050 untuk mendeteksi guncangan nampan
2. Implementasi kontroler PID pada sistem stabilisasi nampan
3. Penggunaan motor servo sebagai aktuator sistem stabilisator
4. Sistem stabilisasi diimplementasikan dalam mikrokontroler Arduino nano

1.5 Metodologi Penelitian

Pembuatan sistem stabiliasasi nampan berbasis IMU Sensor terbagi menjadi lima tahapan. Yaitu studi literature, perancangan sistem, ujicoba serta analisa dan penyusunan laporan. Pada tahap studi literature dilakukan pencarian literature, jurnal atau buku mengenai komponen penyusun sistem terkait yakni sensor MPU6050, Mikrokontroler serta kontroler PID. Data-data yang dicari dari studi literature tersbut antara lain tentang keluaran sensor, cara mengolah sensor serta pemberian feedback sensor dan metode kontrol yang digunakan untuk mencapai kestabilan. Tahap berikutnya adalah perancangan sistem, pada tahap ini dilakukan perakitan setiap komponen yang digunakan menajdi satu kesatuan. Setalah semua menjadi satu, kemudian dirancang suatu program pada Arduino *IDE* yang akan mensinkronkan keseluruhan sistem agar mampu bekerja dengan baik. Tahap ujicoba dilakukan untuk mengetahui apakah keseluruhan sistem bekerja dengan baik ataupun tidak, ujicoba meliputi pengujian keluaran sensor, keluaran *actuator*, ujicoba kontroler PID dan ujicoba keseluruhan sistem dalam menopang beban. Tahap terakhir yang dilakukan adalah penyusunan laporan, pada tahap ini keseluruhan hasil yang didapat dari metode-metode sebelumnya akan dilaporkan pada suatu laporan ilmiah.

1.6 Sistematika Laporan

Pembahasan penelitian ini dibagi menjadi lima bab dengan sistematika sebagai berikut:

Bab I Pendahuluan

Pada bab pendahuluan, menjelaskan mengenai latar belakang pemilihan topik, perumusan masalah dan batasannya. Bab ini juga membahas mengenai tujuan penelitian, metodologi, sistematika laporan, dan relevansi dari penelitian yang dilakukan.

Bab II Teori Dasar

Menjelaskan teori yang berisi teori-teori penunjang yang dijadikan landasan prinsip dasar dan mendukung dalam perencanaan dan pembuatan alat yang dibuat.

Bab III Perancangan Sistem

Membahas perencanaan dan pembuatan tentang perencanaan dan pembuatan perangkat keras (*Hardware*) yang meliputi desain mekanik dan perangkat lunak (*software*) yang meliputi program yang akan digunakan untuk menjalankan alat tersebut.

Bab IV Hasil Implementasi Alat

Membahas pengujian alat dan menganalisa data yang didapat dari pengujian tersebut serta membahas tentang pengukuran, pengujian, dan penganalisaan terhadap alat.

Bab V Penutup

Berisi penutup yang menjelaskan tentang kesimpulan yang didapatdari tugas akhir ini dan saran-saran yang dapat diimplementasikan untuk pengembangan alat ini lebih lanjut.

1.7 Relevansi

Hasil yang diperoleh dari tugas akhir ini diharapkan menjadi referensi lanjutan untuk pengembangan sistem kestabilan dengan IMU Sensor.

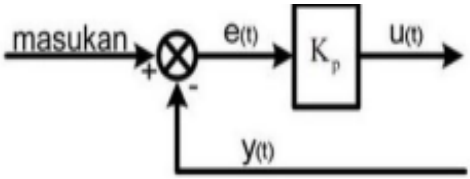
BAB II
TEORI PENUNJANG

Pada bab ini akan dibahas mengenai beberapa teori penunjang yang dipaparkan dalam laporan penelitian ini, diantaranya yaitu kontroler PID, IMU Sensor, *Arduino Uno* Dan *Buck Converter*

2.1 Kontroler PID

Kontroler PID (*Proporsional Integral Derivative*) merupakan suatu metode kontrol yang digunakan untuk mencapai sebuah kesetimbangan [3][4][5][6][7]. Penerapan kontroler PID juga digunakan untuk mengatur suhu suatu sistem [8], juga dapat digunakan untuk mengatur kecepatan motor pada suatu *quadcopter* [9]. Kontrol proporsional berfungsi untuk memperkuat sinyal kesalahan penggerak (sinyal *error*), sehingga akan mempercepat keluaran sistem mencapai titik referensi. Hubungan antara masukan kontroler u(t) dengan sinyal galat e(t), dapat dirumuskan pada persamaan (1). Blok diagram kontroler proporsional dapat kita lihat pada gambar 1 dibawah ini :

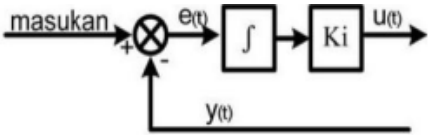
$$K_p \cdot e(t) \tag{2.1}$$



Gambar 2.1 Kontrol Proporsional [9]

Kontrol integral pada prinsipnya bertujuan untuk menghilangkan kesalahan keadaan tunak (*offset*) yang biasanya dihasilkan oleh kontrol proporsional. Hubungan antara keluaran kontrol *integral* u(t) dengan sinyal galat e(t) dapat dirumuskan pada persamaan (2). Blok diagram kontroler proporsional dapat kita lihat pada gambar 2 dibawah ini :

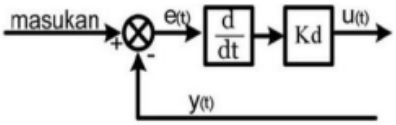
$$K_i \cdot \int_0^t e(t)\,dt \tag{2.2}$$



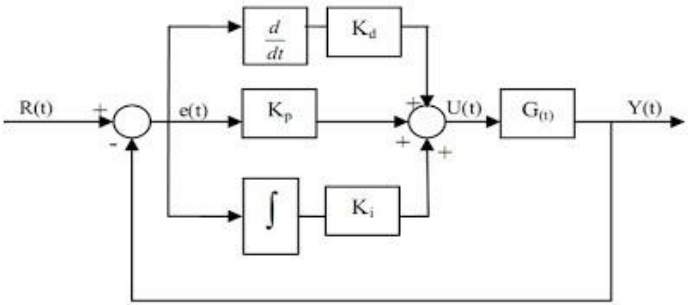
Gambar 2.2 Kontrol Integrator [9]

Kontrol *derivative* dapat disebut pengendali laju, karena keluaran kontroler sebanding dengan laju perubahan sinyal galat. Hubungan antara keluaran kontrol derivatif u(t) dengan sinyal error e(t) dapat dirumuskan pada persamaan (3). Kontrol derivatif tidak akan pernah digunakan sendirian, karena kontroler ini hanya akan aktif pada periode peralihan. Pada periode peralihan, kontrol derivatif menyebabkan adanya redaman pada sistem sehingga lebih memperkecil lonjakan. Seperti pada kontrol proporsional, kontrol derivatif juga tidak dapat menghilangkan *offset*. Blok diagram kontroler proporsional dapat kita lihat pada gambar 3 dibawah ini :

$$K_d \cdot \frac{\delta e(t)}{\delta t} \tag{2.3}$$



Gambar 2.3 Kontrol *Derivative* [9]

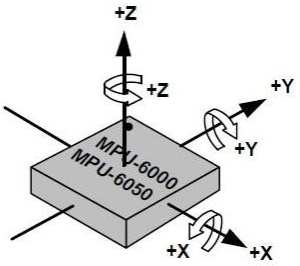


Gambar 2.4 Blok kontrol PID [9]

Gambar 4 merupakan keseluruhan blok diagram dari sistem kontrol PID, berikut merupakan rumus matematis dari kontroler PID .

$$u(t) = K_p\left(e(t) + \frac{1}{T_i}\int_0^t e(t)dt + T_d\frac{de(t)}{dt}\right) \quad (2.4)$$

2.2 IMU Sensor



Gambar 2.5 Komponen Penyusun IMU [10]

Inertial Measurement Unit (IMU) Merupakan suatu sensor yang digunakan untuk mengukur kecepatan, orientasi dan gaya gravitasi dengan menggunakan sensor *accelerometer* dan *gyroscope*. IMU seringkali digunakan dalam suatu sistem pesawat terbang. Komponen penyusun IMU yang pertama adalah sensor *accelerometer*, sensor ini digunakan untuk mengukur percepatan dari suatu benda dengan cara melakukan integral percepatan benda tersebut terhadap waktu. Komponen selanjutnya yang menyusun IMU Sensor adalah sensor *gyro*, cara kerja sensor ini mendeteksi gerakan sesuai gravitasi, atau dengan kata lain mendeteksi gerakan pengguna. *Gyroscope* memiliki keluaran berupa kecepatan sudut dari arah 3 sumbu yaitu: sumbu x / sudut *phi* (kanan dan kiri) dari sumbu y/sudut *theta* (atas dan bawah), dan sumbu z /sudut psi (depan dan belakang) [10] seperti yang ditunjukkan pada gambar 5. Sensor MPU 6050 merupakan salah satu sensor yang menggunakan prinsip dasar IMU Sensor. Pengaturan register pada MPU-6050 dilakukan oleh mikrokontroler dengan komunikasi I2C . Perintah yang digunakan adalah perintah tulis. Dalam pengaturan register MPU-6050 perlu diperhatikan alamat dari MPU-6050 tersebut dan alamat register yang akan diatur. Alamat *default* dari MPU-6050 adalah 0x68, alamat ini dijadikan alamat untuk MPU-6050 pertama sebagai pemberi nilai *set point*. Sedangkan alamat dari MPU-6050 kedua sebagai pemberi nilai

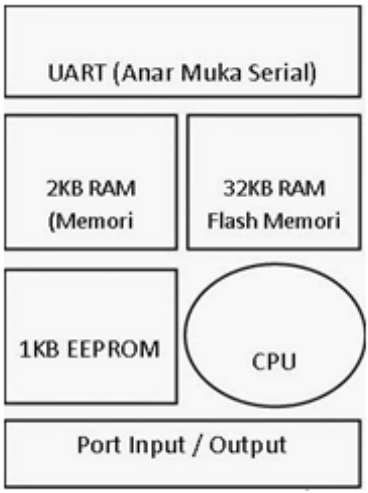
feedback adalah 0x69. Alamat pada MPU-6050 dapat diubah menjadi 0x69 dengan cara memberi tegangan 3.3V pada pin AD0. Pembacaan sensor gyroscope pada MPU-6050 disimpan pada register data. Perangkat luar dapat meminta data tersebut dengan menunjuk alamat dari *register* data tersebut. Data *gyroscope* memiliki lebar data 16-bit yang terdiri dari 8-bit low byte dan 8-bit *high byte*. Karena masing-masing sumbu pada gyroscope register datanya terbagi menjadi dua maka untuk mendapatkan data yang *valid* dari masing-masing sumbu harus dilakukan penggabungan data dari dua buah *register* tersebut. Ukuran variabel yang disediakan pada program mikrokontroler harus 16-bit bertanda, karena data yang dibaca dapat bernilai positif maupun negatif. Proses penggabungan data dilakukan dengan cara menggeser data *high byte* ke kiri sebesar 8-bit kemudian dijumlahkan dengan data *low byte*.

Pada sensor MPU-6050 dapat menggunakan dua buah macam *type* clock, yakni *internal* dan eksternal clock. Pemilihan sumber clock ini tergantung dari mode operasi sistem MPU-6050 yang digunakan. Internal *oscillator* baik digunakan ketika menggunakan DMP (*Digital Motion Processor*) untuk pengolahan data *accelerometer* dan data *gyroscope* dimatikan. Sedangkan bila *gyroscope* aktif dianjurkan menggunakan pemilihan sumber clock dari *gyroscope* untuk keakuratan. Pemilihan sumber clock untuk MPU-6050 dilakukan dengan mengatur CLKSEL (bit [2:0]) dalam *register* PWR_MGMT_1. Alamat *register* PWR_MGMT_1 pada MPU-6050 adalah 0x6B. Dalam sistem ini, sumber clock yang digunakan adalah sumber clock *internal* yang bernilai 8MHz, maka dari itu nilai yang diberikan pada register adalah 0x00. Pengaturan yang dilakukan berkaitan dengan keluaran nilai *gyroscope* dapat diatur pada register *GYRO*_CONFIG dengan alamat register 0x1B. Opsi FS_SEL difungsikan untuk memilih skala penuh yang digunakan pada *gyroscope*. ZG_ST, YG_ST, dan XG_ST digunakan untuk melakukan tes performansi dari masing masing sumbu *gyroscope*. Dalam sistem yang dibuat pengaturan register hanya dilakukan pada FS_SEL [11][12]. Terdapat empat pilihan skala penuh pada gyroscope. Skala penuh tersebut dapat dipilih dengan memberikan nilai bit ke-0 dan bit ke-1 dari FS_SEL. Pada sistem yang dibuat, digunakan skala penuh ±250o/s. Oleh karena itu, register *GYRO_CONFIG* diberikan nilai 0x00. Pada penelitian ini, sensor MPU6050 dipakai karena dirasa mampu dalam membaca perubahan sudut terhadap sumbu X dan Y (*yaw*, *pitch*) dan memberikan keluaran data yang cepat karena sudah menggunaakn komunikasi I2C. I2C merupakan suatu *protocol* komunikasi yang menggunakan dua jalur yakni

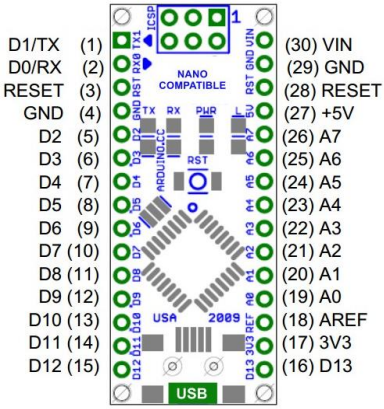
SDA dan SCL, jalur SCL merupakan jalur untuk *clock* sedangakan jalur SDA digunakan untuk data. Jenis komunikasi yang digunakan untuk I2C mempunyai sifat serial *synchronous* half duplex *bidirectional* dimana data yang diterima hanya menggunakan satu jalur SDA saja [13].

2.3 Arduino NANO

Arduino Nano merupakan suatu papan pengembang mikrokontroler yang menggunakan chip ATmega328P, Arduino Nano bekerja pada masukan tegangan 5-7 Volt. Terdapat memori *flash* sebesar 32 KB dan mampu bekerja pada clock 16 Mhz seperti yang ditunjukkan pada gambar 6. Arduino nano dapat diprogram dengan menggunakan Arduino *Integrated Development Environment* (IDE) dan dihubungkan dengan kabel USB type B. Pada arduino nano terdapat 14 buah pin masukan dan keluaran, dimana 6 buah pin diantranya dapat digunakan untuk keluaran pulse width modulation (PWM). Terdapat 8 buah pin analog yakni A1, A2, A3, A4, A5, A6, A7 dan A8, keseluruhan pin analog ini terhubung dengan *Analog to Digital Converter* (ADC) pada internal mikrokontoler. Pada arduino nano juga tersedia dua buah pin SDA dan SCL (masing-masing pada A4 dan A5) yang dapat digunakan untuk komunikasi I2C seperti yang ditunjukkan pada gambar 2.7 [14].



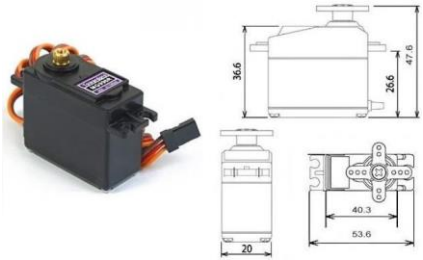
Gambar 2.6 Blok Diagram Arduino Nano [15]



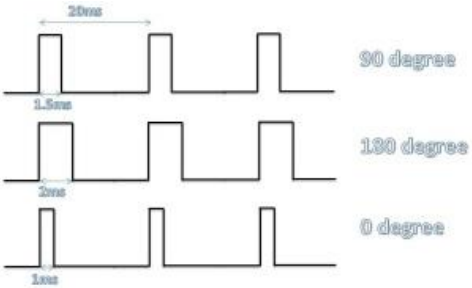
Gambar 2.7 *Pinout* Arduino Nano [15]

2.4 Motor Servo

Motor servo merupakan motor yang bekerja berdasarkan cara kerja closed loop sehingga dapat diatur seberapa besar sudut putaran motor servo. Pengaturan sudut motor servo dapat diatur dengan menggunakan masukan *Pulse Witdh Modulation* (PWM). Besarnya torsi yang digunakan pada tipe motor servo SG90 adalah 9.40 Kg-cm, dengan torsi sebesar itu kiranya sudah cukup digunakan untuk memutar poros motor servo. Untuk mengendalikan motor servo berbeda dengan motor DC biasa. Berikut pada gambar 8 merupakan dimensi dari motor servo SG90.



Gambar 2.8 Motor Servo

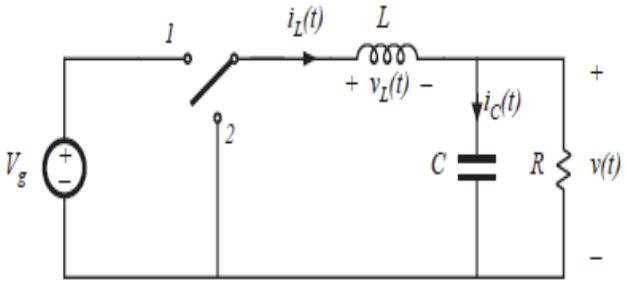


Gambar 2.9 Bentuk Pulsa Kendali Motor Servo

Untuk dapat mengendalikan motor servo perlu adanya PWM (*Pulse Width Modulation*) serperti yang ditunjukkan pada gambar 9. Besarnya tegangan yang digunakan sebagai masukan motor servo bisa bervariasi, besarnya antara 1ms-2ms. 1ms tegangan pulsa yang diberikan akan menghasilkan perputaran pada servo sebesar 0 derajat. Kebanyakan dengan memberikan pulsa PWM dengan lebar 2ms akan menghasilkan putaran sebesar 180 derajat.

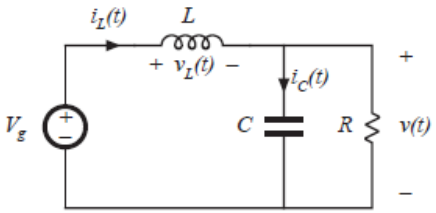
2.5 Buck Converter

Pada rangkaian elektronika, terdapat berbagai rate tegnangan yang umumnya digunakan tergantung dari keperluan suatu kompoenen yang digunakan. Sehingga diperlukan suatu rangakaian yang dapat mengubah level tegangan tersebut, *Buck converter* merupakan suatu rangkaian penurun level tegangan yang bekerja secara terus-menerus (ON-OFF) atau yang biasa disebut dengan PWM (*Pulse Width Modulation*) dan juga *duty cycle* dalam mengendalikan kecepatan (frekuensi) kerja dari switch buck converter tersebut.



Gambar 2.10 Rangkaian *Buck Converter*

Pada gambar diatas dapat kita lihat bahwa pada rangkaian buck terdiri dari inductor, kapasitor serta *switch*. *Switch* berada pada dua posisi berbeda. Berikut merupakan gambar ketika *switch* berada posisi 1.



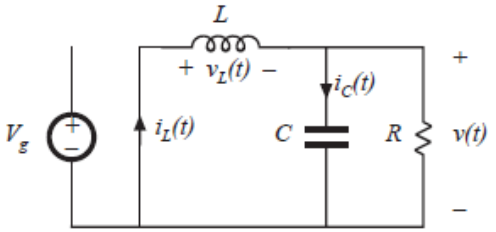
Gambar 2.11 *Switch* pada posisi 1

Pada saat posisi ini inductor *closed* dan menyerap keseluruhan daya, sehingga kapasitor akan *open*. Tegangan pada inductor dapat dirumuskan sebagai berikut :

$$v_L = V_g - v(t) \tag{2.5}$$

Perubahan arus pada imduktor dapat kita rumuskan :

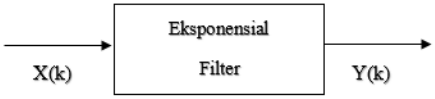
$$\frac{di_L(t)}{dt} = \frac{v_L(t)}{L} \approx \frac{V_g - V}{L} \tag{2.6}$$



Gambar 2.12 Switch berada pada posisi 2

Pada saatkondisi ini catu daya menjadi *open*, dan rangkaian inductor akan mensuplai tegangan yang menuju *vout*.

2.6 Eksponensial Filter



Gambar 2.13 Diagram Blok Eksponensial Filter

Filter eksponensial merupakan suatu filter *low Frequency*, yakni filter ini akan meloloskan frekuensi rendah dan akan menghilangkan frekuensi tinggi. Filter ini akan mengumpan balikkan keluaran dan mengkaliakan dengan suatu konstanta *smoothing* tertentu untuk memperoleh hasil yang lebih bagus. Diagram blok dari filter eksponensial dapat kita lihat pada gambar 2.13

$$Y(k) = a * Y(k-1) + (1-a) * x(k) \tag{2.7}$$

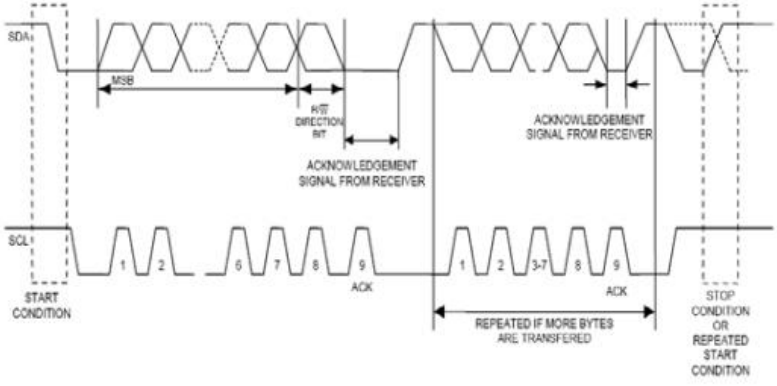
Keterangan :
Y(k) : keluaran yang terfilter pada saat k
X(k) : masukan filter pada saat k
K : konstanta waktu

2.7 Komunikasi I2C

I2C singkatan dari *Inter Integrated Circuit*, adalah sebuah protokol untuk komunikasi serial antar IC, dan sering disebut juga *Two Wire Interface* (TWI). Bus yang digunakan untuk komunikasi antara mikrokontroler dan divais periferal seperti memori, sensor temperatur dan I/O *expander*.

Komunikasi dilakukan melalui dua jalur: SDA (*serial data*) dan SCL (*serial clock*). Setiap divais I2C memiliki 7-bit alamat yang unik. MSB adalah fix dan ditujukan untuk kategori divais. Sebagai contoh, 1010 biner ditujukan untuk serial EEPROM. Tiga bit berikutnya memungkinkan 8 kombinasi alamat I2C, yang berarti, dimungkinkan 8 divais dengan tipe yang sama, beroperasi pada bus I2C yang sama. Pengiriman data hanya dapat dimulai ketika saluran tidak sibuk, ditdai dengan kondisi *HIGH* yang cukup lama pada pin SCL maupun SDA.Selama pengiriman data, saluran data (SDA) harus dalam keadaan stabil ketika saluran clock (SCL) dalam keadaan high. Perubahan kondisi

SDA pada saat SCL high akan dianggap sebagai sinyal-sinyal kendali, seperti: sinyal *START* (*HIGH* ke *LOW*) atau sinyal *STOP* (*LOW* ke *HIGH*).



Gambar 2.14 Prinsip Komunikasi *Serial Bus* I2C

Berikut ini adalah defenisi kondisi bus pada sistem komunikasi serial I2C :

1. Bus tidak sibuk (*bus not busy*): menyatakan pada saat ini bus tidak sibuk yaitu pada saat jalur clock (SCL) dan jalur data (SDA) dua-duanya dalam keadaan *HIGH*.
2. Mulai *transfer* data (*start data transfer*): ditandai dengan perubahan kondisi SDA dari *HIGH* ke *LOW* ketika SCL dalam kondisi *HIGH*.
3. *Stop transfer* data (*stop data transfer*): ditandai dengan perubahan kondisi SDA dari *LOW* ke *HIGH* ketika SCL dalam kondisi *HIGH*.
4. Data *valid*: data yang dikirim bit demi bit dianggap valid jika setelah *START*, kondisi SDA tidak berubah selama SCL *HIGH*, baik SDA *HIGH* maupun SDA *LOW* tergantung dari bit yang ingin ditransfer. Setiap siklus *HIGH* SCL baru menandakan pengiriman bit baru. Duty cycle untuk SCL tidak mesti 50%, tetapi frekuensi kemunculannya hanya ada dua macam, yaitu mode standar 100 kHz dan fast mode atau mode cepat 400 kHz. Setelah SCL mengirimkan sinyal *HIGH* yang kedelapan, arah transfer SDA berubah, sinyal kesembilan pada SDA ini dianggap sebagai *acknowledge* dari *receiver* ke *transmitter*. DS1307 hanya bisa melakukan transfer pada mode standar 100 kHz.
5. Pemberitahuan (*Acknowledge*): setiap *receiver* wajib mengirimkan sinyal *acknowledge* atau sinyal balasan setiap selesai pengiriman 1-

byte (8-bit data). *Master* harus memberikan ekstra clock atau clock tambahan pada SCL, yaitu clock kesembilan untuk memberikan kesempatan receiver mengirimkan sinyal *acknowledge* ke transmitter berupa keadaan *LOW* pada SDA selama SCL *HIGH*. Meskipun *master* berperan sebagai *receiver*, ia tetap sebagai penentu sinyal STOP. Pada bit akhir penerimaan byte terakhir, *master* tidak mengirimkan sinyal acknowledge, SDA dibiarkan *HIGH* oleh *receiver* dalam hal ini *master*, kemudian *master* mengubah SDA dari *LOW* menjadi *HIGH* yang berarti sinyal *STOP*.

2.7.1 Mode Pengoperasian Transfer Data

Mode pengoperasian transfer data berdasarkan kondisi bit R/W, ada dua jenis transfer data yaitu: transfer data dari transmitter *master* ke receiver *slave* dan transfer data dari transmitter slave ke receiver *master*.

A. Transfer Data dari *Transmitter Master* ke *Receiver slave* (*WRITE Mode*).

Byte pertama yang dikirimkan oleh *master* adalah alamat *slave*, setelah itu *master* mengirimkan sejumlah byte data. *slave* atau *receiver* mengirimkan sinyal *acknowledge* setiap kali menerima 1-byte data. Pada tiap byte, bit pertama yang dikirim adalah MSB dan bit yang terakhir adalah LSB. Berikut merupakan aturan dalam mode *write mode :*

1. Setelah sinyal START, *master* mengirim byte pertama yang terdiri dari 7-bit *address* IC MPU6050 dan 1-bit R/W, yaitu *LOW*, karena ini adalah opersai *WRITE*.
2. *Hardware* pada MPU6050 akan membaca *address* yang dikirimkan oleh *master* tersebut, kemudian *slave*, dalam hal ini IC MPU6050 akan *bit-acknowledge* pada SDA.
3. Setelah itu *master* akan mengirimkan *address* tempat data pertama akan diakses. *Address* ini berbeda dengan 7-bit a*ddress* tadi, ini adalah *address* "isi" IC MPU6050, bukan *address* dari IC MPU6050. *Address* ini akan diimpan dalam *register pointer* oleh MPU6050 yang juga mengirim sinyal *acknowledge* ke *master*.
4. Setelah itu *master* dapat mengirimkan sejumlah *byte* ke *slave*, dimana setiap *byte* dibalas dengan *acknowledge* oleh *slave*. Setiap menerima *byte* baru isi *register pointer* ditambah satu sehingga *register* ini menunjuk ke alamat berikutnya dari lokasi data pada MPU6050. Setelah menerima *acknowledge* terakhir, *master* akan mengirim sinyal *STOP* untuk mengakhiri transfer data

B. Transfer Data dari Transmitter *slave* ke Receiver *Master (READ Mode)*

Meskipun *master* berperan sebagai *receiver*, *byte* pertama dikirimkan oleh *master* berupa alamat *slave*. Setelah itu *slave* meengirimkan bit acknowledge, dilanjutkan dengan pengiriman sejumlah byte dari slave ke *master*. *Master* mengirimkan bit *acknowledge* untuk setiap *byte* yang diterimanya kecuali *byte* terakhir. Pada akhir *byte*, *master* mengirimkan sinyal '*not avknowledge'*, setelah itu *master* mengirimkan sinyal *STOP*. Berikut merupakan aturan dalam *write mode :*

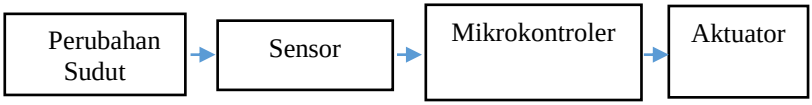
1. Setelah sinyal START, *slave* mengirim byte pertama yang terdiri dari 7-bit *address* IC MPU6050 dan 1-bit R/W, yaitu *High*, karena ini adalah opersai *READ*.
2. *Hardware* pada MPU6050 akan membaca *address* yang dikirimkan oleh *slave* tersebut, kemudian *master*, dalam hal ini mikrokontroler akan *bit-acknowledge* pada SDA.
3. Setelah itu *slave* akan mengirimkan *address* tempat data pertama akan diakses. *Address* ini berbeda dengan 7-bit a*ddress* tadi, ini adalah *address* "isi"
4. Setelah itu *slave* dapat mengirimkan sejumlah *byte* ke *master*, dimana setiap *byte* dibalas dengan *acknowledge* oleh *slave*. Setiap menerima *byte* baru isi *register pointer* ditambah satu sehingga *register* ini menunjuk ke alamat berikutnya dari lokasi data pada MPU6050. Setelah menerima *acknowledge* terakhir, *master* akan mengirim sinyal *STOP* untuk mengakhiri transfer data

BAB III
PERANCANGAN ALAT

Pada bab ini akan dibahas mengenai perancangan perangkat keras dan lunak yang dilakukan dengan metode penelitian berdasarkan pada studi kepustakaan berupa data-data literatur dari masing-masing komponen, informasi dari *internet*, dan konsep-konsep teoritis dari buku-buku penunjang.Perancangan diperlukan agar dalam tahapan selanjutnya berjalan dengan lancar, pada awalnya dilakukan perancangan perangkat keras. Setelah itu akan diuji dengan menggunakan perangkat lunak untuk memastikan keduanya dapat berjalan dengan baik. Pada tahap ini diawali dengan penjelasan diagram fungsional sistem yang bertujuan untuk menjelaskan keseluruhan kinerja sistem. Perancangan hardware memberikan informasi tentang desain dan perancangan hardware yang digunakan dalam sistem stabilisator. Perancangan Elektrik menjelaskan sistem dari sisi elektrik diamana komponen yang digunakan dirancang dan disusun secara elektrik. Perancangan perangkat lunak (program) menjelaskan tentang pemrograman mikrokontroler pada arduino *IDE* untuk pembacaan nilai sensor, pengolahan data, pemberian nilai kontrol serta tentang perubahan servo.

3.1 Blok Fungsional Sistem

Blok Diagram pada gambar 3.1 merupakan representasi dari stabilisator nampan menggunakan IMU Sensor berdasarkan guncangan pada penelitian ini. Perencanaan sistem ini bertujuan agar sistem stabilisator dapat mengkoreksi sinyal galat yang merupakan perubahan sudut yang dideteksi oleh IMU Sensor. Sistem bekerja berdasarkan perubahan sudut yang terjadi secara signifikan.



Gambar 3.1 Diagram Blok Fungsional Sistem

Pada alat ini terdapat beberapa bagian seperti sensor, mikrokontroler serta *actuator*. Pada bagian sensor terdapat IMU Sensor dengan tipe MPU6050 yang akan mendeteksi perubahan sudut akibat adanya guncangan yang terjadi pada sistem. Perubahan sudut yang dapat mampu dideteksi oleh sensor ialah pada sudut *yaw*, *pitch* dan *roll.* Dari data

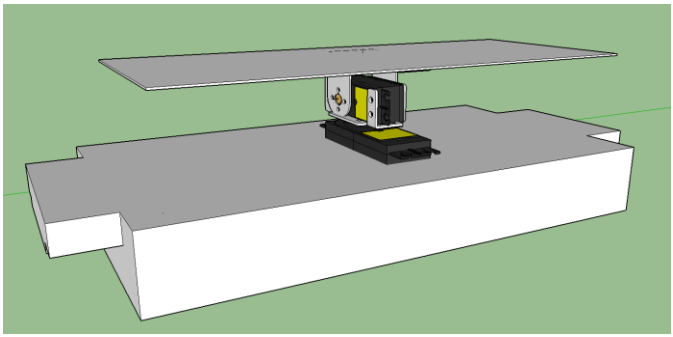
perubahan tersebut data yang diambil hanya pada sudut *yaw* dan *pitch* saja dikarenakan stabilisator hanya menggunakan dua *axis* saja. Sebelum ditambah sinyal kontrol, data dari MPU6050 akan difilter menggunakan exponential filter. *Exponential* filter merupakan filter frekuensi rendah yang menepis frekuensi tinggi pada suatu sinyal. Proses pemberian filter dilakukan oleh mikrokontroler, pada penelitian ini menggunakan mikrokontroler arduino nano. Kemudian *RAW* Data (data mentah) hasil pembacaan MPU6050 akan diproses oleh mikrokontroler untuk dijadikan sudut, pada mikrokontroler juga bertugas untuk memberikan variabel kontrol pada perubahan sudut tersebut agar sinyal kontrol yang diteruskan aktuator. Pada aktuator terdiri dari dua buah motor servo yang masing-masing mewakili sudut *yaw* dan sudut *pitch*, masukan untuk aktuator merupakan sudut koreksi. Aktuator akan memberikan sinyal koreksi secara langsung yang diakibatkan oleh perubahan sudut, sinyal koreksi akan ditambah dengan sinyal kontrol agar mampu mencapai keadaan setimbang lebih cepat.

3.2 Perancangan Perangkat Mekanik

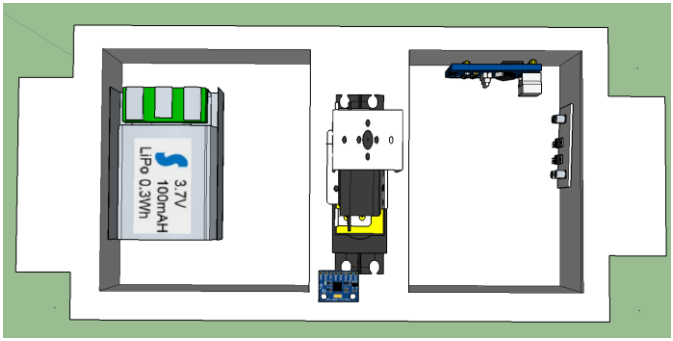
Pada sub bab ini akan membahas tentang perancangan mekanik untuk penelitian ini. Perancangan mekanik berupa perancangan perangkat keras yang berguna untuk mendukung seluruh perancangan dan pembuatan alat. Perancangan mekanik yang akan dibahas yakni mengenai perancangan *base* dan perancangan *bracket* servo.

3.2.1 Perancangan Base Stabilisator

Base yang digunakan pada stabilisator dicetak dengan menggunakan metode 3D Printing, dengan menggunakan bahan plastik ABS (*Acrylonitrile Butadiene Styrene*) yang mempunyai karakteristik kuat dan tidak gampang pecah akan membuat base yang digunakan menjadi kokoh dalam menopang keseluruhan beban pada stabilisator berikut merupakan desain base dapat kita lihat pada gambar 3.2 dan 3.3



Gambar 3.2 Desain *Base* Stabilisator Tampak Samping

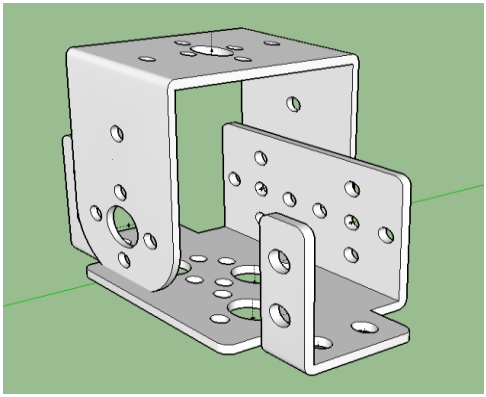


Gambar 3.3 Desain *Base* Stabilisator Tampak Atas

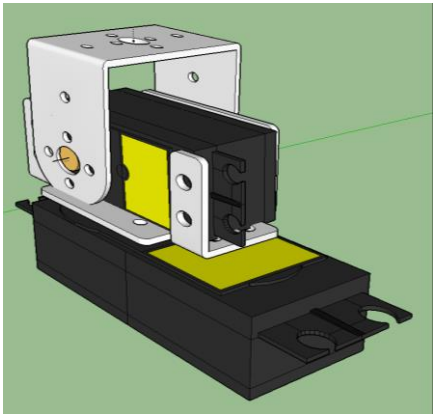
Pada gambar 3.3 dapat kita lihat bahwa terdapat 2 buah ruangan yang berguna dalam meletakkan komponen pendukung. Disamping kiri terdapat ruangan untuk baterai *Li-Po* dan disebelah kanan terdapat ruangan untuk mikrokontroler dan rangakain *buck converter*. Untuk sensor diletakkan sejajar dengan *base*. Pada bagian terluar sebelah kanan dan kiri stabilisator terdapat pegangan yang memudahkan pengguna dalam menggunakan stabilisator. Dua buah motor servo yang berfungsi sebagai *actuator* diletakkan pada tengah-tengah stabilisator agar mendapat jangkauan yang presisi dalam memberikan *feedback*

3.2.1 Perancangan Bracket Servo

Bracket servo digunakan untuk meletakkan servo sehingga servo akan mudah dalam memberikan *feedback*, bracket disusun sedemikian rupa sehingga menghasilkan keluaran servo secara *yaw* dan *pitch*. Bracket dibuat dari besi sehingga akan sangat kuat dalam menopang beban, servo pada bagian bawah akan memberikan *feedback* untuk sumbu *yaw* dan servo pada bagian atas untuk sumbu *pitch*. Berikut merupakan desain *bracket* yang digunakan pada gambar 3.4



Gambar 3.4 *Bracket* Tanpa Servo

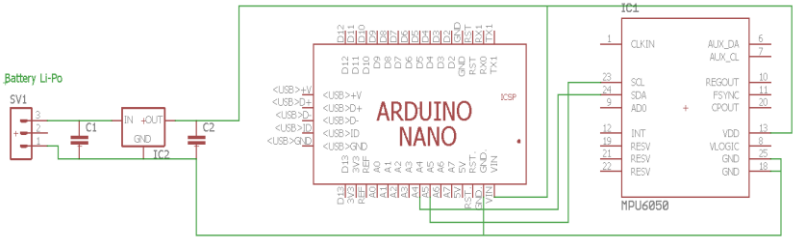


Gambar 3.5 *Bracket* Dengan Servo

3.3 Perancangan Perangkat Elektrik

Pada subbab ini akan dibahas mengenai perancangan elektrik untuk penelitian ini. Perancangan elektrik merupakan perancangan beberapa perangkat keras yang menunjang seluruh perancangan dan pembuatan alat. Perancangan elektrik yang akan dibahas mengenai perancangan sensor MPU6050, rangkaain *power supply*, serta perancangan motor servo untuk aktuator.

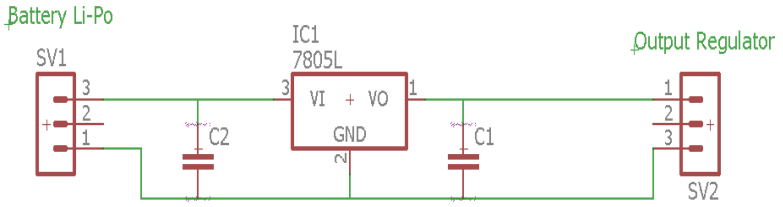
3.3.1 Perancangan Rangkaian IMU Sensor



Gambar 3.6 Rangkaian IMU Sensor Pada Arduino Nano

Pada gambar 3.6 menunjukkan komponen yang digunakan dalam mendeteksi perubahan sudut pada stabilisator, sensor MPU6050 terkoneksi dengan Arduino nano dengan cara menyambungkan pin SDA ke A4 dan SCL ke A5 pada Arduino. Tegangan masukan berasal dari rangkaian *power supply* yang terintegrasi dengan baterai *Li-Po*.

3.3.2 Rangkaian *Power Supply*

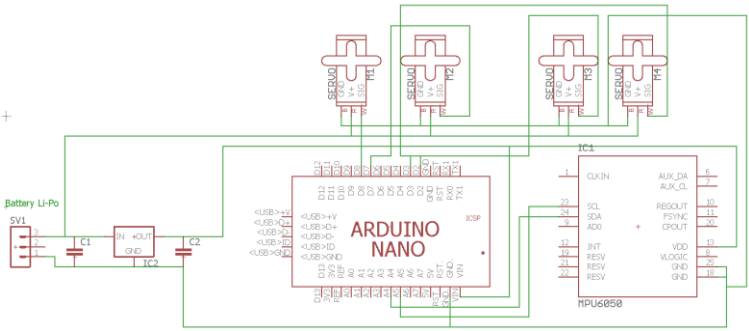


Gambar 3.7 Rangkaian Power Supply

Pada gambar 3.7 menunjukkan komponen yang digunakan dalam mendeteksi perubahan sudut pada stabilisator, sensor MPU6050

terkoneksi dengan Arduino nano dengan cara menyambungkan pin SDA ke A4 dan SCL ke A5 pada Arduino. Tegangan masukan berasal dari rangkaian power supply yang terintegrasi dengan baterai *Li-Po*.

3.3.3 Perancangan Rangkaian Aktuator



Gambar 3.8 Desain Aktuator

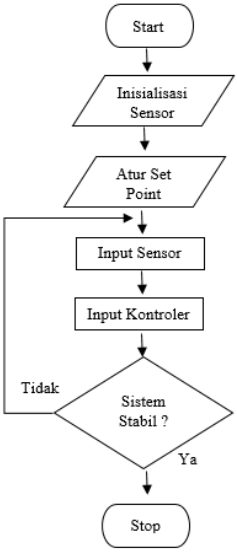
Pada gambar 3.8 menunjukkan komponen keseluruhan termasuk aktuator, pada aktuator digunakan 4 buah motor servo. Masing-masing pin V+ terhubung ke Vout Batterai *Li-Po* serta pin GND terhubung ke pin GND baterai *Li-Po*. Sedangkan untuk pin sinyal motor servo terhubung ke pin D8, D7, D3 dan D2 pada pin Arduino nano.

3.7 Perancangan dan Pembuatan Perangkat Lunak

Perancangan perangkat lunak pada sistem ini yaitu mengontrol permukaan nampan agar tetap stabil ketika mendapat guncangan Perangkat lunak yang digunakan yaitu Arduino IDE. Perangkat lunak diprogram agar mikrokontroler Arduino Nano mampu menghilangkan sinyal galat hasil perubahan sudut pada sensor MPU6050. Sebelum membuat program menggunakan Arduino Nano terlebih dahulu menginstal Arduino Nano pada Arduino IDE, dan memastikan *library* yang dibutuhkan telah terinstall pada Arduino, seperti I2C .

3.7.1 Flowchart

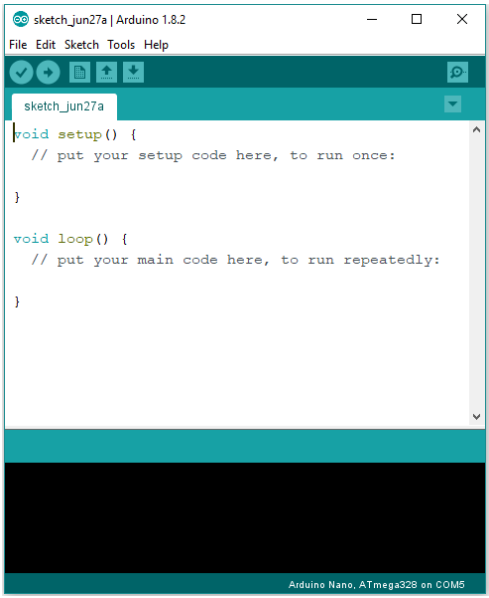
Perangkat lunak (software) dirancang dengan pembuatan *source code* dari integrasi seluruh sistem untuk mengoperasikan stabilisator yang bergerak berdasarkan guncangan, berikut merupakan flowchart sistem :



Gambar 3.9 *Flowchart* Program

Pada gambar 3.9 dapat kita lihat diagram alur *flowchart* pada penelitian ini, dimulai dengan inisialisasi sensor yang digunakan pada penelitian ini yakni IMU Sensor dengan tipe MPU6050. Setelah itu diberikan suatu nilai setpoint sebagai acuan dari pembacaan nilai sensor,

dengan menggunakan suatu nilai variable kontrol maka akan tetap menjaga nampan pada sistem ini tetap stabil. Program pada penelitian ini menggunakan software arduino versi 1.82. gambar 3.13 merupakan tampilan dari program yang digunakan.



Gambar 3.10 Tampilan *Software* Arduino versi 1.82

Berikut merupakan beberapa ikon yang sering digunakan pada penelitian ini yaitu:

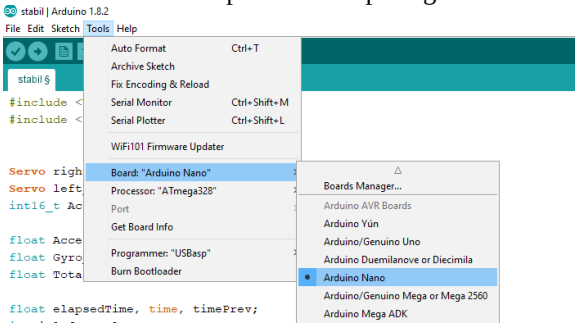
1.  Merupakan *icon* ***create new project***. Berguna untuk membuat project baru
2.  Adalah *icon* menu ***Verify*** berguna untuk memeriksa kesalahan pada program yang telah dibuat
3.  adalah icon Upload, berguna untuk mengirimkan data yang telah diverifikasi menuju ke arduino
4.  adalah icon save, yang berguna untuk menyimpan hasil dari pekerjaan yang telah kita buat

5.  adalah icon serial monitor, yang berguna untuk melihat hasil dari pekerjaan kita. Biasanya digunakan untuk memantau hasil dari keluaran sensor.

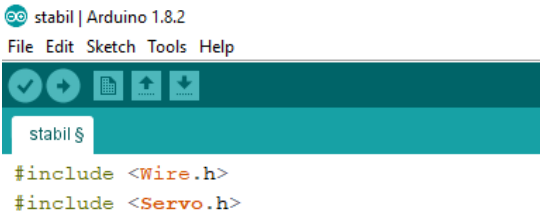
3.7.2 Perancangan Program Arduino

Pada sistem stabilisator nampan menggunakan arduino nano yang harus di *install* terlebih dahulu pada *Board Manager* Arduino lalu inisialisasi tiap pin, serta konfigurasi I2C. Berikut ini adalah langkah-langkah perancangan program pada sistem stabilisator.

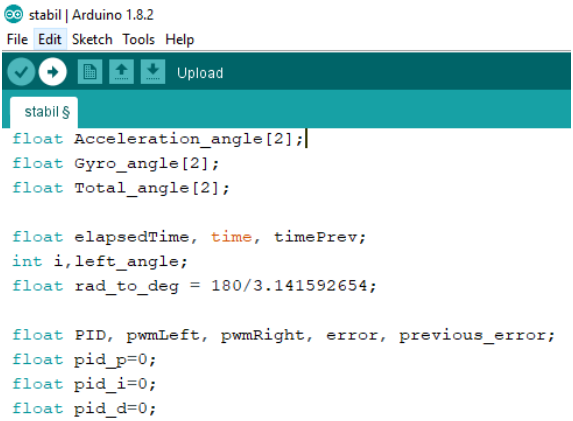
1. Langkah pertama adalah menginstall Arduino Nano pada *software* Arduino IDE dengan cara memilih *tools* - *boards* - *Boards Manager* seperti yang ditunjukkan pada 3.11
2. Setelah Arduino Nano berhasil ditambahkan pada arduino IDE, maka langkah berikutnya adalah menambahkan *library* terkait dengan program yang akan dirancang seperti yang ditunjukkan pada gambar 3.12
3. Langkah berikutnya adalah pembuatan program untuk stabilisator, namun sebelumnya haruslah diatur setuap *variabel* terlebih dahulu. Untuk inisialisasi variabel dapat kita lihat pada gambar 3.13



Gambar 3.11 *Install* Arduino Nano



Gambar 3.12 Pengaturan *Library*



Gambar 3.13 Inisialisasi Variabel

4. Setelah semua variabel terpenuhi, langkah berikutnya adalah menambahkan program pada *void loop*, pada *void loop* ini program akan terus menerus dijalankan seperti layaknya pada diagram alur flowchart yang telah dijelaskan diawal

```
Wire.begin();
Wire.beginTransmission(MPU_addr);
Wire.write(0x6B);
Wire.write(0);
Wire.endTransmission(true);
time = millis();
```

Gambar 3.14 Definisi I2C

Wire begin yaitu perintah untuk memulai I2C. Kemudian wire begintransmission yaitu perintah untuk mengakses MPU address. Kemudian Wire.write(0x6B) digunakan untuk memulai power management dari MPU6050.Kemudian perintah Wire.write(0) digunakan untuk menghidupkan sensor MPU-6050 dan Wire.*endTransmission*(true); yaitu memulai mengambil data pembacaan data *accelerometer*. Hal ini diulang juga untuk data *Gyro*.

```
AcX=Wire.read()<<8|Wire.read();  // 0x3B (ACCEL_XOUT_H) & 0x3C (ACCEL_XOUT_L)
AcY=Wire.read()<<8|Wire.read();  // 0x3D (ACCEL_YOUT_H) & 0x3E (ACCEL_YOUT_L)
AcZ=Wire.read()<<8|Wire.read();  // 0x3F (ACCEL_ZOUT_H) & 0x40 (ACCEL_ZOUT_L)
Tmp=Wire.read()<<8|Wire.read();  // 0x41 (TEMP_OUT_H) & 0x42 (TEMP_OUT_L)
GyX=Wire.read()<<8|Wire.read();  // 0x43 (GYRO_XOUT_H) & 0x44 (GYRO_XOUT_L)
GyY=Wire.read()<<8|Wire.read();  // 0x45 (GYRO_YOUT_H) & 0x46 (GYRO_YOUT_L)
GyZ=Wire.read()<<8|Wire.read();  // 0x47 (GYRO_ZOUT_H) & 0x48 (GYRO_ZOUT_L)
```

Gambar 3.15 pengambilan data MPU6050

Pada gambar 3.15 diatas menunjukkan bahwa ada 3 buah data yang diambil dari sensor yakni *accelerometer*, *gyrometer* dan suhu. Tiap data telah ditempatkan pada variabel-variabel yang telah ditentukan. Data yang diperoleh masih dalam bentuk RAW sehingga untuk mendapatkan data sudut dalam derajat harus dikonversikan lagi. Berikut merupakan cara untuk mengubahnya pada gambar 3.16.

```
float pitch = atan2(AcY,  AcZ)*57.2958;
float roll = atan2(AcX,  AcZ)*57.2958;
float yawds = (float)GyZ/65.536;
```

Gambar 3.16 Konversi Data

Data yang dihasilkan harus difilter lagi supaya hasilnya lebih bagus, untuk filter yang digunakan adalah *exponential* filter. Berikut merupakan programnya pada gambar 3.17

```
yaw = yaw + yawds/250+0.00404;
float rolls = roll_before + 0.2 * (roll - roll_before);
roll_before = rolls;
float pitchs = pitch_before + 0.2 * (pitch - pitch_before);
pitch_before = pitchs;
```

Gambar 3.17 smothing data

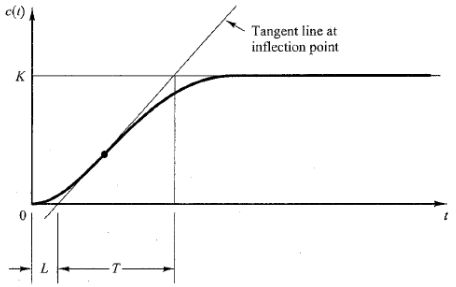
3.8 Perancangan Kontroler Proporsional

Terkadang pemodelan matematis suatu plant susah untuk dilakukan. Jika hal ini terjadi maka perancangan kontroler PID secara analitis tidak mungkin dilakukan sehingga perancangan kontroler PID harus dilakukan secara eksperimental. Pada bagian ini akan dibahas tentang perancangan kontroler PID secara eksperimental dengan menggunakan aturan *Ziegler – Nichols.*

Ziegler dan *Nichols* memberikan aturan untuk menentukan nilai penguatan proporsional Kp, waktu integral τi, dan waktu differensial τd yang didasarkan pada karakteristik respon transien dari *plant*. Terdapat 2 metode penentuan nilai parameter dari *ziegler Nichols*.

1. Metode Pertama *Ziegler Nichols*

   Dalam metode pertama, kita perlu mendapatkan respon plat terhadap masukan sinyal step. Jika plant tidak mengandung integrator atau kutub pasangan komplek yang dominan, maka kurva respon step plant tersebut kelihatan seperti kurva bentuk S. Jika respon plant tidak memiliki kurva berbentuk S, metode ini tidak berlaku. Kurva respon step dapat dihasilkan secara eksperimen atau dari simulasi dinamik sistem.



gambar 3.18 Step Respon Metode *Ziegler Nichols*

Kurva S-bentuk dapat dicirikan oleh dua konstanta, waktu tunda L dan konstanta waktu T. Waktu tunda dan konstanta waktu ditentukan dengan menggambar garis singgung pada titik infleksi kurva berbentuk S dan menentukan persimpangan tangen sejajar dengan sumbu waktu dan garis c (t) = K, seperti yang ditunjukkan pada gambar 3.18.

Tabel 3.1 Aturan *Ziegler Nichols* Berdasarkan Step Respon dari *Plant*

Type of Controller	K_p	T_i	T_d
P	$\frac{T}{L}$	∞	0
PI	$0.9\frac{T}{L}$	$\frac{L}{0.3}$	0
PID	$1.2\frac{T}{L}$	$2L$	$0.5L$

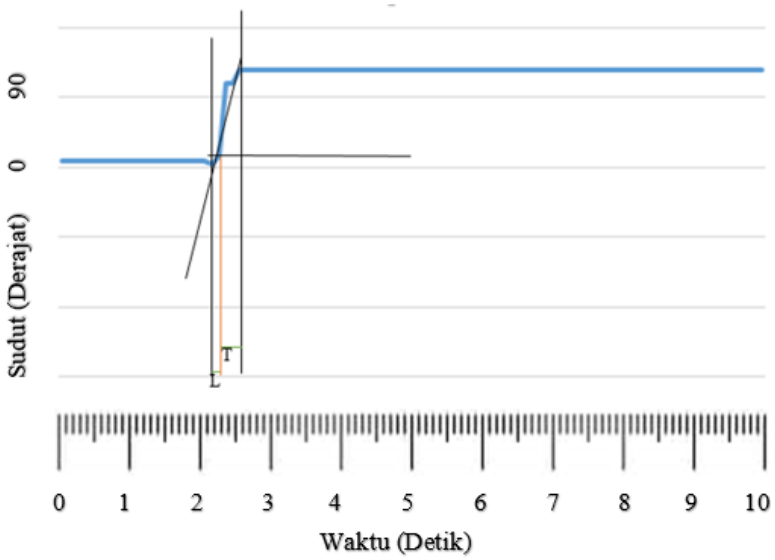
fungsi C (s) / U (s) kemudian dapat didekati oleh sistem orde pertama dengan jeda transportasi sebagai berikut:

$$\frac{C_S}{U_S} = \frac{Ke^{-Ls}}{T_S + 1} \tag{3.1}$$

Ziegler dan Nichols menyarankan untuk menetapkan nilai-nilai Kp τi dan τd sesuai dengan rumus yang ditunjukkan pada tabel 3.2. Perhatikan bahwa kontroler PID disetel oleh metode pertama dari aturan *Ziegler-Nichols.*

$$\begin{aligned} G_C(s) &= K_p\left(1 + \frac{1}{T_i s} + T_d s\right) \\ &= 1.2\frac{T}{L}\left(1 + \frac{1}{2L_s} + 0.5L_s\right) \\ &= 0.6T\frac{\left(S + \frac{1}{L}\right)^2}{s} \end{aligned} \tag{3.2}$$

Dengan demikian, kontroler PID memiliki kutub pada titik asal dan nol ganda pada s = -1 / L,



Gambar 3.19 Respon Sistem Terhadap Waktu Sampling

Pada gambar 3.19 didapatkan grafik respon dari IMU Sensor. Dari grafik tersebut dapat diambil nilai L dan T untuk penentuan parameter kontrol proporsional. Dari grafik tersebut didapatkan nilai L=1.2 dan T=2,5, sehingga jika dimasukkan pada tabel *Ziegler-Nichols* didapatkan nilai seperti pada tabel 3.2.

Tabel 3.2 Hasil Perhitungan Metode *Zieger Nichols*

	kp	**ki**	**kd**
p	2.08	0	0
pi	1.87	4	0
pid	3.36	2.4	0.6

Nilai dari respon sistem yang telah didapat, kemudian dimasukkan pada perhitungan untuk mendapatkan nilai variabel kontrol pada metode zieger nichols seperti pada tabel 3.2. nilai tersebut kemudian akan dimasukkan pada program untuk menghasilkan nilai kontrol.

BAB IV
HASIL IMPLEMENTASI ALAT DAN PENGUJIAN

Pada pembuatan rangkaian elektronika, sebelum dilakukan penyambungan antar rangkaian terlebih dahulu dilakukan pengujian terhadap rangkaian tersebut.Hal ini dilakukan untuk mengetahui hasil dari tiap-tiap rangkaian sebelum dilakukan pengujian secara keseluruhan. Pengujian meliputi pengujian Motor Servo dan pengujian *IMU Sensor*.

4.1 Implementasi Dan Spesifikasi Stabilisator

Setelah melalui beberapa tahapan perancangan dan perhitungan, berikut merupakan speifikasi dari sistem stabilisator yang terdapat pada tabel 4.1

Tabel 4.4.1Spesifikasi Alat

Spesifikasi Stabilisator	
Dimensi	Panjang = 22Cm Lebar = 8Cm Tinggi = 10Cm
Berat	200gram
Ukuran nampan	Panjang = 10Cm Lebar = 10Cm Tinggi = 4 Cm
Power Supply	Baterai Li-Po 12Volt
Mikrokontroler	Arduino Nano
Sudut	*Yaw , pitch*
Beban	500gram
Spesifikasi Servo	
Jenis	MG995S
Torsi	Maks 1.2Kg
Operating Voltage	5Volt
Arah Putaran	CCW-CW
Gear Type	Metal
Spesifikasi Sensor	
Jenis	IMU Sensor
Tipe	MPU6050
Sudut	3 Axis (*yaw, pitch, roll*)
Jenis Komunikasi	I2C

Gambar 4.1Realisasi *Bracket* Dengan Servo

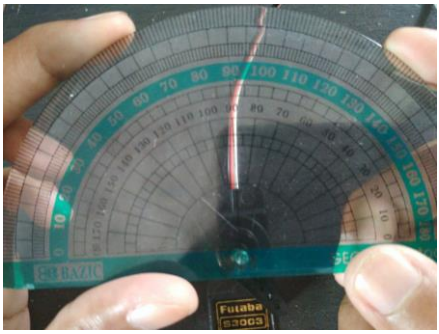
Gambar 4.2 Realisasi *Base* Stabilisator

4.2 Pengujian Motor Servo

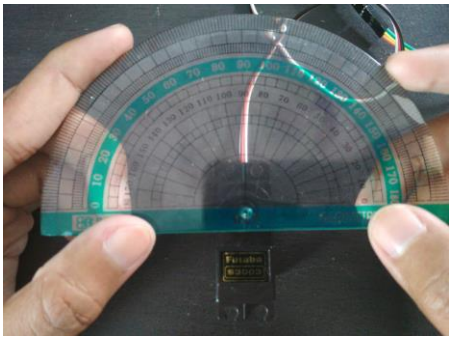
Dalam pengujian ini dilakukan untuk mendapatkan nilai galat motor servo, nilai galat diperoleh dari rumus :

$$\sum galat = ((Output-Input)/Input)/Jumlah\ Data \quad (4.1)$$

Pengukuran dilakukan untuk kedua servo yang berfungsi sebagai actuator, yakni servo A (*actuator yaw*) dan servo B (*actuator pitch*). Cara pengujian ialah dengan memberikan besaran derajat dari arduino dan membandingkan hasil keluaran tersebut dengan menggunakan suatu penggaris busur untuk mengetahui nilai besarnya sudut. Berikut merupakan cara pengukuran serta hasil pengukuran pada tabel 4.2.



Gambar 4.3 Pengukuran 50 Derajat

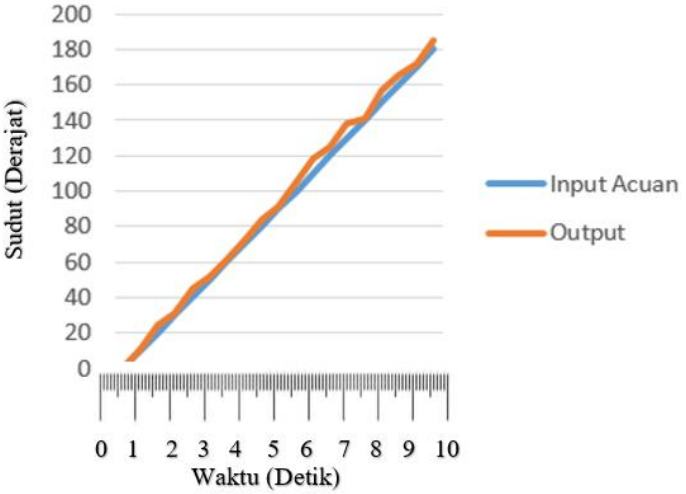


Gambar 4.4 Pengukuran 0 Derajat

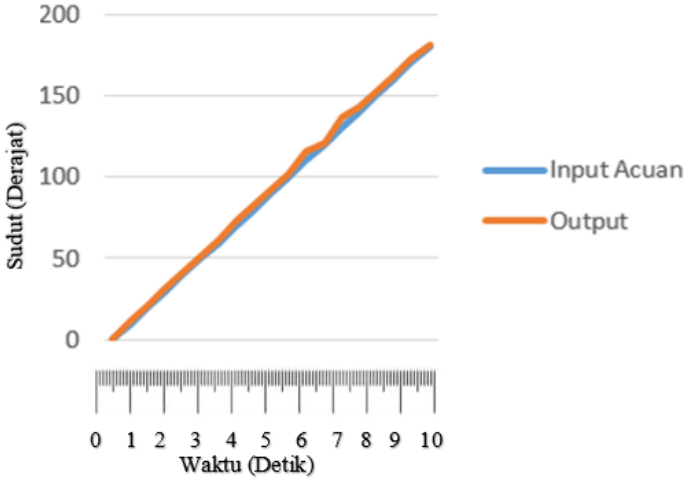
Tabel 4.2 Hasil Pengukuran Servo A dan Servo B

Input A	Output A	Nilai Galat	Input B	Output B	Nilai Galat
0	0	0	0	0	0
10	11	0.1	10	12	0.2
20	24	0.2	20	21	0.05
30	31	0.0333333	30	32	0.0666667
40	45	0.125	40	41	0.025
50	52	0.04	50	51	0.02
60	61	0.0166667	60	62	0.0333333
70	72	0.0285714	70	73	0.0428571
80	84	0.05	80	83	0.0375
90	91	0.0111111	90	92	0.0222222
100	105	0.05	100	102	0.02
110	118	0.0727273	110	115	0.0454545
120	125	0.0416667	120	121	0.0083333
130	138	0.0615385	130	137	0.0538462
140	141	0.0071429	140	143	0.0214286
150	157	0.0466667	150	152	0.0133333
160	165	0.03125	160	162	0.0125
170	172	0.0117647	170	173	0.0176471
180	185	0.0277778	180	181	0.0055556
galat rata-rata		0.0530676	Galat rata-rata		0.0386488

Pada tabel diatas dapat kita lihat bahwa keluaran pada servo mendekati hasil pembacaan dari acuan yang diberikan. Terdapat error yang cukup signifikan. Acuan yang digunakan adalah penggaris busur dan masukan yang diberikan adalah besarnya derajat dari arduino. Pada keluaran busur derajat ditempelkan pada motor servo untuk mengetahui besarnya perubahan sudut yang terjadi.



Gambar 4.5 Linearisasi Servo A



Gambar 4.6 Linearisasi Servo B

Gambar 4.3 dan 4.4 menunjukkan bahwa sudut yang dihasilkan oleh servo tidak jauh berbeda dengan sudut acuan yang digunakan, shingga servo tersebut layak digunakan untuk penelitian ini. Berikut persamaan garis yang didapat dari linieritas kedua servo tersebut :

$$y = 1.0219x + 1.5526 \quad \text{(Linieritas Servo A)} \quad (4.1)$$

$$y = 1.0102x + 1.3474 \quad \text{(Linieritas Servo B)} \quad (4.2)$$

4.3 Pengujian IMU Sensor

MPU6050 terdiri dari 3 buah sensor, pada penelitian ini kita menggunakan accelerometer dan gyro. Langkah pertama yang harus kita lakukan adalah mengambil data RAW keduanya.

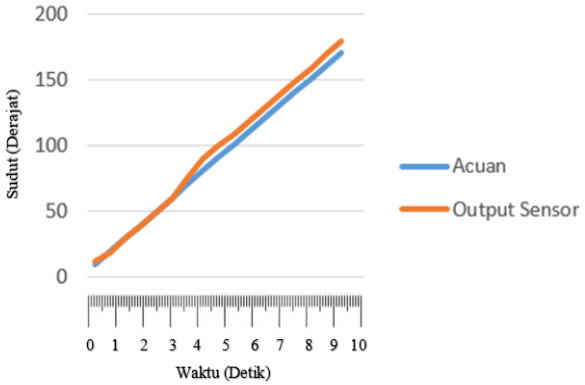
Tabel 4.3 Data Raw Sensor

sudut	GyX	GyY	GyZ	AcX	AcY	AcZ
10	1640	-3228	-111	2864	-1724	13856
20	2987	-6115	-352	5320	-1156	13408
30	2876	-3440	238	7656	-1008	12664
40	-436	-2814	282	10028	-1164	11068
50	-97	-2828	506	12200	-1464	9272
60	-3450	-3485	2543	13716	-1492	7484
70	471	-1512	-623	15572	-2412	4820
80	481	-2559	3473	16392	-1556	2276
90	1314	-3382	1973	17168	-700	-760
100	-151	-4109	1906	16592	-1200	-3872
110	1504	-6977	-1059	16372	-1000	-7656
120	2757	-5372	-138	15256	-716	-10676
130	-424	-5364	48	13608	132	-13436
140	1590	-4770	-1438	11844	612	-15076
150	-114	-772	918	8856	2104	-17032
160	-494	-2494	149	6088	412	-18572
170	152	-3664	714	2888	1592	-19472
180	-454	-4674	1162	108	324	-19320

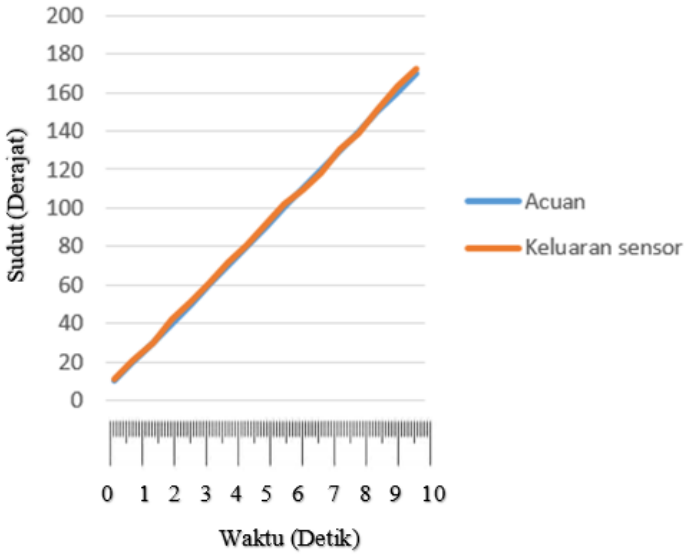
Setelah Data RAW didapat maka sekarang kita mengambil data kesesuaian tiap sudut terhadap alat acuan yakni busur untuk dua buah sudut yang digunakan yakni *yaw* dan *roll.*

Tabel 4.4 Kesesuaian Sudut

Sudut Acuan	Output Sensor Yaw	Output Sensor Roll
10	12	11
20	19	21
30	29	30
40	39	42
50	49	51
60	59	61
70	75	72
80	89	81
90	99	91
100	108	102
110	118	109
120	128	118
130	138	131
140	148	139
150	158	151
160	169	163
170	179	172
180	-18	178



Gambar 4.7 Perbandingan keluaran sensor *Yaw* terhadap acuan



Gambar 4.8 Perbandingan keluaran sensor *Roll* terhadap acuan

Pada gambar 4.7 dan gambar 4.8 menunjukkan bahwa keluaran dari sensor dan busur acuan memiliki linearitas yang cukup tinggi. Nilai galat yang diperoleh *relative* kecil, hal ini dapat kita lihat pada persamaan garis dibawah ini :

Persamaan garis untuk gambar 4.7:

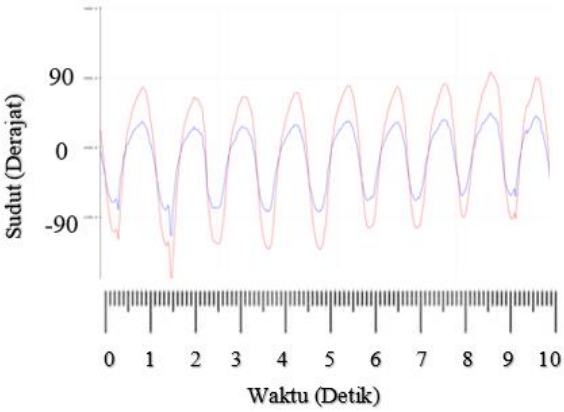
$$y = 1.0718x - 1.4044 \quad (4.3)$$

Persamaan garis untuk gambar 4.7:

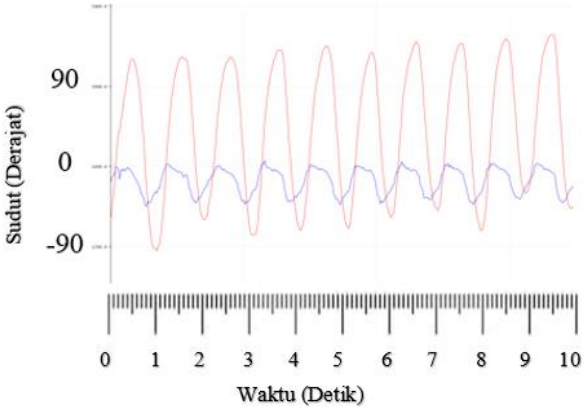
$$y = 0.9998x + 0.9044 \quad (4.4)$$

4.3 Pengujian Kontroler Proporsional

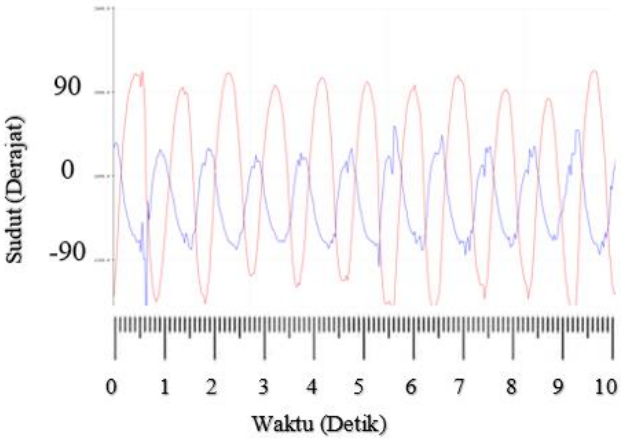
Kontrol proporsional memberikan suatu pengali pada error agar cepat mencapai suatu nilai yang dituju, pemberian nilai kontroler proporsional dapat mempercepat sistem dalam mencapai suatu keadaan yang diinginkan. Besarnya nilai kontroler ditentukan dari cepatnya sistem dalam mencapai suatu keadaan yang diingingkan.



Gambar 4.9 Perbandingan keluaran dan mauskan dengan besar Kp=5



Gambar 4.10 Perbandingan keluaran dan mauskan dengan besar Kp=10



Gambar 4.11 Perbandingan keluaran dan mauskan dengan besar Kp=15

Pengujian dilakukan tanpa beban dan dengan memberikan nilai konstanta proporsional yang berbeda-beda pada sistem. Pada gaambar 4.8 dapat kita lihat bahwa sistem menerima besarnya nilai konstanta proporsional sebesar 5 dan terlihat bahwa sistem masih kurang cepat dalam mencapai keadaan pada gambar 4.9 sistem menerima besarnya nilai kosntanta proporsional sebesar 10 dan berdasarakan gambar terlihat bahwa respon sistem cepat dibandingakan dengan pengujian sebelumnya. Pada gambar 4.10 sistem diberikan besarnya nilai konstanta proporsional sebesar 15 dan terlihat bahwa sistem mengalami osilasi.

4.4 Pengujian Sistem Secara Keseluruhan

Pengujian sistem keseluruhan bertujuan untuk melihat respon sistem yang telah diberi kontroler PID dengan pembacaan perubahan sudut terhadap respon aktuator yang dibaca oleh MPU6050. Pengujian ini dilakukan untuk mengetahui tingkat keberhasilan stabilisator, pengujian dilakukan terhadap 10 subjek yang beerbeda dengan dua kondisi yakni dengan beban dan tanpa beban. Tingkat keberhasilan diperoleh dengan menggunakan rumus sebagai berikut :

$$\sum \textit{Succes Rate} = (\text{Berhasil/Jumlah Data})*100\% \quad (4.5)$$

4.4.1 Pengujian Tanpa Beban

Gambar 4.12 Stabilisator tanpa beban diputar terhadap sumbu *yaw*

Gambar 4.13 Stabilisator tanpa beban diputar terhadap sumbu *pitch*

Pengujian sistem tanpa beban dilakukan dengan menjalanakan stabilisator dan menempatkan sebuah mangkuk diatas nampan guna mengetahui kestabilan. Pengujian dilakukan dengan dua keadaan yakni diam dan berjalan, dan diputar terhadap sumbu *yaw* serta *pitch*. Hasil pengujian dapat dilihat pada tabel 4.5

Tabel 4.5 Uji tanpa beban ketika diam dan berjalan

No	Keadaan diam		Keadaan berjalan	
1	Subjek 1	Berhasil	Subjek 1	Berhasil
2	Subjek 2	Berhasil	Subjek 2	Gagal
3	Subjek 3	Berhasil	Subjek 3	Berhasil
4	Subjek 4	Berhasil	Subjek 4	Berhasil
5	Subjek 5	Berhasil	Subjek 5	Berhasil
6	Subjek 6	Berhasil	Subjek 6	Berhasil
7	Subjek 7	Berhasil	Subjek 7	Berhasil
8	Subjek 8	Berhasil	Subjek 8	Gagal
9	Subjek 9	Berhasil	Subjek 9	Gagal
10	Subjek 10	Berhasil	Subjek 10	Gagal

4.4.2 Pengujian Dengan Beban

Gambar 4.14 Stabilisator dengan beban diputar terhadap sumbu *yaw*

Gambar 4.15 Stabilisator dengan beban diputar terhadap sumbu *pitch*

Pengujian sistem dengan beban dilakukan dengan menjalanakan stabilisator dan menempatkan sebuah mangkuk yang telah diberi mie instan diatas nampan guna mengetahui kestabilan. Berat beban yang

berupa mie instan mempunyai berat sebesar 70 Gram. Pengujian dilakukan dengan dua keadaan yakni diam dan berjalan, dan diputar terhadap sumbu *yaw* serta *pitch*. Hasil pengujian dapat dilihat pada tabel 4.6

Tabel 4.6 Uji dengan beban ketika diam dan berjalan

No	Keadaan diam		Keadaan berjalan	
1	Subjek 1	Berhasil	Subjek 1	Berhasil
2	Subjek 2	Berhasil	Subjek 2	Gagal
3	Subjek 3	Gagal	Subjek 3	Berhasil
4	Subjek 4	Berhasil	Subjek 4	Gagal
5	Subjek 5	Berhasil	Subjek 5	Gagal
6	Subjek 6	Gagal	Subjek 6	Berhasil
7	Subjek 7	Gagal	Subjek 7	Berhasil
8	Subjek 8	Berhasil	Subjek 8	Gagal
9	Subjek 9	Berhasil	Subjek 9	Berhasil
10	Subjek 10	Berhasil	Subjek 10	Berhasil

Berdasarkan hasil uji tanpa beban pada tabel 4.5 didapatkan *success rate* untuk stabilisator ketika sistem tanpa beban adalah sebesar 100% untuk keadaan diam dan 60% ketika keadaan berjalan. Sedangkan pada keadaan dengan beban didapatkan *success rate* sebesar 70% pada keadaan diam dan 60% saat keadaan berjalan, hasil pengujian dapat dilihat pada tabel 4.6.

BAB V
PENUTUP

Setelah melakukan perencanaan, perancangan, dan pengujian alat maka ini dapat mengambil kesimpulan dan memberikan saran demi penyempurnaan penelitian ini.

5.1 Kesimpulan

Berdasarkan data hasil uji implementasi alat stabilisator nampan ini didapatkan kesimpulan antara lain terdapat komponen yang penting dalam sistem ini yakni IMU Sensor dan Motor Servo sebagai actuator. Nilai pembacaan IMU sensor masih harus deprogram lebih lanjut karena nilainya terus naik sedikit sehingga mempengaruhi dalam pembacaan. Besarnya nilai konstanta proporsional untuk sistem ini yakni 10, karena apabila nilai tersebut ditambah maka akan terjadi osilasi pada sistem dan sulit dalam mencapai kestabilan.

MPU6050 mampu mendeteksi perubahan sudut dengan error rata-rata sebesar 3,17 % untuk sumbu pitch sedangkan error rata-rata untuk sumbu yaw sebesar 0,2 %.

Hasil pengujian sistem *success rate* stabilisator ketika sistem tanpa beban adalah sebesar 100% untuk keadaan diam dan 60% ketika keadaan berjalan. Sedangkan pada keadaan dengan beban didapatkan *success rate* sebesar 70% pada keadaan diam dan 60% saat keadaan berjalan.

5.2 Saran

Pemberian sinyal konrol lain diharapkan diberikan agar sistem semakin cepat dan stabil dalam mengkoreksi nilai galat yang disebabkan oleh getaran yang dihasilkan.

-----Halaman ini sengaja dikosongkan-----

DAFTAR PUSTAKA

[1] David G, Cathi A, "Parkinson's Disease Handbook" , Staten Island NY, Medtronic.

[2] Nutt, John G., and G. Frederick Wooten. "Diagnosis and Initial Management of Parkinson's Disease." New England Journal of Medicine 353, no. 10 (2005).

[3] Jing Qiao, Zhixiang Liu, "Gain Scheduling PID Control Of The Quad-Rotor Helicopter" IEEExplore 2017.

[4] Al Sahib R. Alwash Monaf SN, Abdulla M. Din, and Jordan Tafila. "Rotor Position Detection And Control For Spindle Brushless Dc Motors Using Dummy Windings," IEEExplore 2013.

[5] Bhilai, SSTC. "Comparative Study of P, PI and PID Controller forSpeed Control of VSI-Fed Induction Motor,"IEEExplore 2014.

[6] Bolandi, Hossein, Mohammad Rezaei, Reza Mohsenipour, Hossein Nemati, and S. M. Smailzadeh. "Attitude Control of a Quadrotor with Optimized PID Controller." Intelligent Control and Automation 04, no. 03 (2013).

[7] Kritika Bansal, Lilie Dewan, "Stabilization Of A Gimbal Sistem Using Pid Control And Compensator-A Comparison." International Journal Of Electrical. IEEExplore 2015.

[8] Ibrahim, Dogan. Microcontroller-Based Temperature Monitoring and Control. Oxford: Newnes, 2002.

[9] Hartono Nanang Budi, Bambang Sumantri Kemalasari, and Ardik Wijayanto. "Pengaturan Posisi Motor Servo Dc Dengan Metode P, Pi, Dan Pid." POMITS Vol 1 (2008): 1–9.

[10] Datasheet MPU-6050 "MPU-6000 and MPU-6050 product specification", January 2018.

[11] Muhammad Fahrezi Alwi, Muhammad Rivai, and Suwito. "Perancangan Sistem Stabilisasi Kamera Tiga Sumbu Dengan Metode Kontrol Fuzzy Untuk Mobile Surveillance Robot." Jurnal Teknik ITS .

[12] Rodríguez-Martín, Daniel, Carlos Pérez-López, Albert Samà, Joan Cabestany, and Andreu Català. "A Wearable Inertial Measurement Unit for Long-Term Monitoring in the Dependency Care Area." Sensors 13, no. 12 (October 18, 2013).

[13] Daware, Madhuri Hanumanta, and A. S. Patil. "Implementation of I2C Bus Protocol on FPGA." International Journal of Current Engineering and Scientific Research (IJCESR) 2015.

[14] Nurussa'adah,. "Aplikasi Pengenalan Suara Sebagai Pengendali Peralatan Listrik berbasis ArduinoUNO." Jurnal Mahasiswa TEUB 2, no. 5 (2014).

[15] Kurniawan, Agung Andri, Muhammad Rivai, and Fajar Budiman. "Sistem Pemandu Pendaratan Pada Balon Udara Berbasis Pengolahan Citra Dan Kendali PID." Jurnal Teknik ITS 5, no. 2 (2016): A179–A184.

[16] Datasheet Motor Servo "Motor Servo sg90 datasheet" .

[17] Rashid, M. Z. A., M. S. M. Aras, A. A. Radzak, A. M. Kassim, and A. Jamali. "Development of Hexapod Robot with Manoeuvrable Wheel." International Journal of Advanced Science and Technology 49 (2012).**}**

[18] Rivai, Muhammad, Masaji Suwito, Peter Chondro, and Shanq-Jang Ruan. "Design and Implementation of a Submerged Capacitive Sensor in PID Controller to Regulate the Concentration of Non-Denatured Ethyl Alcohol," 45–50. IEEE, 2015.

[19] Wijaya, Putut Dwi, Muhammad Rivai, and Tasripan Tasripan. "Rancang Bangun Mesin Pemotong Styrofoam 3 Axis Menggunakan Hot Cutting Pen Dengan Kontrol PID." *Jurnal Teknik ITS* 6, no. 2 (October 8, 2017).

LAMPIRAN A (Program)

```
#include <Servo.h>
#include<Wire.h>

  const int MPU_addr=0x68;
  int16_t AcX,AcY,AcZ,Tmp,GyX,GyY,GyZ;
  float yaw=0;
  float pitch_before, roll_before;
  int derajat3, derajat4;
Servo base;
Servo elbow;
Servo camera;
int mode=0;
long previousMillis = 0;
long interval = 20;
int kiri=0;
int atas=0;
int kanan=0;
int error = 1;
int bawah=0;
int roll_kiri=0;
int roll_kanan=0;
float error, errori, errord, error_sblmI,error_sblmD, OutP, OutI, OutD,
Kp, Ki, Kd, Tc=0.01 ;
int i=1;
int a=1;
float pid_p=0;
float pid_i=0;
float pid_d=0;
int lapse = 1;
int tambah=1;
int base_deg = 90;
int elbow_deg = 60;
int camera_deg = 80;
float set_point,  previous_error, kp, ki, kd;
int outPID, aduh;
float elapsedTime, time, timePrev;
int imb;
```

```
int imb2;
int imb3;

void setup() {
  // put your setup code here, to run once:
      Wire.begin();
      Wire.beginTransmission(MPU_addr);
      Wire.write(0x6B);  // PWR_MGMT_1 register
      Wire.write(0);     // set to zero (wakes up the MPU-6050)
      Wire.endTransmission(true);
      time = millis();

  kp =150;//3.55
  ki =0.005;//0.003
  kd =2.05;//2.05
  set_point=0;
  imb=0;
  imb2=0;
  imb3=186;

 pinMode(9, OUTPUT);
 pinMode(10, OUTPUT);
 Serial.begin(9600);
 delay(1000);

}

void loop() {
  // put your main code here, to run repeatedly:
    timePrev = time;  // the previous time is stored before the actual time
read
    time = millis();  // actual time read
    elapsedTime = (time - timePrev) / 1000;
     Wire.beginTransmission(MPU_addr);
      Wire.write(0x3B);  // starting with register 0x3B (ACCEL_XOUT_H)
      Wire.endTransmission(false);
      Wire.requestFrom(MPU_addr,14,true);     // request a total of 14
registers
```

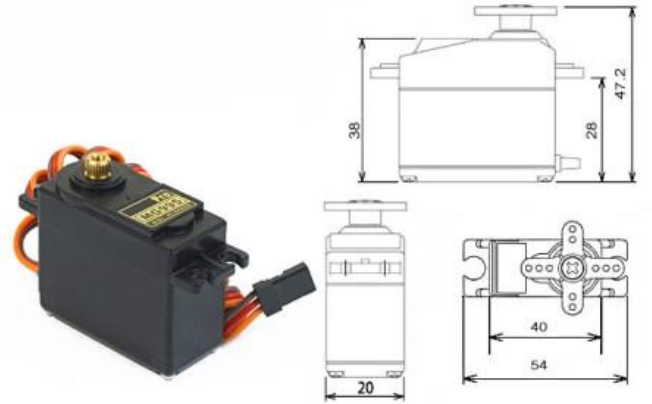
```
    AcX=Wire.read()<<8|Wire.read();    // 0x3B (ACCEL_XOUT_H) &
0x3C (ACCEL_XOUT_L)
    AcY=Wire.read()<<8|Wire.read();    // 0x3D (ACCEL_YOUT_H) &
0x3E (ACCEL_YOUT_L)
    AcZ=Wire.read()<<8|Wire.read();    // 0x3F (ACCEL_ZOUT_H) &
0x40 (ACCEL_ZOUT_L)
    Tmp=Wire.read()<<8|Wire.read();  // 0x41 (TEMP_OUT_H) & 0x42
(TEMP_OUT_L)
    GyX=Wire.read()<<8|Wire.read();    // 0x43 (GYRO_XOUT_H) &
0x44 (GYRO_XOUT_L)
    GyY=Wire.read()<<8|Wire.read();    // 0x45 (GYRO_YOUT_H) &
0x46 (GYRO_YOUT_L)
    GyZ=Wire.read()<<8|Wire.read();    // 0x47 (GYRO_ZOUT_H) &
0x48 (GYRO_ZOUT_L)

     float pitch = atan2(AcY,  AcZ)*57.2958;
     float roll = atan2(AcX,  AcZ)*57.2958;

     float yawds = (float)GyZ/65.536;
     yaw = yaw + yawds/250+0.00404;


    float rolls = roll_before + 0.2 * (roll - roll_before);
    roll_before = rolls;

    float pitchs = pitch_before + 0.2 * (pitch - pitch_before);
    pitch_before = pitchs;

    int derajat1 = (int)map(yaw, -50,50, 500, 2500);
    int derajat2 = (int)map(pitchs,-90,90,2500, 500);
    int derajat3 = (int)map(rolls, -90,90,2500, 500);
    aduh = derajat3 + OutP;

    error = rolls - set_point;

   if(derajat1 <=400 ){
     derajat1=400;
    }
    if(derajat1 >=2200 ){
```

```
      derajat1=2200;
     }
//
   if(derajat3 <=400 ){
     derajat3=400;
     }
    if(derajat3 >=2200 ){
     derajat3=2200;
     }
//
    if(derajat4 <=400 ){
     derajat4=400;
     }
    if(derajat4 >=2200 ){
     derajat4=2200;
     }

    derajat1=derajat1-imb2;


     Serial.print(aduh);Serial.print("\t");
     Serial.print(imb2);Serial.print("\n");

     unsigned long currentMillis = millis();

   if(currentMillis - previousMillis > interval){
    previousMillis = currentMillis;
    updateServo(9, derajat1);
    updateServo(10, aduh);
    //updateServo(11, derajat3);
    }

    imb=imb+1;
       }
void updateServo(int pin, int pulse){
  digitalWrite(pin, HIGH);
  delayMicroseconds(pulse);
  digitalWrite(pin, LOW);
}
```

```
void pid(){

OutP = kp * error;

errori=error;
OutI = (ki*errori*Tc);

errord=error;
OutD = kd*((errord - previous_error)/elapsedTime);
 previous_error = error;
outPID = OutP + OutI + OutD;

if (outPID >= 2500)
outPID = 2500;
if (outPID <= 500)
outPID = 500;
if (error >= 1400 || error <=1600){
outPID = 0;

  }
}
```

LAMPIRAN B (Datasheet)

MG995 High Speed
Metal Gear Dual Ball Bearing Servo



The unit comes complete with 30cm wire and 3 pin 'S' type female header connector that fits most receivers, including Futaba, JR, GWS, Cirrus, Blue Bird, Blue Arrow, Corona, Berg, Spektrum and Hitec.

This high-speed standard servo can rotate approximately 120 degrees (60 in each direction). You can use any servo code, hardware or library to control these servos, so it's great for beginners who want to make stuff move without building a motor controller with feedback & gear box, especially since it will fit in small places. The MG995 Metal Gear Servo also comes with a selection of arms and hardware to get you set up nice and fast!

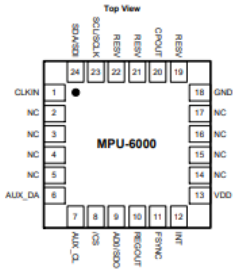
Specifications

- Weight: 55 g
- Dimension: 40.7 x 19.7 x 42.9 mm approx.
- Stall torque: 8.5 kgf·cm (4.8 V), 10 kgf·cm (6 V)
- Operating speed: 0.2 s/60º (4.8 V), 0.16 s/60º (6 V)
- Operating voltage: 4.8 V a 7.2 V
- Dead band width: 5 µs
- Stable and shock proof double ball bearing design
- Temperature range: 0 ºC – 55 ºC

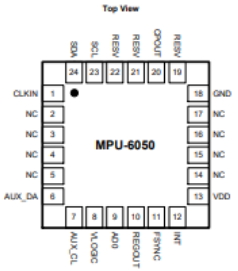
7 Applications Information

7.1 Pin Out and Signal Description

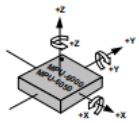
Pin Number	MPU-6000	MPU-6050	Pin Name	Pin Description
1	Y	Y	CLKIN	Optional external reference clock input. Connect to GND if unused.
6	Y	Y	AUX_DA	I^2C master serial data, for connecting to external sensors
7	Y	Y	AUX_CL	I^2C Master serial clock, for connecting to external sensors
8	Y		/CS	SPI chip select (0=SPI mode)
8		Y	VLOGIC	Digital I/O supply voltage
9	Y		AD0 / SDO	I^2C Slave Address LSB (AD0); SPI serial data output (SDO)
9		Y	AD0	I^2C Slave Address LSB (AD0)
10	Y	Y	REGOUT	Regulator filter capacitor connection
11	Y	Y	FSYNC	Frame synchronization digital input. Connect to GND if unused.
12	Y	Y	INT	Interrupt digital output (totem pole or open-drain)
13	Y	Y	VDD	Power supply voltage and Digital I/O supply voltage
18	Y	Y	GND	Power supply ground
19, 21	Y	Y	RESV	Reserved. Do not connect.
20	Y	Y	CPOUT	Charge pump capacitor connection
22	Y	Y	RESV	Reserved. Do not connect.
23	Y		SCL / SCLK	I^2C serial clock (SCL); SPI serial clock (SCLK)
23		Y	SCL	I^2C serial clock (SCL)
24	Y		SDA / SDI	I^2C serial data (SDA); SPI serial data input (SDI)
24		Y	SDA	I^2C serial data (SDA)
2, 3, 4, 5, 14, 15, 16, 17	Y	Y	NC	Not internally connected. May be used for PCB trace routing.



QFN Package
24-pin, 4mm x 4mm x 0.9mm

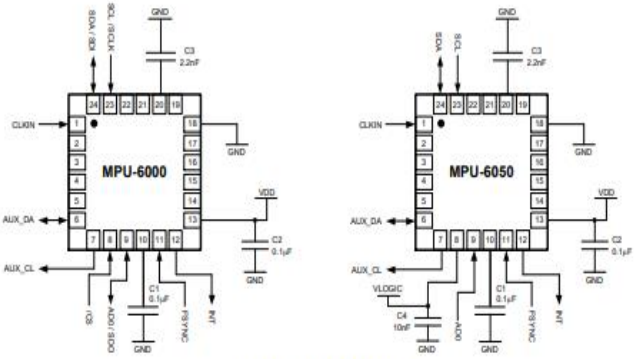


QFN Package
24-pin, 4mm x 4mm x 0.9mm



Orientation of Axes of Sensitivity and Polarity of Rotation

7.2 Typical Operating Circuit



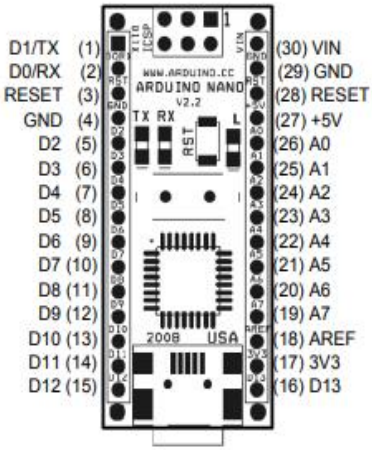
Typical Operating Circuits

7.3 Bill of Materials for External Components

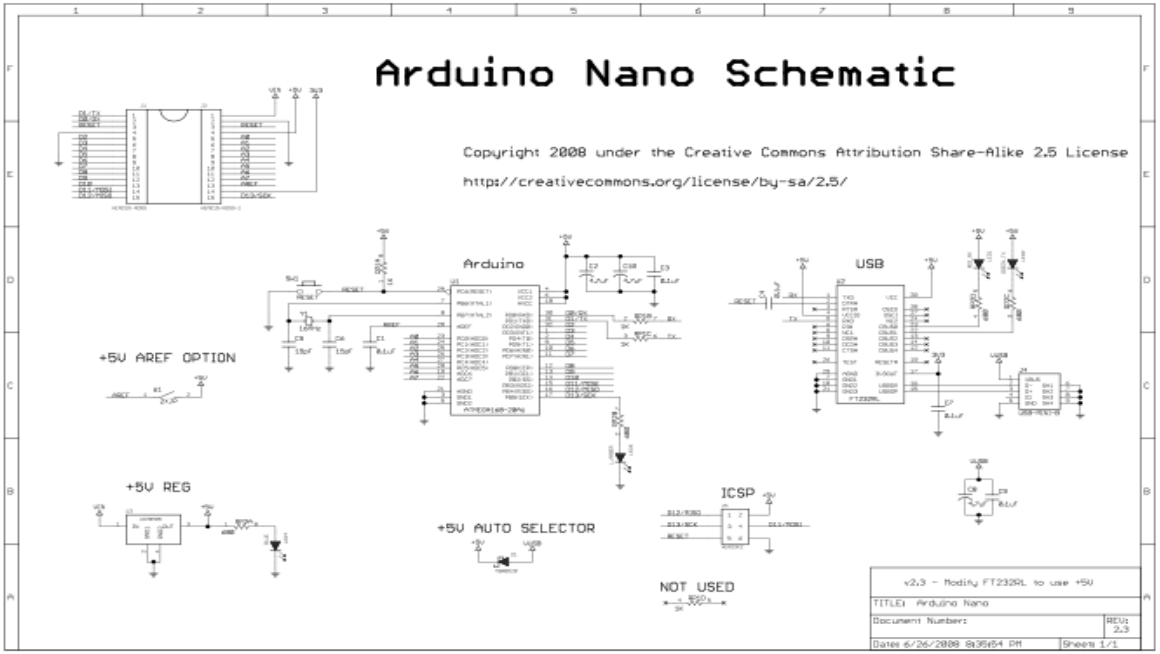
Component	Label	Specification	Quantity
Regulator Filter Capacitor (Pin 10)	C1	Ceramic, X7R, 0.1µF ±10%, 2V	1
VDD Bypass Capacitor (Pin 13)	C2	Ceramic, X7R, 0.1µF ±10%, 4V	1
Charge Pump Capacitor (Pin 20)	C3	Ceramic, X7R, 2.2nF ±10%, 50V	1
VLOGIC Bypass Capacitor (Pin 8)	C4*	Ceramic, X7R, 10nF ±10%, 4V	1

* MPU-6050 Only.

Arduino Nano Pin Layout



Pin No.	Name	Type	Description
1-2, 5-16	D0-D13	I/O	Digital input/output port 0 to 13
3, 28	RESET	Input	Reset (active low)
4, 29	GND	PWR	Supply ground
17	3V3	Output	+3.3V output (from FTDI)
18	AREF	Input	ADC reference
19-26	A7-A0	Input	Analog input channel 0 to 7
27	+5V	Output or Input	+5V output (from on-board regulator) or +5V (input from external power supply)
30	VIN	PWR	Supply voltage


Arduino Nano Schematic
Copyright 2008 under the Creative Commons Attribution Share-Alike 2.5 License
http://creativecommons.org/license/by-sa/2.5/
Arduino
USB
+5V AREF OPTION
+5V REG
+5V AUTO SELECTOR
ICSP
NOT USED
v2.3 - Modify FT232RL to use +5V
TITLE: Arduino Nano
Document Number:
REV: 2.3
Date: 6/26/2008 8:35:54 PM
Sheet 1/1

DAFTAR RIWAYAT HIDUP

Abu Hatim Kurniawan lahir pada 1 Juli 1994 di sidoarjo. Penulis adalah anak pertama dari dua bersaudara. Penulis menyelesaikan pendidikan dasar di SD Negeri 1 Mojoruntut kemudian melanjutkan ke jenjang pendidikan menengah di SMP Negeri 1 Krembung dan dilanjutkan kembali ke jenjang pendidikan atas di SMA Negeri 1 Krembung. Pada jenjang perguruan tinggi penulis memulai pendidikan pada jenjang pendidikan Diploma di D3 Teknik Elektro, Fakultas Teknologi Industri, Institut Teknologi Sepuluh Nopember Surabaya pada tahun 2013. Kemudian penulis menyelesaikan masa pendidikan diploma pada tahun 2016. Penulis kemudian melanjutkan kembali jenjang pendidikan sarjana di S1 Teknik Elektro ITS pada tahun 2016. Semasa kuliah pada jenjang diploma maupun sarjana penulis pernah melaksanakan kerja praktek di PT PJB dan juga PT PLN. Penulis juga pernah menjadi asisten laboratorium komputer pada saat jenjang diploma,

Email : aboehatim@gmail.com

-----Halaman ini sengaja dikosongkan-----