SISTEM KENDALI POSISI DENGAN METODE LINEAR QUADRATIC GAUSSIAN PADA VTOL *(VERTICAL TAKE OFF LANDING)*

SKRIPSI

Diajukan sebagai salah satu syarat

untuk memperoleh gelar

Sarjana Teknik

Oleh :

RILO PAMBUDI ADITYA WARDANI

NIM. I0716028

PROGRAM STUDI TEKNIK ELEKTRO

FAKULTAS TEKNIK UNIVERSITAS SEBELAS MARET

SURAKARTA

2020

HALAMAN PENUGASAN

SURAT PERNYATAAN INTEGRITAS PENULIS

Saya mahasiswa Program Studi Sarjana Teknik Elektro Universitas Sebelas Maret yang bertanda tangan di bawah ini :

Nama	: Rilo Pambudi Aditya Wardani
NIM	: I0716028
Judul tugas akhir	: Sistem Kendali Posisi Dengan Metode Linear Quadratic Gaussian Pada VTOL *(Vertical Take Off Landing)*

Dengan ini menyatakan bahwa tugas akhir yang saya susun tidak mencontoh atau melakukan plagiat dari karya tulis orang lain. Jika terbukti tugas akhir yang saya susun tersebut merupakan hasil plagiat dari karya orang lain maka tugas akhir yang saya susun tersebut dinyatakan batal dan gelar sarjana yang saya peroleh dengan sendirinya dibatalkan atau dicabut.

Demikian surat pernyataan ini saya buat dengan sebenarnya dan apabila dikemudian hari terbukti melakukan kebohongan maka saya sanggup menanggung segala konsekuensinya.

Surakarta, 15 Juli 2020

Rilo Pambudi Aditya Wardani

NIM. I0716028

SISTEM KENDALI POSISI DENGAN METODE LINEAR QUADRATIC GAUSSIAN PADA VTOL (VERTICAL TAKE OFF LANDING)

Disusun oleh

RILO PAMBUDI ADITYA WARDANI

NIM. I0716028

Dosen Pembimbing I	**Dosen Pembimbing II**
Hari Maghfiroh, S.T., M.Eng.	**Chico Hermanu B. A., S.T., M.Eng.**
NIP. 199104132018031001	**NIP. 198804162015041002**

Telah dipertahankan di hadapan Tim Dosen Penguji pada hari, tanggal 2020

1. Hari Maghfiroh, S.T., M.Eng.
NIP. 199104132018031001 .....

2. Chico Hermanu B. A., S.T., M.Eng.
NIP. 198804162015041002 .....

3. Dr.Ir. Augustinus Sujono M.T.
NIP. 1951100120161001 .....

4. Feri Adriyanto, Ph.D.
NIP. 196801161999031001 .....

Mengetahui,

Kepala Prodi Teknik Elektro	Koordinator Tugas Akhir
Feri Adriyanto, Ph.D.	**M. Hamka Ibrahim, S.T., M.Eng.**
NIP. 196801161999031001	NIP. 19680116199903100

SISTEM KENDALI POSISI DENGAN METODE LINEAR QUADRATIC GAUSSIAN PADA VTOL (VERTICAL TAKE OFF LANDING)

Rilo Pambudi Aditya Wardani

Program Studi Teknik Elektro, Fakultas Teknik, Universitas Sebelas Maret

Email : rilopaw@gmail.com

ABSTRAK

Perkembangan kendali pada UAV *(Unmanned Aerial Vehicle)* khususnya jenis VTOL (*Vertical Take Off Landing*) sangat pesat dikarenakan banyaknya inovasi pada penggunaan selain militer. Pada pesawat jenis VTOL diperlukan sistem kendali yang dapat mempertahankan posisi kemiringan untuk bermanuver. Saat ini sistem kendali yang close loop yang sering digunakan adalah PID *(Proportional Integral and Derivative)*, namun hasilnya kurang memuaskan dikarenakan responnya yang buruk. Maka diusulkan sistem kendali posisi menggunakan metode LQG *(Linear Quadratic Gaussian)*, yang bertujuan untuk menghasilkan respon yang cepat dan stabil sehingga mampu mendukung kebutuhan pesawat untuk bermanuver. Pada penelitian ini menggunakan motor BLDC sebagai penggerak secara *sensorless* dan ESC *(Electronic Speed Controller)* sebagai *driver* sehingga bobot menjadi lebih ringan dan mendukung pengujian. Pada sistem kendali ini akan menggunakan mikrokontroler Arduino dengan membandingkan algoritma metode PID dan LQG. Pada hasilnya didapatkan kendali LQG unggul dengan ISE hanya 792 dibanding 3035 pada kendali PID saat pengujian variasi posisi dan 2088 dibanding 2353 saat pengujian simulasi manuver.

Kata Kunci : *Linear Quadratic Gaussian*, *Vertical Take-Off Landing*, Arduino, *Brushless DC*.

SISTEM KENDALI POSISI DENGAN METODE LINEAR QUADRATIC GAUSSIAN PADA VTOL (VERTICAL TAKE OFF LANDING)

Rilo Pambudi Aditya Wardani

Program Studi Teknik Elektro, Fakultas Teknik, Universitas Sebelas Maret

Email : rilopaw@gmail.com

ABSTRACT

The development of controls on UAV (Unmanned Aerial Vehicle), especially the type of VTOL (Vertical Take Off Landing) is very rapid due to many innovations in non-military uses. In VTOL type aircraft, a control system that can maintain the tilt position to maneuver is needed. Currently the close loop control system that is commonly used is PID (Proportional Integral and Derivative), but the results are unacceptable due to poor response. Then the position control system is proposed using the LQG (Linear Quadratic Gaussian) method, which aims to produce a fast and stable response so that it can support the needs of the aircraft to maneuver. In this study using a BLDC motor sensorless method and ESC (Electronic Speed Controller) as a driver so that the weight becomes lighter and supports testing. In this control system will use an Arduino microcontroller by comparing the algorithm of the PID and LQG methods. The results obtained in LQG control with ISE only 792 compared to 3035 in the PID control when position variations and 2088 compared to 2353 when maneuver simulation.

Keywords : Linear Quadratic Gaussian, Vertical Take-Off Landing, Arduino, *Brushless DC.*

KATA PENGANTAR

Puji dan syukur kepada Tuhan Yang Maha Esa, atas rahmat dan karunianya, penulis dapat menyelesaikan penyusunan skripsi dengan judul Sistem Kendali Dengan Metode *Linear Quadratic Gaussian* pada VTOL *(Vertical Take-Off Landing)*. Dalam penyusunan skripsi ini, tentu penulis perlu melewati serangkaian proses, sebagaimana kita ketahui bahwa dalam berproses tidak selamanya jalan yang harus dilalui mulus. Kadang kalanya jalan berlubang dan berliku menjadi bagian dari perjalanan proses yang harus dilalui. Semakin panjang perjalanan yang telah ditempuh, maka semakin jauh kita telah bepergian. Semakin jauh kita bepergian, maka pengalaman yang kita lalui pun semakin banyak. Bahkan setelah satu masalah usai, lahir kembali masalah lainnya. Dan akhirnya, semakin banyak pengalaman akan memberikan semakin banyak pelajaran hidup sebagai bekal masa depan.

Dalam menyusun skripsi ini, empat tahun perkuliahan di Program Studi Teknik Elektro telah penulis lalui. Skripsi bukan tentang menyelesaikan sebuah proyek dan menuliskannya, namun di dalamnya terkandung segala aspek materi dan kebijaksanaan yang telah diasah menjadi lebih runcing selama kurang lebih empat tahun masa perkuliahan di Teknik Elektro. Oleh karena itu, skripsi ini penulis persembahkan bagi segala pihak yang telah menjadi bagian dalam perjalanan kurang lebih empat tahun dalam dunia perkuliahan.

Menyelesaikan tugas akhir atau skripsi ini tentu bukanlah perkara mudah. Sebagai wujud apresiasi, melalui kata pengantar ini, penulis hendak menyampaikan ucapan terima kasih kepada:

1. Bapak Hari Maghfiroh, S.T., M.Eng. selaku Pembimbing I yang setia memberikan dukungan, ide, arahan, bimbingan, dan motivasi selama perkuliahan di Teknik Elektro, khususnya selama mengerjakan tugas akhir ini hingga selesai.

2. Bapak Ir. Chico Hermanu B.A., S.T. ,M.Eng. selaku Pembimbing II yang selalu setia memberikan dukungan, ide, arahan, bimbingan, dan motivasi selama pengerjaan tugas akhir ini.
3. Bapak Dr.Ir. Augustinus Sujono M.T. dan Bapak Feri Adriyanto, Ph.D. selaku Dosen Penguji yang telah memberikan dukungan, ide, arahan, bimbingan, dan motivasi selama pengerjaan tugas akhir ini.
4. Bapak Meiyanto Eko Sulistyo, S.T., M.Eng. selaku Pembimbing Akademik yang selalu tabah memberikan dukungan, bimbingan serta motivasi dalam menjalani perkuliahan di setiap semesternya.
5. Seluruh Dosen Program Studi Teknik Elektro yang telah memberikan ilmu yang bermanfaat, motivasi, dan inspirasi yang luar biasa selama menjalani masa perkuliahan selama kurang lebih 4 tahun ini.
6. Teman-teman Teknik Elektro angkatan 2016 yang tidak bisa disebutkan satu persatu yang telah sama-sama berjuang selama 4 tahun ini atas bantuannya yang luar biasa dan semangat yang diberikan untuk segera mendapatkan gelar Sarjana Teknik ini.
7. Keluarga yang tercinta mama saya Yuliani, bapak saya Suwardi, dan adik saya Reva Pramesly yang selalu memberikan dukungan finansial, motivasi dan semangat serta doa restu sehingga penulisan tugas akhir ini dapat terselesaikan.

Surakarta, 20 Juli 2020

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR SINGKATAN

VTOL	: *Vertical Take Off Landing*
UAV	: *Unmanned Aerial Vehicle*
PID	: *Proportional Integral and Derivative*
LQG	: *Linear Quadratic Gaussian*
LQR	: *Linear Quadratic Regulator*
BLDC	: *Brushless Direct Current*
MIMO	: *Multiple Input Multiple Output*
SISO	: *Single Input Single Output*
MPU	: *Motion Processing Unit*
I2C	: *Inter Integrated Circuit*
DC	: *Direct Current*
PWM	: *Pulse Width Modulation*
PC	: *Personal Computer*
EMF	: *Electromotive Forces*

DAFTAR NOTASI

Kv	: *constant velocity*, atau disebut juga kecepatan konstan motor
V	: Volt, satuan tegangan listrik
A	: Ampere, satuan arus listrik
Rpm	: *revolution per minute* atau disebut juga rotasi per menit
ISE	: *integral squared error,* satuan nilai error

BAB I
PENDAHULUAN

1.1 Latar Belakang

VTOL (*Vertical Take Off Landing*) merupakan salah satu jenis pesawat tanpa awak (*drone*) atau UAV *(Unmanned Aerial Vehicle)* yang perkembangannya sangat pesat. *Drone* memiliki fungsi utama, yaitu melakukan tugas yang terlalu berbahaya untuk manusia, biasanya berhubungan dengan ketinggian atau medan yang sulit dijangkau [1]. Pemanfaatan *drone* diantaranya meliputi penginderaan jauh pada bidang agraria [2], militer [3], industri dan perfilman [4]. Di berbagai negara maju seperti Amerika Serikat sudah memanfaatkan *drone* sebagai kurir untuk pengiriman barang dan makanan. Namun, pemanfaatan *drone* di Indonesia terkendala oleh biaya riset yang cukup tinggi mengenai keandalan performa *drone* tersebut.

Penerbangan *drone* di Indonesia memiliki regulasi yang cukup ketat, berdasarkan Permenhub Nomor 47 Tahun 2016 yaitu mengenai area yang dilarang dilewati seperti kawasan keselamatan operasi penerbangan, kawasan udara terlarang dan kawasan udara terbatas [5]. Juga mengenai batas ketinggian penerbangan tidak lebih dari 150 meter serta izin penerbangan *drone* .

Pada *drone* dengan jenis VTOL membutuhkan motor yang ringan namun dengan kecepatan serta efisiensi yang tinggi, sehingga dapat bermanuver dengan gesit, penggunaan motor BLDC (*Brushless Direct Current*) dapat memenuhi kriteria tersebut. Sesuai namanya, motor BLDC tidak menggunakan sikat untuk pergantian, namun diubah agar beroperasi secara elektronik [6]. Motor BLDC mempunyai banyak kelebihan, seperti : kecepatan yang lebih baik dibandingkan karakteristik torsi; respons dinamis yang tinggi; efisiensi tinggi; masa pengoperasian yang lama; operasi tanpa suara [6]; serta kisaran kecepatan lebih tinggi dibanding dengan motor jenis *brushed* yang menggunakan sikat dan terdapat kontak fisik sehingga cenderung panas saat beroperasi, juga dengan umur penggunaan yang lebih pendek.

Salah satu metode yang dapat digunakan sebagai sistem kendali pada motor BLDC adalah kontroler PID *(Proportional Integral and Derivative)* yang sudah diterapkan dalam berbagai bidang dalam rekayasa karena sistem kontrol kecepatan

yang digunakan dalam kontroler PID konvensional memiliki algoritma yang sederhana, stabil dan sangat andal. Namun, pada kendali PID terdapat kendala yaitu perubahan mendadak pada variasi parameter, sehingga membuat kontroler PID memberikan respon yang buruk [7].

Terdapat metode lain apabila sistem yang digunakan tidak linear, yaitu metode sistem kendali optimal. Ketidakstabilan kecepatan pada motor menyebabkan sifatnya menjadi linear dan non-linear, maka salah satu solusinya menggunakan metode sistem kendali optimal karena metode ini memperhitungkan pemilihan indeks atau kriteria performa dan desain yang dihasilkan pada sistem kendali optimal dalam batas-batas kendali fisik [8]. Sistem kendali optimal menggunakan vektor kontrol sehingga indeks performan diminimumkan atau dimaksimumkan. Secara garis besar teori kendali optimal adalah suatu teori kontrol yang pencarian solusinya didasarkan pada usaha untuk meminimumkan atau memaksimalkan suatu fungsi indeks kinerja [8]. Pada fungsi ini terdiri dari beberapa variabel sistem yang nilainya diminimalkan dengan memberikan matrik bobot yang menyatakan besarnya pembobotan untuk masing-masing variabel sistem tersebut.

Penelitian ini menggunakan metode LQG *(Linear Quadratic Gaussian)* yang merupakan kendali LQR *(Linear Quadratic Regulator)* yang dikombinasikan dengan *Kalman Filter*. LQR bekerja dengan MIMO *(Multiple Input Multiple Output)* yang merupakan kelebihan dibandingkan dengan metode kendali PID yang hanya dapat memproses satu input dan satu output (SISO). Kalman Filter bekerja sebagai estimasi optimal dengan penambahan white noise sehingga pengukuran baru dapat langsung diproses dan mengurangi nilai *mean square error* yang diestimasi.

Pada penelitian ini akan membahas tentang sistem kendali motor BLDC dengan membandingkan metode PID dan LQG sehingga dapat menentukan metode yang akurat dalam pengendalian motor BLDC dalam pengaplikasiannya pada VTOL untuk pengendalian posisi. Dalam penelitian ini menggunakan mikrokontroler Arduino yang dioperasikan menggunakan *data logger* sehingga hasil dapat diamati secara lengkap dan didokumentasikan secara langsung.

1.2 Rumusan Masalah

Pada penelitian ini dirumuskan masalah untuk membandingkan kinerja pada sistem kendali posisi dengan metode PID dan LQG pada VTOL.

1. Bagaimana merancang sistem kendali posisi berbasis mikrokontroler Arduino pada VTOL menggunakan motor listrik BLDC dengan metode PID dan LQG ?
2. Bagaimana performa dari masing – masing metode PID dan LQG terhadap kebutuhan pengaplikasian pada VTOL ?

Dalam penelitian ini, permasalahan akan dibatasi pada perancangan sistem berjenis *Partial VTOL Single Axis System* sehingga hanya menggunakan satu motor BLDC dan satu sumbu sudut.

1.3 Tujuan

Dengan mempertimbangkan hal-hal yang sudah dipaparkan sebelumnya, penelitian ini menitikberatkan pada aspek analisis dan algoritma pada metode PID dan LQG. Adapun tujuan dari membuat penelitian ini adalah untuk :

1. Merancang sistem kendali yang dapat diaplikasikan pada VTOL sebagai metode penyeimbang posisi.
2. Merancang kendali posisi dengan motor BLDC dengan metode *optimal control* (LQG).
3. Membandingkan metode antara PID dan LQG pada sistem kendali posisi pada VTOL, sehingga diperoleh metode kendali yang tepat.

1.4 Manfaat

Manfaat yang diharapkan dari penelitian ini yaitu dapat membuat sistem kendali posisi pada VTOL yang handal. Dengan demikian metode kendali yang tepat akan mempersingkat waktu akselerasi motor untuk mencapai *setpoint* yang diinginkan sehingga dapat diaplikasikan pada VTOL dan dapat menggantikan metode kendali konvensional.

1.5 Sistematika Penulisan

Sistematika penulisan yang digunakan dalam tugas akhir ini agar mudah untuk dipahami dengan rangkaian sebagai berikut.

BAB I PENDAHULUAN

Menguraikan tentang latar belakang masalah, perumusan masalah, tujuan penelitian, manfaat dan luaran penelitian, dan sistematika penulisan.

BAB II TINJAUAN PUSTAKA

Menguraikan tinjauan pustaka dan masalah mengenai pesawat VTOL, Motor BLDC, metode kendali optimal, dan juga menjelaskan tentang landasan teori yang berhubungan dengan penelitian.

BAB III METODOLOGI PENELITIAN

Menguraikan tahap – tahap dalam perancangan kendali posisi dengan LQG berdasarkan diagram alir proses penelitian.

BAB IV HASIL DAN PEMBAHASAN

Berisikan implementasi, data pengujian dan perbandingan baik dalam bentuk data tertulis dan grafik beserta analisis.

BAB V PENUTUP

Menjelaskan tentang kesimpulan dan saran dari penelitian ini.

DAFTAR PUSTAKA

Berisikan tentang sumber-sumber pada referensi yang digunakan dalam menyusun skripsi ini.

BAB II
TINJAUAN PUSTAKA

2.1 Penelitian Sebelumnya

Penelitian yang telah dijalankan sebelumnya dalam topik kendali LQG telah dirangkum dalam Tabel 2.1. Adapun perbedaan dan pembaruan penelitian ini telah dimuat pula dalam tabel yang sama

Tabel 2.1 Penelitian Sebelumnya

No	Pengarang	Judul	Metoda	Hasil
1	A. Dharmawan, *et al* [9]	Translation Movement Stability Control of Quad Tiltrotor Using LQR and LQG	Metode yang digunakan pada pengendalian Quad Tilrotor yaitu dengan LQR (Linear Quadratic Regulator) dan LQG (Linear Quadratic Gaussian).	Dihasilkan redaman rotasi saat translasi, dengan pengaturan sudut kemiringan dan kecepatan masing-masing rotor yang disebabkan oleh proses transisi yang melibatkan perubahan sudut tilt-rotor.
2	B. Kurkcu, *et al* [10]	LQG/LTR Position Control of an BLDC Motor with Experimental Validation	Metode yang digunakan yaitu LQG yang disimulasikan menggunakan Matlab dengan motor 171 W.	Pada penelitian ini dihasilkan kestabilan dalam kecepatan motor, yang mana bergantung dari nilai PWM yang dihasilkan.
3	A. Maddi, *et al* [11]	Using Linear Quadratic Gaussian Optimal Control for Lateral Motion of Aircraft	Metode yang digunakan yaitu kendali LQG dengan Kalman Filter sehingga dapat mengontrol manuver dari pesawat terbang CESSNA-182.	Pada penelitian ini hasil dimodelkan dengan bantuan Kalman Filter sehingga mampu mengurangi steady shift error.

4	R. Eide, *et al* [12]	LQG Control Design for Balancing an Inverted Pendulum Mobile Robot	Metode yang digunakan untuk merealisasikan *inverted pendulum* ini yaitu LQG dengan memanfaatkan Matlab Simulink untuk simulasinya.	Penelitian ini berhasil untuk mensimulasikan reaksi robot terhadap *pendulum* sehingga dapat membuat posisi tegak.
5	V. Y. Vrolov, *et al* [6]	Development of the Sensorless Control System BLDC Motor	Menggunakan FOC (Field Oriented Control) dalam mengendalikan motor BLDC, yang mana cara kerjanya cukup mirip dengan kendali PID.	Sistem berhasil mempertahankan kecepatan yang ditentukan. Pengambilan sampel loop dan kecepatan kontrol arus dapat mengurangi jumlah komputasi CPU tanpa memiliki dampak signifikan pada kualitas manajemen.
6	M. A. Shamseldin, *et al* [7]	Speed Control of BLDC Motor By Using PID Control and Self-tuning Fuzzy PID Controller	Menggunakan tiga metode, yaitu PID, Genetic Algorithm dan Fuzzy PID.	Hasil simulasi menunjukkan bahwa PID fuzzy yang diusulkan memiliki kinerja yang lebih baik dibandingkan dengan dua teknik lainnya.
7	M.S. Mazinder, *et al* [13]	Automatic Control System for The BLDC Motor A2212/13t using Arduino Uno Platform	Metode pengujian menggunakan variasi sinyal PWM terhadap ESC sehingga kecepatan motor yang sesuai dapat dihasilkan.	Hasil pengujian pada kendali motor BLDC dapat proporsional terhadap tegangan yang diberikan.

Pada penelitian ini, penulis merancang sistem kendali posisi dengan metode LQG, namun terdapat beberapa perbedaan dibanding penelitian sebelumnya, yaitu dengan melakukan simulasi terlebih dahulu untuk mendapatkan nilai penguatan yang sesuai lalu dilanjutkan dengan implementasi pada mikrokontroler sehingga dapat meminimalisir biaya dan waktu pengujian. Serta implementasi langsung metode kendali LQG pada mikrokontroler.

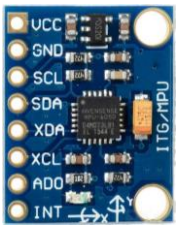
2.2 VTOL (Vertical Take-Off Landing)

Pesawat tanpa awak terdapat berbagai macam jenis, salah satunya yaitu VTOL dimana baling – baling beroperasi pada arah vertikal sehingga perlu sudut kemiringan dan kecepatan tertentu untuk manuver ke arah yang diinginkan [2]. Pesawat jenis ini memerlukan motor yang ringan namun memiliki torsi dan kecepatan tinggi. Pesawat jenis VTOL umumnya juga dikenal sebagai *quadcopter* seperti pada Gambar 2.1, dan juga helikopter. Pada penelitian ini merupakan pengujian pada *partial* VTOL sehingga hanya menggunakan satu motor dalam pengujiannya.

Gambar 2.1 Pesawat tanpa awak jenis VTOL [2]

2.3 Sensor MPU-6050

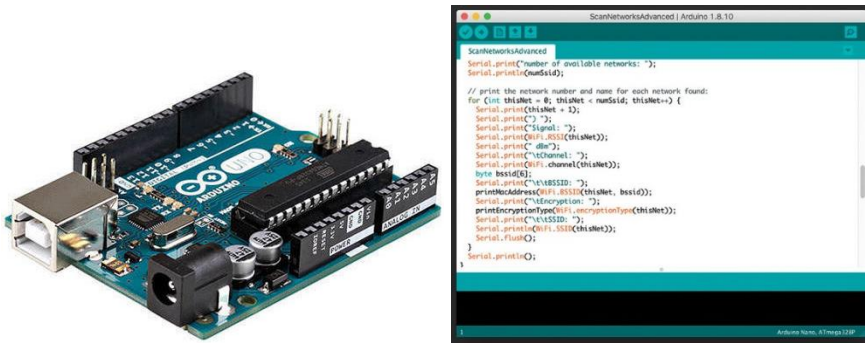
Sensor MPU *(Motion Processing Unit)*-6050 merupakan modul yang terdiri dari *accelerometer* dan *gyroscope* untuk mendeteksi kemiringan suatu benda [14]. Modul ini seperti pada Gambar 2.2 mempunyai 6-*axis* dengan penambahan regulator tegangan sehingga dapat beroperasi dengan tegangan DC 3-5 V dan mempunyai *interface* I2C sehingga dapat dengan mudah dalam koneksinya dengan mikrokontroler. Nilai keluaran sensor ini berupa nilai pembacaan sudut dengan axis X, Y dan Z [14].



Gambar 2.2 Sensor *accelerometer* MPU6050 [14]

2.4 Arduino UNO R3

Arduino UNO R3 adalah sistem mikrokontroler berbasis open-source yang berbasis IC ATmega328P dengan 14 pin *input/output* digital (6 di antaranya dapat digunakan sebagai *output* PWM), dan 6 input analog serta *interface* I2C [13]. Mikrokontroler ini dapat beroperasi dengan tegangan 5 VDC melalui pin yang tersedia maupun dengan kabel USB dan 12 VDC melalui adaptor jack seperti pada Gambar 2.3.



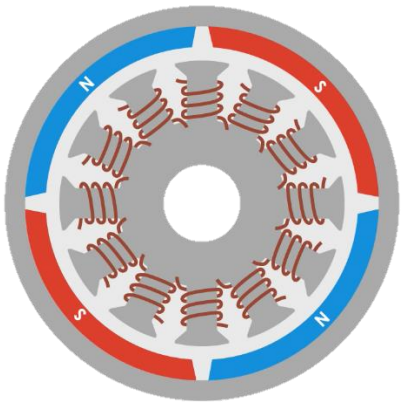
Gambar 2.3 Arduino UNO R3 dan Arduino IDE [13]

Arduino Uno beroperasi menggunakan bahasa pemrograman C dengan Arduino IDE (Integrated Development Environment) sebagai perangkat lunaknya yang digunakan untuk *upload* dan monitor hasil dari pembacaan serial komunikasi. Arduino IDE juga terkoneksi dengan *database* berbagai *library* sehingga dapat mendapatkan pembaruan secara berkala.

2.5 Motor BLDC

Motor BLDC merupakan salah satu jenis motor listrik yang memiliki sistem kontrol secara elektris dan memiliki karakteristik kecepatan, torsi, respon dinamis serta tingkat efisiensi yang lebih baik dibandingkan dengan motor *brushed* (dengan sikat) [6]. Namun, motor BLDC membutuhkan kontroler untuk pengoperasiannya.

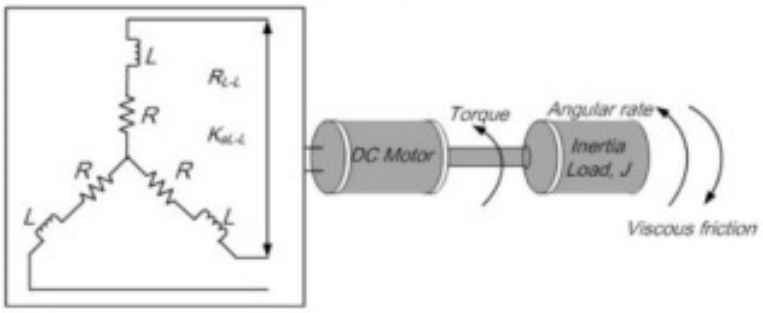
Pada penelitian ini menggunakan motor BLDC jenis *outrunner* (bagian luar yang berputar) dengan konstruksi motor seperti pada Gambar 2.4.



Gambar 2.4 Motor BLDC *Outrunner* [6]

Prinsip kerja dari motor BLDC yaitu adanya gaya tarik - menarik dan tolak – menolak antara magnet permanen pada *rotor* dan elektromagnet pada *stator* (setelah diberi catu daya DC), kutub magnet akan saling tolak menolak dengan kutub yang sejenis begitupun sebaliknya akan saling tarik menarik jika magnetnya berlawanan kutub. Pada penelitian ini menggunakan motor BLDC A2212/6T 2200Kv dengan catu daya baterai Li-Po 11,1 V.

2.5.1 Permodelan Motor BLDC



Gambar 2.5 Model motor BLDC [18]

Motor BLDC dapat dioperasikan secara *sensorless* (tanpa hall sensor) dengan memanfaatkan sinyal back EMF *(electromotive forces)* yang mempunyai gelombang berbentuk trapesium seperti pada Gambar 2.6. Kecepatan motor BLDC diatur melalui tegangan 3 fasa dengan variasi sinyal PWM. Persamaan untuk motor BLDC :

$$V_{app}(t) = L\frac{di(t)}{dt} + R\ i(t) + V_{emf}(t) \quad (2.1)$$

$$V_{emf} = k_b\ \omega(t) \quad (2.2)$$

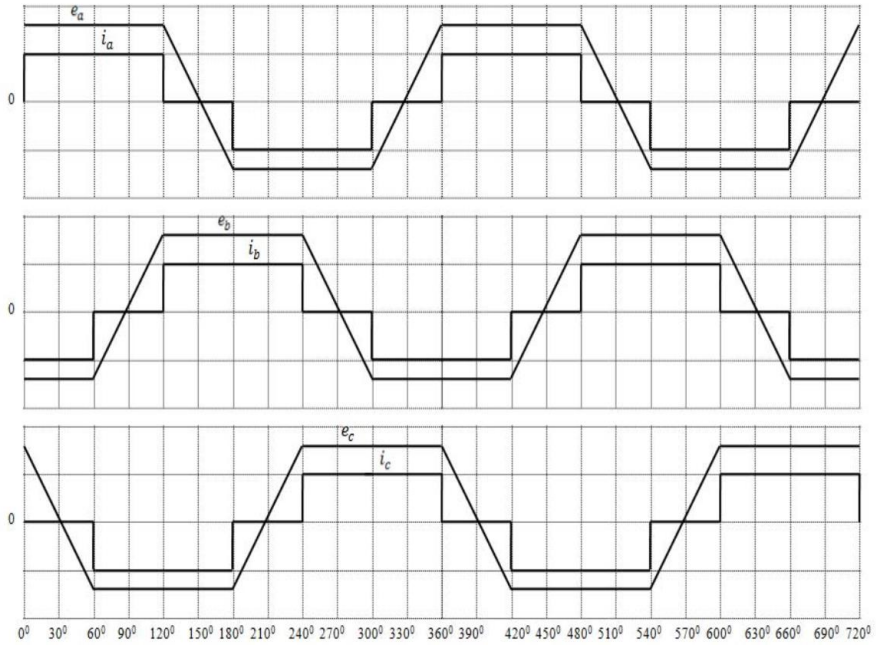
$$T(t) = K_t\ i(t) \quad (2.3)$$

$$T(t) = J\frac{d\omega(t)}{dt} + D\ \omega(t) \quad (2.4)$$

$V_{app}(t)$ merupakan tegangan motor, $\omega(t)$ adalah kecepatan motor, V_{emf} adalah tegangan back EMF, $i(t)$ = arus yang mengalir pada motor, L merupakan induktansi pada stator, T adalah torsi motor, D adalah koefisien viskositas, J adalah momen inersia, k_b adalah konstanta back EMF, serta k_t merupakan konstanta torsi [18]. Sehingga motor BLDC dapat dimodelkan dengan fungsi alih seperti pada persamaan 2.5:

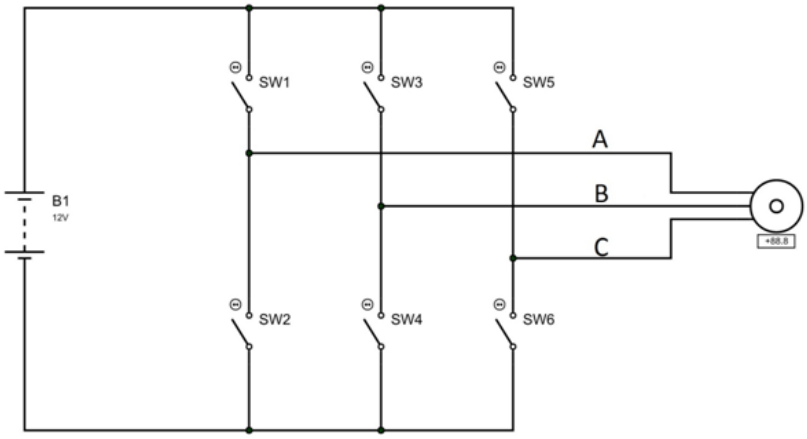
$$G(s) = \frac{\omega(s)}{V(s)} = \frac{k_t}{L\ J\ s^2+(LD+RJ)s+\ k_t k_b} \quad (2.5)$$

Pada persamaan 2.5 dijelaskan bahwa dalam pemodelan kecepatan motor BLDC diperlukan parameter yang telah disebutkan sesuai dengan spesifikasi motor yang akan digunakan untuk penelitian.



Gambar 2.6 Sinyal trapesium pada motor BLDC

2.6 ESC *(Electronic Speed Controller)*



Gambar 2.7 Sistem Kerja *Electronic Speed Controller*

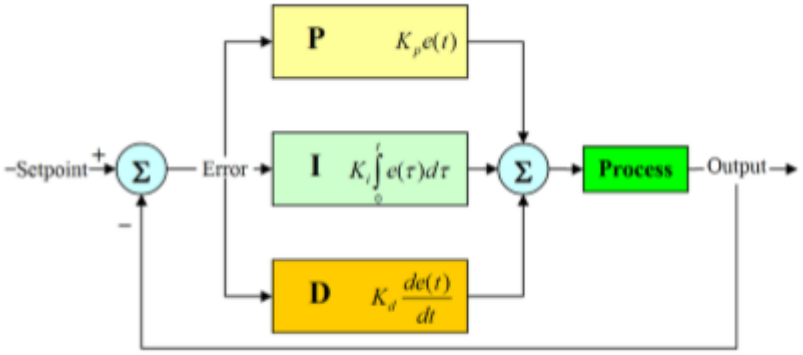
Dikarenakan motor BLDC tidak memiliki *brush* maka diperlukan kontroler tambahan untuk mengatur aliran arus pada stator. Cara kerja dari ESC yaitu dengan menerima perintah berupa sinyal PWM [13] kemudian diterjemahkan untuk melakukan *switching* secara bergantian sehingga masing – masing fasa A,B dan C mendapat tegangan dan motor BLDC akan berputar seperti pada Gambar 2.7, variasi kecepatan yang dihasilkan sesuai dengan sinyal PWM yang diberikan. Pada penelitian ini menggunakan ESC 30A yang beroperasi secara *sensorless*.

Tabel 2.2 Langkah switching ESC

Step	1	2	3	4	5	6
Positif	A	A	B	B	C	C
Ground	B	C	C	A	A	B
Terbuka	C	B	A	C	B	A

2.7 Kendali PID *(Proportional Integral and Derivative)*

Kendali PID adalah sistem pengendali yang umum digunakan pada industri karena mudah digunakan dan paling sederhana [7].



Gambar 2.8 Blok Diagram Kontroller PID [7]

Persamaan nilai keluaran dari kontrol PID, dirumuskan sebagai:

$$u(t) = K_p e(t) + K_i \int_o^t e(t)dt + K_d \frac{d}{dt} e(t) \quad (2.6)$$

Persamaan (2.6) menjelaskan bahwa nilai keluaran u(t), merupakan jumlah dari gain *proportional* (Kp) yang menyebabkan perubahan sinyal keluar berdasarkan perubahan sinyal input, gain *integral* (Ki) yang menghilangkan *steady-state error* menjadi nol, dan gain *derivative* (Kd) yang menggunakan kecepatan perubahan sinyal sebagai parameter kontrol. Masing-masing dipengaruhi oleh error (e) dalam selang waktu (t) tertentu seperti pada Gambar 2.8.

2.8 Sistem Kendali Optimal

Kendali optimal adalah proses menentukan kontrol pada sistem dinamis selama periode waktu tertentu untuk meminimalkan indeks kinerja. Kendali optimal telah diaplikasikan pada berbagai bidang, termasuk *aerospace*, kontrol proses, robotika, bioteknologi, ekonomi, keuangan, dan ilmu manajemen, dan terus menjadi area penelitian aktif dalam teori kontrol [15]. Saat ini dengan teknologi komputer yang lebih baik maka memungkinkan penerapan teori kontrol yang optimal dan metode untuk banyak masalah kompleks [16].

2.8.1 *Transfer Function* (Fungsi Alih)

Fungsi alih merupakan perbandingan antara transformasi laplace dengan keluaran (fungsi respon y) dan transformasi laplace masukan (fungsi referensi u) dengan anggapan bahwa semua syarat awal adalah bernilai nol. Contoh persamaan fungsi alih dapat dilihat pada persamaan 2.7.

$$\frac{1.091z^2}{z^2-0.9102z-0.036} \tag{2.7}$$

Fungsi alih mencakup satuan – satuan yang diperlukan untuk merelasikan masukan dengan keluaran. Namun, fungsi alih tidak memberikan informasi mengenai struktur fisik dari sistem, sehingga sistem fisik yang berbeda dapat memiliki fungsi alih yang identik [15]. Pada fungsi alih, pangkat tertinggi dari s penyebut sama dengan orde suku turunan tertinggi dari s tersebut adalah n, maka sistem tersebut disebut "sistem orde ke-n".

2.8.2 *State Space* (Persamaan Ruang Keadaan)

Persamaan ruang keadaan digunakan untuk menganalisa sistem pengendalian optimal, persamaan ini didapatkan dengan konversi dari fungsi alih yang dapat mewakili kerja sistem.

$$\dot{x}(t) = A(t)x(t) + B(t)u(t) \tag{2.8}$$

$$y(t) = C(t)x(t) + D(t)u(t)$$

Pada persamaan ruang keadaan terdapat 4 matriks, yaitu matriks A sebagai matriks sistem, menghubungkan bagaimana keadaan saat ini mempengaruhi perubahan keadaan x '. Matriks B sebagai matriks kontrol, menentukan bagaimana input sistem memengaruhi perubahan status. Matriks C adalah matriks output, menentukan hubungan antara status sistem dan output sistem. Matriks D adalah matriks umpan maju, memungkinkan input sistem untuk mempengaruhi output sistem secara langsung, oleh karena itu untuk sebagian besar sistem, matriks D adalah matriks nol [15].

2.8.3 Keterkendalian dan Keteramatan

Setelah mendapatkan persamaan ruang keadaan, langkah selanjutnya melakukan uji keterkendalian dan keteramatan untuk mencoba apakah nilai ruang keadaan dapat dikendalikan dan diamati [17].

- Keterkendalian (Controllability)

  Suatu sistem dikatakan dapat dikendalikan jika dimungkinkan untuk mendapatkan suatu vektor kendali (u) yang dalam waktu berhingga dapat

membawa sistem tersebut dari suatu kondisi awal x(0) ke kondisi lain x(f) [17].

Agar Sistem dapat dikendalikan maka :

1. Tidak ada kolom yang merupakan kelipatan kolom lainnya.
2. Nilai determinan tidak sama dengan nol.

Perumusan Matriks keterkendalian:

$$[B \mid AB \mid \ldots\ldots \mid A^{(n-1)}B] \qquad (2.9)$$

- Keteramatan (Observability)

  Suatu sistem dikatakan dapat teramati apabila setiap keadaan awal x(0) dapat ditentukan oleh pengamatan y(T) selama periode waktu terhingga [17].

  Agar Sistem dapat teramati maka :

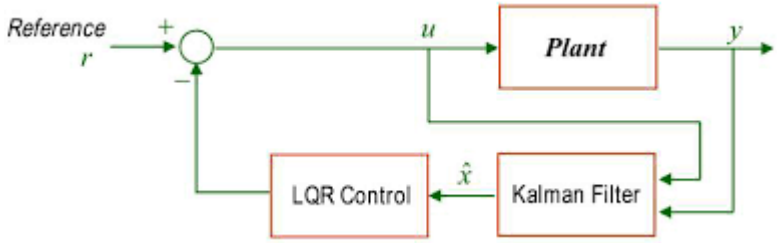
  1. Tidak ada kolom yang merupakan kelipatan kolom lainnya.
  2. Nilai determinan tidak sama dengan nol.

  Perumusan Matriks keteramatan:

$$[C^T \mid A^T C^T \mid \ldots\ldots \mid (A^T)^{n-1} C^T] \qquad (2.10)$$

2.8.4 LQG *(Linear Quadratic Gaussian)*

Linear Quadratic Gaussian adalah sistem kendali optimal yang tujuannya menemukan kendali yang meminimalkan fungsi biaya kuadratik pada dinamika sistem linear [9]. Pada LQG terdapat *noise* Gaussian pada sistem serta outputnya. Maka dapat disimpulkan bahwa kendali LQG merupakan kendali LQR yang ditambahkan *noise* Gaussian [11]. Sehingga dalam rangkaian pengujiannya dapat ditambahkan Kalman Filter [11].



Gambar 2.9 Blok diagram LQG

Maka, dari blok diagram pada Gambar 2.9 diperlukan nilai penguatan pada LQR. Pada LQR, nilai penguatan dapat dihitung menggunakan persamaan matriks Riccati (2.11) dengan nilai P definit positif. Matriks Q merupakan matriks yang mempengaruhi performa dan matriks R merupakan matriks yang mempengaruhi penggunaan energi.

$$K_{lqr} = (R + B^T PB)^{-1} B^T PA \tag{2.11}$$

$$P = A^T PA - (A^T PB)(R + B^T PB)^{-1}(B^T PA) + Q$$

2.8.5 Kalman Filter *(Linear Quadratic Estimator)*

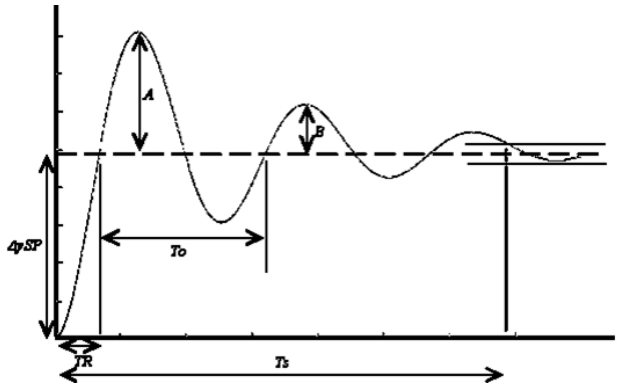
Kalman Filter merupakan model estimasi keadaan yang dapat memperkirakan variable keadaan dengan pengukuran dengan menambahkan *noise*. Jika *noise* bertipe Gaussian, maka variasi kesalahan dapat diminimalisir secara optimal [11]. Filter ini memperkirakan proses dengan menggunakan bentuk kendali umpan balik berupa pengukuran dengan *noise*. Persamaan untuk Kalman filter juga menggunakan persamaan matriks Riccati dikarenakan kendali optimal dan estimasi optimal merupakan masalah ganda matematis, seperti keterkendalian dan keteramatan, sehingga Kalman filter juga dapat ditemukan menggunakan LQR. Pada persamaan 2.12 dapat dilihat untuk perhitungan gain Kf dengan nilai P_e definit positif. Dengan Vn sebagai matriks *noise* dan Vd sebagai matriks *disturbance.*

$$K_f = (V_n + CPC^T)^{-1} CPA^T \tag{2.12}$$

$$P_e = AP_eA^T - (AP_eC^T)(V_n + CP_eC^T)^{-1}(CP_eA^T) + Vd$$

2.9 Kualitas Respons Sistem Kendali

Ada beberapa karakteristik yang dapat digunakan untuk membandingkan kualitas respons sistem kendali



Gambar 2.10 Grafik Respons Sistem [17]

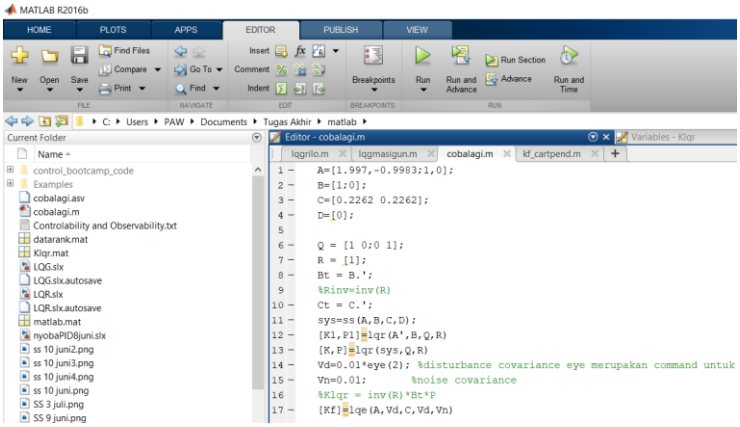
Rise Time merupakan waktu yang diambil untuk respons proses pertama kali mencapai titik acuan, pada Gambar 2.11 ditulis dengan "TR". *Settling Time* merupakan waktu yang dibutuhkan sistem untuk menyesuaikan dalam toleransi di sekitar titik acuan, pada Gambar 2.10 ditulis dengan "Ts". *Overshoot* dan *Undershoot* adalah kuantitas yang relatif untuk respons titik acuan, pada Gambar 2.10 *overshoot* ditulis dengan "A". Sedangkan untuk *undershoot* berada di bawah sumbu x. To merupakan periode osilasi satu siklus, dan B adalah puncak kedua pada respons.

Kualitas respons pada sistem dapat dihitung dengan metode *Integral Squared Error* (ISE), dengan mengkuadratkan error dari waktu ke waktu. ISE dengan nilai minimal merupakan hasil dari respon sistem yang dapat mengatasi *error* dengan cepat [17].

2.10 Perangkat Lunak MATLAB

Matlab merupakan sebuah platform yang dirancang untuk memenuhi kebutuhan komputasi tingkat lanjut yang biasa dilakukan oleh *engineer*, sehingga dapat menampilkan hasil dengan metode komputasi yang beragam dan meminimalisir kesalahan pada pengujian [10]. Pada Matlab terdapat ekstensi Simulink yang dapat digunakan untuk memenuhi kebutuhan simulasi pada sistem, sehingga pengguna

dapat terlebih dahulu menyesuaikan rancangan untuk hasil yang diinginkan dan meminimalisir kesalahan serta biaya pengujian.



Gambar 2.11 Perangkat Lunak Matlab

BAB III
METODOLOGI PENELITIAN

3.1 Waktu dan Tempat Penelitian

Penelitian direncanakan akan dilaksanakan dalam waktu 4 bulan terhitung dari bulan April 2020 hingga Juli 2020 di Laboratorium Instrumentasi dan Kendali Teknik Elektro Universitas Sebelas Maret.

3.2 Alat dan Bahan

Perangkat keras dan perangkat lunak yang digunakan dalam melaksanakan penelitian ini adalah sebagai berikut :

Tabel 3.1 Alat

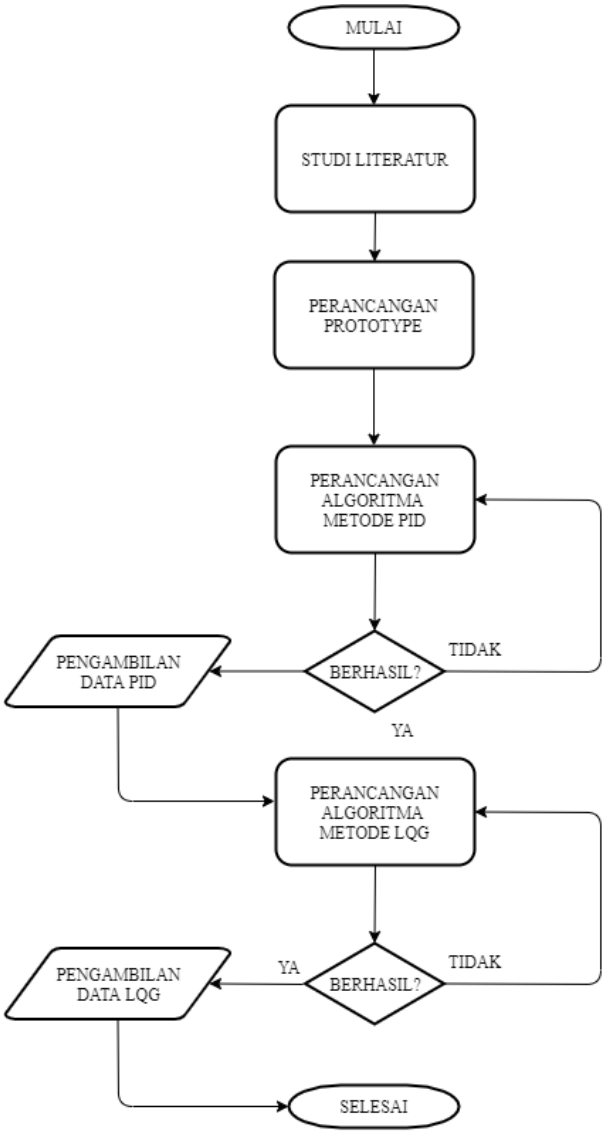
No	Nama Alat / Spesifikasi	Kuantitas	Keterangan
1	Charger iMAX B6AC	1 buah	Sebagai pengisi daya baterai Li-Po
2	Toolset	1 buah	Obeng dan tang lengkap sebagai perkakas pendukung
3	Multimeter	1 buah	Sebagai pengukur tegangan dan arus
4	PC/Laptop	1 buah	Sebagai media upload program dan simulasi
5	Solder	1 buah	Sebagai penghubung rangkaian permanen
6	Tenol	1 buah	Timah untuk menghubungkan komponen

Tabel 3.2 Bahan

No	Nama Alat / Spesifikasi	Kuantitas	Keterangan
1	Motor BLDC A2212/6T 2200Kv	1 buah	Sebagai penggerak
2	Propeller ukuran 5 inch	1 buah	Sebagai baling – baling untuk penerbang
3	Arduino UNO R3	1 buah	Sebagai mikrokontroler untuk aplikasi program LQG
4	Lengan akrilik	1 buah	Sebagai lengan untuk meletakkan motor
5	Sensor MPU-6050	1 buah	Sebagai sensor pembaca input sudut
6	ESC Simonk 30A	1 buah	Sebagai driver motor BLDC
7	Optocoupler PC123	1 buah	Sebagai pemisah ground rangkaian
8	Baterai Li-Po 3S	1 buah	Sebagai catu daya untuk ESC
9	Kabel jumper	20 buah	Sebagai kabel penghubung rangkaian

3.3 Metode Penelitian

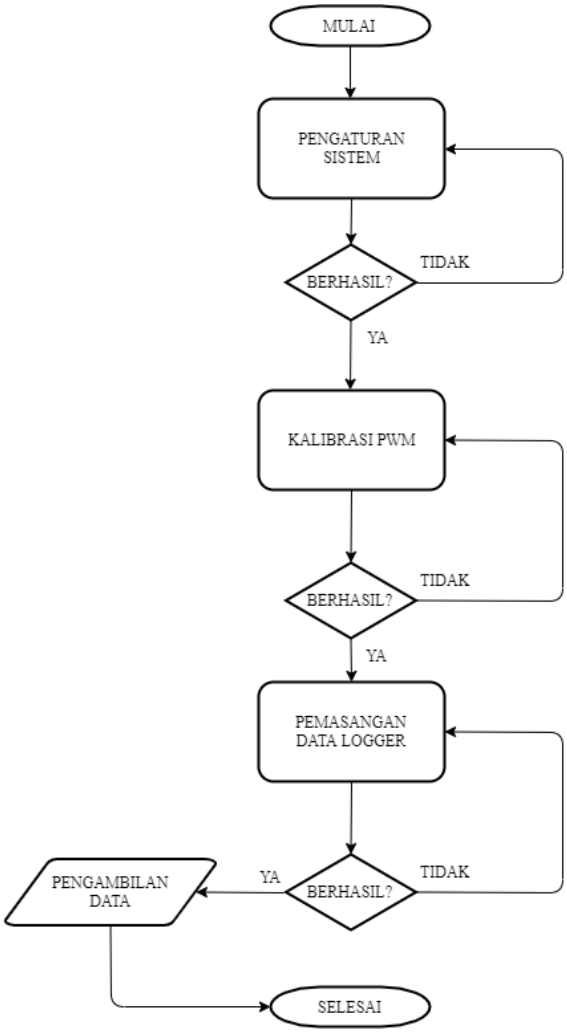
Pada sub-bab ini merupakan langkah langkah penelitian setelah studi literatur. Tahapan – tahapan pada metode penelitian ini diawali dengan perancangan algoritma PWM (*Pulse Width Modulation*) sistem kendali untuk motor BLDC *Sensorless.* Selanjutnya akan dilakukan perancangan algoritma untuk masing – masing metode kendali, serta pengujian performa dari masing – masing metode yaitu PID dan LQG. Diagram alir tahapan penelitian dapat dilihat pada Gambar 3.1.



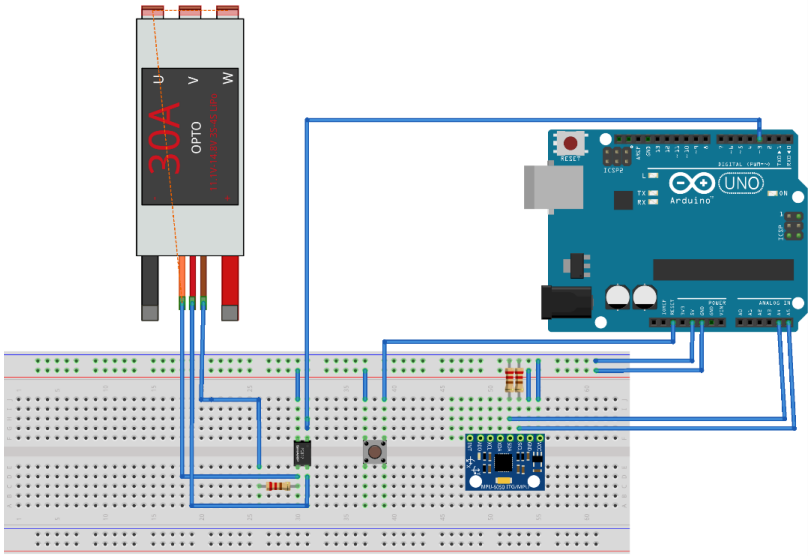
Gambar 3.1 Diagram alir penelitian

3.4 Perancangan *Prototype*

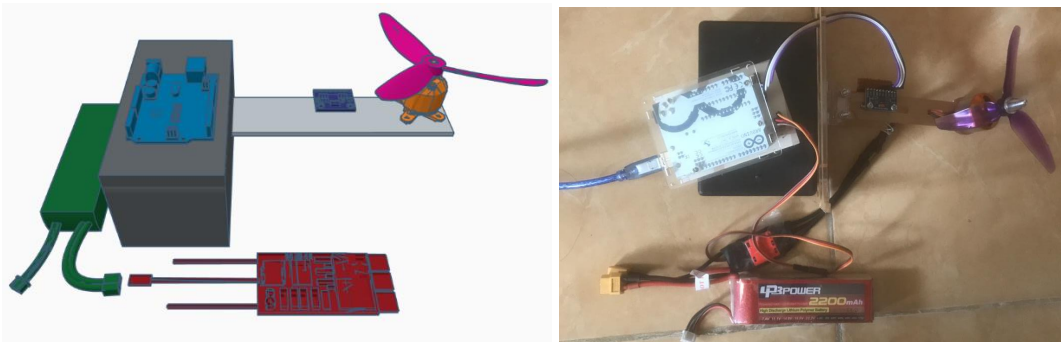
Perancangan *prototype* sistem kendali posisi dimulai dengan dengan pengaturan sistem dan kalibrasi perangkat keras. Pengaturan sistem dan kalibrasi dilakukan dengan cara menjalankan program PWM untuk menggerakan motor BLDC. Selanjutnya dengan menambahkan *data logger* yang terdapat pada modul kendali untuk pengambilan data yang telah diuji. Adapun diagram alir perancangan *prototype* sistem kendali dapat dilihat pada Gambar 3.2 serta rangkaian *prototype* pada Gambar 3.3.



Gambar 3.2 Diagram alir Perancangan *Prototype*



Gambar 3.3 Rangkaian *Prototype* Sistem Kendali



Gambar 3.4 (a) Desain alat dan (b) Hasil Pembuatan *Prototype* Sistem Kendali

Pada Gambar 3.4 merupakan rancangan gambar 3 dimensi dari prototype sistem kendali dengan diagram sirkit sesuai pada Gambar 3.3.

3.4.1 Kalibrasi Sensor MPU-6050

Pembacaan sudut dilakukan dengan menghubungkan koneksi pin pada MPU-6050 ke pin Arduino. Dengan jalur komunikasi I2C yaitu SDA dan SCL pada masing – masing portnya. Nilai pembacaan sensor adalah radian dan akan dirubah menjadi satuan derajat sudut.

```
float rad_to_deg = 180/3.141592654;
```

3.4.2 Kalibrasi ESC

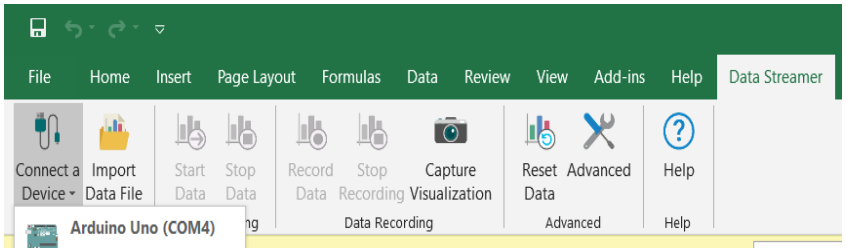
Pada penggunaan ESC menggunakan *library servo* sehingga dapat memudahkan dalam pengiriman sinyal PWM dari Arduino ke ESC. Nilai PWM yang divariasikan bernilai dari 1000 sampai 2000, ESC dihubungkan pada pin 3 Arduino dengan nilai awal 1170 sehingga cukup untuk membuat motor menyala.

```
Servo motor_prop;
double motor=1170;
motor_prop.attach(3);
motor_prop.writeMicroseconds(1000);
```

3.4.3 Pemasangan Data Logger

Pada penelitian ini pengambilan data dilakukan dengan bantuan *addon* dari Microsoft Excel yaitu *Data Streamer* sehingga dapat dengan mudah untuk monitor

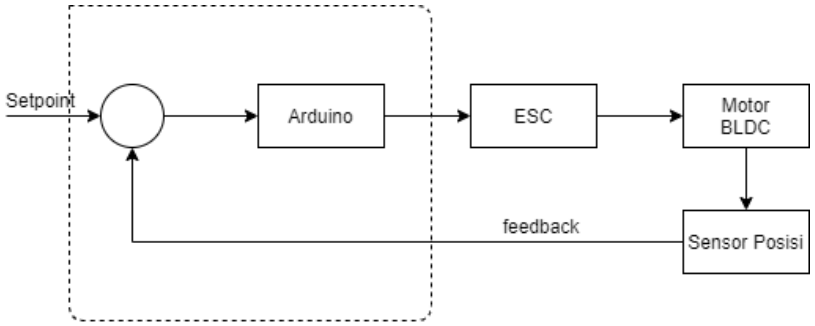
nilai keluaran dari sudut dan PWM untuk proses selanjutnya. Untuk penggunannya dengan mengatur *port* COM pada tab *Data Streamer*.



Gambar 3.5 Microsoft Excel *Data Streamer*

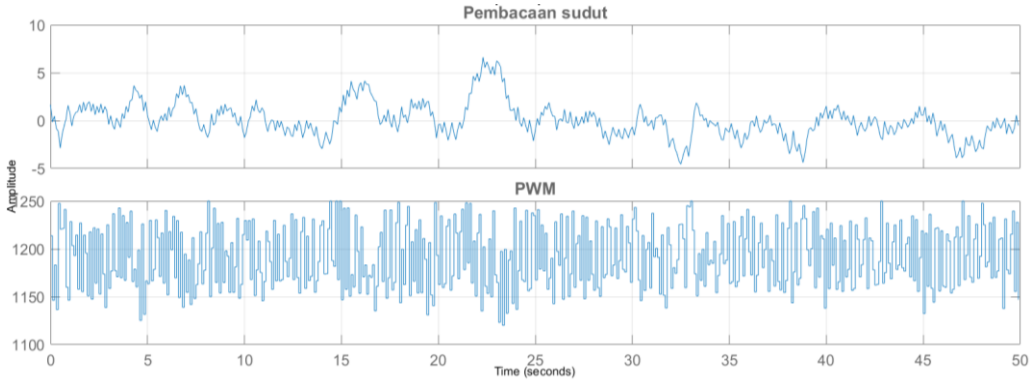
3.4.4 Pemodelan Sistem

Pemodelan adalah upaya untuk menyatakan sistem dari bentuk fisik menjadi bentuk persamaan matematika. Berbagai macam upaya dilakukan dengan menyusun hubungan antara bentuk fisik dari sistem sesungguhnya menggunakan hukum ilmu alam. Bagian terpenting dari model matematis adalah persamaan karakteristik. Persamaan karakteristik sistem menentukan respon sistem tersebut apabila dilakukan simulasi.



Gambar 3.6 Diagram blok sistem kendali

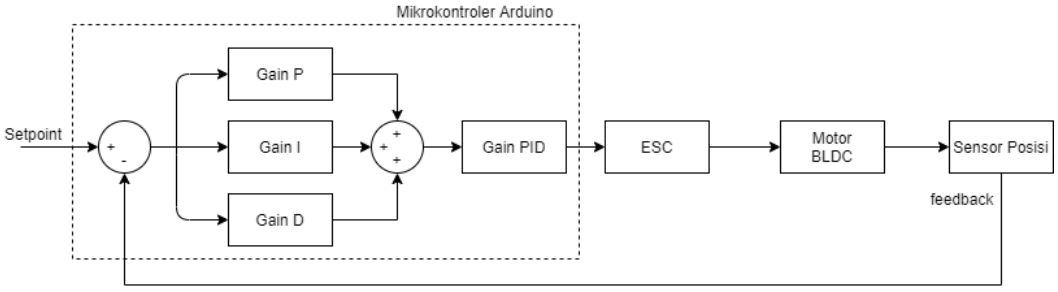
Untuk mendapatkan bentuk persamaan matematika dari plant yang merupakan sensor gyro dan motor BLDC seperti pada Gambar 3.6 dapat menggunakan *System Identification Toolbox* pada Matlab. Fungsi alih dari sensor *gyro* dan motor didapatkan dengan variasi kecepatan motor pada selang waktu tertentu. Hal ini dimaksudkan agar semua kondisi motor dapat terekam. Data yang dimasukkan pada *System Identification Toolbox* adalah data pembacaan sensor dan kecepatan motor. Pada penelitian ini data kecepatan motor diberi label “x” sementara data pembacaan sensor gyro diberi label “y”. Percobaan dilakukan menggunakan sampling time 100 ms. Maka akan didapat data seperti pada Gambar 3.7.



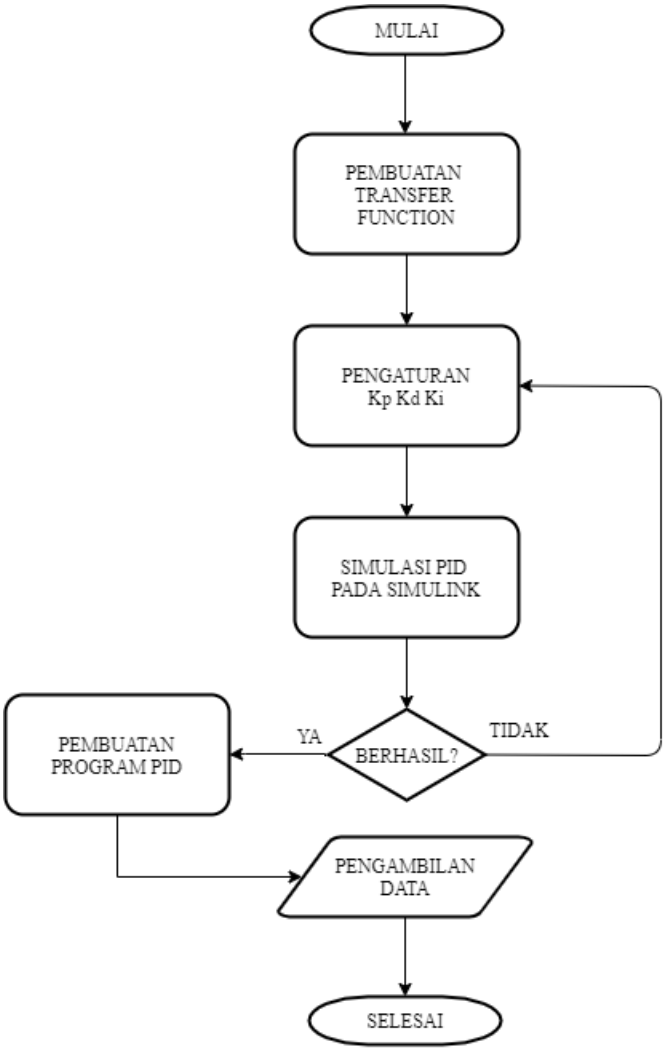
Gambar 3.7 Hasil kalibrasi PWM

3.5 Perancangan Kendali Metode PID

Sistem kendali posisi menggunakan metode PID yang dirancang sesuai dengan diagram blok pada Gambar 3.8. Perancangan kendali PID pada sistem kendali dimulai dengan pembuatan fungsi alih, penentuan parameter PID, yaitu Kp, Kd dan Ki. Parameter tersebut akan mempengaruhi besaran dari kecepatan *rise time* dan *error steady state* yang dihasilkan. Selanjutnya dilakukan simulasi pada Simulink menggunakan parameter PID yang telah ditentukan, apabila hasil *rise time* dan *steady state* tidak sesuai maka dapat mengulangi penentuan parameter PID. Lalu pembuatan kode program yang dapat menghasilkan pulsa PWM untuk menggerakkan motor BLDC. Selanjutnya dilakukan pengambilan data menggunakan *data logger*. Adapun diagram alir perancangan sistem kendali dapat dilihat pada Gambar 3.9.



Gambar 3.8 Diagram blok sistem kendali metode PID



Gambar 3.9 Diagram alir Perancangan Kendali Metode PID

3.5.1 Pembuatan *Transfer Function* (Fungsi Alih)

Untuk mendapatkan fungsi alih dari plant yang merupakan sensor gyro dan motor BLDC dapat menggunakan *System Identification Toolbox* pada Matlab. Fungsi alih didapatkan dari hasil kalibrasi PWM.

- *Poles* dan *Zeros*

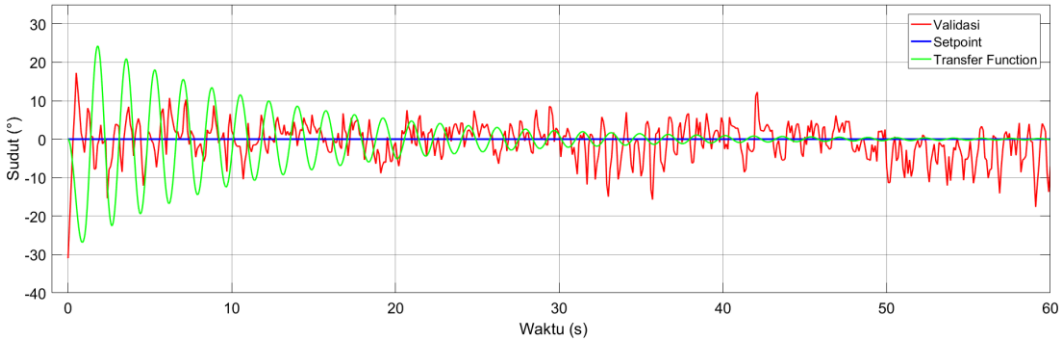
$$H(s) = \frac{N(s)}{D(s)} \tag{3.1}$$

Zeros merupakan akar dari N(s) yang juga disebut sebagai *numerator* dari fungsi alih, akar didapatkan dengan $N(s) = 0$. *Poles* merupakan akar dari D(s) yang juga disebut sebagai *numerator* dari fungsi alih, akar didapatkan dengan

$D(s) = 0$. Maka, jumlah *zeros* tidak boleh melebihi jumlah *poles*. Pada penelitian ini penulis menggunakan *poles* dan *zeros* masing – masing bernilai 2 untuk memodelkan sistem pada Matlab *System Identification Toolbox* sehingga didapat fungsi alih sebagai berikut.

$$\frac{0.2262z+0.2262}{z^2-1.997z+0.9983} \tag{3.2}$$

3.5.2 Validasi *Transfer Function*

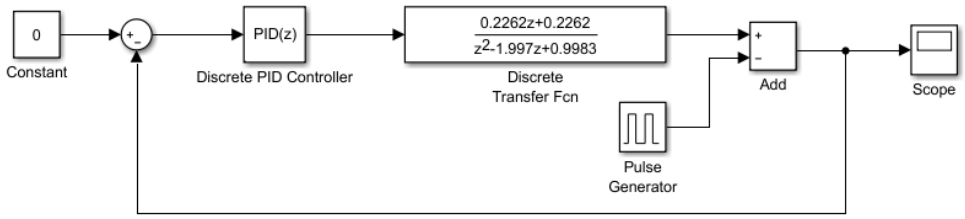


Gambar 3.10 Pengujian Validasi *Transfer Function*

Pada pengujian setelah didapatkan nilai dari fungsi alih dari sistem yang dirancang, maka perlu dilakukan validasi atas model matematis yang didapat dengan membandingkan respon dari nilai fungsi alih dan alat. Pada Gambar 3.10 dapat dilihat bahwa hasil validasi menunjukan kemiripan dari respon yang dihasilkan sehingga dapat melanjutkan langkah pengujian selanjutnya.

3.5.3 Simulasi PID pada Matlab

Hasil Fungsi alih yang didapatkan selanjutnya akan digunakan untuk simulasi pada sistem kendali PID. Pada penelitian ini nilai Kp, Kd dan Ki ditentukan dengan *trial and error* sehingga berhasil untuk mencapai tingkat kestabilan yang diinginkan.

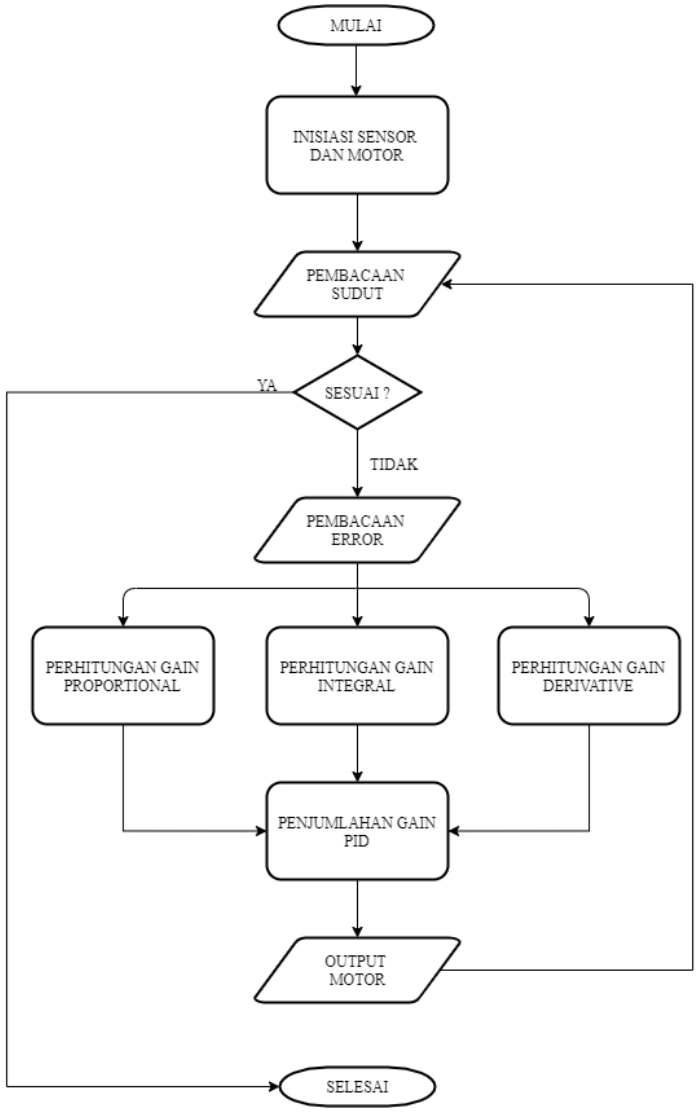


Gambar 3.11 Simulasi sistem kendali PID

Nilai yang ditentukan yaitu Kp = 0.02555 Ki = 0.04024 dan Kd = 0.00405. Pada rangkaian sistem seperti pada Gambar 3.11, ditambahkan sinyal disturbance (pengganggu) untuk membuktikan hasil kestabilan pada sistem.

3.5.4 Pembuatan Program PID

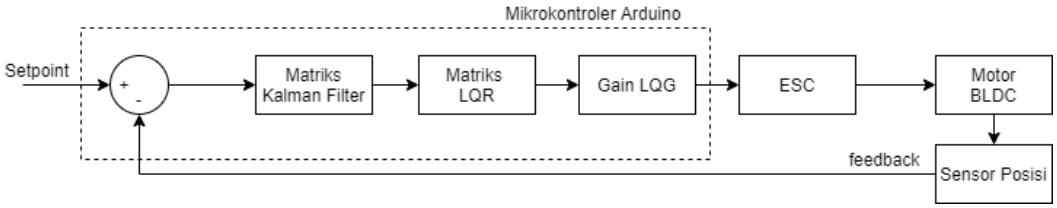
Pada penelitian ini, program PID dibuat untuk diimpelementasikan pada Arduino, sehingga program dibuat pada software Arduino IDE dengan menggunakan bahasa pemrograman C. Alur program untuk kendali posisi menggunakan PID terdiri dari inisiasi sensor dan motor, pembacaan sudut perhitungan error hingga output motor seperti yang terlihat pada diagram alir pada Gambar 3.12.



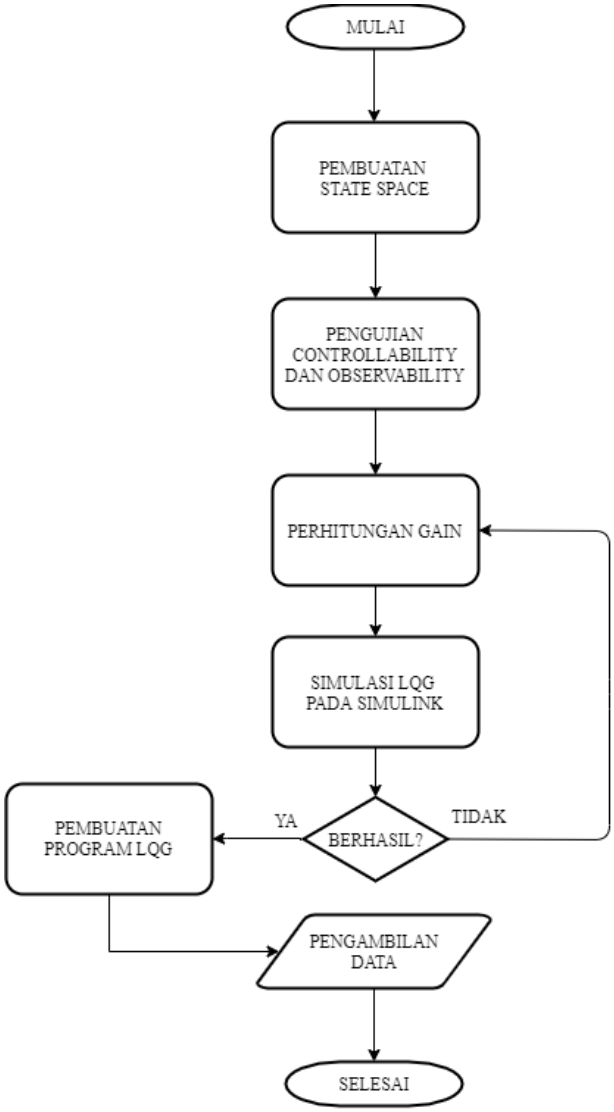
Gambar 3.12 Diagram alir program metode PID untuk kendali posisi

3.6 Perancangan Kendali Metode LQG

Sistem kendali posisi menggunakan metode LQG yang dirancang sesuai dengan diagram blok pada Gambar 3.13. Perancangan kendali LQG pada sistem kendali posisi dimulai dengan pembuatan persamaan ruang keadaan, pengujian keterkendalian dan keteramatan, lalu dilakukan perhitungan *gain*, dilanjutkan dengan simulasi pada Simulink menggunakan nilai *gain* yang telah didapatkan. Selanjutnya dilakukan pembuatan kode program dengan menggunakan parameter *gain* yang telah didapatkan sebelumnya, program tersebut nantinya akan menghasilkan pulsa PWM untuk menggerakkan motor BLDC. Adapun diagram alir perancangan sistem kendali dapat dilihat pada Gambar 3.14.



Gambar 3.13 Diagram blok sistem kendali metode LQG



Gambar 3.14 Diagram alir Perancangan Kendali Metode LQG

3.6.1 Permodelan Matematis Sistem Kendali

Pemodelan adalah upaya untuk menyatakan sistem dari bentuk fisik menjadi bentuk persamaan matematika. Berbagai macam upaya dilakukan dengan menyusun hubungan antara bentuk fisik dari sistem sesungguhnya menggunakan hukum ilmu alam. Bagian terpenting dari model matematis adalah persamaan karakteristik. Persamaan karakteristik sistem menentukan respon sistem tersebut apabila dilakukan simulasi.

Dalam penurunan model matematis dari sistem, terdapat tiga pendekatan yang dapat digunakan, yaitu :

- Persamaan diferensial
- Penekatan fungsi alih (transfer function)
- Pendekatan ruang-keadaan (state-space)

Pendekatan yang digunakan oleh penulis adalah pendekatan menggunakan ruang-keadaan. Pemilihan ini dikarenakan pendekatan ruang-keaaan akan menghasilkan kendali optimal dibandingkan pendekatan lain.

Langkah-langkah yang dilakukan untuk mendapatkan model matematis sistem LQR antara lain adalah sebagai berikut:

1. Menentukan sistem yang akan diteliti dan komponen - komponen yang menyusun sistem tersebut
2. Setelah diketahui komponen-komponen penyusun sistem beserta parameter-parameternya, maka sistem dapat dimodelkan menjadi sebuah persamaan matematis. Dengan memasukkan data-data sistem pada persamaan matematis dengan bantuan Matlab *System Identification Toolbox* yang dimana nilai *output* sensor *gyro* dan kecepatan motor merupakan parameter yang digunakan.
3. Melakukan tuning pada Matlab Identification Toolbox dan mendapatkan nilai fungsi alih terhadap persamaan sebagai berikut :

$$\frac{0.2262z+0.2262}{z^2-1.997z+0.9983} \tag{3.3}$$

4. Maka selanjutnya fungsi alih sistem, akan dirubah menjadi persamaan ruang keadaan (state space) dengan merujuk persamaan pada sub bab Persamaan Ruang Keadaaan maka didapatkan persamaan :

$$\dot{x} = \begin{pmatrix} 0 & 1 \\ -0.9983 & 1.997 \end{pmatrix} x + \begin{pmatrix} 0 \\ 1 \end{pmatrix} u \tag{3.4}$$

dan

$$y = (0.2262\ 0.2262)\, x + (0)u \tag{3.5}$$

Didapat konversi kedalam bentuk *state space* sebagai berikut :

Transfer Function Model Kendali Posisi	Model dalam bentuk State Space
$\frac{0.2262z + 0.2262}{z^2 - 1.997z + 0.9983}$	$\mathrm{A} = \begin{bmatrix} 1.997 & -0.9983 \\ 1 & 0 \end{bmatrix}$ $B = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ $C = [0.2262 \quad 0.2262]$ $\mathrm{D} = 0$

3.6.2 Pengujian Keterkendalian *(Controllability)* dan Keteramatan *(Observability)*

Pada pengujian *Controllability* dan *Observability* peneliti menggunakan bantuan software Matlab, *controllability* merupakan suatu pemeriksaan dimana suatu sistem dikatakan terkendali apabila jika Matriks elemen controllability (ctr) yang berukuran n x n memiliki rank dengan nilai n. Atau jika determinan dari Matriks controllability adalah tidak lah nol. Sedangkan *observability* merupakan ukuran seberapa baik keadaan internal suatu sistem dapat disimpulkan dari pengetahuan tentang output eksternalnya. Matriks elemen *observability* (obsr) dan Matriks elemen *controllability* (ctr) mempunyai rank matriks-n. Rank matriks adalah jumlah baris/kolom yang bebas.

Pada pengujian *controllability* matriks A dengan dimensi nxn harus dipasangkan dengan matriks B dengan dimensi nxm, maka disini peneliti menggunakan matriks A 2x2 dan B 2x1 sehingga sesuai dengan syarat pengujian. Pada pengujian *observability* matriks A dengan dimensi nxn harus dipasangkan dengan matriks C dengan dimensi pxn, maka disini peneliti menggunakan matriks A 2x2 dan C 1x2 sehingga sesuai dengan syarat pengujian. Dengan menggunakan software perintah matlab dapat diperoleh perintah sebagai berikut `rank(ctrb(A,B)` dan `rank(obsv(A,C)`,maka didapatkan perhitungan *controllability* dan *observability* seperti pada Gambar 3.15.

```
>> Qc = ctrb(A,B);
rankQc = rank (Qc);
disp ('Matriks controllable Qc = ');
disp (Qc);
if(rankQc == rank(A))
disp ('Sistem dapat dikendalikan. ');
else
disp ('Sistem tidak dapat dikendalikan. ');
end
Matriks controllable Qc =
    1.0000    1.9997
         0    1.0000

Sistem dapat dikendalikan.
```

```
>> Qb = obsv(A,C);
rankQb = rank (Qb);
disp ('Matriks observable Qb = ');
disp (Qb);
if(rankQc == rank(A))
disp ('Sistem dapat diamati. ');
else
disp ('Sistem tidak dapat diamati. ');
end
Matriks observable Qb =
    0.2262    0.2262
    0.6785   -0.2258

Sistem dapat diamati.
```

Gambar 3.15 Pengujian Controllability dan Observability

Dapat disimpulkan bahwa nilai *state space* yang didapat dapat dikendalikan dan diamati karena memenuhi syarat dari *controllability* dan *observability*.

3.6.3 Perhitungan Gain

Pada tahap ini merupakan tahap dimana persamaan Matematis yang didapat dari bentuk state space dimasukkan kedalam perhitungan *LQR* dan Kalman Filter. Tujuan dari optimasi ini adalah mendapatkan nilai umpan balik (k) optimal, yang mampu meminimumkan cost function J. Kedua perhitungan ini dilakukan dengan jalan memasukkan persamaan Riccati. Sedangkan matriks pembobotan Q dan R ditentukan secara sembarang.

- Pembobotan Matriks Q dan R

Matriks bobot adalah matriks Q dan R. Pemilihan matriks Q dan R dilakukan dengan cara coba-coba (trial and error). Dengan syarat, matriks Q adalah matriks simetris, semidefinit positif dan real ($Q \geq 0$). Matriks Q merupakan matriks berorde 2x2 yang ditulis sebagai persamaan.

$$Q = \begin{bmatrix} q & 0 \\ 0 & q \end{bmatrix} \tag{3.6}$$

Matriks Q adalah matriks diagonal dengan komponen-komponennya q, dan bila diadakan pemisahan akan diperoleh matriks identitas yang dikalikan dengan konstanta q.

Sedangkan matriks R adalah matriks diagonal dengan komponen-komponennya r, dan bila diadakan pemisahan akan diperoleh matriks identitas yang dikalikan dengan konstanta r. matriks R merupakan matrik simetris, definit positif dan real (R > 0). Matriks R merupakan matriks berordo 1×1 yang ditulis sebagai persamaan

$$R = [r] \tag{3.7}$$

Hasil dari pembobotan matriks Q dan R nantinya untuk menghitung besarnya nilai penguatan (gain) optimal k kedalam rumus lqr harus menyelesaikan persamaan Riccati. Sebelum memasuki persamaan riccati, harus memilih matriks bobot Q dan R. Selain itu pedoman untuk pemilihan matriks pembobot Q dan R adalah sebagi berikut:

1. Semakin besar nilai Q, maka akan semakin besar harga elemen penguatan K sehingga mempercepat sistem untuk mencapai keadaan stabil.
2. Semakin besar harga R, maka akan memperkecil harga penguatan K dan memperlambat keadaan tunak.

- Penyelesaian persamaan Riccati LQR

Persamaan riccati digunakan untuk memperoleh matriks P dimana Matriks P merupakan sebuah Matrik Positif definite yang simestris dengan nilai lebih dari 0. Dengan meneyelesaikan Persaamaan Riccati mensubstitusi matriks Q, matriks R, matriks A dan Matriks B ke persamaan riccati maka didapatkan Matriks P. Bentuk umum persamaan riccati adalah sebagai berikut :

$$P = A^T PA - (A^T PB)(R + B^T PB)^{-1}(B^T PA) + Q \tag{3.8}$$

Diketahui bahwa :

$$A = \begin{bmatrix} 1.997 & -0.9983 \\ 1 & 0 \end{bmatrix}$$

$$B = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$R = 1$$

$$P = \begin{bmatrix} P_{11} & P_{12} \\ P_{12} & P_{22} \end{bmatrix}$$

Sehingga apabila matriks A, B, Q dan R disubstitusikan pada persamaan tersebut akan menjadi:

$$\begin{bmatrix} P_{11} & P_{12} \\ P_{12} & P_{22} \end{bmatrix} = \begin{bmatrix} 1.997 & -0.9983 \\ 1 & 0 \end{bmatrix}^T \begin{bmatrix} P_{11} & P_{12} \\ P_{12} & P_{22} \end{bmatrix} \begin{bmatrix} 1.997 & -0.9983 \\ 1 & 0 \end{bmatrix}$$

$$- \left(\begin{bmatrix} 1.997 & -0.9983 \\ 1 & 0 \end{bmatrix}^T \begin{bmatrix} P_{11} & P_{12} \\ P_{12} & P_{22} \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} \right) \left([1] + \begin{bmatrix} 1 \\ 0 \end{bmatrix}^T \begin{bmatrix} P_{11} & P_{12} \\ P_{12} & P_{22} \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} \right)^{-1}$$

$$\left(\begin{bmatrix} 1 \\ 0 \end{bmatrix}^T \begin{bmatrix} P_{11} & P_{12} \\ P_{12} & P_{22} \end{bmatrix} \begin{bmatrix} 1.997 & -0.9983 \\ 1 & 0 \end{bmatrix} \right) + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad (3.9)$$

Maka,

$$P = \begin{bmatrix} 4.8196 & -1.4093 \\ -1.4093 & 1.8254 \end{bmatrix} \quad (3.10)$$

Disubstitusikan pada persamaan

$$K_{lqr} = (R + B^T P B)^{-1} B^T P A \quad (3.11)$$

$$\left([1] + \begin{bmatrix} 1 \\ 0 \end{bmatrix}^T \begin{bmatrix} 4.8196 & -1.4093 \\ -1.4093 & 1.8254 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} \right)^{-1}$$

$$\begin{bmatrix} 1 \\ 0 \end{bmatrix}^T \begin{bmatrix} 4.8196 & -1.4093 \\ -1.4093 & 1.8254 \end{bmatrix} \begin{bmatrix} 1.997 & -0.9983 \\ 1 & 0 \end{bmatrix} \quad (3.12)$$

$$= [1.4116 \quad -0.8267] \quad (3.13)$$

- Penyelesaian persamaan Riccati Kalman Filter

Persamaan riccati digunakan untuk memperoleh matriks penguatan K_f dengan terlebih dahulu mendapatkan matriks P_e. Matriks P_e merupakan sebuah matriks Positif definite yang simestris dengan nilai lebih dari 0. Dengan meneyelesaikan Persaamaan Riccati mensubstitusi matriks V_n, matriks V_d, matriks A dan matriks C ke persamaan riccati maka didapatkan Matriks P_e. Bentuk umum persamaan riccati adalah sebagai berikut :

$$P_e = AP_eA^T - (AP_eC^T)(V_n + CP_eC^T)^{-1}(CP_eA^T) + Vd \quad (3.14)$$

Diketahui bahwa :

$$A = \begin{bmatrix} 1.997 & -0.9983 \\ 1 & 0 \end{bmatrix}$$

$$C = [0.2262 \quad 0.2262]$$

$$V_d = \begin{bmatrix} 0.01 & 0 \\ 0 & 0.01 \end{bmatrix}$$

$$V_n = 0.01$$

$$P_e = \begin{bmatrix} P_{11} & P_{12} \\ P_{12} & P_{22} \end{bmatrix}$$

Sehingga apabila matriks A, C, V_d dan V_n disubstitusikan pada persamaan tersebut akan menjadi :

$$\begin{bmatrix} P_{11} & P_{12} \\ P_{12} & P_{22} \end{bmatrix} = \begin{bmatrix} 1.997 & -0.9983 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} P_{11} & P_{12} \\ P_{12} & P_{22} \end{bmatrix} \begin{bmatrix} 1.997 & -0.9983 \\ 1 & 0 \end{bmatrix}^T$$

$$- \left(\begin{bmatrix} 1.997 & -0.9983 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} P_{11} & P_{12} \\ P_{12} & P_{22} \end{bmatrix} [0.2262 \quad 0.2262] \right)$$

$$\left([0.01] + [0.2262 \quad 0.2262] \begin{bmatrix} P_{11} & P_{12} \\ P_{12} & P_{22} \end{bmatrix} [0.2262 \quad 0.2262]^T \right)^{-1}$$

$$\left([0.2262 \quad 0.2262] \begin{bmatrix} P_{11} & P_{12} \\ P_{12} & P_{22} \end{bmatrix} \begin{bmatrix} 1.997 & -0.9983 \\ 1 & 0 \end{bmatrix}^T \right) + \begin{bmatrix} 0.01 & 0 \\ 0 & 0.01 \end{bmatrix} \quad (3.15)$$

Maka,

$$P = \begin{bmatrix} 0.1841 & 0.0978 \\ 0.0978 & 0.0711 \end{bmatrix} \quad (3.16)$$

Disubstitusikan pada persamaan

$$K_f = (V_n + CPC^T)^{-1} CPA^T \quad (3.17)$$

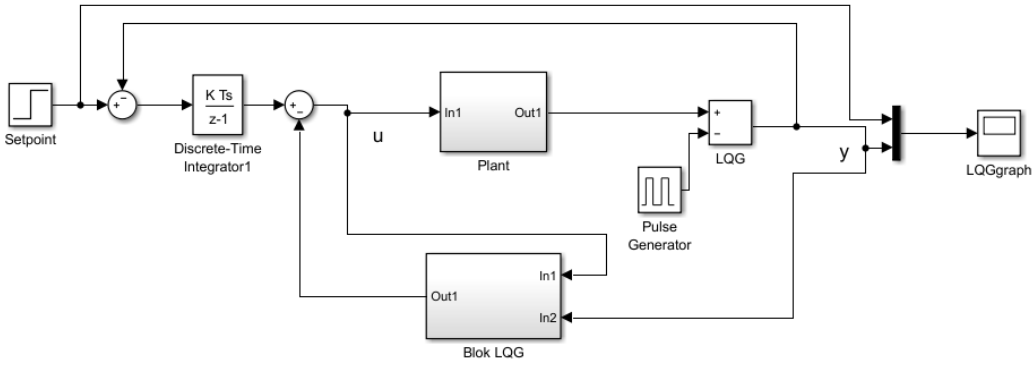
$$\left([0.01] + [0.2262 \quad 0.2262] \begin{bmatrix} 0.1841 & 0.0978 \\ 0.0978 & 0.0711 \end{bmatrix} [0.2262 \quad 0.2262]^T \right)^{-1}$$

$$[0.2262 \quad 0.2262] \begin{bmatrix} 0.1841 & 0.0978 \\ 0.0978 & 0.0711 \end{bmatrix} \begin{bmatrix} 1.997 & -0.9983 \\ 1 & 0 \end{bmatrix}^T \quad (3.18)$$

$$= \begin{bmatrix} 2.6979 \\ 1.9287 \end{bmatrix} \quad (3.19)$$

3.6.4 Simulasi LQG pada Matlab

Hasil dari nilai penguatan yang didapatkan selanjutnya akan digunakan untuk simulasi pada sistem kendali LQG. Pada penelitian ini nilai K1, K2, Kf1 dan Kf2 digunakan pada blok Simulink Matlab.



Gambar 3.16 Simulasi sistem kendali LQG

3.6.5 Pembuatan Program LQG

Pada penelitian ini, program LQG dibuat untuk diimpelementasikan pada Arduino, sehingga program dibuat pada software Arduino IDE dengan menggunakan bahasa pemrograman C. Alur program untuk kendali posisi menggunakan LQG terdiri dari inisiasi sensor dan motor, pembacaan sudut perhitungan error hingga output motor seperti yang terlihat pada diagram alir pada Gambar 3.17.

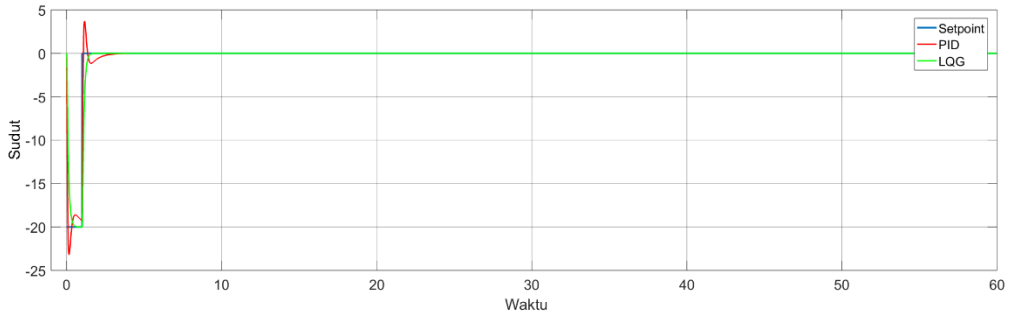
Gambar 3.17 Diagram alir program metode LQG untuk kendali posisi

BAB IV
HASIL DAN PEMBAHASAN

4.1 Hasil Simulasi Metode PID dan LQG pada Simulink

4.1.1 Simulasi Tanpa Gangguan

Pada hasil simulasi untuk kendali PID dan kendali LQG menggunakan Matlab Simulink dengan nilai awal step -20 dapat diperoleh hasil keluaran dapat dilihat pada Gambar 4.1 di bawah ini.



Gambar 4.1 Hasil simulasi sistem kendali PID dan LQG

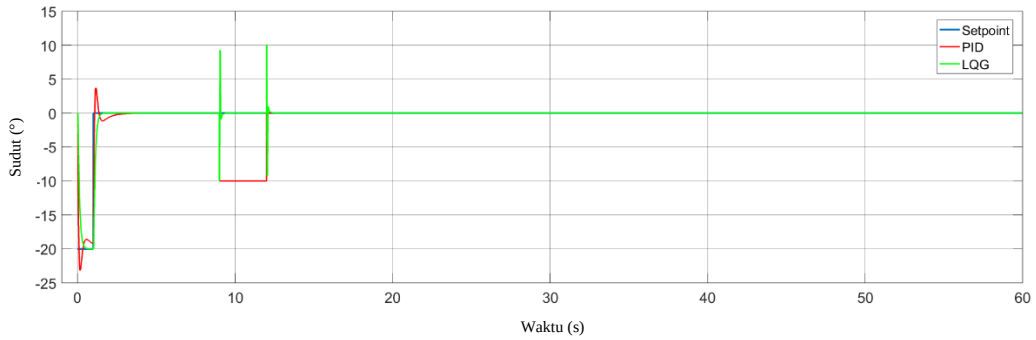
Pada Gambar 4.1 terlihat pada kendali PID (grafik berwarna merah) mempunyai nilai overshoot lebih tinggi dibanding dengan kendali LQG (grafik berwarna hijau), namun keduanya dapat menyeimbangkan posisi sudut pada detik ke-5. Pada simulasi sistem kendali PID dan LQG hasil respon disajikan pada **Tabel 4.1**.

Tabel 4.1 Hasil Simulasi PID dan LQG pada Simulink Matlab

Metode	*Rise Time*	*Settling Time*	*Overshoot*	*Undershoot*	ISE
PID	1,055 s	2,295 s	12,33 %	73,33 %	384,1
LQG	1,218 s	1,427 s	0 %	66,67 %	360,4

4.1.2 Simulasi dengan Sinyal Pengganggu

Pada hasil simulasi untuk kendali PID dan LQG menggunakan Matlab dengan nilai awal step -20 dengan sinyal pengganggu pada detik ke 9 selama 5 detikdapat diperoleh hasil keluaran seperti pada Gambar 4.2 di bawah ini.



Gambar 4.2 Hasil simulasi sistem kendali PID dan LQG dengan gangguan

Pada Gambar 4.1 terlihat pada kendali PID dan LQG sama sama terpengaruh dengan sinyal gangguan yang dikirimkan pada detik ke 9, namun kendali PID mampu menahan sinyal tersebut sehingga tidak menghasilkan overshoot yang terlalu tinggi dibandingkan dengan LQG. Namun keduanya dapat menyeimbangkan posisi sudut, pada simulasi sistem kendali PID dan LQG hasil respon disajikan pada **Tabel 4.2**.

Tabel 4.2 Hasil Simulasi dengan gangguan pada Simulink Matlab

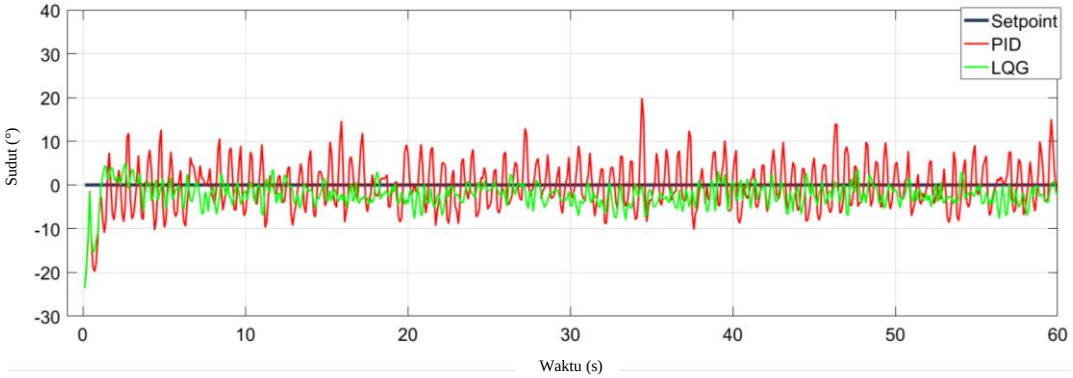
Metode	*Rise Time*	*Settling Time*	*Overshoot*	*Undershoot*	ISE
PID	1,05 s	2,2 s	9,25 %	14,28 %	684,1
LQG	1,257 s	1,427 s	14,28 %	14,28 %	369,5

4.2 Pengujian Sistem Kendali pada Arduino

Pengujian sistem kendali menggunakan mikrokontroler Arduino, motor BLDC 2200 Kv dengan daya 11,1 V melalui baterai Li-Po 3S dengan putaran maksimal 24.420 rpm, namun disini peneliti membatasi kecepatan pada 2900 rpm hingga 6100 rpm. Pada pengujian digunakan waktu sampling 100ms sehingga dalam

pengambilan data selama 60 detik didapat 600 data, lalu menggunakan batas kecepatan motor dengan hasil sebagai berikut :

4.2.1 Pengujian dengan posisi awal -23° dan setpoint 0°



Gambar 4.3 Hasil pengujian sistem kendali posisi awal -23°

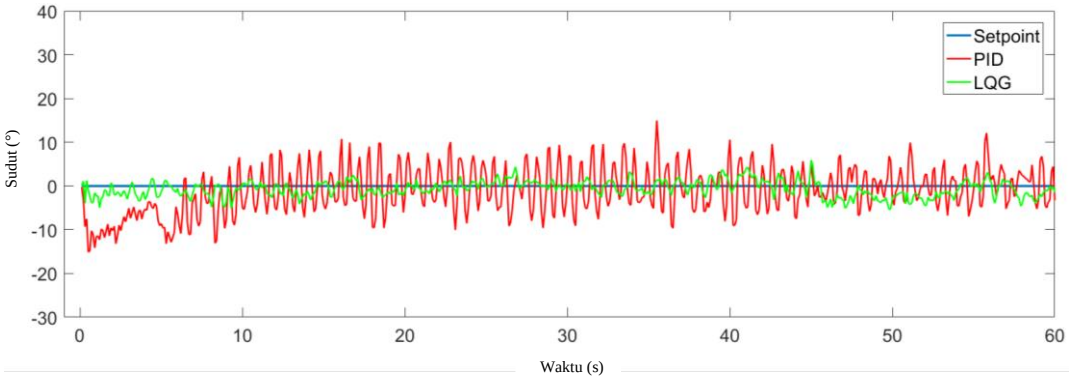
Pada Gambar 4.3 grafik hasil pengujian menunjukan pembacaan sudut yang dimulai dari sudut -23° sehingga hasilnya bergelombang naik turun untuk mempertahankan posisi yang diingingkan, yakni 0° sehingga sistem kendali masing – masing akan memproses nilai pembacaan sensor yang masuk untuk dikonversi menjadi output kecepatan motor yang sesuai.

Tabel 4.3 Hasil Pengujian dengan posisi awal -23°

Metode	*Rise Time*	*Settling Time*	*Overshoot*	*Undershoot*	ISE
PID	1,771 s	3,327 s	28,41 %	15 %	1962
LQG	1,150 s	7,583 s	7,25 %	10,98 %	854,9

Perbandingan kinerja sistem kendali dengan posisi awal -23° disajikan dalam Tabel 4.3. *Rise time* pertama kali dilakukan oleh kendali LQG dengan nilai 1,15 s, namun untuk *settling time* petama kali dicapai oleh kendali PID. Hasil *overshoot* dan *undershoot* terkecil diperoleh kendali LQG dengan masing – masing nilai 7,25% dan 10,98%. Dan nilai *integral squared error* terkecil diperoleh oleh kendali LQG dengan 854,9.

4.2.2 Pengujian dengan posisi awal 0° dan setpoint 0°



Gambar 4.4 Hasil pengujian sistem kendali posisi awal 0°

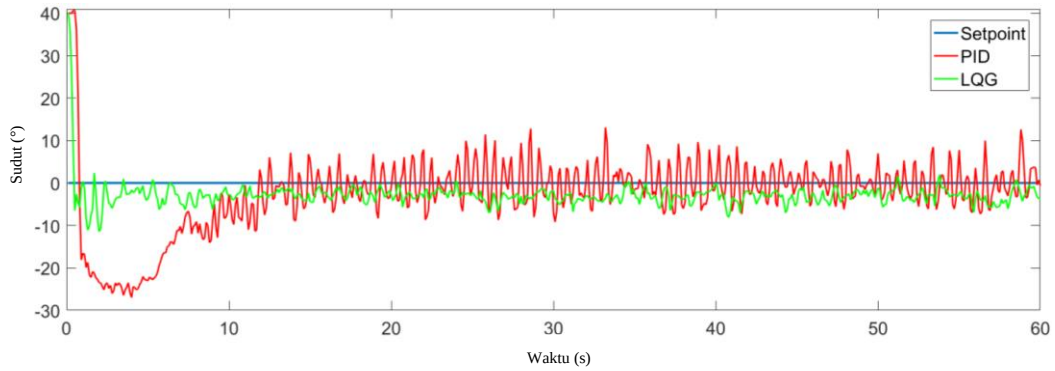
Pada Gambar 4.4 grafik hasil pengujian menunjukan pembacaan sudut yang dimulai dari sudut 0° sehingga hasil 5 detik pertama gelombang yang muncul belum terdapat riak yang tinggi pada kendali LQG, namun pada kendali PID sudah muncul riak sehingga dapat disimpulkan kendali PID cukup kesulitan dalam mempertahankan posisi sudut 0°.

Tabel 4.4 Hasil Pengujian dengan posisi awal 0°

Metode	*Rise Time*	*Settling Time*	*Overshoot*	*Undershoot*	ISE
PID	7,168 s	9,354 s	21,4 %	14,38 %	1812
LQG	1,605 s	3,155 s	8,442 %	7,671 %	226,1

Perbandingan kinerja sistem kendali dengan posisi awal 0° disajikan dalam Tabel 4.4. *Rise time* pertama kali dilakukan oleh kendali LQG dengan nilai 1,605 s, juga denga *settling time* dengan 3,155 s. Hasil *overshoot* dan *undershoot* terkecil juga dihasilkan pada kendali LQG yaitu 8,442 % dan 7,671%. Nilai *integral squared error* terkecil diperoleh oleh kendali LQG dengan 226,1.

4.2.3 Pengujian dengan posisi awal 40° dan setpoint 0°



Gambar 4.5 Hasil pengujian sistem kendali posisi awal 40

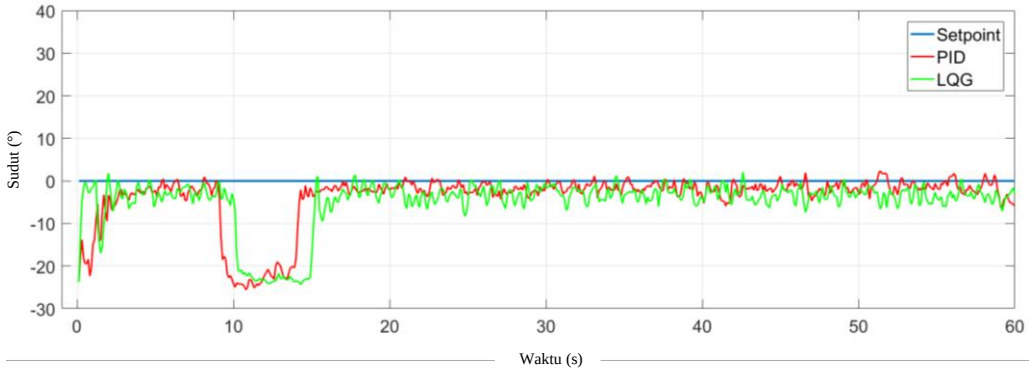
Pada Gambar 4.5 didapatkan grafik hasil pengujian menunjukan pembacaan sudut dari 40° sehingga akan mempengaruhi pembacaan selanjutnya yang menyebabkan kecepatan motor perlu diturunkan seminimal mungkin untuk mempertahankan posisi pada 0°, namun pada kendali PID turun terlalu jauh hingga hampir menyentuh 30°.

Tabel 4.5 Hasil Pengujian dengan posisi awal 40°

Metode	*Rise Time*	*Settling Time*	*Overshoot*	*Undershoot*	ISE
PID	11,845 s	13,284 s	18,685 %	16,357 %	5333
LQG	1,854 s	4,705 s	3,371 %	12,457 %	1297

Perbandingan kinerja masing – masing sistem kendali dengna posisi awal 40° disajikan dalam Tabel 4.5. *Rise time* tercepat dilakukan oleh kendali LQG dengan waktu 1,854 s dengan *settling time* 4,705 s. Juga mempunyai nilai *overshoot* dan *undershoot* yang lebih baik dibanding kendali PID yaitu 3,371% dan 12,457%. Nilai *integral squared error* terkecil diperoleh oleh kendali LQG dengan 1297.

4.2.4 Pengujian simulasi manuver bawah



Gambar 4.6 Hasil pengujian sistem kendali dengan manuver bawah

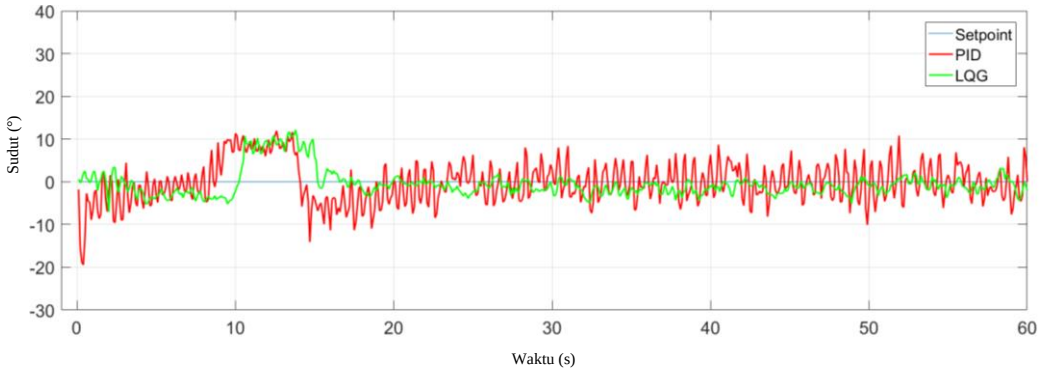
Pada Gambar 4.6 grafik hasil pengujian dengan penambahan sinyal bernilai negatif pada detik ke 9 menampilkan keandalan masing – masing sistem kendali dalam mempertahankan posisinya masing – masing pada sudut 0°. Pada pengujian ini motor dipaksa untuk bertahan pada posisi tersebut selama 5 detik dengan tujuan untuk mengganggu kestabilan dari masing – masing sistem kendali dan juga memberi contoh pembacaan saat pesawat VTOL bermanuver.

Tabel 4.6 Hasil Pengujian sistem kendali dengan manuver bawah

Metode	*Rise Time*	*Settling Time*	*Overshoot*	*Undershoot*	ISE
PID	4,705 s	17,214 s	3,371 %	8,742 %	3267
LQG	4,539 s	17,657 s	2,957 %	13,571 %	3561

Tabel 4.6 disajikan perbandingan kinerja sistem kendali dengan simulasi manuver bawah. *Rise time* tercepat dicapai oleh kendali LQG dengan waktu 4,677 s. Lalu *settling time* pertama kali dicapai oleh kendali PID dengan waktu 17,214 s, namun *overshoot* lebih tinggi yaitu 3,371% dan *undershoot* lebih rendah dibanding kendali LQG yaitu 8,742. Pada kendali PID berhasil mendapatkan nilai *integral squared error* terkecil dengan 3267.

4.2.5 Pengujian simulasi manuver atas



Gambar 4.7 Hasil pengujian sistem kendali dengan manuver atas

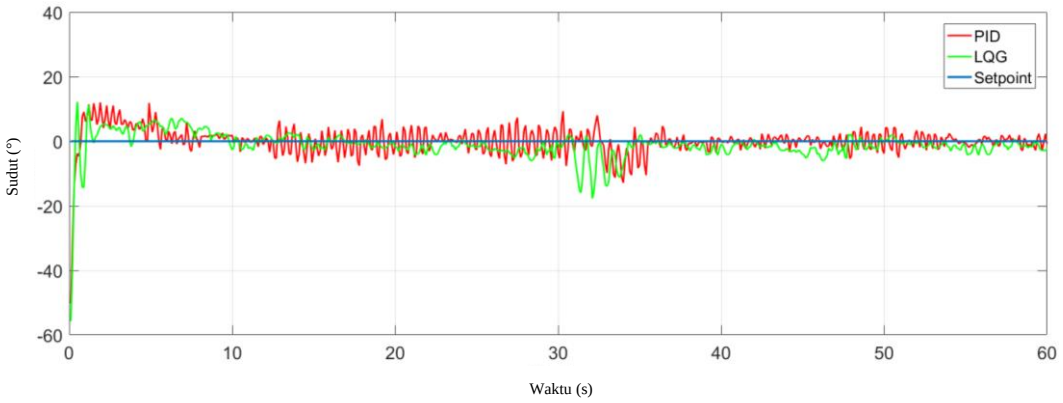
Pada Gambar 4.7 ditampilkan grafik hasil pengujian dengan penambahan sinyal bernilai positif pada detik ke 9 dengan tujuan untuk mengganggu kestabilan dari masing – masing kendali selama 5 detik dan juga menampilkan contoh saat pesawat jenis VTOL ingin melakukan manuver dan kembali ke posisi stabil yaitu 0°.

Tabel 4.7 Hasil Pengujian sistem kendali dengan manuver atas

Metode	*Rise Time*	*Settling Time*	*Overshoot*	*Undershoot*	ISE
PID	1,190 s	23,081 s	15,442 %	20,171 %	1440
LQG	6,365 s	19,124 s	4,485 %	7,371 %	616,6

Pada Tabel 4.7 hasil kinerja sistem kendali dengan simulasi manuver atas ditampilkan. *Rise time* tercepat dicapai oleh kendali PID dengan 1,19 s, namun *settling time* kalah cepat oleh kendali LQG dengan waktu 19,124 s. Nilai *overshoot* dan *undershoot* pada PID juga cukup tinggi dibanding kendali LQG dengan 15,442% dan 20,171%. Namun kendali LQG mendapatkan nilai *integral squared error* terkecil dengan 616,6.

4.2.6 Pengujian dengan beban 15 gram



Gambar 4.8 Hasil pengujian sistem kendali dengan beban 15 gram

Pada Gambar 4.8 ditampilkan grafik hasil pengujian dengan penambahan beban dengan bobot 15 gram pada detik ke 30 dengan tujuan untuk mengganggu kestabilan dari masing – masing kendali sampai akhir pengujian dan juga menampilkan keandalan dari masing – masing sistem.

Tabel 4.8 Hasil Pengujian dengan beban 15 gram

Metode	*Rise Time*	*Settling Time*	*Overshoot*	*Undershoot*	ISE
PID	0,94 s	30,22 s	12,06 %	12,88 %	1255
LQG	1,047 s	29,528 s	12,21 %	15,24 %	1407

Pada Tabel 4.8 hasil kinerja sistem kendali yang diberi beban dengan bobot 15 gram ditampilkan. *Rise time* tercepat dicapai oleh kendali PID dengan 0,94 s, namun *settling time* kalah cepat oleh kendali LQG dengan waktu 29,528 s. Nilai *overshoot* dan *undershoot* pada LQG juga cukup tinggi dibanding kendali PID dengan 12,21% dan 15,24%. Namun kendali PID mendapatkan nilai *integral squared error* terkecil dengan 1255.

4.3 Perbandingan Kinerja Sistem Kendali

Pada sub bab ini dilakukan perbandingan dari hasil masing – masing pengujian pada masing – masing sistem kendali, yaitu PID dan LQG dengan menghitung rata – rata pada masing – masing parameter. Pada Tabel 4.9 adalah nilai rata – rata dari tabel pengujian simulasi(Tabel 4.1 dan 4.2) Pada Tabel 4.10 adalah nilai rata – rata dari tabel perbandingan kinerja sistem kendali dengan posisi awal -23°, 0°, dan 40° (Tabel 4.3, 4.4, dan 4.5). Pada Tabel 4.11 adalah nilai rata – rata dari tabel perbandingan kinerja sistem kendali simulasi manuver atas dan bawah (Tabel 4.6, 4.7). Pada Tabel 4.12 adalah nilai dari tabel perbandingan kinerja sistem kendali dengan penambahan beban 15 gram dan (Tabel 4.8).

Tabel 4.9 Perbandingan kinerja pada simulasi

Metode	*Rise Time*	*Settling Time*	*Overshoot*	*Undershoot*	ISE
PID	1,0525 s	2,2475 s	10,79 %	43,805 %	534,1
LQG	1,2375 s	1,427 s	7,14 %	40,475 %	364,95

Tabel 4.10 Perbandingan kinerja sistem kendali pengujian posisi awal

Metode	*Rise Time*	*Settling Time*	*Overshoot*	*Undershoot*	ISE
PID	6,928 s	8,655 s	22,83 %	13,945 %	3035
LQG	4,609 s	5,147 s	19,063 %	11,669 %	792,67

Tabel 4.11 Perbandingan kinerja sistem manuver

Metode	*Rise Time*	*Settling Time*	*Overshoot*	*Undershoot*	ISE
PID	2,9475 s	20,147 s	9,406 %	14,456 %	2353
LQG	5,452 s	18,390 s	3,721 %	10,471 %	2088

Tabel 4.12 Perbandingan kinerja sistem kendali dengan beban

Metode	*Rise Time*	*Settling Time*	*Overshoot*	*Undershoot*	ISE
PID	0,94 s	30,22 s	12,06 %	12,88 %	1255
LQG	1,047 s	29,528 s	12,21 %	15,24 %	1407

Pada hasil simulasi kendali LQG jauh lebih baik dibanding pada kendali PID apabila melihat dari parameter *integral squared error* di pengujian dengan sinyal pengganggu. Begitu juga dengan pengujian langsung pada mikrokontroler Arduino bahwa kendali LQG jauh lebih unggul. Hal ini dikarenakan pada pengujian Arduino kendali PID masih menampilkan hasil yang terosilasi sehingga mempengaruhi nilai ISE. Selain itu juga dikarenakan kendali LQG merupakan kendali optimal yang berbasis regulator sehingga apabila terdapat pembacaan error akan mengeluarkan nilai *feedback* yang lebih kecil dari kendali lainnya. *Feedback* tersebut juga dibandingkan dengan nilai pembacaan error sebelumnya sehingga tidak menghasilkan lonjakan *feedback* yang signifikan.

BAB V
PENUTUP

5.1 Kesimpulan

Dari penelitian yang telah dilakukan, diperoleh beberapa kesimpulan sebagai berikut :

1. Secara sederhana, sistem kendali posisi pada pesawat VTOL dapat dirancang dengan komponen sistem meliputi Arduino Uno sebagai mikrokontroler, sensor MPU-6050 sebagai pembaca posisi sudut, ESC sebagai *driver* motor, motor BLDC tipe outrunner sebagai penggerak yang diletakkan pada lengan akrilik dan baterai Li-Po 3S sebagai catu daya.
2. Pada penelitian sistem kendali posisi VTOL ini secara keseluruhan didapatkan hasil dengan masing – masing variasi pengujian bahwa kinerja sistem kendali LQG lebih baik dibanding kendali PID dengan rata – rata ISE 792,67 dibanding 3035 saat pengujian variasi awal dan 2088 dibanding 2353 saat pengujian simulasi manuver atas dan bawah.
3. Penyelesaian perhitungan pada kendali optimal LQG dapat diselesaikan di mikrokontroler Arduino, namun memakan cukup banyak memori sehingga sangat mempengaruhi kinerja Arduino. Kelebihan dari sistem kendali posisi pada pesawat VTOL menggunakan mikrokontroler Arduino secara *standalone* yaitu terhindar dari *delay read-write* terhadap program komputer. Namun, kesulitan pada pembuatan program sesuai dengan algoritma menjadi tantangan tersendiri, terutama pada penyelesaian matriks untuk *Kalman Filter* pada kendali LQG yang perlu diselesaikan langsung di mikrokontroler.

5.2 Saran

Berdasarkan penelitian yang telah dilakukan, terdapat beberapa saran untuk pelaksanaan penelitian selanjutnya agar mendapatkan hasil yang lebih baik, yaitu :

1. Menambahkan penggerak motor BLDC menjadi 4 buah sehingga dapat disimulasikan langsung bagaimana pesawat VTOL dapat lepas landas, mendarat serta bermanuver.

2. Menggunakan sensor dengan tingkat akurasi yang lebih presisi sehingga tidak mempengaruhi pembacaan saat diproses pada mikrokontroler.
3. Menggunakan komponen hardware untuk mengisolasi sinyal pembacaan sensor agar tidak terpengaruh oleh sinyal dari ESC untuk menggerakkan motor.

DAFTAR PUSTAKA

[1] United States. Department of Defense. Office of the Secretary of Defense, "Unmanned Aircraft Systems Roadmap 2005-2030", 2005.

[2] C. Rokhmana, "The Potential of UAV-based Remote Sensing for Supporting Precision Agriculture in Indonesia", Procedia Environmental Sciences, vol. 24, pp. 245-253, 2015.

[3] Maddalon, Jeffrey M., Hayhurst, Kelly, J., Koppen, Daniel, M., Upchurch, Jason M., Morris, Allan T., Verstynen, and Harry A, Perspectives on Unmanned Aircraft Classification for Civil Airworthiness Standards. Virginia: NASA Langley Research Center, 2013.

[4] R. K. Lea, R. Allen and S. L.Merry, "A comparative study of control techniques for an underwater flight vehicle," International Journal of Systems Science, Vol. 30, No. 9, pp. 947- 964, 1999.

[5] Menteri Perhubungan Republik Indonesia, "Peraturan Menteri Perhubungan Republik Indonesia Nomor 47 Tahun 2016 TentangPengendalian Pengoperasian Sistem Pesawat Tanpa Awak di Ruang Udara yang Dilayani Indonesia", 2016.

[6] Zhiligotov, R. I., & Frolov, V. Y. (2017, February). Development of the sensorless control system BLDC motor. In 2017 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus) (pp. 1109-1111). IEEE.

[7] Shamseldin, M. A., & EL-Samahy, A. A. (2014, September). Speed control of BLDC motor by using PID control and self-tuning fuzzy PID controller. In 15th International Workshop on Research and Education in Mechatronics (REM) (pp. 1-9). IEEE.

[8] Teknik Energi Politeknik Negeri Semarang, "EKSERGI," Teknik Energi, vol. 6, no. Politeknik Negeri Semarang, pp. Halaman 1-41, 2010.

[9] A. Dharmawan, A. Ashari and A. Putra, "Translation Movement Stability Control of Quad Tiltrotor Using LQR and LQG", International Journal of Intelligent Systems and Applications, vol. 10, no. 3, pp. 10-21, 2018.

[10] B. Kurkcu and C. Kasnakoglu, "LQG/LTR position control of an BLDC motor with experimental validation", IEEE, pp. (pp. 796-800), 2015.

[11] A. Maddi, A. Guessoum and D. Berkani, "Using Linear Quadratic Gaussian Optimal Control for Lateral Motion of Aircraft", International Journal of Aerospace and Mechanical Engineering, vol. 3, 2009.

[12] R. Eide, P. Egelid, A. Stamsø and H. Karimi, "LQG Control Design for Balancing an Inverted Pendulum Mobile Robot", Intelligent Control and Automation, vol. 02, no. 02, pp. 160-166, 2011.

[13] M.S. Mazinder, S. Bordoloi and P. Bordoloi, "Automatic Control System for The BLDC Motor A2212/13t using Arduino Uno Platform", International Journal on Recent Innovation in Instrumentation & Control Engineering, vol. 1, no. 1, 2017.

[14] R. A. Vidi, “Aplikasi Sensor Accelerometer Pada Deteksi Posisi,” Universitas Diponogoro, Semarang, 2009.

[15] M. Athans, Optimal Control : an Introduction to the Theory and Its Applications. Dover Publications, 2013.

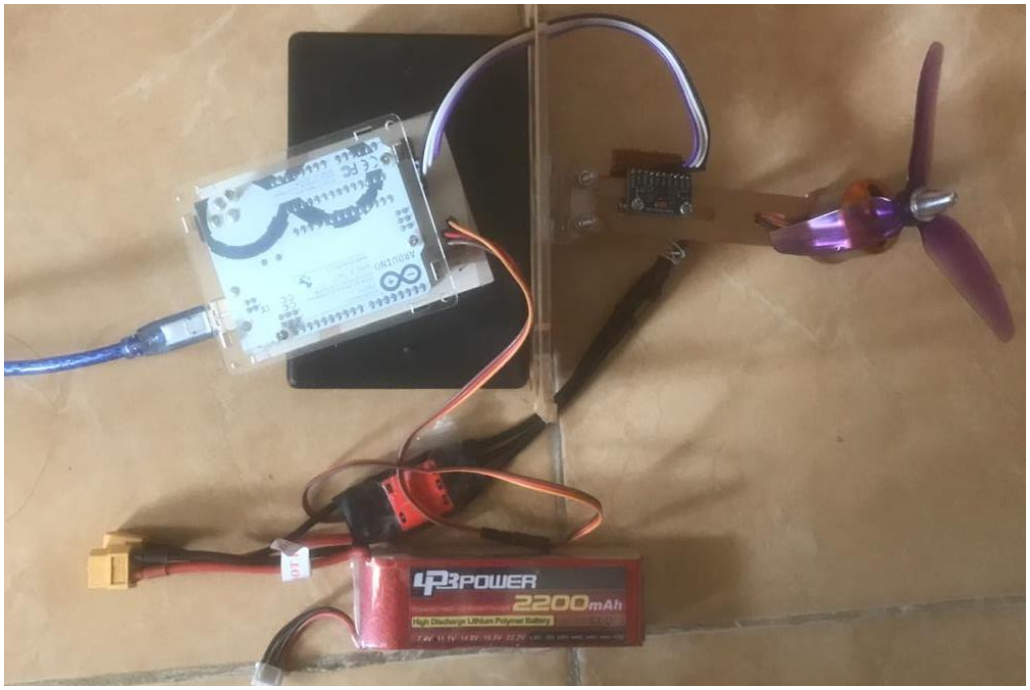
[16] V. Becerra, "Solving Optimal Control Problems With State Constraints Using Nonlinear Programming and Simulation Tools", IEEE Transactions on Education, vol. 47, no. 3, pp. 377-384, 2004.

[17] N. Kutz and S. Brunton, Data-Driven Science and Engineering : Machine Learning, Dynamical Systems, and Control. Chapter 8 : Linear Control Theory 2017.

[18] M. Ibrahim, A. Mahmood and N. Sultan, "Optimal PID controller of a brushless dc motor using genetic algorithm", International Journal of Power Electronics and Drive Systems (IJPEDS), vol. 10, no. 2, p. 822, 2019. Available: 10.11591/ijpeds.v10.i2.pp822-830.

LAMPIRAN

Dokumentasi



Gambar desain alat

Gambar pengujian alat (a) tanpa beban dan (b) dengan beban

Program Sistem Kendali pada Arduino IDE

```
#include <Wire.h>

#include <Servo.h>
```

```
#include <Matrix.h>

//////////////INISIASI SENSOR MPU-6050///////////////

int16_t Acc_rawX, Acc_rawY, Acc_rawZ,Gyr_rawX, Gyr_rawY, Gyr_rawZ;

float Acceleration_angle[2];

float Gyro_angle[2];

float Total_angle[2];

float sudut;

float rad_to_deg = 180/3.141592654;


//////////////INISIASI DATA LOGGER///////////////

String delimiter=",";

int n;

long refresh_time = 15000;

const long SERIAL_REFRESH_TIME = 100;

long upload_time;

float elapsedTime, time, timePrev;


////////////////////////////////////////////////

Servo motor_prop;

float PID, pwmMotor, error, previous_error;

float pid_p=0;

float pid_i=0;

float pid_d=0;


/////////////////PID CONSTANTS/////////////////

double kp=0.02555;

double ki=0.04024;
```

```
double kd=0.00405;
double cumError;

//////////LQG with KF Matrix/////////////////
float arrayA[2][2] = {{1.997,-0.9983},{1,0}};
float arrayB[2][1] = {{1},{0}};
float arrayx1[2][1] = {{0},{0}};
float arrayC[1][2] = {{0.2262,0.2262}};
float arrayKf[2][1] = {{2.7},{1.93}};
float arrayKlqr[1][2] = {{1.4 ,-0.83}};
float array1[1][1] = {1};
double Ki = 10;
float  Kl1;

//////////OPERASI MATRIKS LQG///////////////
Matrix<float> A(2,2,(float*)arrayA);
Matrix<float> B(2,1,(float*)arrayB);
Matrix<float> x1(2,1,(float*)arrayB);
Matrix<float> C(1,2,(float*)arrayC);
Matrix<float> mat1x1(1,1,(float*)array1);
Matrix<float> GainKf(2,1,(float*)arrayKf);
Matrix<float> GainKlqr(1,2,(float*)arrayKlqr);
Matrix<float> a1;
Matrix<float> b1;
Matrix<float> c1;
Matrix<float> x;
Matrix<float> Kf1;
```

```
float kLQG;


///////////////INISIASI MOTOR/////////////////
double motor=1165;                //NILAI AWAL KECEPATAN MOTOR
float desired_angle = 0;          //NILAI SUDUT YANG DIINGINKAN


void setup() {
  Wire.begin();                   //begin the wire comunication
  Wire.beginTransmission(0x68);
  Wire.write(0x6B);
  Wire.write(0);
  Wire.endTransmission(true);
  Serial.begin(9600);
  motor_prop.attach(3);            //PASANG ESC DI PIN 3
  time = millis();
  motor_prop.writeMicroseconds(1000);
  delay(6000);                     //MEMPERSIAPKAN PENGUJIAN
}


void loop() {
///////////I M U/////////////////////////
    timePrev = time;
    time = millis();
    elapsedTime = (time - timePrev) / 1000;
     Wire.beginTransmission(0x68);
     Wire.write(0x3B); // 0x3B register AcX
     Wire.endTransmission(false);
```

```
     Wire.requestFrom(0x68,6,true);
     Acc_rawX=Wire.read()<<8|Wire.read();
     Acc_rawY=Wire.read()<<8|Wire.read();
     Acc_rawZ=Wire.read()<<8|Wire.read();

 /*---X---*/
Acceleration_angle[0]=atan((Acc_rawY/16384.0)/sqrt(pow
((Acc_rawX/16384.0),2)+pow((Acc_rawZ/16384.0),2)))*rad_to_deg;
/*---Y---*/
Acceleration_angle[1]    =    atan(-1*(Acc_rawX/16384.0)/sqrt(pow
((Acc_rawY/16384.0),2)+pow((Acc_rawZ/16384.0),2)))*rad_to_deg;
   Wire.beginTransmission(0x68);
   Wire.write(0x43);
   Wire.endTransmission(false);
   Wire.requestFrom(0x68,4,true);
   Gyr_rawX=Wire.read()<<8|Wire.read();
   Gyr_rawY=Wire.read()<<8|Wire.read();

   /*---X---*/
   Gyro_angle[0] = Gyr_rawX/131.0;
   /*---Y---*/
   Gyro_angle[1] = Gyr_rawY/131.0;

   /*---X axis angle---*/
   Total_angle[0]      =      0.98      *(Total_angle[0]      +
Gyro_angle[0]*elapsedTime) + 0.02*Acceleration_angle[0];
   /*---Y axis angle---*/
```

```
   Total_angle[1]       =       0.98       *(Total_angle[1]       +
Gyro_angle[1]*elapsedTime) + 0.02*Acceleration_angle[1];


//////////////////SERIAL LOOP//////////////

sudut = Total_angle[0];

   if (millis() > upload_time) {

   String printPC = sudut + delimiter + pwmMotor + delimiter +
millis();

    Serial.println(printPC);

    upload_time = millis() + SERIAL_REFRESH_TIME;

   }

////////////////////////////////////////

error = sudut - desired_angle;

//progPID(); //PILIH SALAH SATU

progLQG(); //PILIH SALAH SATU

/////////PENGUJIAN SINYAL PENGGANGGU////////

//n = millis();

//if ((n >=15000) && (n <= 20000)){

//  pwmMotor = 1120;

//  }

////////////////////////////////////////////

//PEMBATASAN PWM MOTOR / KECEPATAN MOTOR

if(pwmMotor < 1120)

{

  pwmMotor= 1120;

}

if(pwmMotor > 1210)

{
```

```
  pwmMotor=1210;
}
motor_prop.writeMicroseconds(pwmMotor);
//previous_error = error; //Remember to store the previous error.

}//end void loop

/*////////////////P I D///////////////*/
void progPID(){ //Perhitungan metode PID berdasarkan blok diagram
pid_p = kp*error;//Perhitungan gain P
pid_i = ki*cumError;  //Perhitungan gain I
pid_d  =  kd*((error  -  previous_error)/elapsedTime);//Perhitungan
gain D
PID = pid_p + pid_i + pid_d;//Penjumlahan nilai P I D

previous_error = error;
cumError += error;

if(PID < -1000)//Pembatasan nilai PID agar tidak melebihi batas PWM
{
  PID=-1000;
}
if(PID > 1000)
{
  PID=1000;
}
//pwmMotor = motor - PID;//Upload nilai PWM setelah gain PID
```

```
pwmMotor = PID;          //+++ kalau error ganti pwmMotor=motor+PID
}// END PID PROGRAM


void progLQG(){ //Perhitungan matriks berdasarkan Blok Diagram
    cumError += error;
    float ei = Ki*cumError;
    float u  = ei-kLQG;
    float e1 = sudut - c1;
    Matrix<float> x = Kf1+a1+b1;    //Setelah penjumlahan
    Matrix<float> b1 = B*u;         //Setelah penguatan B
    Matrix<float> x1  = x;
    Matrix<float> Kf1 = GainKf*e1;  //Setelah penguatan Kf
    Matrix<float> a1 = A*x1;        //Setelah penguatan A
    Matrix<float> c1 = C*x1;        //Setelah penguatan C
    Matrix<float> Kl1 = GainKlqr*x1; //Setelah penguatan Klqr
    float kLQG = (Kl1.getMatrix());
    //Serial.print(u);
    //delay(3000);
    /*==================================
    Matrix<float> y = mat1x1*error;//Output dari plant
    Matrix<float> b1 = B*u; //Setelah penguatan B
    Matrix<float> c1 = x*C; //Setelah penguatan C?
    Matrix<float> Kf1 = GainKf*(y-c1); //Setelah penguatan Kf
    Matrix<float> x = Kf1+a1+b1; //Setelah penjumlahan
    Matrix<float> a1 = A*x;     //Setelah penguatan A
    Matrix<float> Kl1 = GainKlqr*x; //Setelah penguatan Klqr
    float kLQG = Ki*(Kl1.getMatrix());
```

```
   ================================== */
   if(u < -1000)
   {
     u=-1000;
   }
   if(u > 1000)
   {
     u=1000;
   }
// pwmMotor = motor - kLQG; //Upload nilai ke PWM setelah gain LQG
pwmMotor = motor - u;
 }//END LQG
```