TUGAS AKHIR – TE091399

TUNING PARAMETER *LINEAR QUADRATIC TRACKING* MENGGUNAKAN ALGORITMA GENETIKA UNTUK PENGENDALIAN GERAK LATERAL *QUADCOPTER*

Farid Choirul Akbar
NRP 2212 100 008

Dosen Pembimbing
Ir. Rusdhianto Effendie A.K., MT.
Eka Iskandar, ST. MT.

JURUSAN TEKNIK ELEKTRO
Fakultas Teknologi Industri
Institut Teknologi Sepuluh Nopember
Surabaya 2016

FINAL PROJECT – TE091399

PARAMETER TUNING OF LINEAR QUADRATIC TRACKING USING GENETIC ALGORITHM FOR LATERAL MOVEMENT CONTROL OF QUADCOPTER

Farid Choirul Akbar
NRP 2212 100 008

Advisor
Ir. Rusdhianto Effendie A.K., MT.
Eka Iskandar ST., MT.

ELECTRICAL ENGINEERING DEPARTMENT
Faculty of Industrial Technology
Institut Teknologi Sepuluh Nopember
Surabaya 2016

TUNING PARAMETER *LINEAR QUADRATIC TRACKING* MENGGUNAKAN ALGORITMA GENETIKA UNTUK PENGENDALIAN GERAK LATERAL *QUADCOPTER*

TUGAS AKHIR

Diajukan Guna Memenuhi Sebagian Persyaratan
Untuk Memperoleh Gelar Sarjana Teknik
Pada
Bidang Studi Teknik Sistem Pengaturan
Jurusan Teknik Elektro
Institut Teknologi Sepuluh Nopember

Menyetujui :

Dosen Pembimbing I	Dosen Pembimbing II
Ir. Rusdhianto Effendie A.K., MT.	**Eka Iskandar, ST., MT.**
NIP.195704241985021001	NIP.198005282008121001

SURABAYA
JANAURI, 2016

TUNING* PARAMETER *LINEAR QUADRATIC TRACKING* MENGGUNAKAN ALGORITMA GENETIKA UNTUK PENGENDALIAN GERAK LATERAL *QUADCOPTER

Nama : **Farid Choirul Akbar**
Pembimbing I : **Ir. Rusdhianto Effendie A.K., MT.**
Pembimbing II : **Eka Iskandar, ST., MT.**

ABSTRAK

Gerakan lateral *quadcopter* dapat dilakukan apabila *quadcopter* dapat menjaga kestabilan pada saat *hover*, sehingga *quadcopter* dapat melakukan gerak rotasi. Perubahan sudut *roll* akan mengakibatkan gerak translasi pada sumbu Y, sedangkan perubahan sudut *pitch* akan mengakibatkan gerak translasi pada sumbu X. Disisi lain, *quadcopter* merupakan suatu sistem *non-linear* dan memiliki kestabilan yang rendah sehingga rentan terhadap gangguan. Pada penelitian Tugas Akhir ini dirancang pengendalian gerak rotasi *quadcopter* menggunakan *Linear Quadratic Regulator* (LQR) dan *Linear Quadratic Tracking* (LQT) untuk pengendalian gerak translasi. Untuk mendapatkan parameter dari LQT digunakan Algoritma Genetika (GA). Hasil *tuning* GA yang digunakan pada LQT memiliki nilai Qx 700,1884, nilai Qy 700,6315, nilai Rx 0,1568, dan nilai Ry 0,1579. Respon LQT tersebut memiliki RMSE pada sumbu X dan sumbu Y sebesar 1,99 % serta memiliki *time lagging* 0,35 detik. Dengan hasil tersebut *quadcopter* mampu men-*tracking trajectory* berbentuk segitiga.

Kata Kunci: *Quadcopter*, lateral, *Linear Quadratic Regulator*, *Linear Quadratic Tracking*, Algoritma Genetika

PARAMETERS TUNING LINEAR QUADRATIC TRACKING USING GENETIC ALGORITHM FOR CONTROLLING LATERAL MOVEMENT OF QUADCOPTER

Name ***: Farid Choirul Akbar***
Supervisor I ***: Ir. Rusdhianto Effendie A.K., MT.***
Supervisor II ***: Eka Iskandar, ST., MT.***

ABSTRACT

Lateral movement of quadcopter occurs when quadcopter can maintain stability during a hover, because this movement occurs by changing the angle of rotation. Changing of roll angel make translational movement on Y axis and changing of pitch angel make translational movement on X axis. Additionally, quadcopter are a non-linear system, has low stability and susceptible with interference. In this final project designed rotational movement control of quadcopter using Linear Quadratic Regulator (LQR) and Linear Quadratic Tracking (LQT) for controlling the translational movement. To get the parameters of LQT used Genetic Algorithm (GA). From GA tuned, the value of Qx is 700.1884, the value of Qy is 700.6315, the value of Rx is 0.1568 and the value of Ry is 0.1579. The response of LQT has RMSE on the X axis and Y axis about 1.99% and has a lagging time about 0.35 seconds. With these results quadcopter able to follow the triangular trajectory

Key Word: *Quadcopter, lateral movement, Linear Quadratic Regulator, Linear Quadratic Tracking, Genetic Algorithm*

KATA PENGANTAR

Assalamualaikum wr. wb.

Puji dan syukur kehadirat Allah SWT yang telah melimpahkan rahmat dan hidayah-Nya, sehingga penulis dapat menyelesaikan buku Tugas Akhir ini. Shawalat serta salam senantiasa tercurah pula kepada nabi besar baginda Rasulullah Muhammad SAW.

Buku Tugas Akhir ini disusun untuk melengkapi salah satu syarat memperoleh gelar sarjana teknik di jurusan Teknik Elektro ITS. Buku yang berjudul **"*Tuning* Parameter *Linear Quadratic Tracking* Menggunakan Algortima Genetika untuk Pengendalian Gerak Lateral *Quadcopter*"** dipersembahkan juga untuk kemajuan riset dan teknologi Indonesia khusunya untuk ITS, Fakultas Teknologi Industri, Jurusan Teknik Elektro, dan bidang studi Teknik Sistem Pengaturan.

Hambatan dan rintangan selalu ada dalam menyelesaikan menyelesaikan buku ini. Namun dukungan dan bantuan terus mengalir hingga penulis dapat menyelesaikan buku ini. Pada kesempatan ini tak lupa penulis menyampaikan rasa terima kasih kepada beberapa pihak, antara lain:

1. Allah SWT yang telah memlimpahkan ramhat dan hidayah-Nya serta memperlancar dalam pengerjaan Tugas Akhir ini.
2. Orang tua dan kakak tercinta yang secara tidak langsung menjadi sumber semangat dalam menyelesaikan Tugas Akhir ini.
3. Dosen Pembimbing I, Bapak Ir. Rusdhianto Effendie A.K., MT. atas segala bimbingannya kepada penulis dalam pengerjaan Tugas Akhir ini.
4. Dosen Pembimbing II, Bapak Eka Iskandar, ST., MT. atas segala bimbingannya kepada penulis dalam pengerjaan Tugas Akhir ini.
5. Kepala Laboratorium, Bapak Ir. Ali Fathoni, MT. atas segala fasilitas yang dapat penulis gunakan dalam pengerjaan Tugas Akhir ini.
6. Bapak dan Ibu penguji Tugas Akhir yang telah memberi masukan kepada penulis sehingga buku ini menjadi lebih baik.
7. Teman-teman *team* TA *quadcopter*, Yurid, mas Kunto, mas Recho, mas Temmy atas segala kerjasamanya dalam pengerjaan butu Tugas Akhir ini.
8. Khairurizal Alfathdyanto atas segala bantuan dalam pengerjaan Tugas Akhir ini.

9. Teman-teman Lab B105 atas segala semangat dan kebersamaannya dalam pengerjaan Tugas Akhir ini.
10. Teman-teman Lab AJ204, mas Adit, mas Agyls, Mas hungkul, dan mbak Ayak yang telah berjuang bersama dalam pengerjaan Tugas Akhir.
11. Teman-teman *State Official*, Onang Surya Nugroho, Dhuhari Chalis Bani, Diaz Ficry Arfianto, Mohammad Faizal Shultoni, Andrie Wijaya, Rizkha Ajeng Rochmatika, Nabila Ardhana Iswari Azisputri, Asti Rakhmawati, Istiqomah, dan Aulia Haque Qonita yang banyak memberikan penyemangat, nasihat, kebahagiaan, kebersamaan, dan selalu mendukung penulis dalam pengerjaan Tugas Akhir ini.
12. Teman-teman angkatan 2012 Teknik Elektro ITS, khususnya bidang studi Teknik Sistem Pengatuan yang telah berbagi suka dan duka.
13. Semua pihak yang turut membantu dalam pengerjaan Tugas Akhir ini yang tidak dapat disebutkan satu per satu.

Penulis menyadari dan memohon maaf karena masih banyak kekurangan pada Tugas Akhir ini. Kritik dan saran selalu penulis nantikan agar menjadi lebih baik pada masa mendatang. Akhir kata, penulis berharap Tugas Akhir ini dapat bermanfaat dan menjadi acuan dalam penelitian selanjutnya.

Surabaya, Desember 2015

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

--halaman ini sengaja dikosongkan--

BAB I
PENDAHULUAN

Tugas Akhir adalah suatu penelitian yang bersifat mandiri yang dilakukan sebagai persyaratan akademik untuk mendapatkan gelar sarjana teknik di Institut Teknologi Sepuluh Nopember (ITS) Surabaya. Topik yang dibahas dalam Tugas Akhir ini ialah mengenai gerak lateral pada *quadcopter*.

Pada Bab ini membahas mengenai hal-hal yang mendahului pelaksanaan Tugas Akhir. Hal tersebut meliputi latar belakang, permasalahan, batasan masalah, tujuan, metodologi, sistematika penulisan, dan relevansi.

1.1 Latar Belakang

Pada saat ini, teknologi bidang penerbangan berkembang dengan pesat. Salah satu contohnya ialah *Unmanned Aerial Vehicle* (UAV) atau pesawat tanpa awak. UAV banyak mendapat perhatian dari berbagai kalangan karena dapat menggantikan peran manusia sebagai sistem pengendali. Hal tersebut menjadikan UAV banyak dimanfaatkan untuk keperluan militer ataupun sipil.

Secara garis besar UAV dapat dibagi menjadi dua jenis, yaitu *fixed-wing* dan *rotorcraft*. Pesawat *fixed-wing* memanfaatkan gerak translasi sehingga dihasilkan perbedaan tekanan pada sayap pesawat bagian bawah dan atas yang digunakan sebagai gaya angkat. Sedangkan pesawat jenis *rotorcraft* bergerak dengan *rotor* yang berputar dan menghasilkan gaya translasi vertikal akibat adanya aliran udara [1].

Salah satu contoh pesawat *rotorcraft* konvensional adalah helikopter. Halikopter memiliki satu *rotor* yang digunakan untuk bergerak. Untuk melakukan gerak maju atau menyamping, pesawat jenis ini harus memiringkan *propeller* dengan sudut tertentu. Hal tersebut menjadikan helikopter susah untuk dikendalikan.

Untuk mengatasi hal tersebut, dibuat suatu pesawat dengan menggunakan empat buah *rotor* yang digunakan untuk memudahakan penggendalian *rotorcraft*. Pengaturan perubahan sudut posisi *propeller* yang digunakan oleh helikopter untuk bergerak maju atau menyamping dapat digantikan dengan hanya mengatur kecepatan dari keempat *rotor* pada *propeller*. Pesawat *rotorcraft* dengan empat buah *propeller* ini disebut dengan *quadcopter*.

Quadcopter memiliki empat buah *propeller* dengan konfigurasi *plus* (+). *Propeller* depan dan belakang berputar searah jarum jam (*clockwise*), sedangkan *propaller* kanan dan kiri berputar berlawanan arah jarum jam (*counter clockwise*). Hal tersebut digunakan agar *quadcopter* memiliki keseimbangan yang baik pada saat terbang.

Quadcopter memiliki dua macam gerakan, yaitu gerak rotasi dan gerak translasi. Gerak rotasi merupakan gerakan yang terjadi pada poros *quadcopter* di mana terdiri dari gerak *roll*, *pitch*, dan *yaw*. Sedangkan gerak translasi merupakan gerakan yang dihasilkan akibat adanya gerak rotasi. Gerakan ini dibagi menjadi dua, yaitu gerak lateral dan gerak longitudinal. Gerak longitudinal meliputi tiga fase utama, yaitu *take-off* (tinggal landas), *hovering* (melayang), dan *landing* (pendaratan), sedangkan gerak lateral mencakup gerakan dari satu titik ke titik lain secara horisontal (*waypoint*).

Gerakan lateral *quadcopter* dapat dilakukan apabila *quadcopter* dapat menjaga kestabilan saat melakukan gerakan *hover*, karena gerak ini terjadi akibat adanya perubahan sudut dari gerak rotasi *quadcopter*. Perubahan sudut *roll* akan mengakibatkan gerak translasi pada sumbu Y, sedangkan perubahan sudut *pitch* akan mengakibatkan gerak translasi pada sumbu X. Sehingga untuk melakukan gerakan lateral *quadcopter*, harus dilakukan gerak rotasi terlebih dahulu.

Pada penelitian yang sudah ada, banyak digunakan metode kontrol konvensional PI, PID, *Linear Quadratic Control*, dan lain-lain. Pada penelitian dengan menggunkan *Linear Quadratic Control*, hasil yang didapatkan cukup baik. Namun penentuan parameternya masih menggunakan metode *try and error* [2]. Metode ini tidak efektif karena harus mencoba berulang kali untuk mendapatkan hasil kontrol yang optimal. Untuk mendapatkan nilai parameter Q dan R dari LQT yang optimal, dapat digunakan metode *tuning* GA. Pada penelitian ini akan digunakan metode *Linear Quadratic Regulator* (LQR) untuk pengendalian gerak rotasi *quadcopter*. Pada gerak lateral akan digunakan metode *Linear Quadratic Tracking* (LQT) dan Algoritma Genetika (GA) untuk *tuning* parameter dari kontroler. Perancangan kontroler tersebut akan disimulasikan menggunakan *software* MATLAB 2014a.

1.2 Permasalahan

Dalam suatu misi penerbangan, *quadcopter* perlu melewati titik-titik tujuan (*waypoints*) terterntu yang telah ditentukan. Untuk melakukan misi tersebut diperlukan suatu kontrol agar *quadcopter* dapat terbang

secara lateral menuju *waypoints* tersebut. Gerak lateral merupakan salah satu fasa di mana *quadcopter* dapat terbang pada sumbu X dan sumbu Y. Gerakan ini menuntut adanya kestabilan *quadcopter* dalam melakukan *hover* sehingga dapat terjadi perubahan sudut rotasi dari *pitch* atau *roll* yang menyebabkan terjadinya gerak translasi pada sumbu X atau Y.

Disisi lain, *quadcopter* merupakan sistem *non*-linear dan rentan terhadap gangguan. Untuk itu perlu dirancang suatu kontroler yang dapat menjaga kestabilan *quadcopter* ketika mekaukan gerak rotasi dan gerak lateral. Untuk pengendalian gerak rotasi digunakan metode *Linear Quadratic Regulator* (LQR). Sedangkan untuk pengendalian gerak translasi digunakan metode *Linear Quadratic Tracking* (LQT). Besarnya sinyal kontrol dari LQT bergantung pada kombinasi nilai Q dan R. Untuk mendapatkan kombinasi Q dan R dapat digunakan metode *tuning* manual (*try and error*). Metode ini tidak efektif karena harus mencoba berulang kali untuk mendapatkan hasil kontrol yang optimal. Oleh karena itu pada penelitian ini digunakan metode Algoritma Genetika (GA) untuk menggantikan metode *tuning* manual.

1.3 Batasan Masalah

Untuk membuat suatu penelitian diperlukan suatu batasan agar penelitian dapat lebih spesifik. Pada penelitian Tugas Akhir ini digunakan model UAV jenis *quadcopter* QB (*Quadcopter* Bios) yang berada pada Laboratorium Teknik Pengaturan B-105, Jurusan Teknik Elektro ITS.

Frame yang digunakan ialah *frame* talon turnigy di mana dirancang dengan konfigurasi *plus* (+). Berat dari *quadcopter* 1,26 kg dengan jari-jari 20,6 cm. Kontroler yang digunakan ialah Ardupilot Mega 2.6 dengan mikrokontroler ATmega 2560. Sensor *gyro* dan *accelerometer* yang digunakan ialah MPU 6050 yang terdapat dalam Ardupilot Mega 2.6. *Ground Station* yang digunakan ialah *Mission Planner*. Komunikasi dengan *ground station* menggunakan telemetry 433 Mhz.

Pemodelan *quadcopter* dilakukan dengan identifikasi parametrik untuk mendapatkan model matematika gerak rotasi *quadcopter*. Pengendalian dilakukan hanya pada gerak lateral yaitu gerakan pada sumbu X dan sumbu Y. Variabel yang dikontrol berjumlah 4 yaitu posisi *quadcopter* pada sumbu X, posisi *quadcopter* pada sumbu Y, sudut *pitch*, dan sudut *roll* dari *quadcopter*.

Pada pengendalian posisi *quadcopter* digunakan metode *Linear Quadratic Tracking* (LQT) sedangkan pada pengendalian sudut *quadcopter* digunakan metode metode *Linear Quadratic Regulator*

(LQR). Penentuan parameter Q dan R dari LQR dilakukan secara *tuning try and error*. Sedangkan penentuan parameter Q dan R dari LQT dilakukan dengan metode *tuning* Algoritma Genetika (GA). Perancangan GA dibatasi dengan jumlah populasi, jumlah generasi, rasio seleksi, rasio *crossover*, dan rasio mutasi. Perancangan kontroler dan simulasi dilakukan pada *software* MATLAB R2014a.

1.4 Tujuan

Quadcopter merupakan pesawat tanpa awak yang digunakan untuk berbagai misi penerbangan. Hal tersebut menuntut *quadcopter* agar dapat terbang dari suatu titik ke titik lain secara lateral dengan lincah. Oleh karena itu, *quadcopter* memerlukan suatu pengendalian di mana pengendalian *quadcopter* terdiri dari 2 macam, yaitu pengendalian posisi (gerak translasi) dan pengendalian sudut (gerak rotasi).

Untuk mengendalikan gerak translasi dari *quadcopter* digunakan metode LQT. Sedangkan untuk mengendalikan gerak rotasi digunakan LQR. Hasil kontrol LQT dan LQR yang optimal bergantung pada kombinasi dari Q dan R. Untuk mendapatkan kombinasi tersebut dapat digunakan metode *tuning* manual (*try and error*). Namun sering kali didapatkan hasil yang tidak optimal dari *tuning* manual Q dan R. Hasil optimal dapat dilihat melalui indeks performansi dari kontroler yang minimum. Untuk mendapatkan kombinasi Q dan R yang optimal, pada penelitian ini digunakan metode *tuning* GA. GA memiliki fungsi *fitness* yang dapat dicari nilai optimalnya. Pada perancangan GA digunakan fungsi *fitness* yang dapat meminimumkan indeks performansi dan *Root Mean Square Error* (RMSE) dari sinyal referensi.

Dengan menggunakan kontroler tersebut diharapkan *quadcopter* dapat terbang dengan lincah dan dapat men-*tracking trajectory* berbentuk segitiga.

1.5 Metodologi

Dalam penelitian Tugas Akhir ini diperlukan suatu tahapan yang merepresentasikan urutan yang harus dilaksanakan agar sesuai dengan tujuan penelitian. Tahapan tersebut ialah sebagai berikut:

a. Studi literatur

   Tahap awal untuk membuat suatu penelitian ialah mengumpulakan literatur yang berkaitan dengan LQR, LQT, dan GA, pemodelan *quadcopter*, dan perangkat keras yang akan digunakan pada

quadcopter. Literatur berasal dari jurnal, buku cetak, internet, dan lain-lain. Dengan adanya studi literatur, penelitian dapat dilakukan berdasarkan teori-teori yang telah ada sebelumnya.

b. Identifikasi
   Tahap yang kedua ialah mendapatkan model matematika dari *quadcopter*. Model matematika tersebut didapatkan dengan menggunakan identifikasi parametrik. Dengan didapatkannya model matematika dari *quadcopter*, perancangan kontroler dapat dilakukan.
c. Perancangan Kontroler
   Tahap yang ketiga ialah perancangan kontroler dengan menggunakan *software* MATLAB R2014a. Kontroler yang dirancang ialah LQR untuk pengendalian gerak rotasi, LQT untuk pengendalian gerak translasi, dan GA untuk *tuning* parameter LQT.
d. Simulasi dan Evaluasi
   Tahap yang keempat ialah mensimulasikan kontroler yang telah dibuat menggunakan *software* MATLAB R2014a. Tujuan dari simulasi ialah untuk mengetahui performansi dari sistem yang telah diberi kontroler. Jika performansi yang didapatkan tidak sesuai dengan yang inginkan maka dilakukan evaluasi. Evaluasi dilakukan dengan mengubah parameter dari kontroler yang telah dirancang.
e. Penulisan buku Tugas Akhir
   Tahap yang terakhir ialah penulisan laporan/buku Tugas Akhir. Penulisan dilakukan secara intensif bila proses pengujian telah selesai.

1.6 Sistematika Penulisan

Tahap terakhir dari sebuah penelitian adalah penulisan laporan. Pada penulisan laporan/buku Tugas Akhir ini disusun berdasarkan 5 bab, di mana setiap bab berisi mengenai permasalahan dalam penelitian. Bab tersebut ialah sebagai berikut:

BAB I PENDAHULUAN
Berisi mengenai latar belakang, permasalahan, pembatasan masalah, tujuan, metodologi, sistematika penulisan, dan relevansi pembahasan tugas akhir ini.

BAB II TEORI PENUNJANG

Berisi mengenai konsep dasar dan teori yang mendasari perancangan tugas akhir ini, meliputi teori *quadcopter*, kinematika dan dinamika *quadcopter*, spesifikasi dari *quadcopter* yang digunakan, dan perancangan kontroler LQR, LQT, dan GA.

BAB III PERANCANGAN SISTEM

Berisi mengenai spesifikasi sistem, identifikasi parameter, perancangan kontroler LQR, LQT dan GA dengan mengacu teori pada BAB II.

BAB IV PENGUJIAN DAN ANALISIS

Berisi prosedur pelaksanaan pengujian dan analisis data mengenai pengendalian gerak lateral menggunakan kontroler LQT, LQR, dan GA.

BAB V PENUTUP

Berisi mengenai kesimpulan dari penelitian Tugas Akhir dan saran untuk dapat digunakan untuk pengembangan tugas akhir ini untuk lebih lanjut.

1.7 Relevansi

Tugas akhir ini diharapkan dapat menjadi referensi untuk pengembangan UAV khususnya *quadcopter* untuk gerak lateral. Selain itu juga sebagai perbandingan metode kontrol yang lain untuk menentukan kontroler mana yang memiliki hasil paling baik.

BAB II
TEORI PENUNJANG

Suatu penelitian memerlukan teori-teori yang sudah ada sebelumnya untuk dikaji lebih dalam memperkuat argumen penulis. Teori tersebut digunakan untuk membantu penulis dan sebagai dasar dalam membuat suatu penelitian.

Pada bab ini terdapat beberapa teori dasar yang menjadi landasan untuk merumuskan dan menyelesaikan masalah yang akan dibahas pada penelitian ini. Pada bagian awal terdapat tinjauan pustaka yang menggambarkan landasan teori mengenai *quadcopter* secara umum yang akan digunakan pada tugas akhir ini. Pada bagian selanjutnya membahas mengenai teori-teori pendukung, meliputi dinamika dan kinematika *quadcopter*, dan kontroler LQR, LQT dan GA.

2.1 *Quadcopter* [3], [4]

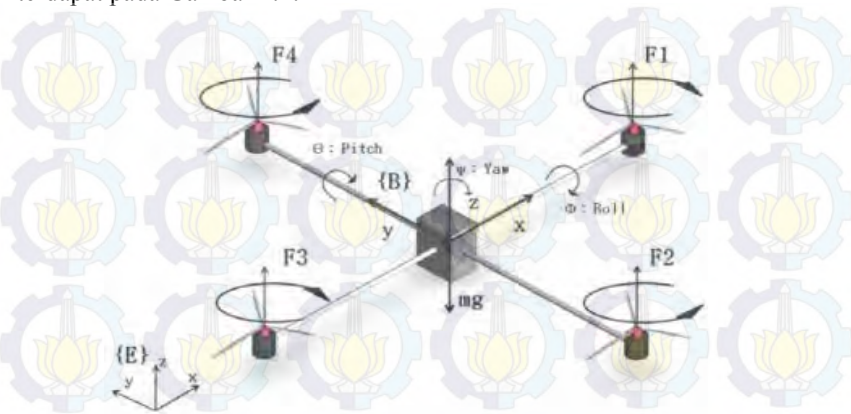
Quadcopter merupakan bagian dari UAV untuk kategori pesawat *rotorcraft* di mana memiliki kemampuan untuk lepas landas dan mendarat secara vertikal. Kemampuan terbang dari *quadcopter* dapat dikendalikan dengan mengatur kecepatan *propaller* yang berputar pada keempat sisi. Bila dibandingkan dengan helikopter, *quadcopter* memiliki beberapa kelebihan, yaitu tidak memerlukan ekor penyeimbang seperti helikopter dan bisa menggunakan *fixed propeller* atau memerlukan pengaturan sudut *propeller* seperti pada helikopter.

2.1.1 Konsep Dasar *Quadcopter* [4]

Quadcopter yang digunakan sebagai robot terbang kecil memiliki model mekanik yang terdiri dari empat *rotor* yang dipasang pada sumbu *plus* (+) simetris. Bentuk ini diharapkan tipis dan kaku, sehingga diperoleh friksi udara yang kecil dan komponen yang bergerak pada *quadcopter* hanyalah putaran *propeller*. Setiap *propeller* pada *quadcopter* diputar oleh satu motor elektrik, sehingga terdapat empat motor sebagai aktuator untuk menghasilkan gaya angkat dari *quadcopter*.

Dengan batasan menggunakan karakteristik motor dan *propeller* yang relatif sama, maka kondisi melayang (*hover*) akan diperoleh kecepatan motor yang sama disetiap *propeller*. Konfigurasi *propeller* terdiri dari dua macam, yaitu dua *propeller* yang bergerak searah jarum jam dan dua *propeller* yang bergerak berlawanan arah jarum jam di mana

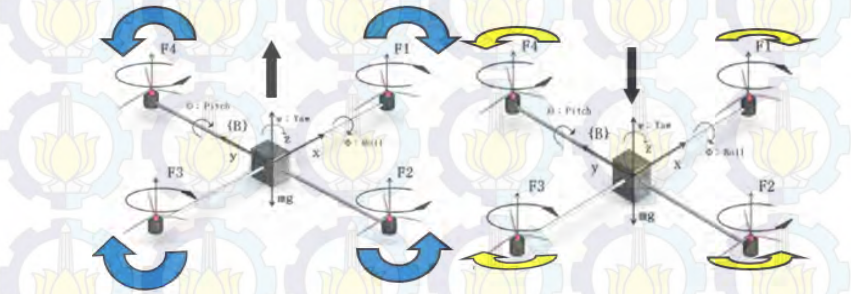
setiap satu sumbu, *propeller* berputar dengan arah yang sama yang terdapat pada Gambar 2.1.



Gambar 2. 1 Konfigurasi *Quadcopter*

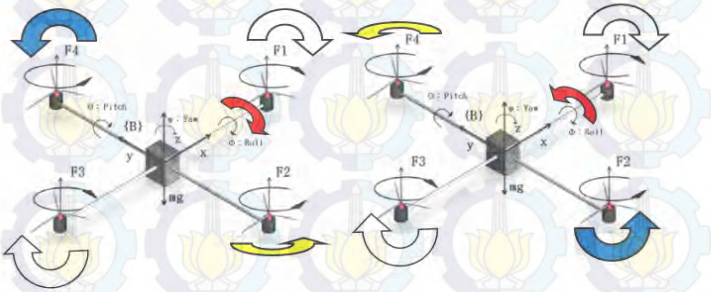
Dengan melakukan pengaturan kecepatan putaran *propeller* akan dihasilkan beberapa komando *input*, diantaranya ialah:

a. Gaya *Thrust* (U_1)
   Dengan mempercepat (biru) dan memperlambat (kuning) kecepatan motor dari *propeller* secara bersamaan akan menghasilkan percepatan vertikal yang ditunjukkan pada Gambar 2.2.



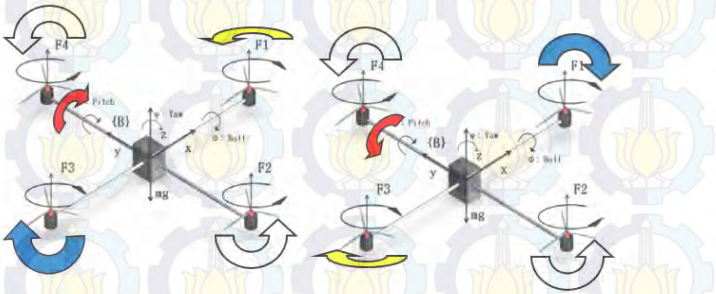
Gambar 2.2 Gaya *Thrust*

b. Torsi *Roll* (U_2)
Dengan mempercepat (biru) atau memperlambat (kuning), akan diperoleh gerak rotasi *roll* dengan sudut tertentu yang ditunjukkan pada Gambar 2.3.



Gambar 2.3 Torsi *Roll*

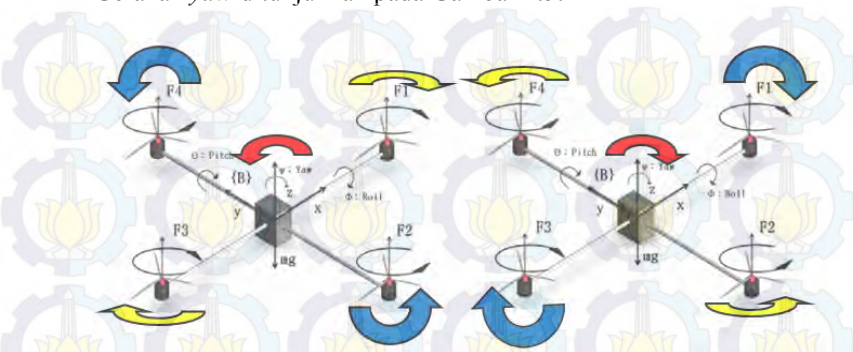
c. Torsi *Pitch* (U_3)
Dengan mempercepat (biru) atau memperlambat (kuning) akan diperoleh gerak rotasi *pitch* dengan sudut tertentu yang ditunjukkan pada Gambar 2.4.



Gambar 2.4 Torsi *Pitch*

d. Torsi *Yaw* (U_4)
Pergerakan ini dilakukan dengan cara meningkatkan atau menurunkan kecepatan putar motor 2 dan 4 pada *quadcopter* dan bersamaan dengan itu, menurunkan atau menaikkan kecepatan

motor 1 dan 3. Gerakan ini berputar dengan acuan pada sumbu Z. Gerakan *yaw* ditunjukkan pada Gambar 2.5.



Gambar 2.5 Torsi *Yaw*

2.1.2 Kinematika *Quadcopter* [4]

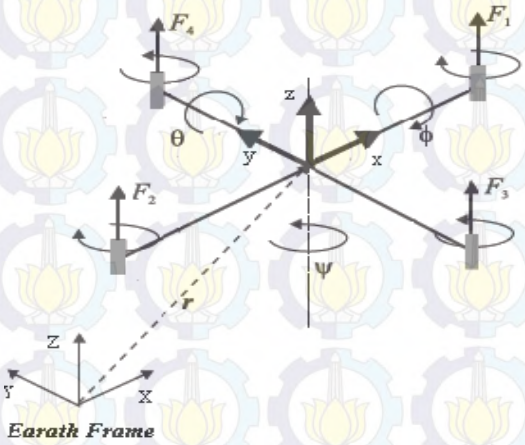
Sebelum melakukan analisis kinematika, dilakukan beberapa penetapan-penetapan yang akan digunakan. Pertama, analisis yang digunakan untuk kinematika memakai bingkai diagram cartesius tiga dimensi (X, Y, Z). Bingkai diagram cartesius dibagi menjadi dua, yaitu bingkai bumi (*earth frame*) yang kaku atau tidak bergerak, serta bingkai *body quadcopter* yang bergerak rotasi dan translasi. Ditetapkan sumbu X-bumi berada pada garis utara-selatan, sumbu Y-bumi berada pada garis timur-barat, sumbu Z-bumi berada pada garis vertikal menuju pusat bumi, sumbu X-*quadcopter* berada pada garis depan-belakang *quadcopter*, sumbu Y-*quadcopter* berada pada garis kiri-kanan *quadcopter*, dan sumbu Z-*quadcopter* berada pada garis atas-bawah *quadcopter*.

Bila didefinisikan sumbu X^+ menuju utara dan depan *quadcopter*, sumbu Z^+ menjauhi pusat bumi dan menuju atas *quadcopter* maka berlandaskan aturan kaidah tangan kanan *cross vector* dapat ditentukan bahwa sumbu Y^+ menuju barat atau kiri *quadcopter* seperti yang ditampilkan pada Gambar 2.6.

Posisi linear *quadcopter* (Γ^E) ditentukan dari koordinat vektor antara origin *B-frame* serta origin dari *E-frame d*engan memperhatikan *E-frame*. Posisi angular *quadcopter* (Θ^E) ditentukan dari orientasi *B-frame* terhadap *E-frame*. Posisi linear dan posisi angular terdapat pada Persamaan (2.1)-(2.2).

$$\Gamma^E = [X \;\; Y \;\; Z]^T \tag{2.1}$$

$$\Theta^E = [\phi \;\; \theta \;\; \Psi]^T \tag{2.2}$$

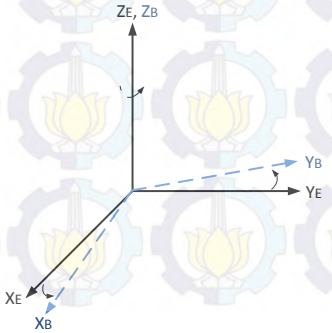


Gambar 2.6 *Earth-frame* dan *Body-frame*

Matriks rotasi dari *quadcopter* diperoleh dengan mengkalikan matriks transformasi untuk rotasi ditiap sumbu.

a. Rotasi Sumbu Z

Quadcopter berotasi pada sumbu Z dan menghasilkan sudut *yaw* yang dilambangkan dengan R *(ψ, z)*. Rotasi sudut *yaw* terdapat pada Gambar 2.7



Gambar 2.7 Rotasi *Quadcopter* Sumbu Z

$$\begin{cases} X_E &= X_B \cos\psi - Y_B \sin\psi \\ Y_E &= X_B \sin\psi + Y_B cos\psi \\ Z_E &= Z_B \end{cases} \quad (2.3)$$

Persamaan (2.3) dapat dibuat dalam bentuk matriks pada Persamaan (2.4).

$$\begin{bmatrix} X_E \\ Y_E \\ Z_E \end{bmatrix} = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_B \\ Y_B \\ Z_B \end{bmatrix} \quad (2.4)$$

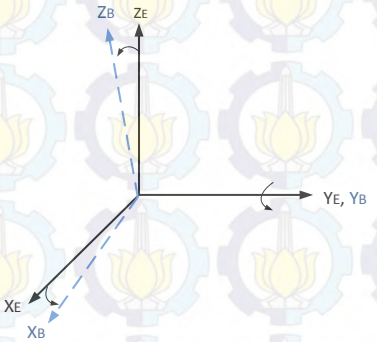
Sehingga didapatkan matriks rotasi pada sumbu X ditunjukkan pada Persamaan (2.5).

.

$$R(\psi) = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (2.5)$$

b. Rotasi Sumbu Y

Quadcopter berotasi pada sumbu Y dan menghasilkan sudut *pitch* yang dilambangkan dengan R (*ɵ, y*). Rotasi sudut *pitch* terdapat pada Gambar 2.8



Gambar 2. 8 Rotasi *Quadcopter* Sumbu Y

$$\begin{cases} X_E &= X_B \cos\theta + Z_B \sin\theta \\ Y_E &= Y_B \\ Z_E &= -X_B \sin\theta + Z_B \cos\theta \end{cases} \quad (2.6)$$

Persamaan (2.6) dapat dibuat dalam bentuk matriks menjadi Persamaan (2.7).

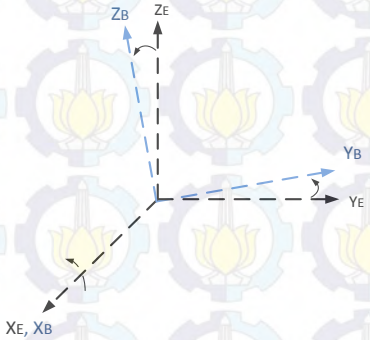
$$\begin{bmatrix} X_E \\ Y_E \\ Z_E \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} X_B \\ Y_B \\ Z_B \end{bmatrix} \quad (2.7)$$

Adapun matriks rotasi pada sumbu Y ditunjukkan pada Persamaan (2.8).

$$R(\theta, y) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \quad (2.8)$$

c. Rotasi Sumbu X

Quadcopter berotasi pada sumbu X dan menghasilkan sudut *roll* yang dilambangkan dengan R (ϕ, x). Rotasi sudut *roll* terdapat pada Gambar 2.7



Gambar 2. 9 Rotasi *Quadcopter* Sumbu X

$$\begin{cases} X_E = X_B \\ Y_E = Y_B \cos\phi - Y_B \sin\phi \\ Z_E = Y_B \sin\phi + Z_B \cos\phi \end{cases} \quad (2.9)$$

Persamaan (2.9) dapat dibuat dalam bentuk matriks menjadi Persamaan (2.10).

$$\begin{bmatrix} X_E \\ Y_E \\ Z_E \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} X_B \\ Y_B \\ Z_B \end{bmatrix} \quad (2.10)$$

Adapun matriks rotasi pada sumbu Z ditunjukkan pada Persamaan (2.11).

$$R(\phi, x) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix} \quad (2.11)$$

Sehingga persamaan rotasi *quadcopter* terdapat pada Persamaan (2.12)-(2.13).

$$R_\Theta = R(\phi)\, R(\theta)\, R(\psi) \quad (2.12)$$

$$R_\Theta = \begin{bmatrix} c_\psi c_\theta & -s_\psi c_\phi + c_\psi s_\theta s_\phi & s_\psi s_\phi + c_\psi s_\theta s_\phi \\ s_\psi c_\theta & c_\psi c_\phi + s_\psi s_\theta c_\phi & -c_\psi s_\phi + s_\psi s_\theta c_\phi \\ -s_\theta & c_\theta s_\phi & c_\theta c_\phi \end{bmatrix} \quad (2.13)$$

Sedangkan untuk kecepatan dalam *quadcopter* diekspresikan terhadap *body frame* (*B-frame*). Kecepatan *quadcopter* terdiri dari kecepatan linear V^B dan kecepatan angular ω^B. Komposisi vektornya disajikan dalam Persamaan (2.14)-(2.15).

$$\omega^B = [p \quad q \quad r]^T \quad (2.15)$$

Diperlukan kombinasi nilai linear dan angular untuk memberikan representasi yang lengkap dalam *space*. $\boldsymbol{\xi}$ merupakan komposisi dari vektor posisi linear Γ^E (m) dan vektor posisi sudut Θ^E (rad) terhadap *earth frame* (*E-frame*) seperti terlihat pada Persamaan (2.16).

$$\boldsymbol{\xi} = [\Gamma^E \quad \Theta^E]^T = [X \quad Y \quad Z \quad \phi \quad \theta \quad \psi]^T \quad (2.16)$$

V merupakan generalisasi dari vektor kecepatan linear *quadcopter* V^B (m s^{-1}) dan kecepatan angular *quadcopter* $\boldsymbol{\omega}^B$ (rad s^{-1}) pada *body frame* (*B-frame*) yang terdapat pada Persamaan (2.17)

$$V = [V^B \ \omega^B]^T = [\, u \quad v \quad w \quad p \quad q \quad r]^T \tag{2.17}$$

Hubungan antara kecepatan linear pada *body frame* (*B-frame*) dan salah satu factor pada *earth frame* (*E-frame*) V^E (atau $\dot{\Gamma^E}$) [m/s] dapat dilihat pada Persamaan (2.18).

$$V^E = \dot{\Gamma^E} = R_\Theta V^B \tag{2.18}$$

Dimana R_Θ adalah matrik rotasi dari *body frame* (*B-frame*) ke *earth frame* (*E-frame*). Seperti pada kecepatan linear, hal tersebut juga berlaku untuk menghubungkan kecepatan angular pada *E-frame* (atau kecepatan Euler) $\dot{\Theta^E}$ [rad/s] ke *B-frame* $\boldsymbol{\omega}^B$ atau sebaliknya. Hubungan tersebut terdapat pada Persamaan (2.19).

$$\omega^B = T_\Theta^{-1} \dot{\Theta^E} \tag{2.19}$$

$$\dot{\Theta^E} = T_\Theta \omega^B \tag{2.20}$$

Dimana T_Θ adalah matriks transformasi. Matriks transformasi T_Θ dapat ditetapkan dengan menggunakan kecepatan Euler dalam *B-frame*, dengan membalik pola perputaran sudut dari *roll*, *pitch* dan *yaw* seperti pada Persamaan (2.21).

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} \dot{\phi} \\ 0 \\ 0 \end{bmatrix} + R(\phi)^{-1} \begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + R(\phi)^{-1} R(\theta)^{-1} \begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix} \tag{2.21}$$

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix} = T_\Theta^{-1} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \tag{2.22}$$

Persamaan (2.13)-(2.19) maka diperoleh matriks transformasi dari *body frame* (*B-frame*) menuju *earth frame* (*E-frame*).

$$T_\Theta^{-1} = \begin{bmatrix} 1 & 0 & -s_\theta \\ 0 & c_\phi & c_\theta s_\phi \\ 0 & -s_\phi & c_\phi c_\theta \end{bmatrix} \tag{2.23}$$

$$T_{\boldsymbol{\Theta}} = \begin{bmatrix} 1 & s_\phi t_\theta & c_\phi t_\theta \\ 0 & c_\phi & -s_\phi \\ 0 & s_\phi / c_\theta & c_\phi / c_\theta \end{bmatrix} \tag{2.24}$$

Persamaan yang sudah didapat, maka dibentuk suatu hubungan antara kecepatan terhadap *earth frame* (*E-frame*) dan *body frame* (*B-frame*).

$$\dot{\xi} = J_\Theta v \tag{2.25}$$

Dimana $\dot{\xi}$ adalah vektor kecepatan yang mengacu pada *earth frame* (*E-frame*) , ν adalah vektor kecepatan mengacu *body frame* (*B-frame*) dan J_Θ adalah matrik *jacobian*. Matrik *jacobian* terdiri dari 4 *sub*-matrik sebagaimana Persamaan (2.26).

$$J_\Theta = \begin{bmatrix} R_\Theta & 0_{3x3} \\ 0_{3x3} & T_\Theta \end{bmatrix} \tag{2.26}$$

2.1.3 Dinamika *Quadcopter* [4]

Berdasarkan aksioma pertama Euler dari hukum kedua Newton, turunan dari komponen linear dari gerakan suatu benda dapat dilihat pada Persamaan (2.25).

$$\begin{cases} m\,\ddot{\boldsymbol{\Gamma}}^{\mathbf{E}} = \boldsymbol{F}^{\boldsymbol{E}} \\ m\,\dot{\overbrace{\mathbf{R}_{\boldsymbol{\Theta}}\mathbf{V}^{\mathbf{B}}}} = \boldsymbol{R}_{\boldsymbol{\Theta}}\,\boldsymbol{F}^{\boldsymbol{B}} \\ m\left(\mathbf{R}_{\boldsymbol{\Theta}}\dot{\mathbf{V}}^{\mathbf{B}} + \dot{\mathbf{R}}_{\boldsymbol{\Theta}}\mathbf{V}^{\mathbf{B}}\right) = \boldsymbol{F}^{\boldsymbol{B}} \\ m\,\mathbf{R}_{\boldsymbol{\Theta}}\left(\dot{\mathbf{V}}^{\mathbf{B}} + \boldsymbol{\omega}^{\mathbf{B}}\,\mathbf{x}\,\mathbf{V}^{\mathbf{B}}\right) = \mathbf{R}_{\boldsymbol{\Theta}}\boldsymbol{F}^{\boldsymbol{B}} \\ m\left(\dot{\mathbf{V}}^{\mathbf{B}} + \boldsymbol{\omega}^{\mathbf{B}}\,\mathbf{x}\,\mathbf{V}^{\mathbf{B}}\right) = \boldsymbol{F}^{\boldsymbol{B}} \end{cases} \tag{2.25}$$

Dengan m (kg) adalah massa *quadcopter*, $\ddot{\boldsymbol{\Gamma}}^{\mathbf{E}}$ (m s^{-2}) adalah vektor percepatan linear *quadcopter* yang mengacu pada E-*frame*, $\boldsymbol{F}^{E}$ (N) adalah vektor gaya *quadcopter* terhadap E-*frame*, $\dot{\mathbf{V}}^{\mathbf{B}}$ (m s^{-2}) adalah percepatan linear *quadcopter* terhadap B-*frame*, dan $\dot{\mathbf{R}}_{\boldsymbol{\Theta}}$ adalah turunan pertama matriks rotasi. Dan simbol × merupakan perkalian produk suatu vektor.

Berdasarkan aksioma kedua Euler dari hukum kedua Newton,

dengan cara yang sama, turunan dari gerakan angular komponen dari suatu benda dapat dilihat pada Persamaan (2.26).

$$\begin{cases} \boldsymbol{I\ddot{\Theta}^{E}} = \boldsymbol{\tau^{E}} \\ \boldsymbol{I\ \dot{\mathbf{T}}_{\Theta}\omega^{B}} = \mathbf{T}_{\Theta}\boldsymbol{\tau^{B}} \\ \boldsymbol{I\dot{\omega}^{B}} + \boldsymbol{\omega^{B}\, x\, (I\omega^{B})} = \boldsymbol{\tau^{B}} \end{cases} \tag{2.26}$$

Pada Persamaan (2.26), $\boldsymbol{I}$ (N m s^2) adalah matriks inersia *quadcopter* (pada B-*frame*), $\ddot{\boldsymbol{\Theta}}^{\boldsymbol{E}}$ (rad s^{-2}) adalah vektor percepatan sudut *quadcopter* terhadap E-*frame*, $\dot{\boldsymbol{\omega}}^{\boldsymbol{B}}$ (rad s^{-2}) adalah vektor percepatan sudut *quadcopter* terhadap B-*frame*, dan $\boldsymbol{\tau}^{E}$ (N m) adalah torsi *quadcopter* terhadap E-*frame*.

Dari Persamaan (2.25) dan (2.26), dapat dibentuk persamaan benda kaku dengan 6 derajat kebebasan (DOF). Persamaan (2.27) menunjukkan formulasi matriks dari dinamika sistem.

$$\begin{bmatrix} m\,\boldsymbol{I_{3\,x\,3}} & 0_{3x3} \\ 0_{3x3} & \boldsymbol{I} \end{bmatrix} \begin{bmatrix} \dot{\mathbf{V}}^{\mathbf{B}} \\ \dot{\boldsymbol{\omega}}^{\mathbf{B}} \end{bmatrix} + \begin{bmatrix} \boldsymbol{\omega}^{\mathbf{B}} \text{ x } (\boldsymbol{m}\mathbf{V}^{\mathbf{B}}) \\ \boldsymbol{\omega}^{\mathbf{B}} \text{ x } (\boldsymbol{I}\boldsymbol{\omega}^{\mathbf{B}}) \end{bmatrix} = \begin{bmatrix} \boldsymbol{F^{B}} \\ \boldsymbol{\tau^{B}} \end{bmatrix} \tag{2.27}$$

Dimana notasi I_{3x3} adalah matriks identitas 3x3. Dan 0_{3x3} adalah matriks nol 3x3. *quadcopter* memiliki empat buah motor yang menghasilkan gaya dorong seperti pada Gambar 2.6.

Vektor gaya yang terjadi pada *quadcopter* dapat ditentukan berdasarkan Persamaan (2.28).

$$\boldsymbol{\Lambda} = [\boldsymbol{F^{B}}\ \ \boldsymbol{\tau^{B}}]^{T} = [\boldsymbol{F_x}\ \ \boldsymbol{F_y}\ \ \boldsymbol{F_z}\ \ \boldsymbol{\tau_x}\ \ \boldsymbol{\tau_y}\ \ \boldsymbol{\tau_z}]^{T} \tag{2.28}$$

Persamaan (2.28) dapat juga ditulis dalam bentuk formulasi matriks seperti pada Persamaan (2.29).

$$\boldsymbol{M_B\dot{v}} + \boldsymbol{C_B(v)v} = \boldsymbol{\Lambda} \tag{2.29}$$

Dengan $\dot{\boldsymbol{v}}$ adalah vektor percepatan *quadcopter* terhadap B-*frame*. $\boldsymbol{M_B}$ adalah matriks inersia sistem dan $\boldsymbol{C_B}$ adalah adalah matriks sentripetal Coriolis. Persamaan (2.30) menunjukkan matriks $\boldsymbol{M_B}$.

$$\boldsymbol{M_B} = \begin{bmatrix} m\, I_{3x3} & 0_{3x3} \\ 0_{3x3} & I \end{bmatrix} = \begin{bmatrix} m & 0 & 0 & 0 & 0 & 0 \\ 0 & m & 0 & 0 & 0 & 0 \\ 0 & 0 & m & 0 & 0 & 0 \\ 0 & 0 & 0 & I_{XX} & 0 & 0 \\ 0 & 0 & 0 & 0 & I_{YY} & 0 \\ 0 & 0 & 0 & 0 & 0 & I_{ZZ} \end{bmatrix} \quad (2.30)$$

Dapat dilihat bahwa matriks $\boldsymbol{M_B}$ adalah matriks diagonal, sedangkan matriks $\boldsymbol{C_B}$ ditunjukkan oleh Persamaan (2.31).

$$\boldsymbol{C_B} = \begin{bmatrix} \mathbf{0_{3x3}} & -m\, \boldsymbol{S}(\mathbf{V_B}) \\ \mathbf{0_{3x3}} & -\boldsymbol{S}(\boldsymbol{I}\, \boldsymbol{\omega_B}) \end{bmatrix}$$

$$\boldsymbol{C_B} = \begin{bmatrix} 0 & 0 & 0 & 0 & m\,w & -m\,v \\ 0 & 0 & 0 & -m\,w & 0 & m\,u \\ 0 & 0 & 0 & m\,v & -m\,u & 0 \\ 0 & 0 & 0 & 0 & I_{ZZ}\,r & -I_{YY}\,q \\ 0 & 0 & 0 & -I_{ZZ}\,r & 0 & I_{XX}\,p \\ 0 & 0 & 0 & I_{YY}\,q & -I_{XX}\,p & 0 \end{bmatrix} \quad (2.31)$$

Pada persamaan ini, diadopsi operator *skew-symmetric* $\boldsymbol{S(.)}$. Untuk vektor 3 dimensi **k**, matriks *skew-symmetric* dari **k** ($\boldsymbol{S(k)}$) ditunjukkan pada Persamaan (2.32).

$$\boldsymbol{S(k)} = -\boldsymbol{S^T(k)} = \begin{bmatrix} 0 & -k3 & k1 \\ k3 & 0 & -k1 \\ -k2 & k1 & 0 \end{bmatrix}, \boldsymbol{k} = \begin{bmatrix} k1 \\ k2 \\ k3 \end{bmatrix} \quad (2.32)$$

Matriks $\boldsymbol{\Lambda}$ dapat dibagi menjadi tiga komponen menurut kontribusi sifat dasar *quadcopter*. Percepatan gravitasi hanya berlaku pada persamaan linear *quadcopter*. Persamaan matriks $\boldsymbol{G_B}(\boldsymbol{\xi})$ dituliskan pada Persamaan (2.33).

$$\boldsymbol{G_B}(\boldsymbol{\xi}) = \begin{bmatrix} F_{GB} \\ 0_{3x1} \end{bmatrix} = \begin{bmatrix} \mathrm{R}_{\Theta}{}^{-1} F_{GE} \\ 0_{3x1} \end{bmatrix}$$

$$= \begin{bmatrix} \mathbf{R_\Theta}^{\mathbf{T}} \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} \\ 0_{3x1} \end{bmatrix} = \begin{bmatrix} mg\, sin\, \theta \\ -mg\, cos\, \theta\, sin\, \phi \\ -\, mg\, cos\, \theta\, sin\, \phi \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad (2.33)$$

di mana $\boldsymbol{F_{GB}}$ (N) adalah vektor gaya gravitasi terhadap B-*frame* dan $\boldsymbol{F_{GE}}$ (N) terhadap E-*frame*. $\mathbf{R_\Theta}$ adalah matriks ortogonal, sehingga matriks inversnya sama dengan matriks transposnya.

Kontribusi yang kedua mengenai efek giroskopis yang dihasilkan oleh perputaran *propeller*. Selama dua *propeller* yang berhadapan berputar searah jarum jam dan dua *propeller* lainnya berputar berlawanan arah jarum jam, terjadi ketidak-seimbangan saat jumlah aljabar dari kecepatan motor tidak sama dengan nol. Jika *roll* dan *pitch* juga tidak nol, *quadcopter* akan mengalami torsi efek giroskopis yang ditunjukkan pada Persamaan (2.34).

$$\boldsymbol{O_B(v)\Omega} = \begin{bmatrix} 0_{3x1} \\ -\sum_{k=1}^{4} \boldsymbol{J_{TP}} \left(\boldsymbol{\omega^B} \times \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \right) (-1)^k \boldsymbol{\Omega_k} \end{bmatrix} = \begin{bmatrix} \mathbf{0_{3x1}} \\ \boldsymbol{J_{TP}} \begin{bmatrix} -q \\ p \\ 0 \end{bmatrix} \boldsymbol{\Omega} \end{bmatrix}$$

$$= \boldsymbol{J_{TP}} \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -q & -q & -q & -q \\ -p & p & -p & p \\ 0 & 0 & 0 & 0 \end{bmatrix} \boldsymbol{\Omega} \qquad (2.34)$$

$\boldsymbol{O_B}$ adalah matriks giroskopis dari *propeller* dan J_{TP} (N m s^2) adalah momen inersia total di sekitar sumbu *propeller*. Efek giroskopis yang dihasilkan oleh putaran *propeller* hanya berhubungan dengan persamaan angular, dan tidak mempengaruhi persamaan linear. $\boldsymbol{\Omega}$ (rad s^{-1}) merupakan vektor kecepatan putar *propeller*, yang ditunjukkan oleh Persamaan (2.35).

$$\boldsymbol{\Omega} = -\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4, \qquad \boldsymbol{\Omega} = \begin{bmatrix} \Omega_1 \\ \Omega_2 \\ \Omega_3 \\ \Omega_4 \end{bmatrix} \qquad (2.35)$$

di mana Ω_1, Ω_2, Ω_3, dan Ω_4 (rad s^{-1}) adalah kecepatan *propeller* depan, kanan, belakang, dan kiri.

Kontribusi yang ketiga adalah perhitungan gaya dan torsi yang dihasilkan oleh pergerakan *input*. Pergerakan matriks $\boldsymbol{E_B}$ dikali dengan $\boldsymbol{\Omega^2}$ untuk memperoleh vektor perpindahan $\mathbf{U_B(\Omega)}$. Efek aerodinamis

(factor *thrust* b (N s^2) dan *drag* d (N m s^2)) berpengaruh pada gaya dan torsi yang dihasilkan. Persamaan vektor perpindahan pada dinamika *quadcopter* ditunjukkan pada Persamaan (2.36).

$$\boldsymbol{U_B(\Omega)} = \boldsymbol{E_B \Omega^2} = \begin{bmatrix} 0 \\ 0 \\ U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ b(\boldsymbol{\Omega_1}^2 + \boldsymbol{\Omega_2}^2 + \boldsymbol{\Omega_3}^2 + \boldsymbol{\Omega_4}^2) \\ bl(-\boldsymbol{\Omega_2}^2 + \boldsymbol{\Omega_4}^2) \\ bl(-\boldsymbol{\Omega_1}^2 + \boldsymbol{\Omega_3}^2) \\ d(-\boldsymbol{\Omega_1}^2 + \boldsymbol{\Omega_2}^2 - \boldsymbol{\Omega_3}^2 + \boldsymbol{\Omega_4}^2) \end{bmatrix} \quad (2.36)$$

di mana l (m) adalah jarak antara pusat *quadcopter* dengan pusat *propeller*. U_1, U_2, U_3, dan U_4 adalah komponen vektor gerak. Hubungan komponen vektor tersebut dengan *propeller* dapat diperoleh dari perhitungan aerodinamis.

Dari persamaan sebelumnya, dapat diketahui konstanta matriks $\boldsymbol{E_B}$ yang dikalikan dengan kuadrat dari kecepatan *propeller* $\boldsymbol{\Omega}^2$ sehingga menghasilkan $\boldsymbol{U_B(\Omega)}$. Persamaan (2.37) menunjukkan matriks $\boldsymbol{E_B}$.

$$\boldsymbol{E_B} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ b & b & b & b \\ 0 & -bl & 0 & bl \\ -bl & 0 & bl & 0 \\ -d & d & -d & d \end{bmatrix} \quad (2.37)$$

Dari Persamaan (2.29), dinamika *quadcopter* dapat dibuat dengan mempertimbangkan ketiga kontribusi menurut Persamaan (2.38).

$$\boldsymbol{M_B \dot{v} + C_B(v)v = G_B(\xi) + O_B(v)\,\Omega + E_B \Omega^2} \quad (2.38)$$

Dengan mengatur ulang Persamaan (2.38) diatas, turunan dari vektor kecepatan terhadap B-*frame* dapat dirumuskan sebagai berikut:

$$\boldsymbol{\dot{v} = M_B^{-1}\,(C_B(v)v + G_B(\xi) + O_B(v)\,\Omega + E_B \Omega^2)} \quad (2.39)$$

Persamaan (2.40) menunjukkan persamaan-persamaan sebelumnya bukan dalam bentuk matriks, melainkan dalam bentuk sistem Persamaan (2.40).

$$\begin{cases} \dot{u} = (v\,r - w\,q) + g\,s_\theta \\ \dot{v} = (w\,p - u\,r) - g\,c_\theta s_\phi \\ \dot{w} = (u\,q - v\,p) - g\,c_\theta s_\phi + \frac{U_1}{m} \\ \dot{p} = \frac{I_{YY} - I_{ZZ}}{I_{XX}} q\,r - \frac{J_{TP}}{I_{XX}} q\,\Omega + \frac{U_2}{I_{XX}} \\ \dot{q} = \frac{I_{ZZ} - I_{XX}}{I_{YY}} p\,r - \frac{J_{TP}}{I_{YY}} p\,\Omega + \frac{U_3}{I_{YY}} \\ \dot{r} = \frac{I_{XX} - I_{YY}}{I_{ZZ}} p\,q - \frac{U_4}{I_{ZZ}} \end{cases} \quad (2.40)$$

Persamaan *input* kecepatan *propeller* (baling-baling) dapat dilihat dalam Persamaan (2.41).

$$\begin{cases} U_1 = b(\boldsymbol{\Omega}_1^2 + \boldsymbol{\Omega}_2^2 + \boldsymbol{\Omega}_3^2 + \boldsymbol{\Omega}_4^2) \\ U_2 = bl(-\boldsymbol{\Omega}_2^2 + \boldsymbol{\Omega}_4^2) \\ U_3 = bl(-\boldsymbol{\Omega}_1^2 + \boldsymbol{\Omega}_3^2) \\ U_4 = d(-\boldsymbol{\Omega}_1^2 + \boldsymbol{\Omega}_2^2 - \boldsymbol{\Omega}_3^2 + \boldsymbol{\Omega}_4^2) \\ \Omega = -\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4 \end{cases} \quad (2.41)$$

Persamaan sistem dinamik *quadcopter* (2.40) diperoleh dari B-*frame*. Dalam pemodelannya, dibutuhkan persamaan dinamik sistem yang mencakup sistem *hybrid* yang disusun oleh persamaan linear dari E-*frame* dan persamaan angular dari B-*frame*.Hal ini dilakukan untuk mempermudah pengaturan. Persamaan-persamaan berikut akan disajikan dalam kerangka "*hybrid*", atau H-*frame*. Persamaan (2.42) menunjukkan vektor kecepatan *quadcopter* yang digeneralisasi pada H-*frame*.

$$\dot{\boldsymbol{\xi}} = [\dot{\boldsymbol{\Gamma}}^{\mathbf{E}} \quad \boldsymbol{\omega}^{\boldsymbol{B}}]^{\boldsymbol{T}} = [\dot{X} \quad \dot{Y} \quad \dot{Z} \quad p \quad q \quad r]^{\boldsymbol{T}} \quad (2.42)$$

Dinamika sistem pada H-*frame* dapat dituliskan dalam bentuk martriks menurut Persamaan (2.43).

$$\boldsymbol{M_H}\dot{\boldsymbol{\xi}} + \boldsymbol{C_H}(\boldsymbol{\xi})\boldsymbol{\xi} = \boldsymbol{G_H} + \boldsymbol{O_H}(\boldsymbol{\xi})\,\boldsymbol{\Omega} + \boldsymbol{E_H}(\boldsymbol{\xi})\boldsymbol{\Omega^2} \quad (2.43)$$

Berikut ini akan ditentukan semua matriks dan vektor yang ditunjukkan oleh persamaan di atas. Matriks inersia sistem terhadap H-

frame $\boldsymbol{M}_H$ sama dengan matriks inersia sistem terhadap B-*frame*, dan ditentukan seperti pada Persamaan (2.44).

$$\boldsymbol{M_H} = \boldsymbol{M_B} = \begin{bmatrix} m & 0 & 0 & 0 & 0 & 0 \\ 0 & m & 0 & 0 & 0 & 0 \\ 0 & 0 & m & 0 & 0 & 0 \\ 0 & 0 & 0 & I_{XX} & 0 & 0 \\ 0 & 0 & 0 & 0 & I_{YY} & 0 \\ 0 & 0 & 0 & 0 & 0 & I_{ZZ} \end{bmatrix} \tag{2.44}$$

Namun, matriks sentripetal Coriolis terhadap H-*frame* $\boldsymbol{C}_H(\boldsymbol{\zeta})$ tidak sama dengan matriks sentripetal Coriolis terhadap B-*frame* dan ditentukan seperti pada Persamaan (2.45).

$$\boldsymbol{C_H}(\boldsymbol{\xi}) = \begin{bmatrix} 0_{3x3} & 0_{3x3} \\ 0_{3x3} & -S(I\ \omega_B) \end{bmatrix} \tag{2.45}$$

Vektor gravitasi terhadap H-*frame* $\boldsymbol{G}_H$ dituliskan pada Persamaan (2.46). Dapat dilihat bahwa gravitasi mempengaruhi ketiga persamaan linear, namun lebih berpengaruh terhadap ketinggian *quadcopter*.

$$\boldsymbol{G_H} = \begin{bmatrix} F_G^E \\ 0_{3x1} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -mg \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{2.46}$$

Efek giroskopis yang dihasilkan oleh putaran *propeller* tidak berubah karena hanya mempengaruhi persamaan angular yang mengacu kepada B-*frame*. Oleh karena itu, matriks giroskopis terhadap H-*frame* $\boldsymbol{O}_H(\boldsymbol{\zeta})$ dibuat sama dengan Persamaan (2.47).

$$\boldsymbol{O_H}(\boldsymbol{\xi})\boldsymbol{\Omega} = \boldsymbol{O_B}(\boldsymbol{v})\,\boldsymbol{\Omega} = \begin{bmatrix} 0_{3x1} \\ J_{TP} \begin{bmatrix} -q \\ p \\ 0 \end{bmatrix} \Omega \end{bmatrix}$$

$$= \boldsymbol{J_{TP}} \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ q & -q & q & -q \\ -p & p & -p & p \\ 0 & 0 & 0 & 0 \end{bmatrix} \boldsymbol{\Omega} \tag{2.47}$$

Matriks perpindahan terhadap H-*frame* $\boldsymbol{E_H(\xi)}$ berbeda dengan perpindahan terhadap B-*frame* karena *input* U_1 mempengaruhi semua persamaan linear dengan matriks rotasi $\mathbf{R_\Theta}$. Perkalian produk antara matriks perpindahan dengan kuadrat kecepatan *propeller* ditunjukkan pada Persamaan (2.48).

$$\boldsymbol{E_H(\xi)\Omega^2} = \begin{bmatrix} \boldsymbol{R_\Theta} & \boldsymbol{0_{3x3}} \\ \boldsymbol{0_{3x3}} & \boldsymbol{I_{3x3}} \end{bmatrix} \boldsymbol{E_B\Omega^2}$$
$$= \begin{bmatrix} (\boldsymbol{s_\psi s_\phi + c_\psi s_\theta c_\phi})\boldsymbol{U_1} \\ (\boldsymbol{-c_\psi s_\phi + s_\psi s_\theta c_\phi})\boldsymbol{U_1} \\ (\boldsymbol{c_\theta c_\phi})\,\boldsymbol{U_1} \\ \boldsymbol{U_2} \\ \boldsymbol{U_3} \\ \boldsymbol{U_4} \end{bmatrix} \tag{2.48}$$

Dengan menyusun kembali Persamaan (2.43), dapat ditemukan rumus untuk turunan vektor kecepatan yang digeneralisasi terhadap H-*frame* yang dapat dilihat pada Persamaan (2.49).

$$\dot{\boldsymbol{\xi}} = \boldsymbol{M_H}^{-1}(\boldsymbol{C_H(\xi)\xi + G_H + O_H(\xi)\,\Omega + E_H(\xi)\Omega^2}) \tag{2.49}$$

2.1.4 Model Matematika [4]

Dari analisis kinematika dan dinamika diperoleh Persamaan model matematika dari *quadcopter* seperti pada Persamaan (2.50).

Dengan melihat secara sederhana pada Persamaan (2.50), posisi pada sumbu Z, dan posisi sudut *roll*, *pitch*, *yaw* dapat dikontrol secara langsung, berturut-turut dengan menggunakan U_1, U_2, U_3, dan U_4. Kontrol pada posisi maju (X), dan menyamping (Y) dapat dilakukan dengan mengatur sudut *pitch* dan *roll* dengan syarat gaya angkat (U_1) tidak sama dengan nol.

$$\begin{cases} \ddot{X} = \dfrac{U_1}{m}(\cos\Psi \sin\theta\cos\phi + \sin\Psi \sin\phi) \\ \ddot{Y} = \dfrac{U_1}{m}(\sin\Psi \sin\theta\cos\phi - \cos\Psi \sin\phi) \\ \ddot{Z} = -g + \dfrac{U_1}{m}(\cos\theta\cos\phi) \\ \dot{p} = \dfrac{I_{yy} - I_{zz}}{I_{xx}} qr - \dfrac{I_r}{I_{xx}} q\Omega + \dfrac{U_2}{I_{xx}} \\ \dot{q} = \dfrac{I_{zz} - I_{xx}}{I_{yy}} pr + \dfrac{I_r}{I_{yy}} q\Omega + \dfrac{U_2}{I_{yy}} \\ \dot{r} = \dfrac{I_{xx} - I_{yy}}{I_{zz}} pq + \dfrac{U_4}{I_{zz}} \end{cases} \quad (2.50)$$

Nilai *input* dari *quadcopter* merupakan gaya angkat tiap *propeller*, yang telah dimodelkan secara teoritis dalam Persamaan (2.51), adalah sebagai berikut:

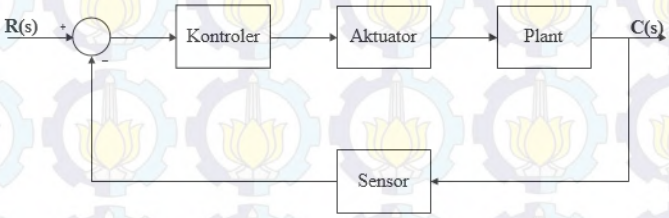
$$\begin{cases} U_1 = b(\Omega_1^{\ 2} + \Omega_2^{\ 2} + \Omega_3^{\ 2} + \Omega_4^{\ 2}) \\ U_2 = bl(-\Omega_2^{\ 2} + \Omega_4^{\ 2}) \\ U_3 = bl(-\Omega_1^{\ 2} + \Omega_3^{\ 2}) \\ U_4 = d(-\Omega_1^{\ 2} + \Omega_2^{\ 2} - \Omega_3^{\ 2} + \Omega_4^{\ 2}) \\ \Omega = -\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4 \end{cases} \quad (2.51)$$

2.2 Kontroler

Dalam sebuah sistem, kontroler memiliki kontribusi yang besar terhadap perilaku sistem. Pada prinsipnya hal itu disebabkan oleh tidak dapat diubahnya komponen penyusun sistem tersebut. Dengan artian bahwa karakteristik *plant* harus diterima sebagaimana adanya, sehingga perubahan perilaku sistem hanya dapat dilakukan melalui penambahan suatu *sub*-sistem, yaitu kontroler. Salah satu tugas komponen kontroler adalah meminimalkan sinyal kesalahan, yaitu perbedaan antara sinyal referensi dan sinyal aktual. Hal ini sesuai dengan tujuan sistem kontrol yaitu memperoleh sinyal aktual yang sama dengan sinyal referensi.

Semakin cepat reaksi sistem (sinyal aktual) mengikuti sinyal referensi dan semakin kecil kesalahan yang terjadi, semakin baik kinerja

sistem kontrol yang diterapkan. Letak kontroler dapat dilihat pada blok diagram *closed loop* pada Gambar 2.10. Namun hal tersebut dapat bervariasi sesuai dengan kebutuhan desain.



Gambar 2. 10 Blok Diagram *Closed Loop*

Pada penelitian ini digunakan kontroler *Linear Quadratic Regulator* (LQR) yang digunakan untuk pengendalian gerak rotasi. Sedangkan untuk pengendalian gerak translasi digunakan kontroler *Linear Quadratic Tracking* (LQT) dan *Algortima Genetika* (GA) yang digunakan untuk *tuning* parameter dari kontroler tersebut.

2.2.1 *Linear Quadratic Regulator* (LQR) [5]

LQR merupakan salah satu kontrol optimal di mana mempunyai maksud hasil paling baik yang dapat dicapai dengan memperhatikan kondisi dan kendala dari suatu sistem. Dalam sistem kontrol optimal, istilah optimal sering kali merujuk pada minimal, misalnya meminimalkan bahan bakar (*input*), waktu, dan kesalahan (*error*). Kontrol optimal secara umum digunakan untuk memilih *input* plant *u* dengan indeks performansi yang minimum.

LQR disebut linear karena model dan bentuk kontrolernya berupa linear. Sedangkan disebut kuadratik karena *cost function*-nya adalah kuadratik dan karena referensinya bukanlah berupa fungsi waktu maka disebut regulator.

Sistem kontrol yang baik adalah sistem kontrol yang mempunyai daya tanggap yang cepat dan stabil, tetapi tidak memerlukan energi yang berlebihan. Sistem kontrol demikian dapat dicapai melalui pengaturan indeks performansi yang tepat. Sistem kontrol yang dirancang berdasarkan optimasi indeks performansi disebut sistem kontrol optimal.

Pada suatu sistem, indeks performansi dipilih sesuai dengan bagian yang akan dioptimalkan. Bentuk umum dari persamaan *state* sistem linear ditunjukkan oleh Persamaan (2.52).

$$\dot{x}(t) = Ax(t) + Bu(t)$$
$$y(t) = Cx(t) \tag{2.52}$$

Dimana memiliki maksud sebagai berikut

x_{n+1}	: *State* Sistem
u_{m*n}	: *State input*
y_{l+1}	: *State output*
A	: Matriks Sistem A_{n*n}
B	: Matriks *Input* B_{n*m}
C	: Matriks *Output* C_{l*n}

Indeks performansi dari minimum energi (*cost function/quadratic function*) pada interval $[t_0\ t_f]$ ditunjukkan oleh Persamaan (2.53).

$$J = \frac{1}{2}\int_0^{\infty} (x^T Q + u^T Ru)\, dt \tag{2.53}$$

Di mana memiliki keterangan sebagai berikut

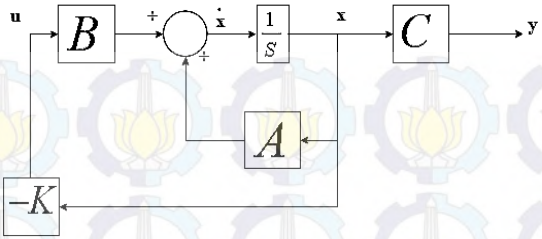
t_0 : waktu awal
∞ : waktu akhir
Q : matriks semidefinit positif
R : matriks definit positif

Pemilihan pembobot Q dan R berpedoman pada:

a. Semakin besar harga Q, semakin memperbesar harga elemen penguatan K sehingga mempercepat sistem untuk mencapai keadaan tunak (*intermediate state cost function*).
b. Semakin besar harga R, maka akan memperkecil harga penguatan K dan memperlambat keadaan tunak (*energy drive*).

Persamaan regulator dapat diselesaikan dengan menyelesaikan persamaan aljabar *Riccati* sesuai dengan Persamaan (2.54).

$$A^T P + PA - PBR^{-1}B^T P + Q = 0 \tag{2.54}$$



Gambar 2.11 Blok Diagram LQR

Blok diagram dari LQR ditunjukkan oleh Gambar 2.11 di mana nilai *gain feedback* dapat dicari dengan menggunakan Persamaan (2.55) sehingga dapat diketaui nilai sinyal kontrol sesuai dengan Persamaan (2.56).

$$-K=R^{-1}B^{T}P \tag{2.55}$$

$$u=-Kx \tag{2.56}$$

2.2.2 *Linear Quadratic Tracking* (LQT) [5]

LQT merupakan sistem pengaturan linear yang *output* sistem mengikuti referensi (*trajectory*) yang diinginkan. Suatu sistem mempunyai persamaan *state* seperti Persamaan (2.52) vektor kesalahan seperti Persamaan (2.57).

$$e(t)=z(t)-y(t) \tag{2.57}$$

Di mana *e(t)* merupakan *error* yang merupakan selisih antara *input* referensi yang diberikan *z(t)* dan *output* dari sistem *y(t)*. Indeks performansi didefinisikan pada Persamaan (2.58).

$$J=\frac{1}{2}e'(t_f)F(t_f)e(t_f)+\frac{1}{2}\int_{t_0}^{t_f}\left[e'(t)Q(t)e(t)+u'(t)R(t)u(t)\right]dt \tag{2.58}$$

Diasumsikan $F(t_f)$ dan $Q(t)$ adalah skalar atau matriks semidefinit positif yang simetris dengan dimensi ($m \times m$), $R(t)$ adalah matriks definit

positif yang juga simetris ($R \times R$) matrix Q dan R adalah matriks pembobot yang menentukan kinerja dari sistem yang akan dikontrol.

Setelah mendapatkan model matematika sistem dalam bentuk *state-space* dan matriks *A(t)* dan *B(t)*, matriks penyelesaian persamaan *differential Riccati* dapat didapatkan dengan Persamaan (2.59) untuk *finite-time case* dan Persamaan (2.60) untuk *infinite-time case*.

$$\dot{P}(t) = -P(t)A(t) - A'(t)P(t) + P(t)B(t)R^{-1}(t)B'(t)P(t) + C'(t)Q(t)C(t) \tag{2.59}$$

$$0 = -P(t)A(t) - A'(t)P(t) + P(t)B(t)R^{-1}(t)B'(t)P(t) + C'(t)Q(t)C(t) \tag{2.60}$$

Matriks Q dan R dapat diasumsikan sesuai dengan performansi yang diinginkan sistem. Setelah mendapatkan persamaan Riccati, persamaan diferensial vektor *non*-homogen dapat dicari menggunakan Persamaan (2.61).

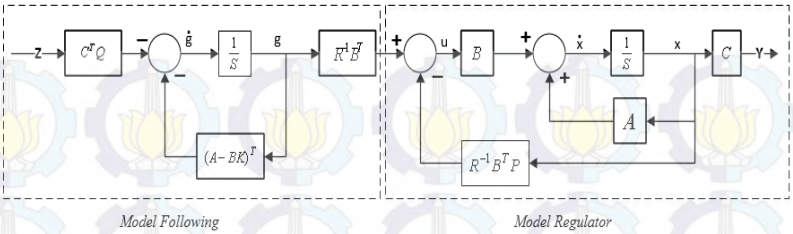
$$\dot{g}(t) = -[A(t) - B(t)R^{-1}B'(t)P(t)]'g(t) - C'(t)Q(t)z(t) \tag{2.61}$$

Setelah mendapatkan matriks *P(t)* yang merupakan matriks definit positif yang simetris dan *g(t)* , nilai *gain feedback K(t),* dan $u^*(t)$ dapat dicari dengan menggunakan Persamaan (2.62)-(2.63).

$$K(t) = R^{-1}(t)B'(t)P(t) \tag{2.62}$$

$$u^*(t) = -K(t)x^*(t) + R^{-1}(t)B'(t)g(t) \tag{2.63}$$

Blok diagram LQT ditunjukkan oleh Gambar 2.12 di mana terdiri dari 2 bagian, yaitu *model following* dan *model regulator*.



Gambar 2. 12 Blok Diagram LQT

2.2.3 Algoritma Genetika (GA) [6] [7]

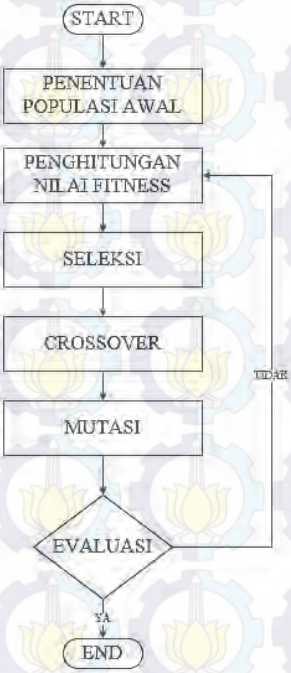
Algoritma genetika adalah algoritma komputasi yang diinspirasi dari teori evolusi, di mana kemudian diadopsi menjadi algoritma komputasi untuk mencari solusi suatu permasalahan optimasi. GA pertama kali dicetuskan oleh John Holland, Profesor University of Michigan pada tahun 1970 dan dikemabangkan oleh Goldberg.

GA yang dikembangkan oleh Goldberg adalah yang menyatakan bahwa kelangsungan hidup suatu makhluk dipengaruhi aturan "yang kuat adalah yang menang". Darwin juga menyatakan bahwa kelangsungan tersebut dapat dipertahankan melalui proses reproduksi, *crossover*, dan mutasi dan kemudian diadopsi menjadi algoritma komputasi untuk mencari solusi suatu permasalahan dengan cara yang lebih alamiah.

Kromosom baru yang disebut dengan *off spring*, dibentuk dengan cara melakukan perkawinan antar kromosom-kromosom dalam satu generasi yang disebut sebagai proses *crossover*. Jumlah kromosom dalam populasi yang mengalami *crossover* ditetukan oleh paramater yang disebut dengan *crossover rate*. Mekanisme perubahan susunan genetika akibat adanya faktor alam disebut dengan mutasi. Jumlah gen dalam populasi yang mengalami mutasi ditentukan oleh parameter yang dinamakan *mutation rate*. Setelah beberapa generasi akan dihasilkan kromosom yang nilainya konvergen pada suatu nilai, di mana merupakan solusi terbaik yang dihasilkan oleh GA terhadap permasalahan yang ingin diselesaikan.

Sebuah kromosom dibentuk dari komponen-komponen penyusun yang disebut sebagai gen dan nilainya dapat berupa bilangan numerik, biner, simbol, ataupun karakter tergantung dari permasalahan yang ingin diselesaikan. Kromosom akan berevolusi secara berkelanjutan yang disebut dengan generasi. Dalam tiap generasi kromosom tersebut dievaluasi tingkat keberhasilan nilai solusinya terhadap masalah yang

ingin diselesaikan (fungsi objektif) menggunakan ukuran yang disebut dengan *fitness*. Fungsi *fitness* merupakan suatu fungsi yang ingin dicari nilai optimalnya. Untuk memilih kromosom yang tetap dipertahankan untuk generasi selanjutnya, dilakukan proses yang disebut dengan seleksi. Proses seleksi kromosom menggunakan konsep aturan evolusi Darwin yang telah disebutkan sebelumnya yaitu kromosm yang mempunyai nilai *fitness* tinggi akan memiliki peluang lebih besar untuk terpilih lagi pada generasi selanjutnya. *Flowchart* pada proses GA ditunjukkan pada Gambar 2.13.



Gambar 2. 13 *Flowchart* Proses GA

Pada bidang kontrol, GA dapat digunakan untuk meningkatkan performansi sistem. Salah satunya ialah pada metode *Linear Quadratic Tracking* (LQT). Tujuan dari metode LQT ialah untuk memperoleh aksi kontrol optimal yang meminimalisasi indeks performansi dan memerintahkan *plant* agar dapat melakukan *tracking* sesuai dengan

model (*trajectory*) yang telah ditentukan. Untuk mendapatkan aksi kontrol yang optimal bergantung pada besarnya nilai Q dan R. Nilai Q dan R didapatkan dengan hasil *tuning* manual (*try and error*).

Metode *tuning try and error* sering kali didapatkan hasil kontrol yang tidak optimal. Untuk mendapatkan kombinasi nilai Q dan R digunakan metode *tuning* menggunakan GA. Setelah didapatkan nilai Q dan R yang optimal, maka akan didapatkan nilai *gain feedback* yang optimal sehingga didapatkan hasil desain yang optimal.

2.3 Linearisasi [8]

Suatu sistem yang kompleks memiliki hubungan antara besaran-besaran fisik dalam sistem bersifat nonlinear. Untuk menyelesaikan persoalan nonlinear, dapat digunakan dengan menggunakan pendekatan persamaan linear. Salah satu kharakteristik dari sistem nonlinear adalah ketergantungan perilaku respon sistem pada besaran dan *input* dari sistem.

Quadcopter merupakan sistem yang memiliki hubungan antar variable bersifat nonlinear. Pada Persamaan (2.27) terlihat bahwa persamaan tersebut terdapat hubungan antar variable bersifat nonlinear karena terdapat fungsi trigonometri. Untuk dapat melakukan proses kontrol yang menggunakan kontroler linear yaitu LQR dan LQT, maka model matematika tersebut harus dilinearisasi. Teori linearisasi digunakan untuk mengatasi permasalahan sistem nonlinear agar dapat didekati dengan sistem linear.

Pada penelitian ini digunakan teknik linearisasi *feedback linearization* di mana merupakan salah satu teknik linearisasi dengan melakukan modifikasi pada variabel yang dikontrol sehingga dapat didekati dengan sistem linear. Misalkan suatu sistem mempunyai persamaan *state* seperti Persamaan (2.64).

$$\begin{aligned} \dot{x} &= f(x) + G(x)u \\ y &= h(x) \end{aligned} \tag{2.64}$$

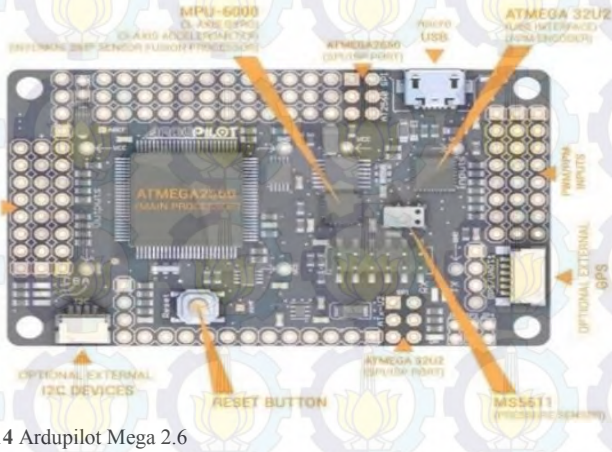
Domain tersebut mengandung *origin*, sehingga akan didapatkan sinyal kontrol modifikasi pada Persamaan (2.65).

$$u = \alpha(x) + \beta(x)v \tag{2.65}$$

2.4 Ardupilot Mega 2.6 [9]

Mikrokontroler akan diprogram menggunakan bahasa C. Mikrokontroler akan dipadukan menjadi modul *Ardupilot Mega* pada Gambar 2.14 Spesifikasi dari mikrokontroler ATMega 2560 adalah:

a. Memori : 256 KByte
b. Frekuensi *Clock* : Maksimum 16Mhz
c. I/O : 86 *port*
d. ADC : 10 bit/16 *port*
e. *Timer* 8 bit : 2 buah
f. *Timer* 16 Bit : 4 buah
g. Supplai tegangan : 4,5-5,5 VDC



Gambar 2. 14 Ardupilot Mega 2.6

Ardupilot Mega 2.6 merupakan modul yang terdapat sensor didalamnya, yaitu *gyro* dan *accelerometer* MPU 6050. Sensor tersebut akan digunakan secara bersamaan untuk menghitung sudut yang berotasi pada sumbu X, Y, dan Z. Sensor *accelerometer* dan *gyro* yang digunakan adalah MPU6000. Posisi sensor tersebut dapat dilihat pada Gambar 2.14. Sensor tersebut memiliki spesifikasi sebagai berikut:

a. Batas operasi : ± 3,6 g
b. Sensitivitas : 300 mV/g
c. Batas suplai tegangan : 1,8 – 5 Volt.

2.5 *Transmitter* dan *Receiver Radio* [10]

Transmitter dan *receiver* digunakan untuk operasi manual melakukan manuver pada *quadcopter*.

Gambar 2. 15 *Transmitter* dan *Receiver*

Transmitter berada pada sisi pengguna yaitu pada *remote control* untuk memberikan nilai *input* kepada *quadcopter*, sinyal tersebut akan diterima *receiver* dan diteruskan ke mikrokontroler berupa pulsa. Tampilan *transmitter* dan *receiver* radio seperti pada Gambar 2.15. Spesifikasi dari radio *transmiter* adalah:

a. Nama *Transmitter* : FrSky V8FR-II 2.4 Ghz 8 CH
b. *Channel* : 8 buah
c. Ukuran : 63,9 x 48,5 x 36,5 mm
d. Arus : 50 mA
e. Konsumsi daya : 60 mW

Disamping itu, spesifikasi untuk *receiver* adalah:

a. Nama *receiver* : FrSky V8FR-II 2.4 Ghz 8 CH
b. *Channel* : 8 buah
c. Ukuran : 44 x 24 x 14 mm
d. Berat : 9,3 gram
e. *Range* tegangan : 3,0 – 16 V
f. Arus : 30 mA
g. Range : 1,5 – 2,5 km

2.6 *Remote Control* Turnigy [11]

Remote control merupakan salah satu perangkat yang digunakan untuk mode penerbangan manual ditunjukkan oleh Gambar 2.16.

Gambar 2. 16 *Remote Control* (*Transmitter*) Turnigy 9XR

Remote control tersebut disambungkan dengan dengan radio *transmitter* sehingga dapat mengirimkan data pada mikrokontroler dan diterima oleh *radio receiver*. Spesifikasi dari *remote control* adalah

a. Nama : Turnigy 9XR Radio *Transmitter*
b. *Channel* : 24 buah
c. *Display* : 128 x 64 LCD
d. *Battery* : 3 *cell* lipo
e. *Stick mode* : 1, 2, 3, 4

2.7 *Electronic Speed Controller* (ESC) [12]

ESC merupakan *driver* dari motor *brushless* DC. ESC di-*trigger* oleh sinyal PWM yang dikendalikan oleh mikrokontroler ATMega 2560 situnjukkan oleh Gambar 2.17. Sinyal PWM tersebut akan men-*drive* motor dengan kecepatan yang linear dengan besar pulsa yang diberikan. Spesifikasi dari ESC adalah:

a. Nama : ESC TBS Bulletproof 30 *Ampere*
b. *Battery* : 2 – 4 cell
c. Arus Maksimum : 30 A
d. Tegangan : 6,4 – 16,8 V

Gambar 2. 17 ESC TBS Bulletproof 30 *Ampere*

2.8 Motor BLDC dan *Propeller* [13]

Motor yang digunakan sebagai aktuator adalah motor DC tanpa sikat SunnySk*y* X*2212 KV980* yang sumbunya dipasang *propeller* berukuran 10 cm x 4,5 cm. Bentuk motor DC tanpa sikat dan *propeller* terlihat seperti pada Gambar 2.18.

Gambar 2. 18 Motor *Quadcopter* dan *Propeller*

2.9 *Ground Station* [14]

Untuk dapat mengetahui data penerbangan dari *quadcopter* dapat digunakan *Ground station*. Data dari mikrokontroler dikirimkan secara *wireless* menuju komputer *ground station* dengan komunikasi serial menggunakan *telemetry radio 433 Mhz*. Data serial langsung dibaca pada *software Mission Planner* pada *ground station* yang ditunjukkan oleh Gambar 2.19.

Gambar 2. 19 *Telemetry* dan *Ground Station*

BAB III
PERANCANGAN SISTEM

Pada Bab ini membahas mengenai perancangan sistem kontrol *quadcopter* pada gerak lateral. Perancangan sistem terdiri dari spesifikasi sistem, identifikasi kebutuhan, perancang mekanik, perancangan elekronik, dan perancangan kontrol LQR, LQT, dan GA.

3.1 Spesifikasi Sistem

Pada penelitian Tugas Akhir ini menggunakan *quadcopter hobby*/rakitan yang memiliki 6 dof (*degree of freedom*), yang terdiri dari 3 dof rotasi dan 3 dof translasi yang akan dibuat menjadi spesifikasi sistem tertentu secara *hardware* maupun simulasi pada matlab dapat diuraikan sebagai berikut:

a. *Quadcopter* dapat bergerak rotasi maksimal 5,7 derajat
b. *Quadcopter* dapat melakukan gerak lateral sesuai dengan referensi yang diberikan
c. *Quadcopter* dapat digerakkan secara manual dengan menggunakan *remote control*.
d. Data-data sensor dan aktuator dikirimkan ke *ground station*.

3.2 Identifikasi Kebutuhan

Quadcopter yang digunakan harus mampu memenuhi spesifikasi yang ada. Untuk memenuhi spesifikasi sistem tersebut maka terdapat kebutuhan yang harus dipenuhi antara lain:

a. *Quadcopter* harus dibuat simetris, seimbang, dan ringan untuk menghindari kelembaman yang sangat besar.
b. Baterai yang digunakan minimal dapat membuat *quadcopter* terbang sekitar 15 menit.
c. *Remote control* minimal terdapat 4 kanal sebagai pergerakan naik/turun, *roll*, *pitch*, dan *yaw*
d. *Ground station* menggunakan *software Mission Planner*

3.3 Perancangan Mekanik

Sistem mekanik yang baik akan mendukung pergerakan *quadcopter* menjadi lebih baik. Untuk mendukung pergerakan tersebut, perancangan mekanik harus dibuat simetris, seimbang, dan ringan. Dalam hal ini *Center of Grafity* (COG) dari *quadcopter* harus diperhatikan. Berat

dari keempat sisi *quadcopter* juga harus sama untuk memenuhi kriteria COG. Selain itu juga posisi baterai harus terletak tepat di tengah titik massa *quadcopter* agar seimbang. Beban yang diberikan kepada *quadcopter* tidak boleh melebihi daya angkat dari *quadcopter* agar dapat terbang.

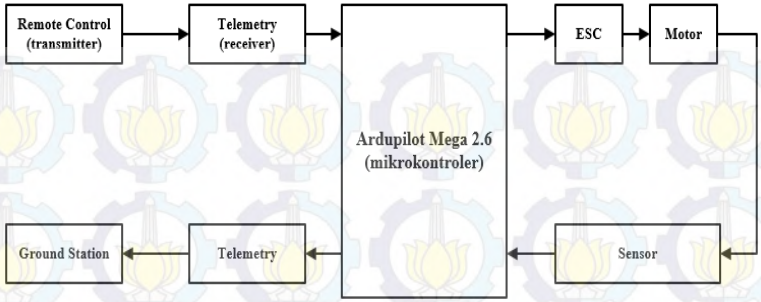
Pada penelitian Tugas Akhir ini, desain dari *quadcopter* yang telah dirancang ditunjukkan pada Gambar 3.1. *Quadcopter* tersebut merupakan *quadcopter* rakitan/*hobby* yang terdapat pada Lab B105 Teknik Pengaturan.

Gambar 3. 1 Desain Mekanik *Quadcopter* yang Telah Dirancang

3.4 Prancangan Elektronik

Perancangan elektronika yang ada pada *quadcopter* terdiri dari sistem kontroler yang berupa Ardupilot Mega 2.6 di mana telah sudah dilengkapi beberapa sensor dan *port* yang memudahkan untuk penggunaan sebagai komunikasi data.

Board Ardupilot Mega 2.6 ini dilengkapi dengan ATMega 2560 sebagai mikrokontroler dari *flight controller*. Selain itu juga ditambahkan GPS yang digunakan untuk mengetahui posisi dari *quadcopter*, *telemetry* 433 Mhz yang digunakan sebagai komunikasi data antara kontroler Ardupilot Mega 2.6 dengan *ground station* yang menggunakan *software Mission Planner*. Perancangan keseluruhan dari sistem *quadcopter* ditunjukkan pada Gambar 3.2.



Gambar 3. 2 Perancangan Sistem Pengendalian *Quadcopter*

3.5 Pemodelan Motor dan *Propeller*

Untuk mengetahui model linear dari *quadcopter*, maka dilakukan beberapa pengukuran, yaitu pengukuran untuk mengetahui model linear dari hubungan pulsa dengan kecepatan motor (rpm dan rad/s) dan pengukuran untuk mengetahui hubungan antara pulsa dengan gaya angkat motor (kg).

3.5.1 Pengukuran Kecepatan Motor

Gambar 3. 3 Pengkuruan Kecepatan Motor Terhadap Sinyal PWM

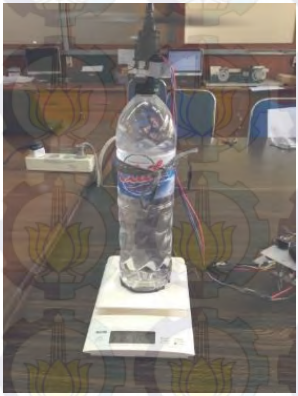
Pengukuran ini dilakukan dengan menggunakan tachometer laser dan menggunakan sinyal masukan berupa PWM dari Arduino.

Pengukuran ini dilakukan seperti pada Gambar 3.3 dan hasil pengukuran kecepatan motor ditampilkan pada Lampiran A1.

Dari hasil pengukuran diperoleh persamaan linear hubungan pulsa motor terhadap kecepatan (rad/s) pada masing-masing motor terdapat pada Persamaan (3.1).

$$\begin{aligned} y_1 &= 44,851x - 1760,5 \\ y_2 &= 44,10x - 1603,7 \\ y_3 &= 44,8x - 1670,4 \\ y_4 &= 45,047x - 1618,7 \end{aligned} \tag{3.1}$$

3.5.2 Pengukuran Gaya Angkat Motor



Gambar 3. 4 Pengkuran Gaya Angkat Motor Terhadap Sinyal PWM

Langkah-langkah dalam pengukuran gaya angkat motor adalah dengan meletakkan motor dan *propeller* tepat diatas beban sehingga memiliki berat total 1,618 kg. Beban tersebut merupakan botol aqua 1,5 liter dengan diisi air, terdapat pada Gambar 3.4. Kemudian diletakkan diatas timbangan digital dan motor dijalankan dengan memberikan pulsa atau sinyal PWM dari arduino. Hasil pembacaan berat pada timbangan dikurangi dengan beban 1,618 kg adalah gaya angkat pada masing-masing motor. Pada eksperimen ini dilakukan pengukuran gaya angkat pada masing-masing motor *propeller*.

Dari pengukuran diperoleh persamaan linear hubungan pulsa motor terhadap gaya angkat kg pada masing-masing motor terdapat pada Persamaan (3.2).

$$\begin{aligned} y_1 &= 0,0076x - 0,6097 \\ y_2 &= 0,0076x - 0,5968 \\ y_3 &= 0,0076x - 0,5968 \\ y_4 &= 0,0078x - 0,6026 \end{aligned} \tag{3.2}$$

3.6 Identifikasi dan Validasi Sistem

Model matematika dari *quadcopter* pada bab sebelumnya diketahui bahwa terdapat beberapa parameter yang harus dicari nilainya. parameter tersebut ialah sebagai berikut:

a) Massa
b) Panjang lengan *quadcopter*
c) Momen inersia
d) Pemetaan gaya angkat terhadap pulsa
e) Konstanta *thrust*
f) Konstanta *drag*.

Massa dari sistem keseluruhan diukur menggunakan timbangan digital. Momen inersia dari sistem pada sumbu X, sumbu Y, dan sumbu Z diperoleh menggunakan identifikasi parametrik dengan cara menerbangkan *quadcopter* lalu didapatkan data keluaran dari sensor yaitu berupa kecepatan sudut, serta nilai persentase *throttle* dari *remote control*.

Sedangkan, untuk mendapatkan nilai konstanta *thrust* dan konstanta *drag,* dilakukan pengujian pada masing-masing motor dengan diberi sinyal PWM, lalu diukur kecepatan dari keempat motor tersebut, dan dihitung konstanta *thrust* dan *drag* dari *quadcopter*.

Untuk mendapatkan konstanta *thrust, quadcopter* diterbangkan dalam posisi *hover* karena pada posisi tersebut, gaya yang dihasilkan oleh motor untuk mengangkat *quadcopter* sama dengan gaya yang mendorong *quadcopter* menuju *ground*. Dengan asumsi gaya gravitasi tetap, maka konstanta *thrust* dapat dihitung menggunakan Persamaan (3.3).

$$b = \frac{m.g}{\sum_{i=1}^{4} \Omega_i^{\,2}} \tag{3.3}$$

Konstanta *drag* dihitung dengan persamaan gerak lurus berubah beraturan. Pengukuran konstanta *drag* dilakukan dengan mengambil data penerbangan *quadcopter* pada saat *take-off*. Gaya yang terjadi saat *quadcopter* bergerak ke atas dapat diperoleh menggunakan Persamaan (3.4).

$$\sum F = Thrust - mg - Drag \tag{3.4}$$

$$ma = (\tau_1 + \tau_2 + \tau_3 + \tau_4) - mg - D \tag{3.5}$$

Jika Persamaan (3.5) disubstitusikan pada Persamaan (3.4) maka Persamaan (3.4) menjadi Persamaan (3.6).

$$D = ma - ((\tau_1 + \tau_2 + \tau_3 + \tau_4) - mg) \tag{3.6}$$

Di mana percepatan dan konstanta *drag* dari *quadcopter* dapat dihitung menggunakan Persamaan (3.7)-(3.8).

.

$$a = \frac{s - v_0 t}{0,5t^2} \tag{3.7}$$

$$CD = \frac{D}{\Omega^2} \tag{3.8}$$

Dengan ketentuan sebagai berikut:

CD = Konstanta *Drag*
v_0 = Kecepatan awal (m/s)
t = Waktu (s)
a = Percepatan (m/s^2)
s = Jarak yang ditempuh (m)
τ_i = Gaya angkat, i=1,2,3,4
D = Gaya *drag*
Ω = Kecepatan motor

Parameter hasil identifikasi menggunakan pengukuran terdapat pada Tabel 3.1.

Tabel 3. 1 Parameter *Quadcopter* Hasil Pengukuran

No	Parameter	Nilai	Satuan
1	Massa *quadcopter* (*m*)	1,26	kg
2	Jari-jari *quadcopter* (*l*)	0,206	meter
3	Konstanta *Thrust*	1,68918 x 10^{-5}	N.sec^2
4	Konstanta *Drag*	4,19 x 10^{-6}	Nm.sec^2

Pada model matematika rotasi *quadcopter*, terdapat beberapa parameter yang belum diketahui. Untuk mendapatkannya dapat dicari menggunakan Pendekatan Penyelesaian Persamaan Linear Simultan (P3LS) dengan memodifikasi model rotasi *quadcopter* menjadi Persamaan (3.9)-(3.11).

$$\dot{p} = a_1 qr + b_1 q\Omega + c_1 U_2 \tag{3.9}$$

$$\dot{q} = a_2 pr + b_2 p\Omega + c_2 U_3 \tag{3.10}$$

$$\dot{r} = a_3 pq + b_3 U_4 \tag{3.11}$$

Dimana data parameter tersebut diperoleh dari data penerbangan. Dari Persamaan (3.9) dapat dibuat suatu persamaan matriks pada Persamaan (3.12) dan nilai a_1, b_1 ,c_1 dapat dicari nilainya dengan menggunakan Persamaan (3.13).

$$\begin{bmatrix} qr & q\Omega & U_2 \end{bmatrix} \begin{bmatrix} a_1 \\ b_1 \\ c_1 \end{bmatrix} = \dot{p} \tag{3.12}$$

$$\begin{bmatrix} a_1 \\ b_1 \\ c_1 \end{bmatrix} = \left(\begin{bmatrix} qr \\ q\Omega \\ U_2 \end{bmatrix} \begin{bmatrix} qr & q\Omega & U_2 \end{bmatrix} \right)^{-1} \begin{bmatrix} qr \\ q\Omega \\ U_2 \end{bmatrix} \dot{p} \tag{3.13}$$

Persamaan (3.10) dapat dibuat suatu persamaan matriks pada Persamaan (3.14) dan nilai a_2, b_2 ,c_2 dapat dicari nilainya dengan menggunakan Persamaan (3.15).

$$\begin{bmatrix} pr & p\Omega & U_3 \end{bmatrix} \begin{bmatrix} a_2 \\ b_2 \\ c_2 \end{bmatrix} = \dot{q} \tag{3.14}$$

$$\begin{bmatrix} a_2 \\ b_2 \\ c_2 \end{bmatrix} = \left(\begin{bmatrix} pr \\ p\Omega \\ U_3 \end{bmatrix} \begin{bmatrix} pr & p\Omega & U_3 \end{bmatrix} \right)^{-1} \begin{bmatrix} pr \\ p\Omega \\ U_3 \end{bmatrix} \dot{q} \tag{3.15}$$

Persamaan (3.11) dapat dibuat suatu persamaan matriks pada Persamaan (3.16) dan nilai a_3 dan b_3 dapat dicari dengan menggunakan Persamaan (3.17).

$$\begin{bmatrix} pq & U_4 \end{bmatrix} \begin{bmatrix} a_3 \\ b_3 \end{bmatrix} = \dot{r} \tag{3.16}$$

$$\begin{bmatrix} a_3 \\ b_3 \end{bmatrix} = \left(\begin{bmatrix} pq \\ U_4 \end{bmatrix} \begin{bmatrix} pq & U_4 \end{bmatrix} \right)^{-1} \begin{bmatrix} pq \\ U_4 \end{bmatrix} \dot{r} \tag{3.17}$$

Dari hasil Persamaan (3.13)-(3.17) didapatkan nilai parameter yang ditunjukkan pada Tabel (3.2).

Tabel 3. 2 Parameter *Quadcopter* Hasil Identifikasi Parametrik

No	Konstanta	Nilai
1	a_1	-0,5495
2	b_1	-0,0017
3	c_1	0,2052
4	a_2	0,1675
5	b_2	-0,0094
6	c_2	2,955
7	a_3	-2,0257
8	b_3	0,0594

Identifikasi model *quadcopter* juga dilakukan dengan menggunakan identifikasi fisik, berupa penghitungan momen inersia dan

konstanta motor *propeller*. Penghitungan identifikasi fisik terdapat pada lampiran A3.

Hasil dari identifikasi fisik terdapat terdapat pada Tabel 3.3.

Tabel 3. 3 Parameter *Quadcopter* Hasil Identifikasi Fisik

No	Konstanta	Nilai
1	a_1	0.25327
2	b_1	-0.0198
3	c_1	595.174
4	a_2	-0.2537
5	b_2	0.0198
6	c_2	797.04
7	a_3	0
8	b_3	595.174

Selanjutnya akan dilakukan proses validasi untuk membandingkan hasil identifikasi fisik dan parametrik dengan menggunakan Persamaan (3.9)-(3.10) di mana telah diketahui nilai parameter hasil penerbangkan.

Hasil proses validasi dengan menggunakan RMSE sebagai pembanding terdapat pada Tabel 3.4.

Tabel 3. 4 Validasi Hasil Identifikasi Parametrik dan Identifikasi Fisik

Besaran	RMSE Identifikasi Parametrik	RMSE Identifikasi Fisik
$\dot{p}$	3,2232%	190,34%
$\dot{q}$	1,3507%	177,53%
$\dot{r}$	5,3405%	92,525%

3.7 Model Matematika Hasil Identifikasi

Setelah diperoleh parameter dari sistem, maka dapat dituliskan kembali model matematika dari sistem *quadcopter* pada Persamaan (3.18)-(3.23).

$$\ddot{X} = \frac{U_1}{1{,}26}(\cos\Psi \sin\theta\cos\phi + \sin\Psi \sin\phi) \quad (3.18)$$

$$\ddot{Y} = \frac{U_1}{1{,}26}(\sin\Psi \sin\theta\cos\phi - \cos\Psi \sin\phi) \quad (3.19)$$

$$\ddot{Z} = -9{,}81 + \frac{U_1}{1.26}(\cos\theta\cos\phi) \quad (3.20)$$

$$\dot{p} = -0{,}5495qr - 0{,}00017q\Omega + 0{,}2052U_2 \quad (3.21)$$

$$\dot{q} = 0{,}1675pr - 0{,}0094p\Omega + 2{,}955U_3 \quad (3.22)$$

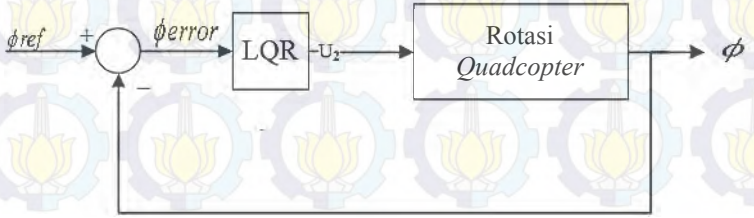
$$\dot{r} = -2{,}0257pq + 0{,}0954U_4 \quad (3.23)$$

3.8 Perancangan Linear Quadratic Regulator (LQR)

LQR digunakan untuk mengendalikan gerak rotasi. Gerak rotasi terdiri dari 3 macam, yaitu gerak rotasi sudut *roll*, gerak rotasi sudut *pitch*, dan gerak rotasi sudut *yaw*. Untuk merancang kontroler LQR untuk gerak rotasi, diperlukan suatu linearisasi karena *quadcopter* merupakan salah satu sistem nonlinear. Metode linearisasi yang digunakan pada perancangan kontroler LQR ini ialah *feedback linearization* dengan memodifikasi sinyal kontrol untuk menghilangkan efek nonlinear dari sistem.

3.8.1 Perancangan Kontroler LQR Sudut *Roll*

Pada perancangan kontroler sudut *roll* digunakan Persamaan (3.21). Sudut *roll* akan menyebabkan *quadcopter* bergerak pada sumbu Y. Blok diagram pengontrolan sudut *roll* ditunjukkan pada Gambar 3.5 di mana ϕ *ref* adalah sinyal kontrol dari kontroler translasi sumbu Y. Hal tersebut dikarenakan untuk mengatur gerak translasi harus dikontrol secara *cascade*.



Gambar 3. 5 Blok Diagram Pengendalian Sudut *Roll*

Untuk perancangan LQR sudut *roll*, langkah pertama ialah mendefinisikan hubungan linear antar *variable* pada model matematika sudut *roll*. Hubungan linear tersebut terdapat pada Persamaan (3.24)-(3.26).

$$\dot{\phi} = p \tag{3.24}$$

$$\ddot{\phi} = \dot{p} \tag{3.25}$$

$$\ddot{\phi} = -0{,}5495qr - 0{,}00017q\Omega + 0{,}2052U_2 \tag{3.26}$$

Untuk membuat *state space* pengendalian sudut *roll*, maka Persamaan (3.26) harus dimodifikasi pada Persamaan (3.27)-(3.28).

$$\ddot{\phi} = 0{,}2052U_2 + -0{,}5495qr + -0{,}0001q\Omega \tag{3.27}$$

$$\ddot{\phi} = 0{,}2052\left(U_2 + \frac{1}{0{,}2052}\left(-0{,}5495qr - 0{,}00017q\Omega\right)\right) \tag{3.28}$$

Jika dibuat permisalan pada Persamaan (3.29), maka Persamaan (3.28) akan menjadi Persamaan (3.30).

$$U_2^* = U_2 + \frac{1}{0{,}2052}\left(-0{,}5495qr - 0{,}00017q\Omega\right) \tag{3.29}$$

$$\ddot{\phi} = 0{,}2052U_2^* \tag{3.30}$$

Dari Persamaan (3.30) dapat dibuat *state space* pada Persamaan (3.31)-(3.32).

$$\begin{bmatrix} \dot{\phi} \\ \ddot{\phi} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \phi \\ \dot{\phi} \end{bmatrix} + \begin{bmatrix} 0 \\ c_1 \end{bmatrix} U_2^* \tag{3.31}$$

$$Y = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} \phi \\ \dot{\phi} \end{bmatrix} \tag{3.32}$$

Dari Persamaan (3.31)-(3.32) didapatkan matriks A, B, dan C pada Persamaan (3.33).

$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 \\ c_1 \end{bmatrix} \quad C = \begin{bmatrix} 1 & 0 \end{bmatrix} \tag{3.33}$$

Besarnya nilai sinyal kontrol LQR bergantung pada nilai Q dan R. Pada Persamaan (2.53) dijelaskan bahwa indeks performansi LQR bergantung pada nilai Q dan R. Untuk mencari nilai Q dan R digunakan metode *tuning* manual (*try and error*) pada Tabel 3.5.

Tabel 3. 5 Nilai Q dan R Kontroler LQR Sudut *Roll*

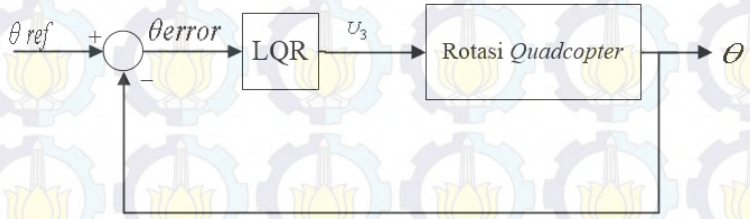
No	Parameter	Nilai
1	Q-*roll*	$\begin{bmatrix} 1000 & 0 \\ 0 & 1 \end{bmatrix}$
2	R-*roll*	0,00001

Dari Tabel 3.5 dapat diketahi bahwa Q bernilai besar dan R bernilai kecil. Akibatnya nilai penguatan K menjadi besar dan akan mempercepat sistem untuk mencapai keadaan tunak. Persoalan ini dianggap sebagai *infinite time* LQR di mana tidak terikat oleh waktu dengan $t_f=\infty$ sehingga nilai K dapat diketahui menggunakan Persamaan (2.54)-(2.55). Nilai K terdapat pada Persamaan (3.35).

$$P = \begin{bmatrix} 44{,}4371 & 0{,}4873 \\ 0{,}4873 & 0{,}0217 \end{bmatrix} \tag{3.34}$$

$$K = [10^4 \quad 0{,}044] \tag{3.35}$$

3.8.2 Perancangan Kontroler LQR Sudut *Pitch*



Gambar 3. 6 Blok Diagram Pengendalian Sudut *Pitch*

Pengontrolan gerak rotasi sudut *pitch* menggunakan Persamaan (3.22). Sudut *pitch* tersebut akan menyebabkan *quadcopter* bergerak pada sumbu X. Blok diagram pengendalian sudut *pitch* ditunjukkan oleh Gambar 3.6 di mana *θref* adalah sinyal kontrol dari kontroler translasi pada sumbu X.

Hubungan antar *variable* pada model matematika gerak rotasi sudut *pitch* yang terdapat pada Persamaan (3.36)-(3.38).

$$\dot{\theta} = q \tag{3.36}$$

$$\ddot{\theta} = \dot{q} \tag{3.37}$$

$$\ddot{\theta} = 0{,}1675\, pr - 0{,}0094\, p\Omega + 2{,}955 U_3 \tag{3.38}$$

State space untuk LQR pengendalian sudut *pitch* didapatkan dengan memodifikasi Persamaan (3.38) pada Persamaan (3.39)-(3.40).

$$\ddot{\theta} = 2{,}955 U_3 + 0{,}1675\, pr - 0{,}0094\, p\Omega \tag{3.39}$$

$$\ddot{\theta} = 2{,}955\left(U_3 + \frac{1}{2{,}955}\left(2{,}955\, pr + 0{,}1675\, p\Omega\right)\right) \tag{3.40}$$

Jika dibuat permisalan pada Persamaan (3.41), maka Persamaan (3.40) akan menjadi Persamaan (3.42).

$$U_3^* = U_3 + \frac{1}{2{,}955}\left(2{,}955\, pr + 0{,}1675\, p\Omega\right) \tag{3.41}$$

$$\ddot{\theta} = 2{,}955 U_3^* \tag{3.42}$$

Dari Persamaan (3.42) dapat dibuat *state space* pada Persamaan (3.43)-(3.44).

$$\begin{bmatrix} \dot{\theta} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \theta \\ \dot{\theta} \end{bmatrix} + \begin{bmatrix} 0 \\ c_2 \end{bmatrix} U_3^* \tag{3.43}$$

$$Y = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} \theta \\ \dot{\theta} \end{bmatrix} \tag{3.44}$$

Dari Persamaan (3.43)-(3.44) didapatkan matriks A, B, dan C pada Persamaan (3.45).

$$A=\begin{bmatrix}0 & 1\\ 0 & 0\end{bmatrix} \quad B=\begin{bmatrix}0\\ c_2\end{bmatrix} \quad C=\begin{bmatrix}1 & 0\end{bmatrix} \tag{3.45}$$

Penentuan nilai Q dan R pada LQR pengendalian sudut *pitch* sama dengan pengendalian sudut *roll,* yaitu secara *tuning* manual (*try and error*) pada Tabel 3.6.

Tabel 3. 6 Nilai Q dan R Kontroler LQR Sudut *Pitch*

No	Parameter	Nilai
1	Q-*pitch*	$\begin{bmatrix}1000 & 0\\ 0 & 4\end{bmatrix}$
2	R-*pitch*	0,1

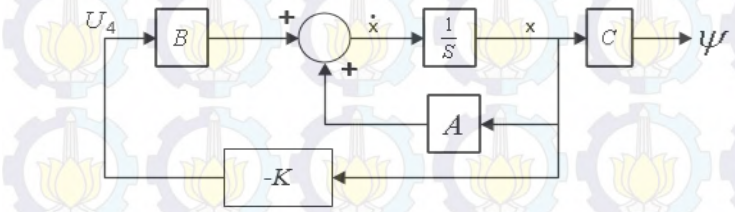
Pada persoalan ini juga dianggap sebagai *infinite time* LQR. Sehingga nilai K dapat diketahui pada Persamaan (3.47).

$$P=\begin{bmatrix}103{,}7644 & 3{,}3835\\ 3{,}3835 & 0{,}3511\end{bmatrix} \tag{3.46}$$

$$K=[100 \quad 10{,}3764] \tag{3.47}$$

3.8.3 Perancangan Kontroler LQR Sudut *Yaw*

Perancangan LQR pada pengendalian sudut *yaw* menggunakan Persamaan (3.23).



Gambar 3. 7 Blok Diagram Pengendalian Sudut *Yaw*

Kontroler sudut *yaw* dirancang sebagai regulator, di mana sudut *yaw* akan dipertahankan pada sekitar 0 radian. Blok diagram pengendalian sudut *yaw* ditunjukkan pada Gambar 3.7.

Hubungan antar *variable* pada model matematika gerak rotasi sudut *yaw* terdapat pada Persamaan (3.48)-(3.50)

$$\dot{\psi} = r \tag{3.48}$$

$$\ddot{\psi} = \dot{r} \tag{3.49}$$

$$\ddot{\psi} = -2{,}0257\, pq + 0{,}0954 U_4 \tag{3.50}$$

State space dari pengendalian sudut *yaw* didapatkan dengan memodifikasi Persamaan (3.50) pada Persamaan (3.51)-(3.52).

$$\ddot{\psi} = 0{,}0954 U_4 - 2{,}0257\, pq \tag{3.51}$$

$$\ddot{\psi} = 0{,}0954 \left(U_4 + \frac{1}{0{,}0954} \left(0{,}0954\, pq \right) \right) \tag{3.52}$$

Jika dibuat permisalan pada Persamaan (3.53), maka Persamaan (3.52) akan menjadi Persamaan (3.54).

$$U_4^* = U_4 + \frac{1}{0{,}0954} \left(0{,}0954\, pq \right) \tag{3.53}$$

$$\ddot{\psi} = 0{,}0954 U_4^* \tag{3.54}$$

Dari Persamaan (3.54) dapat dibuat *state space* pada Persamaan (3.55)-(3.56).

$$\begin{bmatrix} \dot{\psi} \\ \ddot{\psi} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \psi \\ \dot{\psi} \end{bmatrix} + \begin{bmatrix} 0 \\ b_3 \end{bmatrix} U_4^* \tag{3.55}$$

$$Y = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} \psi \\ \dot{\psi} \end{bmatrix} \tag{3.56}$$

Dari Persamaan (3.55)-(3.56) didapatkan matriks A, B, dan C pada Persamaan (3.57).

$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 \\ b_3 \end{bmatrix} \quad C = \begin{bmatrix} 1 & 0 \end{bmatrix} \tag{3.57}$$

Pemilihan pembobot Q dan R dilakukan secara *tuning* manual (*try and error*) pada Tabel 3.7. Nilai K dari LQR sudut *yaw* terdapat pada Persamaan (3.59).

Tabel 3. 7 Nilai Q dan R Kontroler LQR Sudut *Yaw*

No	Parameter	Nilai
1	Q-*yaw*	$\begin{bmatrix} 1000 & 0 \\ 0 & 4 \end{bmatrix}$
2	R-*yaw*	0,00001

$$P = \begin{bmatrix} 121{,}0264 & 5{,}3237 \\ 5{,}3237 & 0{,}6443 \end{bmatrix} \tag{3.58}$$

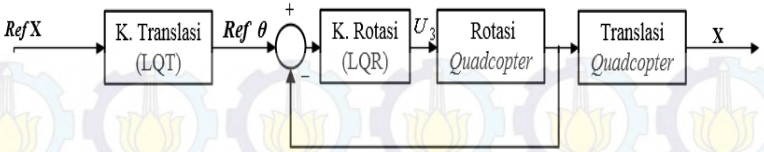
$$K = \begin{bmatrix} 3{,}162 \times 10^3 & 0{,}3827 \end{bmatrix} \tag{3.59}$$

3.9 Perancangan *Linear Quadratic Tracking* (LQT)

LQT digunakan untuk pengendalian gerak translasi. Gerak translasi tersebut terdiri dari 3 macam, yaitu gerak translasi pada sumbu X, gerak translasi pada sumbu Y, dan gerak translasi pada sumbu Z. Sama halnya dengan perancangan LQR, perancangan LQT juga memerlukan linearisasi karena *quadcopter* merupakan sistem nonlinear. Perancangan LQT ini menggunakan *feedback linearization* yang digunakan untuk menghilangkan pengaruh nonlinear dari sistem tersebut.

3.9.1 Perancangan Kontroler LQT Pengendalian Sumbu X

Untuk dapat mengendalikan sumbu X, maka *quadcopter* harus dalam keadaan stabil sehingga dapat melakukan perubahan sudut *pitch*. Oleh karena itu perancangan gerak translasi digunakan metode pengendalian secara *cascade* antara kontroler gerak translasi dan kontroler gerak rotasi. Blok diagram pengendalian gerak translasi sumbu X ditunjukkan oleh Gambar 3.8.



Gambar 3. 8 Blok Diagram Pengendalian Gerak Translasi *Quadcopter* pada Posisi X

Pada perancangan LQT untuk posisi sumbu X digunakan Persamaan (3.18). Persamaan tersebut memiliki tiga *variable* ϕ, θ, ψ di mana gerak pada sumbu X bergantung pada besarnya nilai sudut θ dan juga mendapat pengaruh secara langsung oleh sudut ϕ. Dimisalkan bahwa *quadcopter* bersifat *rigid* untuk sudut ψ sehingga besarnya perubahan sudut *yaw* bernilai sangat kecil atau $\psi \approx 0$.

Dari permisalan tersebut, Persamaan (3.18) akan menjadi Persamaan (3.60).

$$\ddot{X} = (\sin\theta\cos\phi)\frac{U_1}{1{,}26} \tag{3.60}$$

Untuk menghilangkan efek nonlinear dari sistem, maka sinyal kontrol harus dimodifikasi pada Persamaan (3.61) dan persamaan *state* sistem menjadi Persamaan (3.62).

$$U_x^* = (\sin\theta\cos\phi)\frac{U_1}{1{,}26} \tag{3.61}$$

$$\ddot{X} = U_x^* \tag{3.62}$$

Karena sinyal kontrol LQT merupakan θref maka dilakukan konversi sinyal kontrol $U_x{}^*$ menjadi θ pada Persamaan (3.64).

$$\sin\theta = \frac{(1{,}26)U_x^*}{U_1\cos\phi} \tag{3.63}$$

$$\theta = arc\sin\left(\frac{(1{,}26)U_x^*}{U_1\cos\phi}\right) \tag{3.64}$$

State space dari Persamaan (3.62) terdapat pada Persamaan (3.65)-(3.66).

$$\begin{bmatrix} \dot{X} \\ \ddot{X} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} X \\ \dot{X} \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} U_x^* \tag{3.65}$$

$$Y = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ \dot{X} \end{bmatrix} \tag{3.66}$$

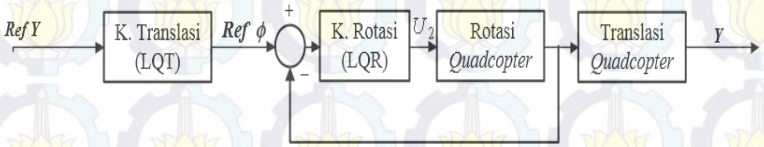
Dari Persamaan (3.65)-(3.66) didapatkan matriks A, B, dan C pada Persamaan (3.67).

$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad C = \begin{bmatrix} 1 & 0 \end{bmatrix} \tag{3.67}$$

Dengan memodifikasi sinyal kontrol tersebut maka sinyal kontrol LQT akan menjadi *θref* yang mana merupakan *setpoint* (referensi) dari kontroler LQR sudut *pitch* sesuai blok diagram pada Gambar 3.8. Nilai Q dan R akan dicari dengan menggunakan *tuning* GA.

3.9.2 Perancangan Kontroler LQT Pengendalian Sumbu Y

Gerak translasi sumbu Y dapat terjadi karena adanya gerak rotasi sudut *roll*. Sistem yang digunakan untuk gerak translasi sumbu Y juga menggunakan sistem *cascade* yaitu seperti pada Gambar 3.9.



Gambar 3. 9 Blok Diagram Pengendalian Gerak Translasi *Quadcopter* pada Posisi Y

Pada perancangan LQT untuk posisi sumbu Y digunakan Persamaan (3.19). Persamaan tersebut memiliki tiga variabel ϕ, θ, ψ di mana gerak pada sumbu Y bergantung pada besarnya nilai sudut ϕ dan juga mendapat pengaruh secara langsung oleh sudut θ. Dimisalkan bahwa *quadcopter* bersifat *rigid* untuk sudut ψ sehingga besarnya perubahan sudut *yaw* bernilai sangat kecil atau $\psi \approx 0$.

Dari permisalan tersebut, Persamaan (3.19) akan menjadi Persamaan (3.68).

$$\ddot{Y} = -(\sin \phi)\frac{U_1}{1{,}26} \tag{3.68}$$

Untuk menghilangkan efek nonlinear dari sistem, maka sinyal kontrol harus dimodifikasi pada Persamaan (3.69) dan persamaan *state* sistem menjadi Persamaan (3.70)

$$U_y^* = -(\sin \phi)\frac{U_1}{1{,}26} \tag{3.69}$$

$$\ddot{Y} = U_y^* \tag{3.70}$$

Karena sinyal kontrol LQT merupakan ϕref maka dilakukan konversi sinyal kontrol U_y^* menjadi ϕ pada Persamaan (3.72).

$$\sin \phi = -\frac{(1{,}26)U_y^*}{U_1} \tag{3.71}$$

$$\phi = arc\sin\left(-\frac{(1{,}26)U_y^*}{U_1}\right) \tag{3.72}$$

State space dari Persamaan (3.70) terdapat pada Persamaan (3.73)-(3.74).

$$\begin{bmatrix} \dot{Y} \\ \ddot{Y} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}\begin{bmatrix} Y \\ \dot{Y} \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} U_y^* \tag{3.73}$$

$$Y = \begin{bmatrix} 1 & 0 \end{bmatrix}\begin{bmatrix} Y \\ \dot{Y} \end{bmatrix} \tag{3.74}$$

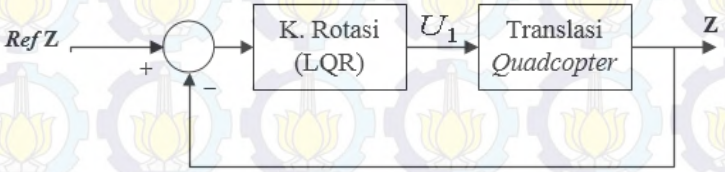
Dari Persamaan (3.72)-(3.73) didapatkan matriks A, B, dan C pada Persamaan (3.75).

$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad C = \begin{bmatrix} 1 & 0 \end{bmatrix} \tag{3.75}$$

Dengan memodifikasi sinyak kontrol tersebut maka sinyal kontrol LQT menjadi ϕref sesuai diagram blok pada Gambar 3.9. Penentuan nilai Q dan R akan dicari menggunakan GA.

3.9.3 Perancangan Kontroler LQT Pengendalian Sumbu Z

Pada perancangan LQT sumbu Z berbeda dengan perancangan LQT pada sumbu X maupun sumbu Y. Untuk dapat bergerak pada sumbu Z (*take off*) tidak memerlukan gerak rotasi. Blok diagram LQT pada sumbu Z ditunjukkan pada Gambar 3.8.



Gambar 3. 10 Blok Diagram Pengendalian Gerak Translasi *Quadcopter* pada Sumbu Z

Perancangan LQT sumbu Z menggunakan Persamaan (3.20). Untuk menghilangkan efek nonlinear dari sistem, maka sinyal kontrol harus dimodifikasi pada Persamaan (3.77) dan persamaan *state* sistem menjadi Persamaan (3.78).

$$\ddot{Z} = -9{,}81 + (\cos\theta\cos\phi)\frac{U_1}{1{,}26} \tag{3.76}$$

$$U_1^* = -9{,}81 + (\cos\theta\cos\phi)\frac{U_1}{1{,}26} \tag{3.77}$$

$$\ddot{Z} = U_1^* \tag{3.78}$$

Persamaan (3.78) dapat dibuat *state space* pada Persamaan (3.79)-(3.80).

$$\begin{bmatrix} \dot{Z} \\ \ddot{Z} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} Z \\ \dot{Z} \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} U_1^* \tag{3.79}$$

$$Y = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} Z \\ \dot{Z} \end{bmatrix} \tag{3.80}$$

Persamaan (3.79)-(3.80) didapatkan matriks A, B, dan C pada Persamaan (3.81).

$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad C = \begin{bmatrix} 1 & 0 \end{bmatrix} \tag{3.81}$$

Besarnya sinyal kontrol bergantung pada nilai Q dan R. Pada perancangan LQT pada sumbu Z digunakan metode *tuning* manual (*try and error*) untuk menenentukan Q dan R yang ditunjukkan pada Tabel 3.8.

Tabel 3. 8 Nilai Q dan R Kontroler LQT Pengendalian *Quadcopter* pada Sumbu Z

No	Parameter	Nilai
1	Qz	1000
2	Rz	0.0001

3.10 Perancangan Algoritma Genetika (GA)

LQT merupakan salah satu metode yang digunakan pada penelitian Tugas Akhir ini. Besarnya sinyal kontrol dari LQT bergantung pada nilai pembobotan Q dan R. Untuk mendapatkan nilai pembobotan Q dan R yang optimal dapat digunakan metode *tuning* manual (*try and error*). Penelitian ini akan menggantikan metode *tuning* manual dengan *tuning* menggunakan GA. Perancangan GA didasarkan pada *flowchart* yang ditunjukkan oleh Gambar 2.13.

Pada perancangan GA didefinisikan individu Q dan R pada sebuah individu pada Gambar 3.11. Nilai *fitness* yang digunakan ditunjukkan pada Persamaan (3.82).

Qx	Qy	Rx	Ry

Gambar 3. 11 Perancangan Individu GA

$$f = \frac{1}{IPx * 0.2 + RMSEx * 0.3 + IPy * 0.2 + RMSEy * 0.3} \quad (3.82)$$

Dimana:

IP : indeks performansi minimum energi

RMSE : *Root Mean Square Error*

Penghitungan nilai indeks performansi minimum energi terdapat pada Persamaan (3.83).

$$J_i = \int_{t_0}^{t_f} \left[x'(t) Q_i x(t) + u(t) R_i u(t) \right] dt \quad (3.83)$$

Sedangkan penghitungan nilai RMSE terdapat pada Persamaan (3.84).

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (X_i - Y_i)^2} \quad (3.84)$$

Setelah menentukan fungsi *fitness* yang ingin dicari, sesuai dengan *flowchart* pada Gambar 2.13, langkah pertama ialah pembangkitan populasi dan generasi. Pada penelitian ini, pembangkitan nilai gen dari individu dilakukan secara *random* dengan *range* kerja dari setiap gen adalah sebagai berikut:

a. Qx = bernilai 700 hingga 1000
b. Qy = bernilai 700 hingga 1000
c. Rx = bernilai 0,02 hingga 1
d. Ry = bernilai 0,02 hingga 1

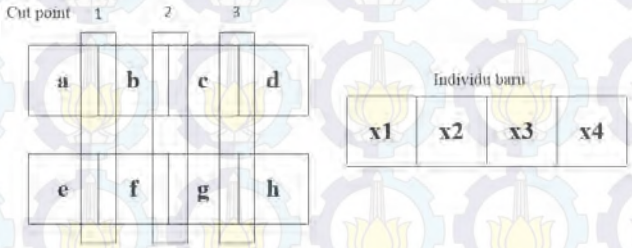
Selanjutnya dilakukan perhitungan nilai *fitness* dengan menggunakan populasi yang telah dibangkitkan sehingga setiap individu memiliki nilai *fitness* yang berbeda-beda.

Seleksi pada individu digunakan untuk memilih individu yang memiliki nilai *fitness* tinggi. Seleksi ini dirancang dengan menggunakan 2 macam, yaitu seleksi *truncking* dan seleksi mesin *roullete*. Seleksi *truncking* digunakan untuk memilih individu yang memiliki nilai *fitness*

tinggi. Banyaknya individu yang hilang bergantung pada besarnya rasio dari seleksi (*selection rate*). Individu yang tersisa selanjutnya diseleksi secara acak dengan menggunakan mesin *roullete*.

Crossover dilakukan pada semua individu. Metode yang digunakan ialah dengan membangkitkan nilai *random integer* yang merepresentasikan nilai *cut poin*, dijelaskan pada Gambar 3.12. Individu baru tersebut bergantung pada resio *crossover*.

Jika *cut point* 1 yang terpilih, maka individu baru tersebut akan sesuai dengan Persamaan (3.85)-(3.86).



Gambar 3. 12 Perancangan Proses *Crossover* GA

$$x1 = a \times rc + e \times (1 - rc) \quad (3.85)$$

$$x2 = b,\ x3 = c,\ x4 = d \quad (3.86)$$

Jika *cut point* 2 yang terpilih, maka individu baru tersebut akan menjadi Persamaan (3.87)-(3.89).

$$x1 = a \times rc + e \times (1 - rc) \quad (3.87)$$

$$x2 = b \times rc + f \times (1 - rc) \quad (3.88)$$

$$x3 = c,\ x4 = d \quad (3.89)$$

Jika *cut point* 3 yang terpilih, maka individu baru tersebut akan menjadi Persamaan (3.90)-(3.93).

$$x1 = a \times rc + e \times (1 - rc) \quad (3.90)$$

$$x2 = b \times rc + f \times (1 - rc) \quad (3.91)$$

$$x3 = c \times rc + g \times (1 - rc) \quad (3.92)$$

$$x4 = d \quad (3.93)$$

Setelah proses *crossover* selesai, langkah selanjutnya ialah proses mutasi. Banyaknya gen yang mengalami mutasi bergantung pada rasio mutasi. Untuk memilih posisi gen yang mengalami mutasi, dibangkitkan nilai *random* dari banyaknya gen yang mengalami mutasi.

Setelah mengalami mutasi, proses selanjutnya ialah menghitung ulang nilai *fitness*. Setelah menghitung nilai *fitness*, proses dilanjutkan seperti *flowchart* hingga memenuhi generasi yang telah ditentukan.

BAB IV
PENGUJIAN DAN ANALISIS

Berdasarkan spesifikasi dan perancangan sistem yang telah dijelakan pada bab sebelumnya, maka dilakukan pengujian dan analisis terhadap perilaku sistem. Tujuan dari Pengujian adalah untuk mengetahui performansi sistem hasil perancangan. Jika performansi yang diharapkan belum sesuai, maka perlu diadakan analisis untuk penyempurnaan kinerja sistem sekaligus dapat digunakan untuk pengembangan lebih lanjut.

Pada bab ini dipaparkan mengenai hasil simulasi dari perancangan gerak lateral *quadcopter* menggunakan kontroler LQT di mana parameter dari kontroler akan didapatkan dengan *tuning* menggunakan GA.

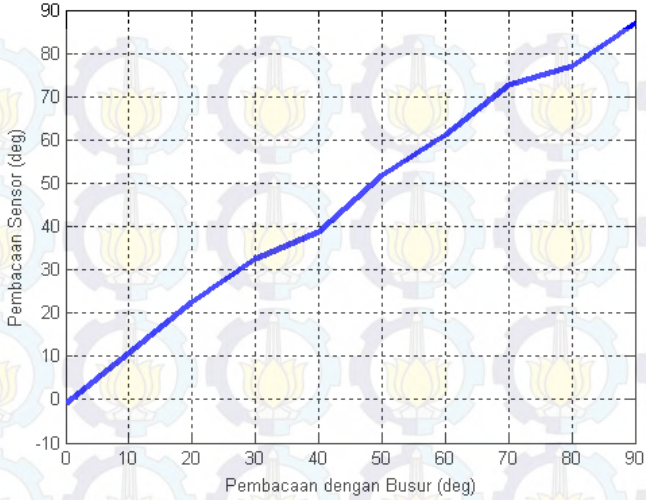
4.1 Pengujian Sensor

Sensor merupakan salah satu hal terpenting dalam *quadcopter* di mana pembacaan sensor merupakan data yang akan diproses oleh kontroler. Pengujian ini dilakukan pada sensor yang ada pada Ardupilot Mega 2.6, yaitu sensor *gyro* dan *accelerometer*. Hasil Pengujian sensor terhadap sudut *roll* ditunjukkan oleh Tabel 4.1.

Tabel 4. 1 Pengujian Sudut *Roll*

No	Pembacaan Busur (deg)	Pembacaan Sensor (deg)
1	0	-0,68
2	10	10,82
3	20	22,7
4	30	32,6
5	40	38,7
6	50	51,8
7	60	61,3
8	70	72,8
9	80	77,06
10	90	87,1

Dari Tabel 4.1, dapat ketahui bahwa sensor mampu mengukur sudut *roll* dengan RMSE sebesar 4,802 %. Grafik pembacaan sensor terhadap pembacaan busur ditunjukkan pada Gambar 4.1.



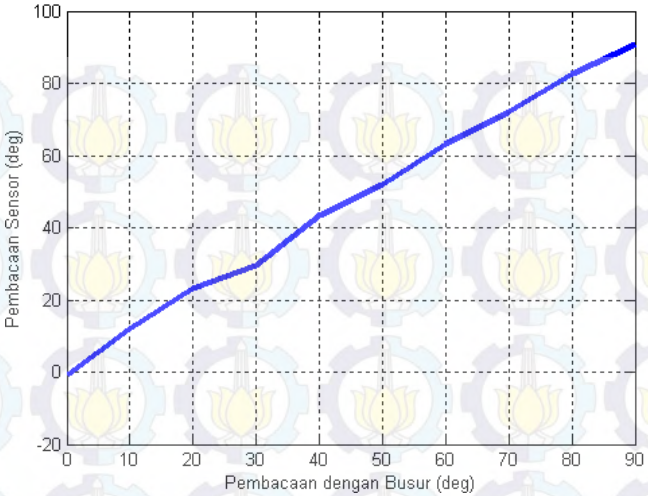
Gambar 4. 1 Pengujian Sudut *Roll*

Sedangkan hasil pengujian sensor terhadap sudut *pitch* ditunjukakn pada Tabel 4.2.

Tabel 4. 2 Pengujian Sudut *Pitch*

No	Pembacaan Busur (deg)	Pembacaan Sensor (deg)
1	0	-0,8
2	10	12
3	20	23,1
4	30	29,67
5	40	43,2
6	50	51,8
7	60	63,2
8	70	72
9	80	82,46
10	90	90,7

Dari Tabel 4.2 dapat diketahui bahwa sensor mampu mengukur sudut *pitch* dengan RMSE sebesar 4,875 %. Grafik pembacaan sensor terhadap pembacaan busur ditunjukkan pada Gambar 4.2.



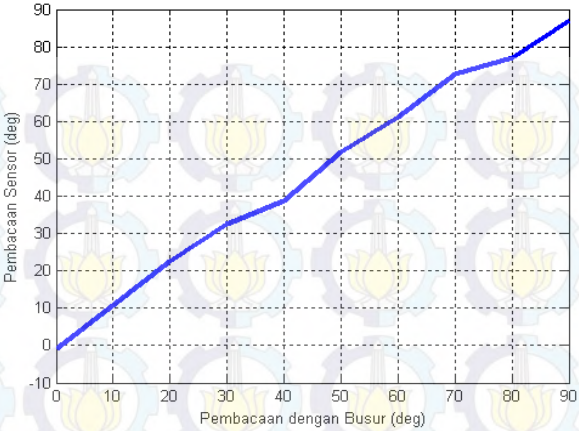
Gambar 4. 2 Pengujian Sudut *Pitch*

Sedangkan hasil pengujian sensor terhadap sudut *yaw* ditunjukkan pada Tabel 4.3

Tabel 4. 3 Pembacaan Sudut *Yaw*

No	Pembacaan Busur (deg)	Pembacaan Sensor (deg)
1	0	-0,68
2	10	10,82
3	20	22,7
4	30	32,6
5	40	38,7
6	50	51,8
7	60	61,3
8	70	72,8
9	80	77,06
10	90	87,1

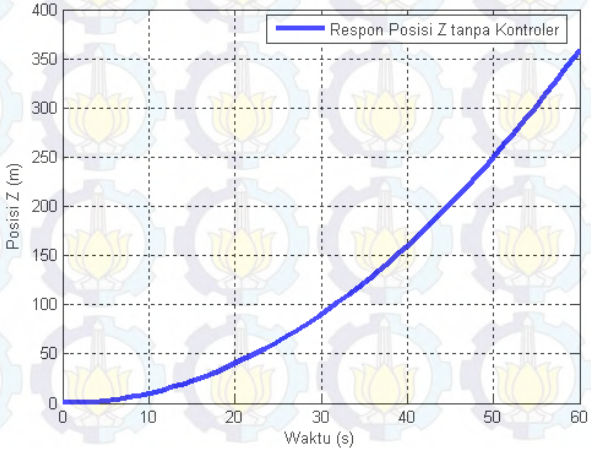
Dari Tabel 4.3, dapat diketahui bahwa sensor mampu mengukur sudut *yaw* dengan RMSE sebesar 3,992 %. Grafik pembacaan sensor terhadap pembacaan busur ditunjukkan pada Gambar 4.3.



Gambar 4. 3 Pengujian Sudut *Yaw*

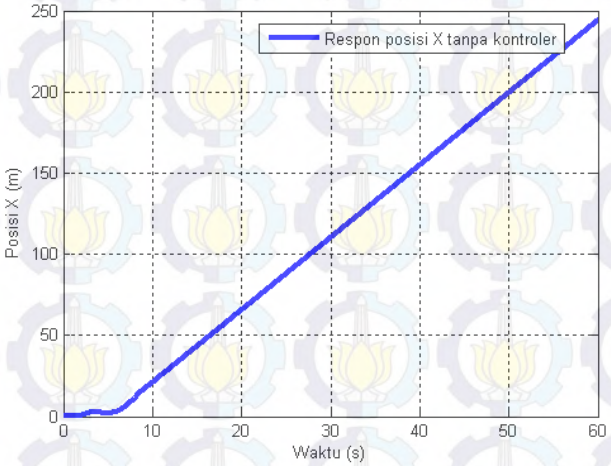
Dari pengujian tersebut dapat diketahui bahwa kesalahan pembacaan sensor tidak lebih dari 5 derajat atau RMSE dari setiap pembacaan sudut tidak melebihi 5%, sehingga sensor masih dapat berfungsi dengan baik.

4.2 Simulasi *Open Loop* Sistem

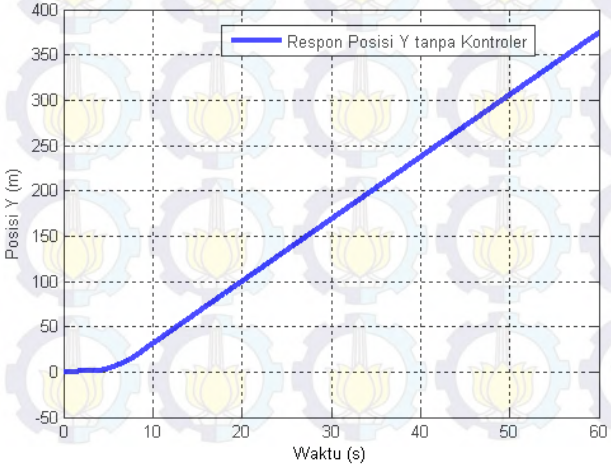


Gambar 4. 4 Respon *Open Loop* Posisi Z Tanpa Kontroler

Untuk mengetahu karakteristik dari *quadcopter* sebelum dilakukan perancangan kontroler, dilakuakan pengujian *open loop*. Pengujian dilakukan dengan memberi referensi posisi X, Y, dan Z dengan sinyal *step* yang diberi nilai akhir 1. Respon posisi *quadcopter* pada sumbu Z ditunjukkan pada Gambar 4.4.



Gambar 4. 5 Respon *Open Loop* Posisi X Tanpa Kontroler



Gambar 4. 6 Respon O*pen Loop* Posisi Y Tanpa Kontroler

Dari Gambar 4.4 terlihat bahwa *quadcopter* akan terbang hingga menuju tak terhingga. Dari respon tersebut dapat dikatakan bahwa *quadcopter* tanpa kontroler merupakan suatu sistem yang tidak stabil.

Untuk dapat bergerak translasi pada sumbu X atau sumbu Y, *quadcopter* harus dapat mempertahankan posisinya. Sedangkan dari Gambar 4.4 terlihat bahwa *quadcopter* akan terbang menuju tak terhingga. Hal tersebut berdampak pada respon posisi X ataupun posisi Y, ditunjukkan pada Gambar 4.5 dan Gambar 4.6, di mana respon akan menuju tak terhingga karena mendapat pengaruh dari gaya U_1.

4.3 Simulasi Pengujian LQR dengan Nilai Q Berbeda

Perancagnan kontrol LQR membutuhkan nilai matriks Q dan R untuk memperoleh penyelesaian optimal. Pada bab 2 telah dijelaskan bahwa semakin besar nilai Q, maka akan memperbesar nilai *Gain Feedback* K, sehingga respon akan semakin cepat mencapai keadaan tunak. Sedangkan jika memperbesar nilai R, maka akan menghemat energi tetapi memperkecil nilai *gain feedback* K sehingga memperlambat sistem untuk mencapai keadaan tunak.

Dalam perancangan kontrol LQR untuk mengatur kestabilan gerak rotasi, dipilih memperbesar nilai Q daripada nilai R dengan menghiraukan besarnya energi. Hal tersebut dikarenakan gerak rotasi *quadcopter* merupakan salah satu faktor penting dalam melakukan gerak translasi, sehingga dibutuhkan respon yang cepat menuju keadaan stabil.

4.3.1 Simulasi Pengujian LQR Pengendalian Sudut *Roll* dengan Nilai Q Berbeda

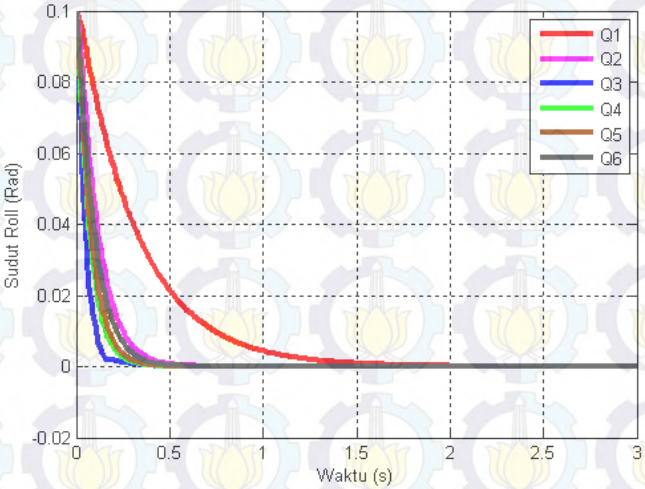
Pada pengujian ini ditetapkan nilai R sebesar 0,00001 dan beberapa kondisi nilai Q sesuai dengan Tabel 4.4. Pemilihan nilai matriks Q lebih dititikberatkan pada *state* pertama, yaitu *state* sudut.

Tabel 4. 4 Variasi Nilai Q pada LQR Pengendalian Sudut *Roll*

No	Nilai Matriks Q	Nilai K
1	$\begin{bmatrix} 10 & 0 \\ 0 & 1 \end{bmatrix}$	[1000 331,281]
2	$\begin{bmatrix} 100 & 0 \\ 0 & 1 \end{bmatrix}$	[3,162 x 10^3 0,362]

No	Nilai Matriks Q	Nilai K
3	$\begin{bmatrix} 1000 & 0 \\ 0 & 1 \end{bmatrix}$	$[10^4 \quad 0{,}044]$
4	$\begin{bmatrix} 1000 & 0 \\ 0 & 4 \end{bmatrix}$	$[10^4 \quad 0{,}071]$
5	$\begin{bmatrix} 800 & 0 \\ 0 & 4 \end{bmatrix}$	$[9{,}94 \times 10^3 \quad 0{,}698]$
6	$\begin{bmatrix} 500 & 0 \\ 0 & 4 \end{bmatrix}$	$[7{,}07 \times 10^3 \quad 0{,}685]$

Hasil respon dari variasi nilai Q ditunjukkan pada Gambar 4.7 dan spesifikasi dari respon ditunjukkan pada Tabel 4.5



Gambar 4. 7 Respon Variasi Nilai Q pada Pengendalian LQR Sudut *Roll*

Dari Tabel 4.5 diketahui bahwa nilai Q3 memiliki hasil yang paling bagus. Sehingga dapat diketahui semakin besar nilai Q yang digunakan, semakin cepat respon menuju keadaan tunak.

Tabel 4. 5 Karakteristik Respon Variasi Nilai Q pada LQR Pengendalian Sudut *Roll*

Variasi Q	*Time Constant* (s)	*Settling Time* (s)	*Rise Time* (s)
Q1	0,3322	0,9966	0,9781
Q2	0,1105	0,3315	0,3253
Q3	0,0487	0,1461	0,1434
Q4	0,073	0,219	0,2149
Q5	0,0806	0,2418	0,2373
Q6	0,0994	0,2982	0,2927
Ket: t_s (±5%), t_r (5%-95%)			

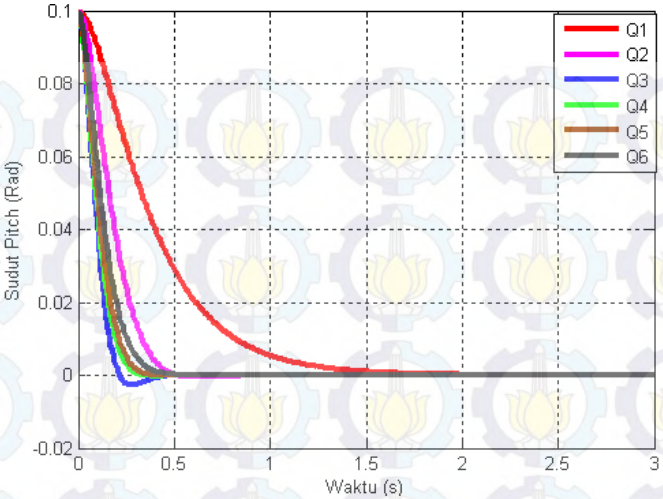
4.3.1 Simulasi Pengujian LQR Pengendalian Sudut *Pitch* dengan Nilai Q Berbeda

Pada pengujian ini ditetapkan nilai R sebesar 0.1 dan beberapa kondisi nilai Q sesuai dengan Tabel 4.6.

Tabel 4. 6 Variasi Nilai Q pada LQR Pengendalian Sudut *Pitch*

No	Nilai Matriks Q	Nilai K
1	$\begin{bmatrix} 10 & 0 \\ 0 & 1 \end{bmatrix}$	[10 4,095]
2	$\begin{bmatrix} 100 & 0 \\ 0 & 1 \end{bmatrix}$	[31,623 5,604]
3	$\begin{bmatrix} 1000 & 0 \\ 0 & 1 \end{bmatrix}$	[100 8,813]
4	$\begin{bmatrix} 1000 & 0 \\ 0 & 4 \end{bmatrix}$	[100 10,3764]
5	$\begin{bmatrix} 800 & 0 \\ 0 & 4 \end{bmatrix}$	[89,443 10,026]
6	$\begin{bmatrix} 500 & 0 \\ 0 & 4 \end{bmatrix}$	[70,711 9,373]

Hasil respon dari variasi nilai Q ditunjukkan pada Gambar 4.8 dan spesifikasi dari respon ditunjukkan pada Tabel 4.7



Gambar 4. 8 Respon Variasi Nilai Q pada Pengendalian LQR Sudut *Pitch*

Tabel 4. 7 Karakteristik Respon Variasi Nilai Q pada LQR Pengendalian Sudut *Pitch*

Variasi Q	*Time Constant* (s)	*Settling Time* (s)	*Rise Time* (s)
Q1	0,428	1,284	1,26
Q2	0,2006	0,602	0,591
Q3	0,1055	0,3165	0,311
Q4	0,1157	0,3471	0,341
Q5	0,124	0,372	0,365
Q6	0,1442	0,4326	0,425
Ket: t_s (±5%), t_r (5%-95%)			

Dari Tabel 4.7 diketahui bahwa nilai Q3 memiliki hasil yang paling cepat menuju keadaan tunak. Namun jika diamati pada Gambar 4.8, respon Q3 memiliki *overshoot* sebesar 0,25%. Sehingga untuk pengendalian sudut *pitch*, dipilih nilai Q4.

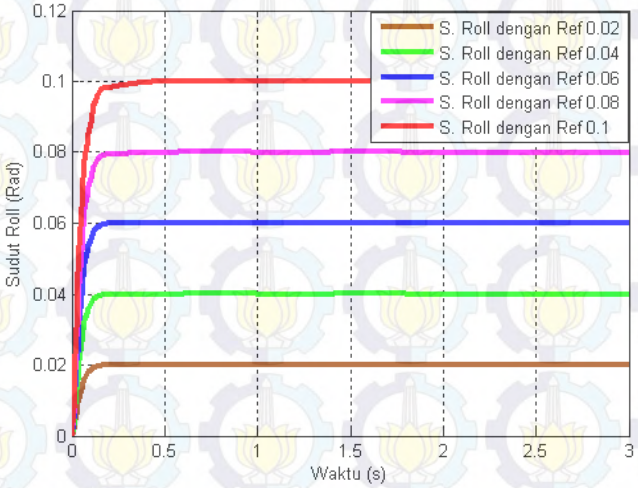
4.4 Simulasi Pengujian LQR untuk Gerak Rotasi *Quadcopter*

Gerak rotasi merupakan gerakan *quadcopter* pada porosnya. Gerak rotasi terdiri dari 3 macam, yaitu gerak rotasi sudut *roll*, gerak rotasi sudut *pitch*, dan gerak rotasi sudut *yaw*. Pada penelitian Tugas

Akhir ini hanya menggunakan gerak rotasi sudut *roll* dan *pitch* karena gerak sudut *yaw* dianggap *rigit*. Sehingga pada pengujian ini hanya membahas mengenai simulasi pengendalian gerak rotasi sudut *roll* dan simulasi pengendalian gerak rotasi sudut *pitch*.

4.4.1 Simulasi Pengujian LQR untuk Pengendalian Sudut *Roll*

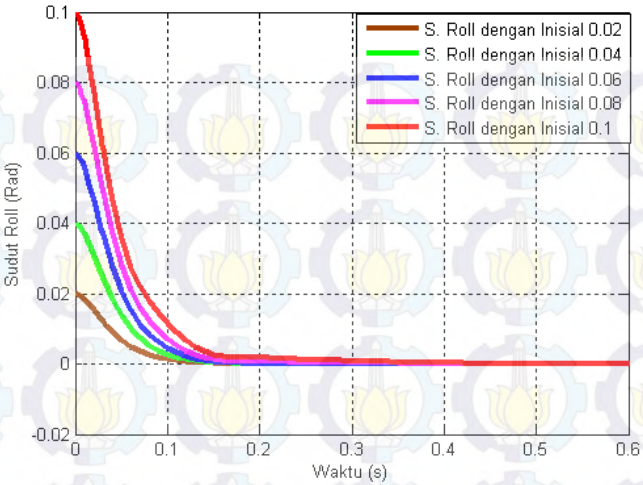
Simulasi pada sudut *roll* dilakukan dengan memberi referensi sinyal *step* dengan nilai maksimal 0,1 atau bernilai ±5,7°. Pada penelitian ini, nilai perubahan sudut dibatasi bernilai maksimum 0,1 dengan asumsi bahwa nilai tersebut dipilih karena sudut rotasi dari *quadcopter* harus mendekati nol untuk menjaga agar *quadcopter* tidak jatuh.



Gambar 4. 9 Respon Simulasi Sudut *Roll* dengan Referensi Berbeda

Pada pengujian pertama diberi nilai referensi yang berbeda-beda. Jika respon yang diberikan mengikuti referensi tersebut, dapat dikatakan bahwa kontroler LQR bekerja dengan baik.

Tujuan dari pengujian ini ialah pada Gambar 3.9 diketahui bahwa kontroler sudut *roll* mendapatkan referensi dari kontroler gerak translasi sumbu Y. Sehingga kontroler sudut *roll* harus dapat mengikuti referensi yang diberikan. Pengujian ini menggunakan sinyal step dengan variasi referensi 0,02 radian hingga 0,1 radian. Hasil dari pengujian terdapat pada Gambar 4.9.



Gambar 4. 10 Respon Simulasi Sudut *Roll* dengan Nilai Awal Berbeda

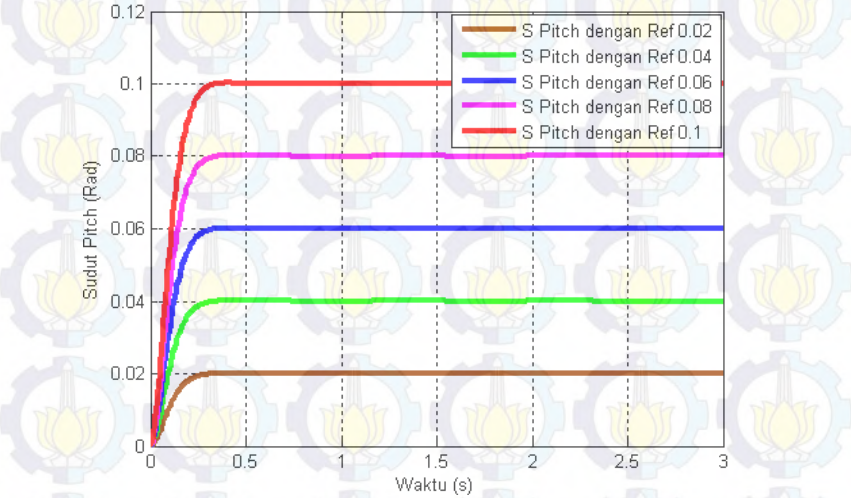
Selain itu juga dilakukan pengujian sudut *roll* dengan memberikan nilai awal yang berbeda-beda. Hasil pengujian terdapat pada Gambar 4.10. Tujuannya ialah untuk mengetahui apakah kontroler sudut *roll* tersebut dapat meregulasi ke titik awal. Jika kontroler tidak dapat meregulasi ke titik awal, maka *quadcopter* akan melakukan gerak translasi secara terus menerus dan menyebabkan *quadcopter* keluar dari lintasan (*trajectory*) yang telah ditentukan. Oleh karena itu pada penelitian ini pengendalian gerak rotasi *quadcopter* menggunakan kontroler LQR agar respon sistem dapat meregulasi kembali ke titik awal.

Pada Gambar 4.9 dan Gambar 4.10 dapat diketahu bahwa pemberian referensi dan nilai awal yang berbeda tidak berpengaruh terhadap kestabilan sudut *roll* pada *quadcopter*. Dengan demikian bahwa perfomansi dari kontrol LQR untuk pengendalian sudut *roll* yang telah dirancang dapat sesuai dengan performansi yang diinginkan untuk menstabilkan sudut *roll* dari *quadcopter.*

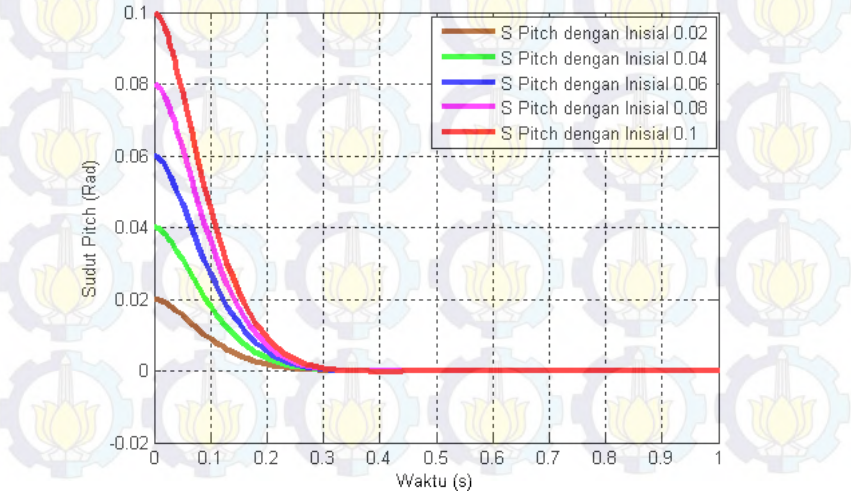
4.4.2 Simulasi Pengujian LQR untuk Pengendalian Sudut *Pitch*

Pengujian sudut *pitch* dilakukan sama seperti pengujian sudut *roll.* Pengujian yang pertama dilakukan dengan memberikan nilai referensi yang berbeda. Pengujian sudut *pitch* terdapat pada Gambar 4.11 dan Gambar 4.12. Pada Gambar 3.8 dijelaskan bahwa kontroler sudut *pitch*

mendapat referensi dari kontroler gerak translasi sumbu X karena untuk pengendalian posisi dari *quadcopter* dibutuhkan *cascade controller*. Nilai referensi yang diberikan sama seperti pengujian pada sudut *roll*.



Gambar 4. 11 Respon Simulasi Sudut *Pitch* dengan Referensi Berbeda



Gambar 4. 12 Respon Simulasi Sudut *Pitch* dengan Nilai Awal Berbeda

Gambar 4.11 dan Gambar 4.12 dapat diketahu bahwa pemberian referensi dan nilai awal yang berbeda tidak berpengaruh terhadap kestabilan sudut *pitch* pada *quadcopter*. Sehingga dapat dikatakan bahwa perancangan LQR pengendalian sudut *pitch* telah sesuai dengan performansi yang diinginkan untuk menjaga kestabilan sudut *pitch* dari *quadcopter*

4.5 Simulasi Pengujian LQT dengan *Tuning* GA pada Gerak Lateral *Quadcopter*

Pada penelitian Tugas Akhir ini digunakan metode LQT untuk pengendalian gerak translasi. Untuk mendapatkan nilai parameter Q dan R dari LQT yang optimal, digunakan metode *tuning* GA. Fungsi *fitness* yang digunakan ditunjukkan pada Persamaan (3.82).

Tuning GA dilakukan dengan variasi parameter, diantaranya ialah jumlah populasi (P), jumlah generasi (G), rasio seleksi (RS), rasio *crossover* (RC), dan rasio mutasi (RM). Pada pengujian ini digunakan *signal builder* yang merepresentasikan pergerakan *quadcopter* dengan lintasan berbentuk segitiga.

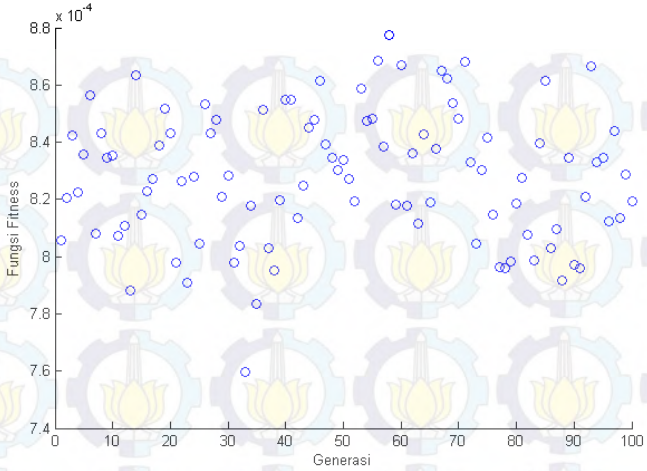
4.5.1 Simulasi Pengujian LQT dengan Kombinasi GA-1

Untuk mendapatkan hasil yang optimal, dilakukan beberapa pengujian dengan beberapa variasi parameter GA yang terdapat pada Tabel 4.8.

Tabel 4. 8 Parameter GA-1

No	Parameter GA	Nilai
1	Populasi (P)	100
2	Generasi (G)	100
3	Rasio Seleksi (RS)	0,5
4	Rasio *Crossover* (RC)	0,5
5	Rasio Mutasi (RM)	0,5

Parameter GA-1 dipilih untuk melihat kondisi pada saat RS, RC, dan RM memiliki rasio yang sama. Rasio tersebut merupakan bobot dari setiap proses pada GA yang mempengaruhi terbentuknya individu baru. Hasil *tuning* LQT dengan menggunakan data tersebut terdapat pada Tabel 4.9.



Gambar 4. 13 Nilai *Fitness* GA-1

Tabel 4. 9 Parameter Hasil *Tuning* GA-1

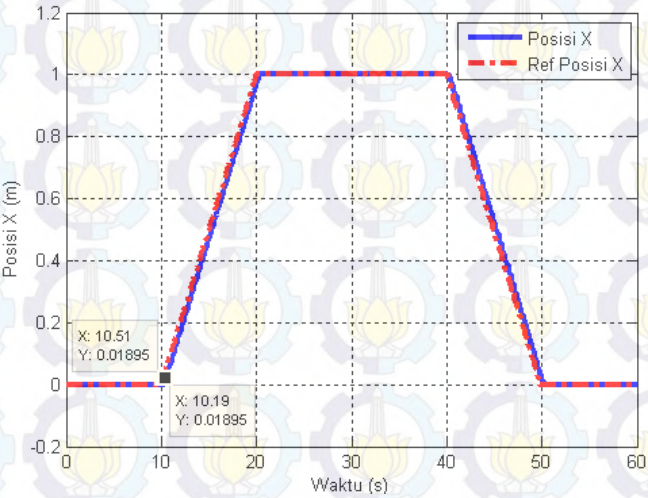
No	Hasil *Tuning* GA-1	Nilai
1	Qx	714,7735
2	Qy	756,9026
3	Rx	0,1026
4	Ry	0,1402
5	RMSEx	1,78 %
6	RMSEy	1,89 %
7	IPx	68,7717
8	IPy	82,0097
9	Nilai *Fitness*	8,1935 x 10^{-4}

Persebaran nilai *fitness* dari setiap generasi ditunjukkan pada Gambar 4.13 di mana dapat diketahui bahwa banyak yang memiliki nilai *fitness* lebih besar daripada nilai *fitness* pada generasi paling akhir dan belum menuju satu nilai yang sama.

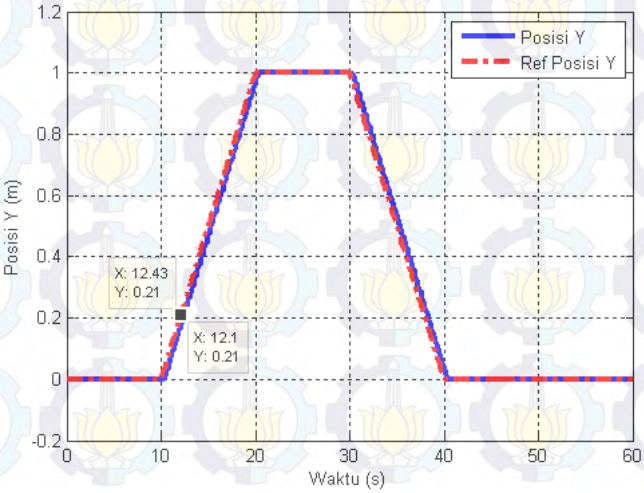
Hasil respon simulasi LQT pada pergerakan *quadcopter* dengan menggunakan GA-1 dapat diketahui pada Gambar 4.14 untuk sumbu X dan Gambar 4.15 untuk sumbu Y.

Dari Gambar 4.14 dan Gambar 4.15 dapat diketahui karakteristik dari respon LQT pergerakan *quadcopter* terhadap sumbu X dan sumbu Y

di mana pada sumbu X memiliki *time lagging* sebesar 0,32 detik dan 0,33 detik untuk sumbu Y.



Gambar 4. 14 Respon Simulasi LQT Pergerakan *Quadcopter* pada Sumbu X dengan Menggunakan Parameter GA-1



Gambar 4. 15 Respon Simulasi LQT Pergerakan *Quadcopter* pada Sumbu Y dengan Menggunakan Parameter GA-1

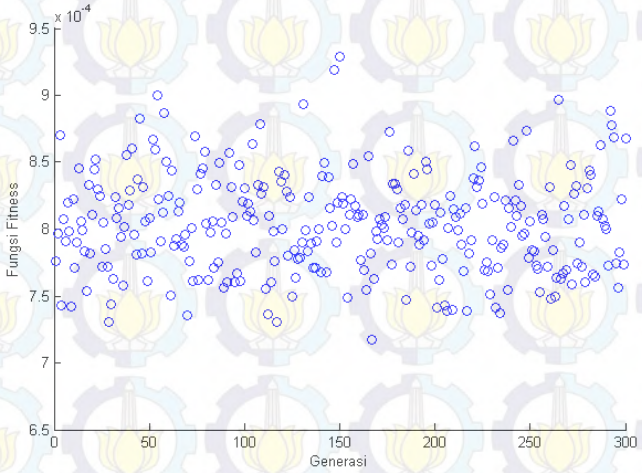
4.5.2 Simulasi Pengujian LQT dengan Kombinasi GA-2

Pada simulasi pengujian LQT dengan GA-1 didapatkan hasil nilai *fitness* yang belum konvergen. Pada pengujian GA-2, jumlah generasi akan ditingkatkan 3 kali lebih besar dan nilai RS, RC, dan RM akan ditingkatkan juga sesuai dengan Tabel 4.10

Tabel 4. 10 Parameter GA-2

No	Parameter GA	Nilai
1	Populasi (P)	100
2	Generasi (G)	300
3	Rasio Seleksi (RS)	0,7
4	Rasio *Crossover* (RC)	0,7
5	Rasio Mutasi (RM)	0,7

Hasil *tuning* LQT menggunakan parameter GA-2 ditunjukkan pada Tabel 4.11 dan persebaran nilai *fitness* GA-2 ditunjukkan pada Gambar 4.16



Gambar 4. 16 Nilai *Fitness* GA-2

Tabel 4. 11 Parameter Hasil *Tuning* GA-2

No	Hasil *Tuning* GA-2	Nilai
1	Qx	714,7735
2	Qy	756,9026
3	Rx	0,1470

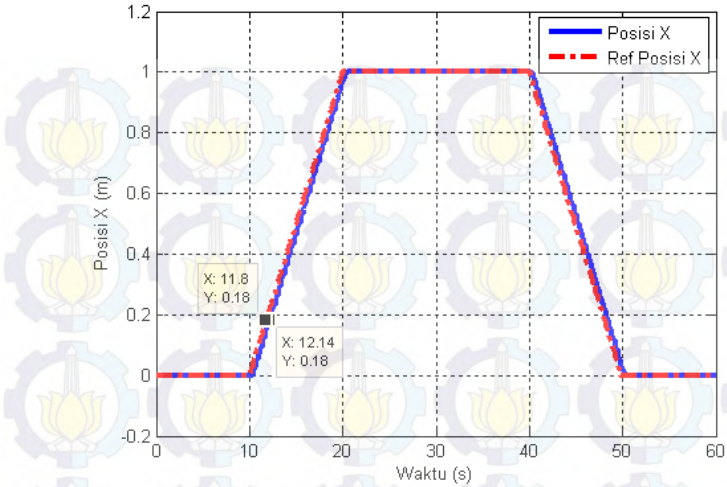
No	Hasil *Tuning* GA-2	Nilai
4	Ry	0,1275
5	RMSEx	1,95 %
6	RMSEy	1,85 %
7	IPx	82,2829
8	IPy	78,2114
9	Nilai *Fitness*	$8{,}6755 \times 10^{-4}$

Dari Gambar 4.16 dapat diketahui bahwa nilai *fitness* dari GA-2 belum menunjukkan nilai yang belum konvergen. Nilai *fitness* setiap generasi dapat berubah-ubah seiring dengan berjalannya program. Hal tersebut dapat terjadi karena GA merupakan suatu algoritma yang didasarkan pada teori genetika pada makhluk hidup. Dimana individu yang kuat/yang memiliki nilai *fitness* tinggi dapat bertahan dari seleksi alam.

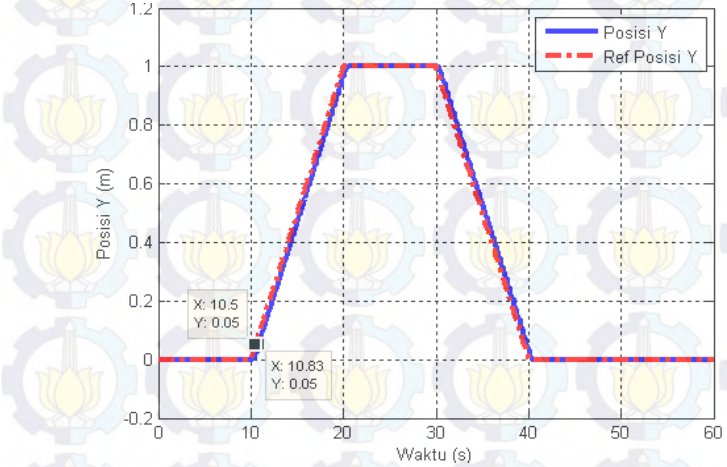
Pada penelitian Tugas Akhir ini, proses seleksi, *crossover*, dan mutasi dirancang dengan menggunakan bobot berupa rasio dari masing-masing proses. Jika nilai *fitness* yang ditunjukkan tiap generasi belum konvergen, ada kemungkinan bahwa semakin besar nilai ketiga bobot akan membuat gen dengan sifat dominan akan terbuang dan tergantikan oleh gen baru.

Pada proses seleksi dirancang dengan menggunkan dua macam, yaitu *truncking* dan *roullete*. Seleksi *truncking* merupakan seleksi dari individu yang memiliki nilai *fitness* tertinggi dengan menggunakan parameter rasio seleksi. Setelah itu dilakukan proses seleksi secara acak dengan menggunkaan mesin *roullete* untuk mengembalikan kedalam jumlah populasi awal. Jika rasio seleksi besar, maka semakin banyak individu yang terseleksi dengan urutan nilai *fitness* yang paling tinggi. Selanjutnya akan dilakukan seleksi *roullete* dengan menggunkan individu yang terseleksi tersebut untuk mengembalikan ke jumlah populasi awal.

Jika nilai rasio mutasi kecil, maka seleksi *truncking* akan menghasilkan individu terpilih sedikit dengan urutan nilai *fitness* tertinggi. Jika dibandingkan dengan nilai rasio mutasi besar, maka hasil seleksi *roullete* dengan rasio seleksi kecil memberikan variasi nilai *fitness* yang lebih bagus dan besar karena jumlah individu terpilih *trucking* lebih sedikit dan memiliki urutan nilai *fitness* yang besar. Hal tersebut juga terjadi pada proses *crossover* dan mutasi.



Gambar 4. 17 Respon Simulasi LQT Pergerakan *Quadcopter* pada Sumbu X dengan Menggunakan Parameter GA-2



Gambar 4. 18 Respon Simulasi LQT Pergerakan *Quadcopter* pada Sumbu Y dengan Menggunakan Parameter GA-2

Hasil respon simulasi LQT untuk pergerakan *quadcopter* dengan menggunakan parameter GA-2 terdapat pada Gambar 4.17 untuk sumbu

X di mana respon tersebut memiliki *time lagging* sebesar 0,34 detik dan Gambar 4.18 untuk sumbu Y dengan *time lagging* sebesar 0,33 detik.

4.5.3 Simulasi Pengujian LQT dengan Kombinasi GA-3

Pada simulasi pengujian LQT menggunakan GA-1 dan GA-2 hasil yang didapatkan belum optimal karena nilai *fitness* pada setiap generasi belum menuju satu nilai yang sama. Pengujian GA-3 menggunakan parameter pada Tabel 4.12.

Tabel 4. 12 Parameter GA-3

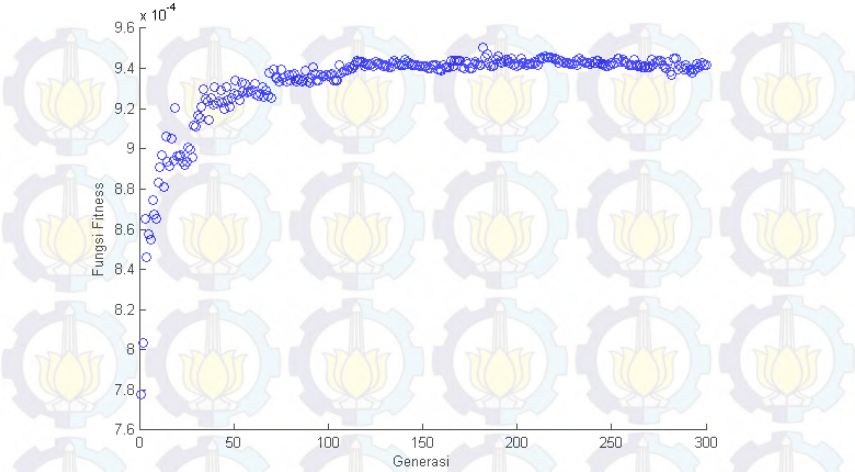
No	Parameter GA	Nilai
1	Populasi (P)	100
2	Generasi (G)	300
3	Rasio Seleksi (RS)	0,3
4	Rasio *Crossover* (RC)	0,3
5	Rasio Mutasi (RM)	0,3

Parameter GA-3 dirancang dengan membuat RS, RC, dan RM kecil. Hasil *tuning* LQT menggunakan parameter GA-3 terdapat pada Tabel 4.13.

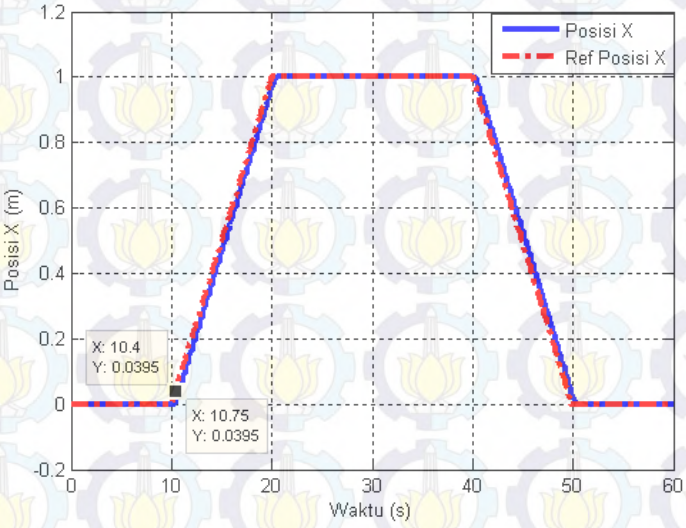
Tabel 4. 13 Parameter Hasil *Tuning* GA-3

No	Hasil *Tuning* GA-3	Nilai
1	Qx	711,0913
2	Qy	703,2811
3	Rx	0,1504
4	Ry	0,1584
5	RMSEx	1,99 %
6	RMSEy	1,99 %
7	IPx	83,0022
8	IPy	84,0136
9	Nilai *Fitness*	$9{,}4148 \times 10^{-4}$

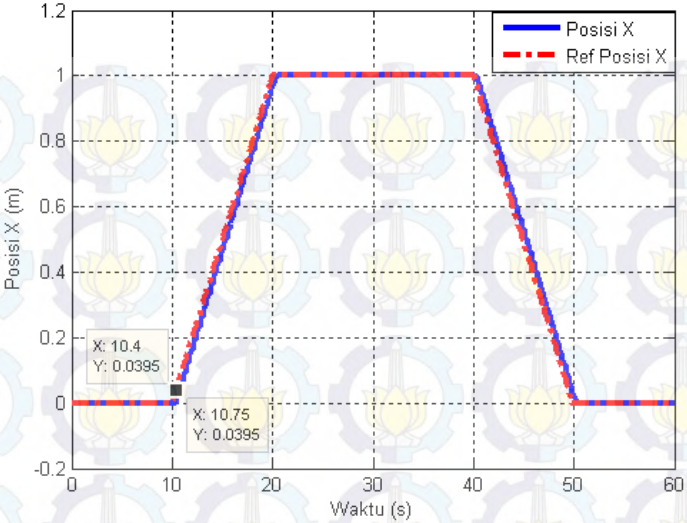
Hasil persebaran nilai *fitness* dari GA-3 terdapat pada Gambar 4.19, yaitu bernilai pada sekitar $9{,}4 \times 10^{-4}$. Pada pengujian sebelumnya telah dijelaskan bahwa pemilihan nilai RS, RC, dan RM yang besar mengakibatkan semakin besar kemungkinan hilangnya gen dengan sifat-sifat dominan. Nilai *fitness* yang konvergen menandakan bahwa hasil parameter Q dan R dari setiap generasi memiliki nilai yang hampir sama, di mana telah mendekati nilai yang optimal.



Gambar 4. 19 Nilai *Fitness* GA-3



Gambar 4. 20 Respon Simulasi LQT Pergerakan *Quadcopter* pada Sumbu X dengan Menggunakan Parameter GA-3



Gambar 4. 21 Respon Simulasi LQT Pergerakan *Quadcopter* pada Sumbu Y dengan Menggunakan Parameter GA-3

Hasil respon simulasi LQT pada pergerakan *quadcopter* dengan menggunakan parameter GA-3 pada sumbu X terdapat pada Gambar 4.20 dan sumbu Y terdapat pada Gambar 4.21 di mana kedua respon memiliki *lagging time* sebesar 0,35 detik

4.5.4 Simulasi Pengujian LQT dengan Kombinasi GA-4

Untuk mengetahui pengaruh RS, RC dan RM jika dibuat lebih kecil lagi, maka pada Pengujian GA-4 digunakan nilai parameter pada Tabel 4.14.

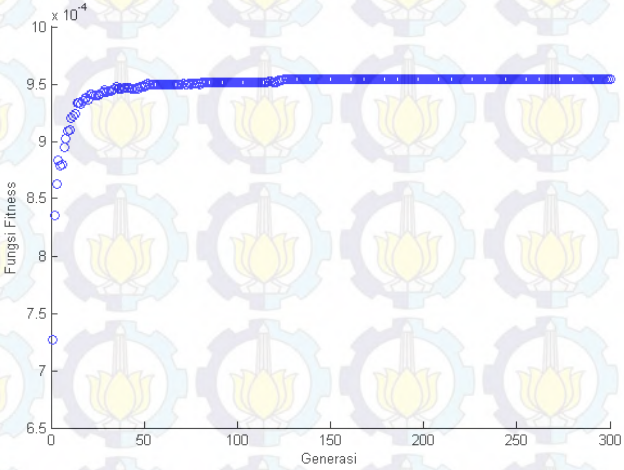
Tabel 4. 14 Parameter GA-4

No	Parameter GA	Nilai
1	Populasi (P)	100
2	Generasi (G)	300
3	Rasio Seleksi (RS)	0,2
4	Rasio *Crossover* (RC)	0,2
5	Rasio Mutasi (RM)	0,2

Hasil *tuning* LQT dengan menggunakan GA-4 terdapat pada Tabel 4.15 dan persebaran nilai *fitness* terdapat pada Gambar 4.22 di mana bernilai sama pada generasi ke 111 hingga akhir dari program. Hasil *tuning* GA dengan menggunakan parameter 3 dan 4 tidak memiliki perbedaan yang begitu besar. Keduanya memiliki nilai yang hampir sama karena telah mendekati hasil yang optimal.

Tabel 4. 15 Parameter Hasil *Tuning* GA-4

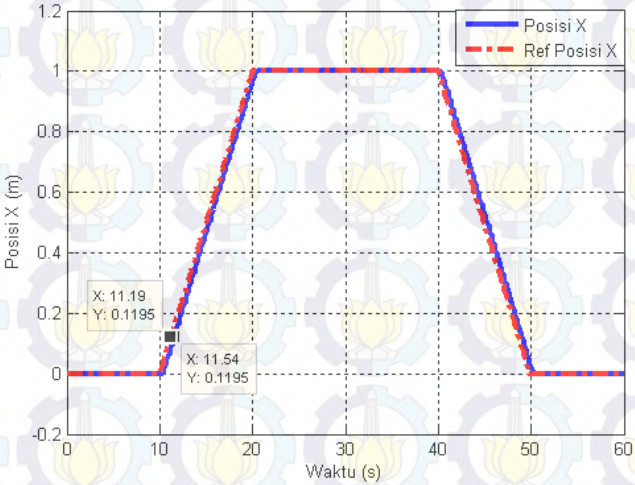
No	Hasil *Tuning* GA-4	Nilai
1	Qx	701,1289
2	Qy	700,5470
3	Rx	0,1575
4	Ry	0,1572
5	RMSEx	1,99 %
6	RMSEy	1,99 %
7	IPx	84,3374
8	IPy	83,5321
9	Nilai *Fitness*	9,5439 x 10^{-4}



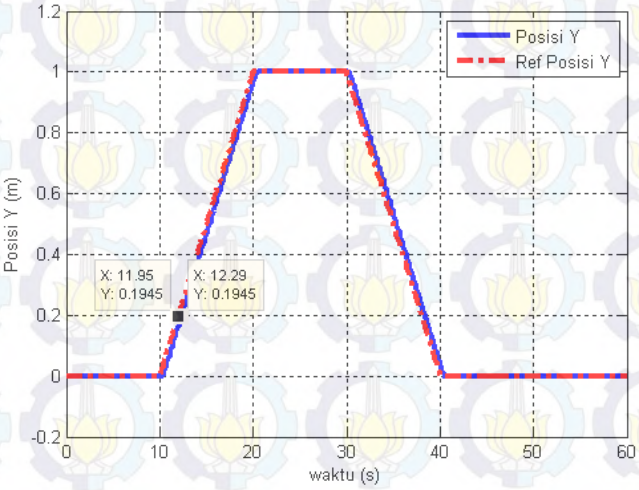
Gambar 4. 22 Nilai *Fitness* GA-4

Hasil respon simulasi LQT pada pergerakan *quadcopter* dengan menggunakan parameter GA-4 pada sumbu X dan sumbu Y dapat dilihat pada Gambar 4.23 dan Gambar 4.24. Karakteristik dari GA-3 dan GA-4

memiliki hasil yang sama yaitu memiliki *time lagging* dan RMSE yang hampir sama untuk sumbu X dan sumbu Y.



Gambar 4. 23 Respon Simulasi LQT Pergerakan *Quadcopter* pada Sumbu X dengan Menggunakan Parameter GA-4



Gambar 4. 24 Respon Simulasi LQT Pergerakan *Quadcopter* pada Sumbu Y dengan Menggunakan Parameter GA-4

4.5.5 Simulasi Pengujian LQT dengan Kombinasi GA-5

Untuk mengetahui pengaruh RS, RC, dan RM bernilai sangat kecil, maka pada pengjian ini digunakan parameter sesuai dengan Tabel 4.16.

Tabel 4. 16 Parameter GA-5

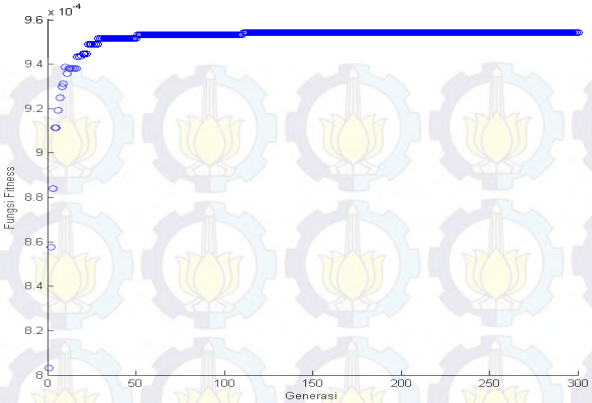
No	Parameter GA	Nilai
1	Populasi (P)	100
2	Generasi (G)	300
3	Rasio Seleksi (RS)	0,1
4	Rasio *Crossover* (RC)	0,1
5	Rasio Mutasi (RM)	0,1

Hasil *tuning* LQT dengan menggunakan parameer GA-5 terdapat pada Tabel 4.17.

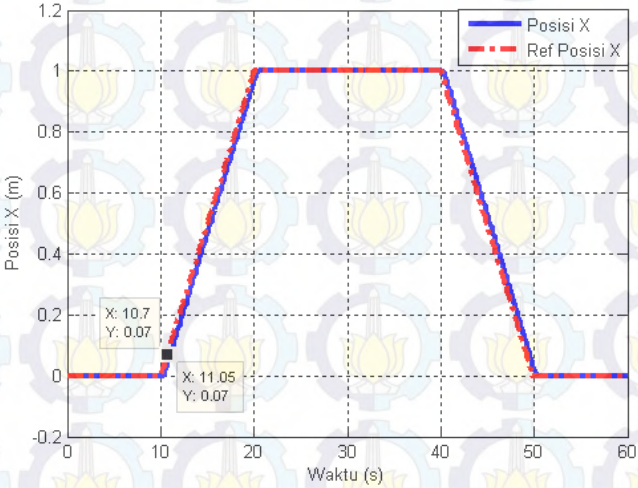
Tabel 4. 17 Parameter Hasil *Tuning* GA-5

No	Hasil *Tuning* GA-5	Nilai
1	Qx	700,1884
2	Qy	700,6315
3	Rx	0,1568
4	Ry	0,1579
5	RMSEx	1,99 %
6	RMSEy	1,99 %
7	IPx	84,0934
8	IPy	83,5227
9	Nilai *Fitness*	$9{,}5447 \times 10^{-4}$

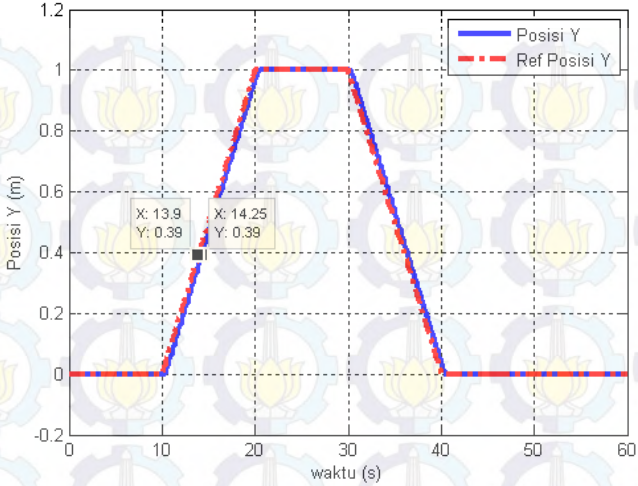
Persebaran nilai *fitness* terdapat pada Gambar 4.25 di mana nilai tersebut telah konvergen pada generasi ke 111. Hasil respon simulasi LQT pada pergerakan *quadcopter* dengan menggunakan parameter GA-5 pada sumbu X dan sumbu Y dapat dilihat pada Gambar 4.26 dan Gambar 4.27. Karakteristik dari GA-3, GA-4, GA-5 memiliki hasil yang sama yaitu memiliki *time lagging* dan RMSE yang sama untuk sumbu X dan sumbu Y. Hal tersebut dikarenakan ketiganya telah memiliki nilai *fitness* yang sama dan telah konvergen pada satu nilai. Sehingga dapat dikatakan bahwa GA-1, GA-2, dan GA-3 menghasilkan individu dengan solusi terbaik.



Gambar 4. 25 Nilai *Fitness* GA-5



Gambar 4. 26 Respon Simulasi LQT Pergerakan *Quadcopter* pada Sumbu X dengan Menggunakan Parameter GA-5



Gambar 4. 27 Respon Simulasi LQT Pergerakan *Quadcopter* pada Sumbu X dengan Menggunakan Parameter GA-5

Data perbandingan *tuning* GA yang digunakan sebagai parameter LQT terdapat pada Tabel 4.18.

Tabel 4. 18 Data Perbandingan Hasil *Tuning* LQT Menggunakan GA

Parameter		GA-1	GA-2	GA-3	GA-4	GA-5
INPUT	Populasi	100	100	100	100	100
	Generasi	100	300	300	300	300
	R.Seleksi	0,5	0,7	0,3	0,2	0,1
	R.*Crossover*	0,5	0,7	0,3	0,2	0,1
	R.Mutasi	0,5	0,7	0,3	0,2	0,1
OUTPUT	Qx	714,773	714,773	711,091	701,128	700,188
	Qy	756,902	756,902	703,281	700,547	700,631
	Rx	0,1026	0,1470	0,1504	0,1575	0,1568
	Ry	0,1402	0,1275	0,1584	0,1572	0,1579
	RMSEx	1,78 %	1,95 %	1,99 %	1,99 %	1,99 %
	RMSEy	1,89 %	1,85 %	1,99 %	1,99 %	1,99 %
	IPx	68,7717	82,2829	83,0022	84,3374	84,0934
	iPy	82,0097	78,2114	84,0136	83,5321	83,5227

Dari hasil pengujian *tuning* LQT menggunakan GA pada Tabel 4.18, didapatkan hasil paling optimal terdapat pada GA-5. Perancangan nilai *fitness* pada Persamaan (3.82) terdapat dua variabel yang digunakan yaitu indeks performansi dengan bobot 0,2 dan RMSE dengan bobot 0,3. Sehingga hasil *tuning* yang digunakan ialah hasil dari parameter GA-5 di mana memiliki nilai RMSE dan indeks performansi yang lebih kecil dari lainnya. Karakteristik LQT hasil respon GA-5 memiliki *time lagging* 0,35 detik. Namun nilai tersebut masih dapat ditoleransi karena bernilai sangat kecil.

4.6 Simulasi Pengujian Gerak Lateral *Quadcopter* pada Lintasan Berbentuk Segitiga

Penelitian Tugas Akhir ini membahas mengenai pengendalian *quadcopter* pada gerak lateral menggunakan kontroler LQT. Pengujian gerak lateral memiliki tujuan untuk memplot sumbu X dan sumbu Y dari pergerakan *quadcopter* sehingga membentuk lintasan segitiga. Kontroler LQT harus bisa men-*tracking trajectory* yang telah diberikan.

Gerak lateral atau gerak translasi terdiri dari dua macam yaitu gerak translasi sumbu X dan gerak translasi sumbu Y. Gerak translasi dapat dilaksanakan jika *quadcopter* dapat menjaga kestabilan di udara dan selanjutnya bergerak rotasi. Pengujian dilakukan dengan mensimulasikan sistem yang telah dirancang dengan menggunakan *signal builder*. Hal tersebut dikarenakan untuk men-*tracking* suatu *trajecktory*, *quadcopter* memerlukan proses untuk menuju *trajectory* tersebut. Pada *signal builder* dapat dibuat sinyal sesuai dengan keinginan. Sinyal yang dirancang berbentuk sinyal *ramp*.

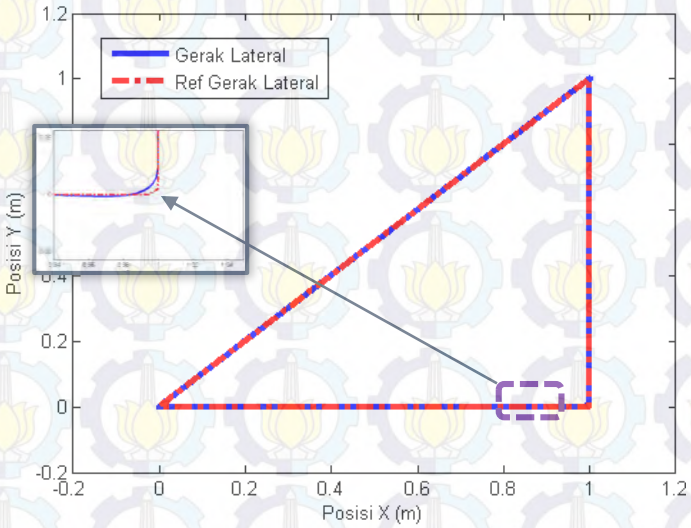
Lintasan segitiga dipilih karena memiliki sisi miring, di mana untuk melewati sisi miring tersebut, *quadcopter* harus melakukan perubahan sudut kurang dari 90°. Jika pergerakan sudut tersebut tidak seimbang, maka akan menyebabkan *quadcopter* bergerak pada satu sumbu dan dapat menyebabkan *quadcopter* jatuh.

Prosedur dari pergerakan *quadcopter* adalah sebagai berikut:

a. Waktu simulasi yang digunakan adalah 60 detik
b. *Quadcopter* telah melakukan *take off* pada detik ke 5 dan terbang *hover* pada ketinggian 1 meter dengan kordinat (0,0) selama 5 detik hingga detik ke 10

c. Pergerakan *quadcopter* dimulai dari kordinat (0,0) melewati sisi miring menuju kordinat (1,1) dari detik ke 10 hingga detik ke 20
d. *Quadcopter* terbang *hover* pada kordinat (1,1) selama 10 detik
e. *Quadcopter* bergerak pada sumbu Y dari kordinat (1,1) menuju kordinat (0,1) dari detik ke 30 hingga detik ke 40
f. *Quadcopter* akan bergerak pada sumbu X dari kordinat (0,1) menuju posisi awal (0,0) dari detik ke 40 hingga detik ke 50 dan terbang *hover* dari detik 50 hingga detik 60.

Pengujian ini dilakukan dengan menghiraukan gangguan yang ada atau bersifat ideal. Hasil respon simulasi gerak lateral *quadcopter* sesuai prosedur yang telah dibuat terdapat pada Gambar 4.28.



Gambar 4. 28 Respon Simulasi Gerak Lateral *Quadcopter* pada Lintasan Berbentuk Segitiga

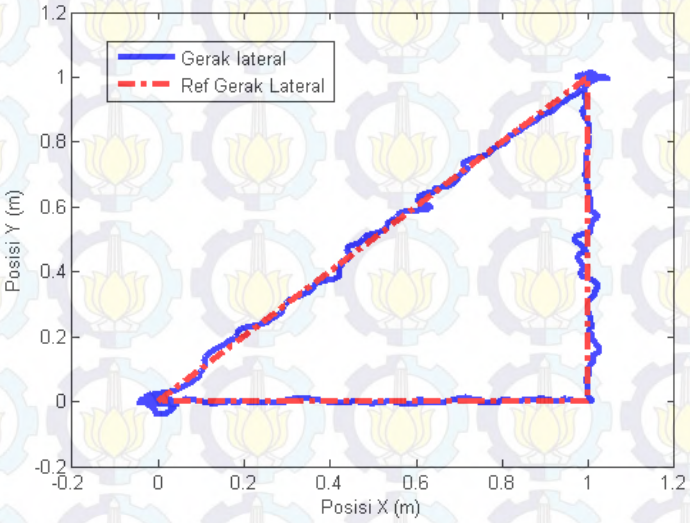
Dari Gambar 4.28 dapat diketahui bahwa *quadcopter* mampu mengikuti lintasan segitiga. Jika ditinjau pada kordinat (1,0) respon dari *quadcopter* tidak sesuai dengan referensi. Hal tersebut karena adanya perubahan sudut *roll* menjadi sudut *pitch*. Untuk melakukan gerak translasi pada sumbu X tanpa diikuti *hover* terlebih dahulu. Sedangkan pada kordinat (1,1) *quadcopter* akan melakukan *hover* terlebih dahulu

selama 10 detik sebelum melakukan perubahan sudut *roll* untuk pergerakan pada sumbu Y.

4.7 Simulasi Pengujian Gerak Lateral *Quadcopter* pada Lintasan Berbentuk Segitiga dengan *Noise*

Untuk menguji kemampuan kontroler yang telah dirancang, pada pengujian simulasi ini ditambahkan *noise* pada sinyal kontrol. *Noise* yang digunakan merupakan *noise random*. Penambahan *noise* ini dapat direpresentasikan dengan angin yang mendorong *quadcopter* dari samping.

Hasil penambahan *noise* sinyal *random* pada gerak lateral *quadcopter* terdapat pada Gambar 4. 31. Sinyal *random* ditambahkan pada sinyal kontrol LQT yang merupakan sinyal referensi dari kontroler LQR dengan parameter *noise* memiliki nilai minimum -0,1 dan maksimum 0,1. Hasil respon gerak lateral *quadcopter* dengan ditambahkan *noise* terdapat pada Gambar 4.29.

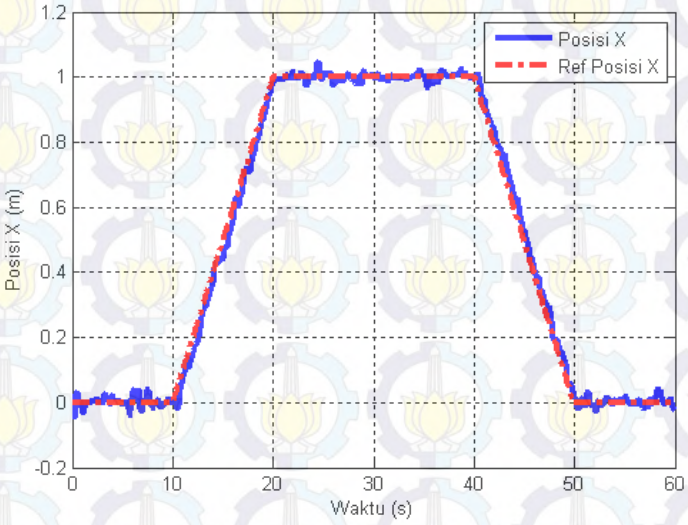


Gambar 4. 29 Respon Simulasi Gerak Lateral *Quadcopter* pada Lintasan Berbentuk Segitiga dengan *Noise* Sinyal *Random*

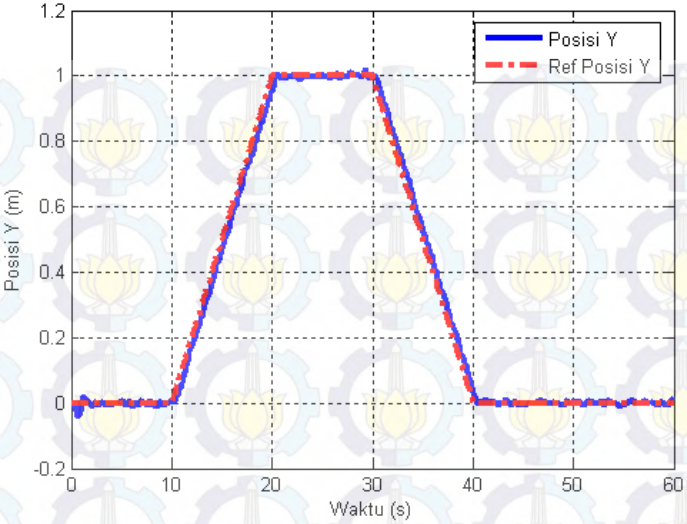
Pada Gambar 4.29 terlihat bahwa penambahan *noise* tidak membuat *quadcopter* keluar dari lintasan yang digunakan. *Noise* tersebut

mengakibatkan adanya *error* pada pergerakan *quadcopter* karena kontroler LQR mendapatkan referensi secara terus menerus dari *noise*. Sehingga mengakibatkan terjadi berubahan sudut *quadcopter* yang mengakibatkan adanya *error* dalam pergerakan. Namun kontroler yang telah dirancang masih dapat mengendalikan *quadcopter* untuk mengikuti referensi berbentuk segitiga.

Hasil respon pergerakan *quadcopter* pada sumbu X dan Y dengan penambahan *noise* sinyal *random* terdapat pada Gambar 4.30 dan Gambar 4.31 di mana pada gerak sumbu X memiliki RMSE sebesar 2,38% dan pada gerak sumbu Y memiliki RMSE sebesar 2,06%. *Error* tersebut masih dapat ditoleransi karena bernilai kecil dan tidak menyebabkan *quadcopter* keluar dari lintasan segitiga yang telah ditentukan



Gambar 4. 30 Respon Simulasi Pergerakan *Quadcopter* pada Sumbu X pada Lintasan Berbentuk Segitiga dengan *Noise* Sinyal *Random*

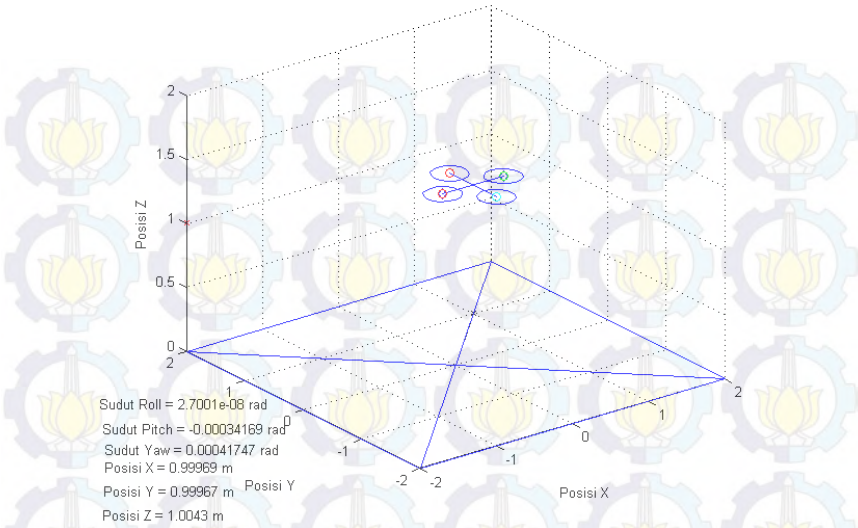


Gambar 4. 31 Respon Simulasi Pergerakan *Quadcopter* pada Sumbu Y pada Lintasan Berbentuk Segitiga dengan *Noise* Sinyal *Random*

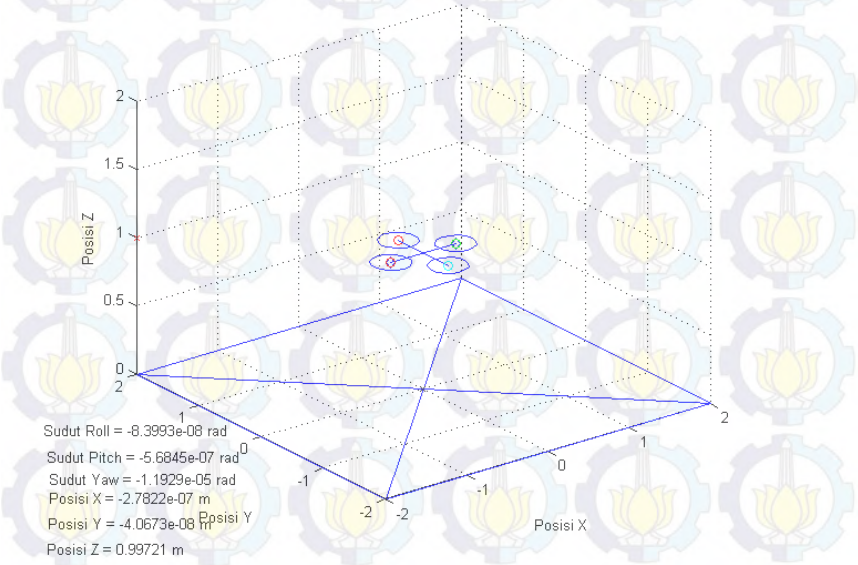
4.8 Simulasi 3D Gerak Lateral *Quadcopter* pada Lintasan Berbentuk Segitiga

Simulasi ini digunakan simulasi 3D untuk mengetahui pergerakan *quadcopter* yang sesungguhnya. Simulasi 3D ini menggunakan *trajectory* berbentuk segitiga. Pada simulasi 3D dibatasi sumbu X, Y, Z bernilai 2 meter untuk mempermudah dalam pembuatan dan ditunjukkan pada Gambar 4.32 dan Gambar 4.33.

Gambar 4.34 adalah salah satu bagian dari simulasi 3D yaitu ketika *quadcopter* bergerak pada sumbu Y pada kordinat (1,1) dengan ketinggian 1 meter. Pada Gambar 4.35 merupakan bagian dari simulasi 3D di mana *quadcopter* telah selesai dalam melakukan pergerakan. Gambar tersebut menunjukkan bahwa *quadcopter* dalam keadaan *hover* pada kordinat (0,0) dengan ketinggian 1 meter.



Gambar 4. 32 Simulasi 3D *Quadcopter* pada Saat *Quadcopter* Berada di Kordinat (1,1)



Gambar 4. 33 Simulasi 3D *Quadcopter* pada Saat *Quadcopter* Berada di Kordinat (0,0)

BAB V
PENUTUP

Hasil dari perancangan dan penelitian Tugas Akhir dirangkum dan dirumuskan kesimpulan. Kesimpulan ini menerangkan hasil dari pengujian dan simulasi yang telah dilaksanakan.

Selama proses perancangan dan penelitian, terdapat banyak kendala yang dihadapi. Kendala tersebut telah penulis rangkum dan dirumuskan dalam bentuk saran untuk penyempuranaan dan penelitian lebih lanjut

5.1 Kesimpulan

Berdasarkan hasil pengujian dan analisis, dapat diperoleh kesimpulan sebagai berikut:

a) Metode *Linear Quadratic Tracking* dengan *tuning* menggunakan Algoritma Genetika dapat digunakan untuk mengendalikan gerak lateral *quadcopter* pada lintasan berbentuk segitiga.
b) Metode *tuning* menggunakan GA dapat menggantikan *tuning* manual (*try and error*). Hasil *tuning* GA memberikan hasil yang optimal pada variabel fungsi *fitness* yang digunakan, yaitu minimum indeks performansi sistem dan RMSE dari setiap pergerakan gerak lateral sumbu X dan sumbu Y.
c) Penggunaan bobot (rasio) pada variabel yang digunakan dalam perancangan GA mempengaruhi nilai *fitness* setiap generasi. Nilai *fitness* yang optimal didapatkan dengan nilai bobot yang rendah, yaitu bernilai 0,1 untuk rasio seleksi, rasio *crossover*, dan rasio mutasi dengan *fitness* $9{,}5447 \times 10^{-4}$
d) Hasil *tuning* GA yang digunakan pada LQT memiliki nilai Qx 700,1884, Qy 700,6315, Rx 0,1568, dan Ry 0,1579. Respon LQT tersebut memiliki RMSE pada sumbu X dan sumbu Y sebesar 1,99 % serta memiliki *time lagging* 0,35 detik.
e) Kontroler yang telah dirancang belum dapat diimplementasikan karena Ardupilot 2.6 tidak dapat diubah *script*-nya dan belum dapat membentuk komunikasi antara MATLAB dengan kontroler.

5.2 Saran

Dari hasil penelitian yang dilakukan, untuk pengembangan berikutnya, disarankan beberapa hal berikut ini:

a) Pemodelan dan pemahaman tentang *quadcopter* akan membantu perancangan sistem yang lebih baik dan mendapatkan model matematika yang lebih akurat.
b) Pengunaan program GA masih bisa didapatkan nilai yang lebih optimal dengan mengubah nilai dari rasio seleksi, *crossover*, dan mutasi.
c) Penggunaan metode LQT dan LQR perlu adanya perbaikan untuk menghilangkan *delay* pada respon.
d) Perlu adanya pembanding antara metode *tuning* GA dan metode *tuning* lainnya seperti *neural network* dll. Hal tersebut dimaksudkan untuk mencari nilai parameter *Linear Quadratic Control* yang lebih optimal.

DAFTAR PUSTAKA

[1] Bouabdallah, S., Murrieri, P., and Siegwart, R., "Towards Autonomous Indoor Micro VTOL", *Autonomous Robots*, vol. 18, 2005

[2] Argentim, L.M., Rezende, W.C., Santos, P.E., and Aguiar, R.A., "PID, LQR, and LQR-PID on a Quadcopter Platform", *IEEE Transaction on Aerospace and Electrical Systems*, 2013

[3] Tommaso Bresciani, "Modelling, Identification and Control of a Quadrotor Helicopter". Department of Automatic Control Lund University, *Thesis*, 2008

[4] Spong, M.W, Vidyasagar, M. "*Robot Dynamic and Control*", Prentice Hall, New Jersey, 2004

[5] Naidu, Subbaram D, "*Optimal Control Systems*", CRC Press Idaho, Ch.3-4, 2002

[6] M. Yuhendri, M. Ashari and M. Purnomo, "Linear Quadratic Regulator Design For Modular Matrix Converter Using Genetic Algorithm," *IEEE Control Systems Magazines*, pp. 175-179, 2011

[7] Vishal, Ohri Jyoti, "GA Tuned LQR and PID Controller for Aircraft Pitch Control," *IEEE 6th India International Conference on Power Electronics (IICPE)*, 2014

[8] H.K. Khalil and JW Grizzle, "*Nonlinear systems",* Prentice Hall, New Jersey, 2002

[9] ……."*Ardupliot Mega 2.6 Controller*", www.readymaderc.com. diakses pada tanggal 10 Januari 2016

[10] ……."*FrSky V8FR-II*", www.frsky-rc.com, diakses pada tanggal 10 Januari 2016

[11] ……."*Turnigy 9XR*", www.turnigy.com, diakses pada tanggal 10 Januari 2016

[12] ……."*ESC TBS Bulletproof 30A*", www.team-blacksheep.com, diakses pada tanggal 10 Januari 2016

[13] ……."*Sunnysky v2216 Brushless Motor 900KV*", www.multirotorsuperstore.com, diakses pada tanggal 10 Januari 2016

[14] ……."*Mission Planner*", www.planner.ardupilot.com, diakses pada tanggal 10 Januari 2016

--halaman ini sengaja dikosongkan--

LAMPIRAN A

A1. Pengukuran Kecepatan Motor

Pulsa	Motor 1	Motor 2	Motor 3	Motor 4
75	114.505	111.888	109.376	113.668
80	152.29	152.604	148.836	154.069
85	187.563	192.900	187.772	193.633
90	219.276	225.242	221.684	229.429
95	252.77	258.108	256.747	263.76
100	281.03	290.973	287.31	295.578
105	313.267	325.722	322.373	330.223
110	342.26	355.552	351.156	360.262
115	369.892	384.964	380.986	390.93
120	399.198	416.468	412.386	422.958
125	427.04	447.031	440.646	452.16
130	453.625	471.732	466.29	475.396
135	479.164	500.411	496.957	508.575
140	502.295	503.97	519.042	533.172
145	525.531	523.752	543.848	558.292
150	546.569	548.872	566.037	582.784
155	570.538	572.212	574.410	606.334
160	590.529	595.344	592.832	630.721
165	611.567	615.126	609.997	654.271
170	633.024	629.465	627.790	656.050
175	651.236	653.434	650.817	661.702
180	673.111	670.076	672.378	673.844
185	676.356	672.169	675.937	676.565

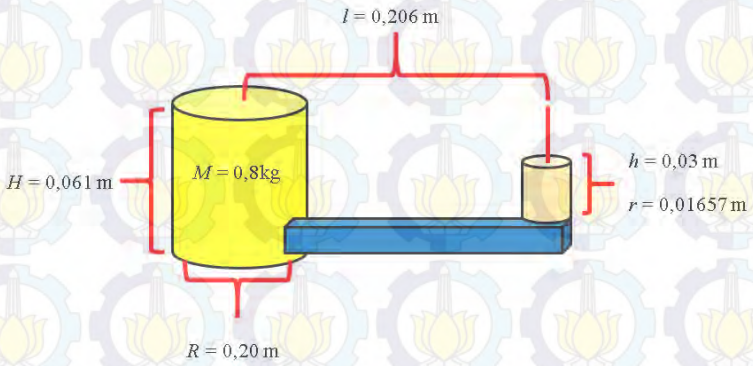
Pulsa	Motor 1	Motor 2	Motor 3	Motor 4
190	673.006	672.483	673.948	675.832
195	672.902	674.053	673.00	675.1
200	673.006	674.262	672.797	675.623

A2 Pengukuran Gaya Angkat Motor

Pulsa	Gaya Angkat 1 (kg)	Gaya Angkat 2 (kg)	Gaya Angkat 3 (kg)	Gaya Angkat 4 (kg)
75	0.02	0.011	0.017	0.014
80	0.036	0.031	0.033	0.039
85	0.058	0.055	0.055	0.061
90	0.078	0.08	0.077	0.084
95	0.104	0.107	0.102	0.115
100	0.131	0.138	0.134	0.145
105	0.165	0.177	0.17	0.184
110	0.198	0.207	0.209	0.221
115	0.227	0.252	0.243	0.264
120	0.27	0.297	0.285	0.306
125	0.31	0.34	0.328	0.357
130	0.352	0.39	0.374	0.402
135	0.396	0.432	0.426	0.452
140	0.438	0.446	0.478	0.498
145	0.488	0.475	0.522	0.553
150	0.533	0.54	0.578	0.611
155	0.582	0.605	0.613	0.664
160	0.63	0.626	0.619	0.718
165	0.672	0.654	0.646	0.771

Pulsa	Gaya Angkat 1 (kg)	Gaya Angkat 2 (kg)	Gaya Angkat 3 (kg)	Gaya Angkat 4 (kg)
170	0.723	0.715	0.704	0.783
175	0.772	0.784	0.762	0.791
180	0.834	0.832	0.826	0.825
185	0.842	0.846	0.84	0.847
190	0.833	0.833	0.836	0.841
195	0.832	0.834	0.834	0.845
200	0.834	0.836	0.837	0.845

A3. Identifikasi Fisik *Quadcopter*



Identifikasi fisik konstanta

$$I_{z.frame1,4} = \frac{m_l}{12}(p^2 + l^2)$$

$$I_{z.frame2,3} = \frac{m_l}{12}(p^2 + t^2)$$

$$I_{z.m1,2,3,4} = \frac{m_m}{2} r^2 + (p_l)^2 . M$$

$$I_{z.batt} = \frac{m_b}{12}(p^2 + l^2)$$

$$I_{z.1} = \frac{m_{mika}}{12}(p^2 + l^2)$$

$$I_{z.2} = \frac{m_{mika}}{12}(2.l^2)$$

$$I_{z.1,2,3,4} = \frac{m_p}{2} r^2 + (p_1)^2 . M_p$$

$$I_{x.frame1,2,3} = \frac{m_l}{12}(l^2 + t^2)$$

$$I_{x.frame4,5,6} = \frac{m_l}{12}(p^2 + l^2)$$

$$I_{z.m2,4} = m_m \left(\frac{r^2}{4} + \frac{h}{12} \right) + [(0.5h)^2 + (p_1 - p_2)^2] \, m$$

$$I_{z.m1,3} = m_m \left(\frac{r^2}{4} + \frac{h}{12} \right) + (0.5h)^2 \, m$$

$$I_{x.batt} = \frac{m_b}{12}(l^2 + t^2) + (0.5t)^2$$

$$I_{y.batt} = \frac{m_b}{12}(p^2 + t^2) + (0.5t)^2$$

$$I_{x.mika1} = \frac{m_{mika}}{12}(p^2 + t^2) + (t^2) \, m_{mika}$$

$$I_{x.mika2,3} = \frac{m_{mika}}{12}(l^2 + t^2) + (t^2) \, m_{mika}$$

$$I_{y.mika1} = \frac{m_{mika}}{12}(l^2 + t^2) + (t^2) \, m_{mika}$$

$$I_{x.mika2,3} = \frac{m_{mika}}{12}(l^2 + t^2) + (t^2 + l^2) \, m_{mika}$$

$$I_{x.1,2} = m_p \left(\frac{{p_2}^2}{6} + \frac{t^2}{12} + {p_1}^2 + (0.5t^2 + h)^2 \right)$$

$I_{x.3,4} = m_p \left(\frac{{p_2}^2}{6} + \frac{t^2}{12} \right)$

Perhitungan I_{xx}, I_{yy}, I_{zz} dan J_{TP}

$$I_{xx} = \frac{mr^2}{4} + \frac{mh^2}{6} + 2mr^2 + \frac{MR^2}{4} + \frac{MH^2}{12}$$

$I_{xx} = 1{,}68 \times 10^{-3}$

$$I_{yy} = \frac{mr^2}{4} + \frac{mh^2}{6} + 2mr^2 + \frac{MR^2}{4} + \frac{MH^2}{12}$$

$I_{yy} = 1{,}68 \times 10^{-3}$

$$I_{zz} = \frac{MR^2}{2} + 4mr^2$$

$Izz = 1{,}25 \times 10^{-3}$

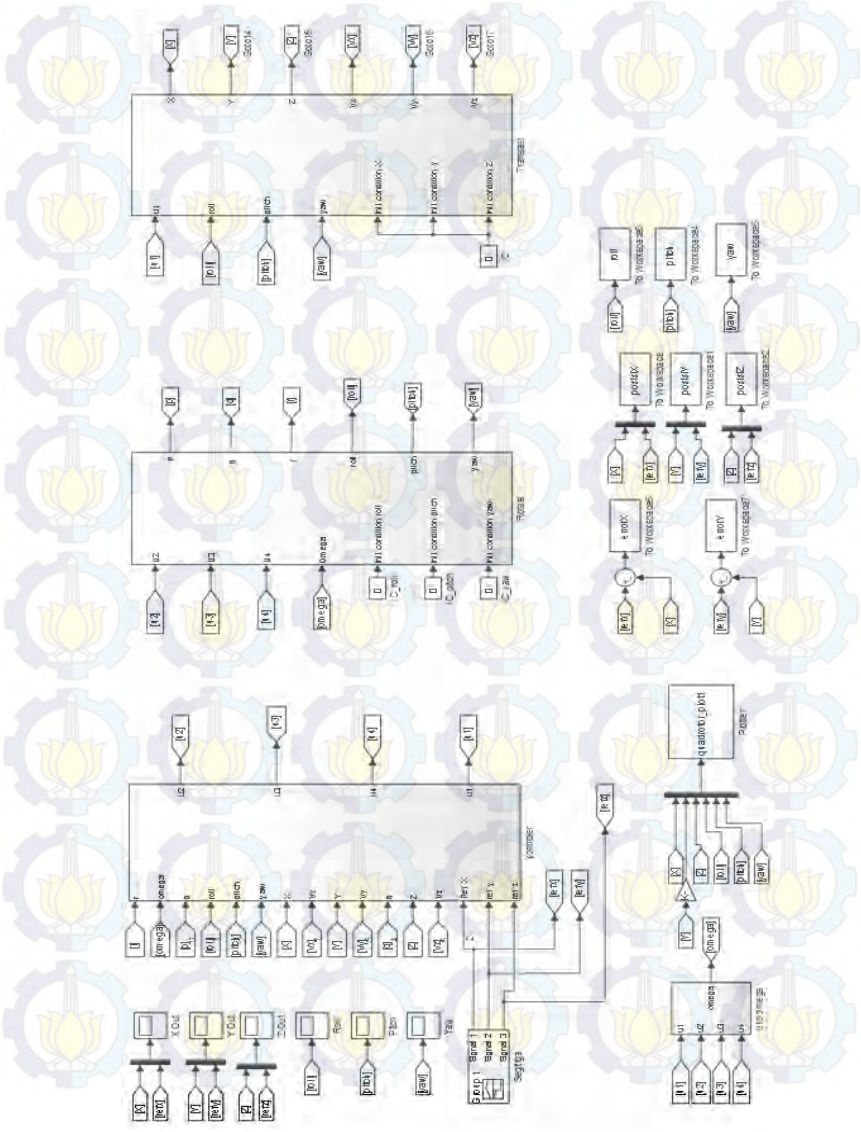
$$J_{TP} = \frac{4 \times \left(mr^2 \right)}{2}$$

$J_{TP} = 0{,}21175 \times 10^{-4}$

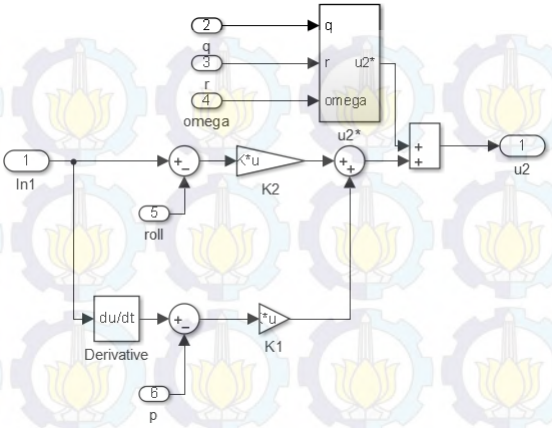
--halaman ini sengaja dikosongkan--

LAMPIRAN B

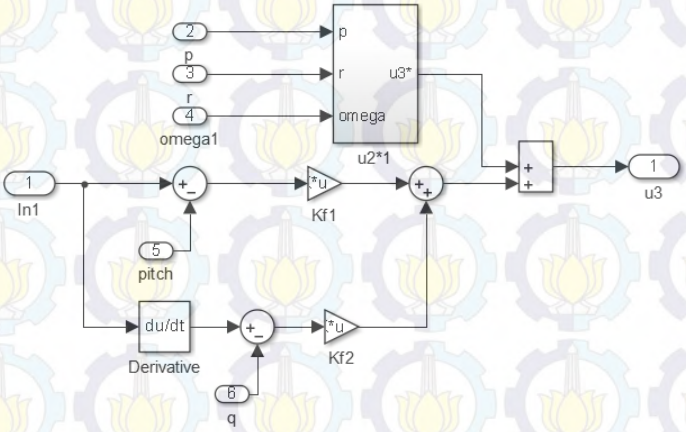
B1. Simulink Keseluruhan Sistem



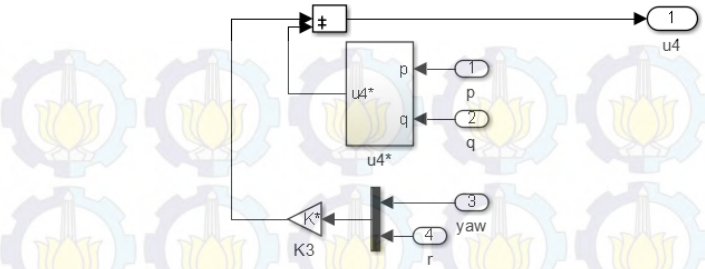
B2 Simulink Pengendalian Sudut *Roll*



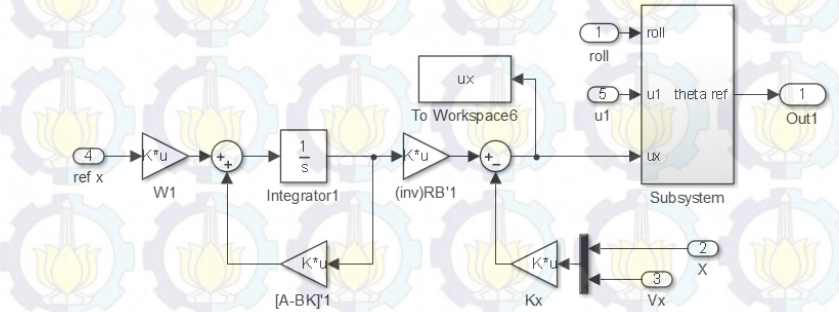
B3. Simulink Pengendalian Sudut *Pitch*



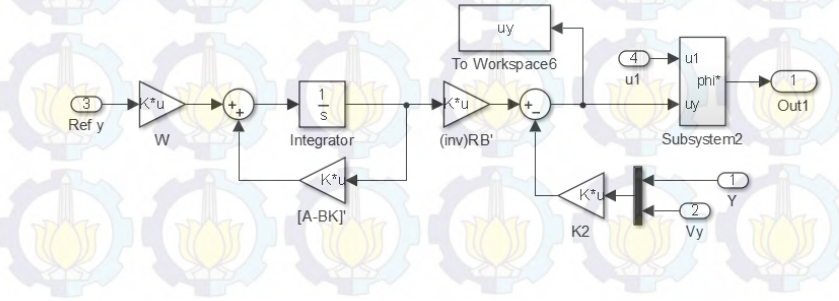
B4. Simulink Pengendalian Sudut *Yaw*



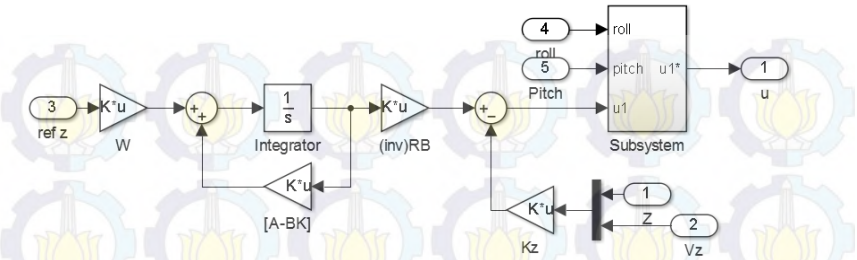
B5. Simulink Pengendalian Posisi X



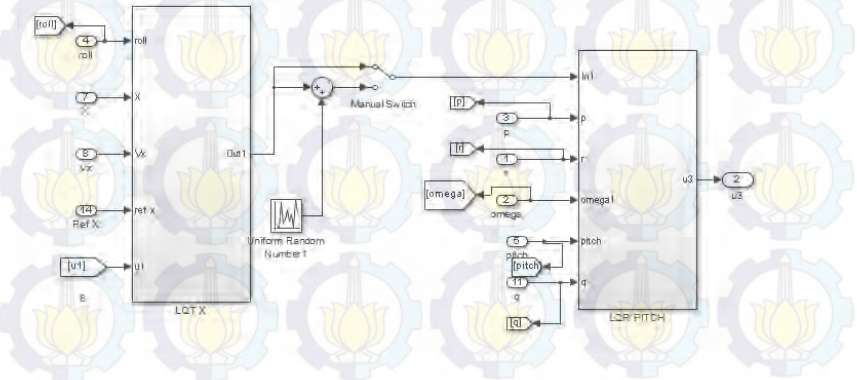
B6. Simulink Pengendalian Posisi Y



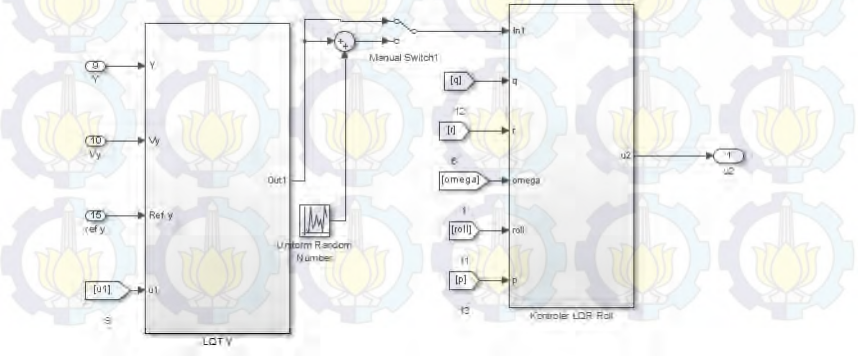
B7. Simulink Pengendalian Posisi Z



B8. Simulink *Cascade* LQT LQR Sumbu X



B9. Simulink *Cascade* LQT LQR Sumbu Y



LAMPIRAN C

C1. Program MATLAB Pencarian Nilai Parameter $\dot{p}$

```
p(1)=[];
pdot(1)=[];
r(1)=[];
q(1)=[];
u2(1)=[];
omega(1)=[];

qr2=sum((q.*r).^2);
qro=sum((q.^2).*r.*omega);
qru=sum(q.*r.*u2);
qo2=sum((q.*omega).^2);
qou=sum((q.^2).*u2.*omega);
u22=sum(u2.^2);
pqr=sum(pdot.*q.*r);
pqo=sum(pdot.*q.*omega);
pu2=sum(pdot.*u2);
M=[qr2 qro qru;qro qo2 qou;qru qou u22];
x=[pqr pqo pu2]*inv(M);
```

C2. Program MATLAB Pencarian Nilai Parameter $\dot{q}$

```
p(1)=[];
qdot(1)=[];
r(1)=[];
q(1)=[];
u3(1)=[];
omega(1)=[];

pr2=sum((p.*r).^2);
pro=sum((p.^2).*r.*omega);
pru=sum(p.*r.*u3);
po2=sum((p.*omega).^2);
pou=sum((p.^2).*u3.*omega);
u32=sum(u3.^2);
pqr=sum(qdot.*p.*r);
pqo=sum(qdot.*p.*omega);
pu3=sum(qdot.*u3);
M=[pr2 pro pru;pro po2 pou;pru pou u32];
x=[pqr pqo pu3]*inv(M);
```

C3. Program MATLAB Pencarian Nilai Parameter $\dot{r}$

```
p(1)=[];
```

```
rdot(1)=[];
r(1)=[];
q(1)=[];
u4(1)=[];

pq2=sum((p.*q).^2);
pqu=sum(p.*q.*u4);
u42=sum(u4.^2);
pqr=sum(rdot.*p.*q);
ru3=sum(rdot.*u4);
M=[pq2 pqu;pqu u42];
x=[pqr ru3]*inv(M);
```

C4. Program LQR LQT pada MATLAB

```
%=====================================================%
parameter rotasi plant!
%parameter pdot!
a1=-0.5495;
b1=-0.0017;
c1=0.2052;
%parameter qdot-------------------------------------
a2=0.1675;
b2=-0.0094;
c2=2.9555;
%parameter zdot-------------------------------------
a3=-2.0257;
b3=0.0594;
%parameter quadcopter------------------------------
m=1.26;
%=====================================================
%LQR!
%LQR pitch!
Apitch=[0 1;0 0];
Bpitch=[0 2.9555]';
Cpitch=[1 0];
Qpitch=[1000 0;0 4];
Rpitch=0.1;
Xpitch,Lpitch,Kpitch]=care(Apitch,Bpitch,Qpitch,Rpitc;
Kipitch=Kpitch(1);
Kfpitch=[0 Kpitch(2)];

%LQR roll!
Aroll=[0 1;0 0];
Broll=[0 0.2052]';
Croll=[1 0];
Qroll=[800 0;0 4];
Rroll=0.00001;
Xroll,Lroll,Kroll]=care(Aroll,Broll,Qroll,Rroll);
Kiroll=Kroll(1);
```

```
Kfroll=[0 Kroll(2)];

%LQR yaw!
Ayaw=[0 1;0 0];
Byaw=[0 0.0594]';
Cyaw=[1 0];
Qyaw=[89 0;0 89];
Ryaw=1;
Xyaw,Lyaw,Kyaw]=care(Ayaw,Byaw,Qyaw,Ryaw);
Kiyaw=Kyaw(1);
Kfyaw=[0 Kyaw(2)];
%=====================================================
%LQT!
%LQT Z doble dot!
Az=[0 1;0 0];
Bz=[0 1]';
Cz=[1 0];
Qz=1000;
Rz=0.00001;
Sz,oz,mz,nz]=care(Az,Bz,Cz'*Qz*Cz,Rz);
Kz=inv(Rz)*Bz'*Sz;
ACLz=(Az-Bz*Kz)';
Lz=inv(Rz)*Bz';
%-----------------------------------------------------
%LQT x!
Ax=[0 1;0 0];
Bx=[0 1]';
Cx=[1 0];
Qx=101.5438;
Rx=10.2777;
Sx,ox,mx,nx]=care(Ax,Bx,Cx'*Qx*Cx,Rx);
Kx=inv(Rx)*Bx'*Sx;
ACLx=(Ax-Bx*Kx)';
Lx=inv(Rx)*Bx';
%-----------------------------------------------------
%LQT y doble dot!
Ay=[0 1;0 0];
By=[0 1]';
Cy=[1 0];
Qy=101.1275;
Ry=10.0238;
Sy,oy,my,ny]=care(Ay,By,Cy'*Qy*Cy,Ry);
Ky=inv(Ry)*By'*Sy;
ACLy=(Ay-By*Ky)';
Ly=inv(Ry)*By';
```

C5. Program GA pada MATLAB

```
%Penghitungan dengan menggunakan Algoritma Genetika!
%=====================================================!
```

```
Npop=input('Jumlah Populasi=');
jumlah_literasi=input('Jumlah Generasi=');
rasio_seleksi=input('Rasio Seleksi=');
p_crossover=input('Rasio Crossover=');
p_mutasi=input('Rasio Mutasi=');
Nbits=4;
literasi=1;

%Pembangkitan populasi awal
POP=generate3(Npop,Nbits);

while literasi<jumlah_literasi
    literasi=literasi+1;
    %Penghitungan nilai fitness
    [POP1]=fitnesscoba(POP,Npop);
    %plot hasil bagian 1
    dscatter(literasi-1,1)=literasi-1;
    dscatter(literasi-1,2)=POP1(1,5);
    %Seleksi Truncking dan Mesin Roullet
    [POP3]=seleksi(POP1,rasio_seleksi);
    %Crossover
    [POP4]=crossover(POP3,p_crossover);
    %Mutasi
    [POP]=mutasi(POP4,p_mutasi,Npop);
    assignin('base','fff',POP);
end;
%Penghitungan nilai fitness bagian II
[POP6]=fitnesscoba(POP,Npop);
%plot hasil bagian 2
dscatter(literasi,1)=literasi;
dscatter(literasi,2)=POP6(1,5);
%menampilkan nilai Q dan R paling optimal
Qx=POP6(1,1);
Qy=POP6(1,2);
Rx=POP6(1,3);
Ry=POP6(1,4);

%=====================================================!
%parameter plant!
%parameter pdot!
a1=-0.5495;
b1=-0.0017;
c1=0.2052;
%parameter qdot!
a2=0.1675;
b2=-0.0094;
c2=2.9555;
%parameter zdot
a3=-2.0257;
b3=0.0594;
```

```
m=1.26;

%=====================================================!
%Kontroler ROTASI!
%LQR!
%LQR pitch!
Apitch=[0 1;0 0];
Bpitch=[0 2.9555]';
Cpitch=[1 0];
Qpitch=[1000 0;0 4];
Rpitch=0.1;
[Xpitch,Lpitch,Kpitch]=care(Apitch,Bpitch,Qpitch,Rpitc
h);
Kipitch=Kpitch(1);
Kfpitch=[0 Kpitch(2)];
%LQR roll!
Aroll=[0 1;0 0];
Broll=[0 0.2052]';
Croll=[1 0];
Qroll=[1000 0;0 1];
Rroll=0.00001;
[Xroll,Lroll,Kroll]=care(Aroll,Broll,Qroll,Rroll);
Kiroll=Kroll(1);
Kfroll=[0 Kroll(2)];
%LQR yaw!
Ayaw=[0 1;0 0];
Byaw=[0 0.0594]';
Cyaw=[1 0];
Qyaw=[1000 0;0 4];
Ryaw=0.0001;
[Xyaw,Lyaw,Kyaw]=care(Ayaw,Byaw,Qyaw,Ryaw);
Kiyaw=Kyaw(1);
Kfyaw=[0 Kyaw(2)];

%=====================================================!
%Kontroler TRANSLASI!
%LQT Z!
Az=[0 1;0 0];
Bz=[0 0.7936]';
Cz=[1 0];
Qz=1000;
Rz=0.0001;
[Sz,oz,mz,nz]=care(Az,Bz,Cz'*Qz*Cz,Rz);
Kz=inv(Rz)*Bz'*Sz;
ACLz=(Az-Bz*Kz)';
Lz=inv(Rz)*Bz';
%LQT X!
Ax=[0 1;0 0];
Bx=[0 1]';
Cx=[1 0];
```

```
[Sx,ox,mx,nx]=care(Ax,Bx,Cx'*Qx*Cx,Rx);
Kgx=inv(Rx)*Bx'*Sx;
ACLx=(Ax-Bx*Kgx)';
Lx=inv(Rx)*Bx';
%LQT Y!
Ay=[0 1;0 0];
By=[0 1]';
Cy=[1 0];
[Sy,oy,my,ny]=care(Ay,By,Cy'*Qy*Cy,Ry);
Kgy=inv(Ry)*By'*Sy;
ACLy=(Ay-By*Kgy)';
Ly=inv(Ry)*By';
%=====================================================!
%running Program!
simopt=simset('solver','ode45','srcWorkspace','current
','dstWorkspace','current');
y=sim('cobaGA',[],simopt);
tX=posisiX.time;%plot(tX,posisiX.signals.values(:,1))!
tY=posisiY.time;%plot(tY,posisiY.signals.values(:,1))!
tZ=posisiZ.time;%plot(tZ,posisiZ.signals.values(:,1))!
troll=roll.time;
tpitch=pitch.time;
yaw=yaw.time;
```

C6. Program Menentukan Nilai *Fitness* GA pada MATLAB

```
%Fungsi Fitness
function [POP1]=fitnesscoba(POP,Npop)
%parameter pdot!
a1=-0.5495;
b1=-0.0017;
c1=0.2052;
%parameter qdot!
a2=0.1675;
b2=-0.0094;
c2=2.9555;
%parameter zdot
a3=-2.0257;
b3=0.0594;
m=1.26;

%LQTX!
Ax=[0 1;0 0];
Bx=[0 1]';
Cx=[1 0];
%LQTY!
Ay=[0 1;0 0];
By=[0 1]';
Cy=[1 0];
%LQTZ!
```

```
Az=[0 1;0 0];
Bz=[0 0.7936]';
Cz=[1 0];
Qz=1000;
Rz=0.0001;
[Sz,oz,mz,nz]=care(Az,Bz,Cz'*Qz*Cz,Rz);
Kz=inv(Rz)*Bz'*Sz;
ACLz=(Az-Bz*Kz)';
Lz=inv(Rz)*Bz';

%LQR!
%LQR pitch!
Apitch=[0 1;0 0];
Bpitch=[0 2.9555]';
Cpitch=[1 0];
Qpitch=[1000 0;0 4];
Rpitch=0.1;
[Xpitch,Lpitch,Kpitch]=care(Apitch,Bpitch,Qpitch,Rpitch);
Kipitch=Kpitch(1);
Kfpitch=[0 Kpitch(2)];

%LQR roll!
Aroll=[0 1;0 0];
Broll=[0 0.2052]';
Croll=[1 0];
Qroll=[1000 0;0 1];
Rroll=0.00001;
[Xroll,Lroll,Kroll]=care(Aroll,Broll,Qroll,Rroll);
Kiroll=Kroll(1);
Kfroll=[0 Kroll(2)];

%LQR yaw!
Ayaw=[0 1;0 0];
Byaw=[0 0.0594]';
Cyaw=[1 0];
Qyaw=[1000 0;0 4];
Ryaw=0.0001;
[Xyaw,Lyaw,Kyaw]=care(Ayaw,Byaw,Qyaw,Ryaw);
Kiyaw=Kyaw(1);
Kfyaw=[0 Kyaw(2)];

%inisialisasi!
Qx=0;
Qy=0;
Rx=0;
Ry=0;
for i=1:Npop;
    Qx=POP(i,1);
    Qy=POP(i,2);
    Rx=POP(i,3);
```

```
    Ry=POP(i,4);
    [Sx,ox,mx,nx]=care(Ax,Bx,Cx'*Qx*Cx,Rx);
    [Sy,oy,my,ny]=care(Ay,By,Cy'*Qy*Cy,Ry);
    Kgx=inv(Rx)*Bx'*Sx;
    Kgy=inv(Ry)*By'*Sy;
    ACLx=(Ax-Bx*Kgx)';
    Lx=inv(Rx)*Bx';
    ACLy=(Ay-By*Kgy)';
    Ly=inv(Ry)*By';
    assignin('base','Kgx',Kgx);
    assignin('base','Kgy',Kgy);

simopt=simset('solver','ode45','srcWorkspace','current
','dstWorkspace','current');
    y=sim('cobaGA',[],simopt);
    assignin('base','pX',posisiX.signals.values);
    assignin('base','eX',errorX.signals.values);
    assignin('base','ux',ux);
    assignin('base','pY',posisiY.signals.values);
    assignin('base','eY',errorY.signals.values);
    assignin('base','uy',uy);

    %Indeks Performansi X;

Jxx=(0.5*((errorX.signals.values')*(Qx)*(errorX.signal
s.values))+0.5*((ux')*(Rx)*(ux)));
    xxx=size(posisiX.signals.values);
    E2x=(posisiX.signals.values(:,1)-
posisiX.signals.values(:,2)).^2;
    RMSX=sqrt(sum(E2x)/xxx(1));

    %Indeks Performansi y;

Jyy=(0.5*((errorY.signals.values')*(Qy)*(errorY.signal
s.values))+0.5*((uy')*(Ry)*(uy)));
    yyy=size(posisiY.signals.values);
    E2y=(posisiY.signals.values(:,1)-
posisiY.signals.values(:,2)).^2;
    RMSY=sqrt(sum(E2y)/yyy(1));

    %nilai fitness
    u(i,1)=1/(Jxx*0.2+RMSX*0.3+Jyy*0.2+RMSY*0.3);

end
POP1=[POP u];
v=[];
% mengurutkan populasi berdasarkan nilai fitness
for w=1:Npop
    for x=1:Npop
        if POP1(w,5)>POP1(x,5)
            v=POP1(w,:);
```

```
                POP1(w,:)=POP1(x,:);
                POP1(x,:)=v;
            end;
        end;
    end;
    end
```

C7. Program Pembangkitan Populasi GA pada MATLAB

```
%Pembangkitan Populasi
function POP=generate3(Npop,Nbits);
for k=1:Npop
    POP(k,:)=randint(1,Nbits);  %pembangkitan nilai
random!
    POP(k,1)=(rand(1,1)+randint(1,1,[700,1000]));
%pembangkitan nilai random untuk Qx!
    POP(k,2)=(rand(1,1)+randint(1,1,[700,1000]));
%pembangkitan nilai random untuk Qy!
    POP(k,3)=(rand(1,1)); %pembangkitan nilai random
untuk Rx!
    POP(k,4)=(rand(1,1)); % pembangkitan nilai random
untuk Ry!
    while sum(POP(k,:))==0
        POP(k,:)=randint(1,Nbits,[0,1]);
    end;
end;
```

C8. Program seleksi GA pada MATLAB

```
%Mutasi Random
function [POP5]=mutasi(POP4,p_mutasi,Npop)
POP5=POP4;
jumlah=floor(Npop*4*p_mutasi); %mencari jumlah yang
akan dimutasi!
posisi=randint(1,jumlah,[1,Npop*4]); %mencari posisi
mana yang akan di mutasi!
for i=1:jumlah;
    if mod(posisi(1,i),4)==0;
        POP5(posisi(1,i)/4,4)=(rand(1,1));
    elseif mod(posisi(1,i),4)==3;
        POP5(ceil(posisi(1,i)/4),3)=(rand(1,1));
    elseif mod(posisi(1,i),4)==2;
        POP5(ceil(posisi(1,i)/4),2)=(rand(1,1)+randint(
        1,1,[700,1000]));
    else
        POP5(ceil(posisi(1,i)/4),1)=(rand(1,1)+randint(
        1,1,[700,1000]));
    end;
end;
```

C9. Program *Crossover* GA pada MATLAB

```
%Perkawinan Silang
function [POP4]=crossover(POP3,p_crossover)
i=0;
[a b]=size(POP3);
anak=zeros(a,b-1);
while i<(a-1)
%diiterasi selama jumlah iterasi lebih kecil daripada
jumlah populsi)
    i=i+1;
    cut_point=randint(1,1,[1,3]);
    if cut_point==1

anak(i,:)=[(POP3(i,1)*p_crossover)+(POP3(i+1,1)*(1-
p_crossover)) POP3(i,2:4)];
    elseif cut_point==2

anak(i,:)=[(POP3(i,1)*p_crossover)+(POP3(i+1,1)*(1-
p_crossover))  (POP3(i,2)*p_crossover)+(POP3(i+1,2)*(1-
p_crossover)) POP3(i,3:4)];
    else

anak(i,:)=[(POP3(i,1)*p_crossover)+(POP3(i+1,1)*(1-
p_crossover))  (POP3(i,2)*p_crossover)+(POP3(i+1,2)*(1-
p_crossover))  (POP3(i,3)*p_crossover)+(POP3(i+1,3)*(1-
p_crossover)) POP3(i,4)];
    end;
end;
for i=a
    cut_point=randint(1,1,[1,3]);
    if cut_point==1

anak(i,:)=[(POP3(i,1)*p_crossover)+(POP3(1,1)*(1-
p_crossover)) POP3(i,2:4)];
    elseif cut_point==2

anak(i,:)=[(POP3(i,1)*p_crossover)+(POP3(1,1)*(1-
p_crossover))    (POP3(i,2)*p_crossover)+(POP3(1,2)*(1-
p_crossover)) POP3(i,3:4)];
    else

anak(i,:)=[(POP3(i,1)*p_crossover)+(POP3(1,1)*(1-
p_crossover))    (POP3(i,2)*p_crossover)+(POP3(1,2)*(1-
p_crossover))    (POP3(i,3)*p_crossover)+(POP3(1,3)*(1-
p_crossover)) POP3(i,4)];
    end;
end;
POP4=anak;
end
```

C10. Program Mutasi GA pada MATLAB

```
%Mutasi Random
function [POP5]=mutasi(POP4,p_mutasi,Npop)
POP5=POP4;
jumlah=floor(Npop*4*p_mutasi); %mencari jumlah yang
akan dimutasi!
posisi=randint(1,jumlah,[1,Npop*4]); %mencari posisi
mana yang akan di mutasi!
for i=1:jumlah;
    if mod(posisi(1,i),4)==0;
        POP5(posisi(1,i)/4,4)=(rand(1,1));
    elseif mod(posisi(1,i),4)==3;
        POP5(ceil(posisi(1,i)/4),3)=(rand(1,1));
    elseif mod(posisi(1,i),4)==2;
        POP5(ceil(posisi(1,i)/4),2)=(rand(1,1)+randint(
        1,1,[700,1000]));
    else
        POP5(ceil(posisi(1,i)/4),1)=(rand(1,1)+randint(
        1,1,[700,1000]));
    end;
end;
```

RIWAYAT PENULIS

Farid Choirul Akbar yang biasa dipanggil Farid oleh teman-temannya lahir di Mojokerto, 29 Nopember 1993. Farid merupakan anak kedua dari pasangan Fakih dan Ninik Sutanti. Lulus dari SDN Miji IV Mojokerto pada tahun 2006, kemudian melanjutkan studi ke jenjang lebih lanjut di SMPN 1 Mojokerto dan lulus pada tahun 2009. Kemudian melanjutkan ke SMAN 1 Sooko Mojokerto dan lulus pada tahun 2012. Setelah menembuh studi pada tingkat SMA, penulis melanjutkan ketingkat lebih lanjut, yaitu di Institut Teknologi Sepuluh Nopember Surabaya jurusan Teknik Elektro pada tahun 2012. Pada bulan Januati 2015 penulis mengikuti seminar dan ujian Tugas Akhir sebagai salah satu syarat untuk memperoleh gelar Sarjana Teknik Elektro dari Institut Teknologi Sepuluh Nopember Surabaya.