TESIS-TE142599

KOMPENSASI KESALAHAN AKTUATOR QUADROTOR PADA *CONTROL ASSEMBLER* MENGGUNAKAN *MODIFIED LEAST SQUARE*

ANISA ULYA DARAJAT
2214202002

DOSEN PEMBIMBING
Dr. Trihastuti Agustinah, S.T., M.T.

PROGRAM MAGISTER
BIDANG KEAHLIAN SISTEM PENGATURAN
JURUSAN TEKNIK ELEKTRO
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA
2016

TESIS-TE142599

COMPENSATION OF QUADROTOR ACTUATOR FAULT BY USING MODIFIED LEAST SQUARE IN CONTROL ASSEMBLER

ANISA ULYA DARAJAT
2214202002

SUPERVISOR
Dr. Trihastuti Agustinah, S.T., M.T.

MAGISTER PROGRAM
CONTROL SYSTEM ENGINEERING
DEPARTMENT OF ELECTRICAL ENGINEERING
FACULTY OF INDUSTRIAL TECHNOLOGY
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA
2016

Tesis disusun untuk memenuhi salah satu syarat memperoleh gelar
Magister Teknik (M.T.)
di
Institut Teknologi Sepuluh Nopember

Oleh:
Anisa Ulya Darajat
NRP. 2214202002

Tanggal Ujian : 20 Juli 2016
Periode Wisuda : September 2016

Disetujui Oleh:

1. **Dr. Trihastuti Agustinah, S.T., M.T.** **(Pembimbing I)**
   NIP: 1968 0812 1994 0320 01

2. **Prof. Dr. Ir. Achmad Jazidie, M.Eng.** **(Penguji)**
   NIP: 1959 0219 1986 1010 01

3. **Dr. Ir. Mochammad Rameli** **(Penguji)**
   NIP: 1954 1227 1981 0310 02

4. **Ir. Rusdhianto Effendi AK, M.T.** **(Penguji)**
   NIP:1957 0424 1985 0210 01

Direktur Program Pascasarjana,

Prof. Ir. Djauhar Manfaat, M.Sc, Ph.D.
NIP. 19601202 198701 1 001

KOMPENSASI KESALAHAN AKTUATOR QUADROTOR PADA *CONTROL ASSEMBLER* MENGGUNAKAN *MODIFIED LEAST SQUARE*

Nama	: Anisa Ulya Darajat
NRP	: 2214202002
Dosen Pembimbing	: Dr. Trihastuti Agustinah, S.T., M.T.

ABSTRAK

Quadrotor sebagian besar membahas penerapan kontroler saat kondisi penerbangan normal. Untuk quadrotor saat terjadi kesalahan pada salah satu atau lebih aktuator dapat menyebabkan quadrotor tidak dapat mengikuti sinyal referensi dan menyebabkan *tracking error* semakin besar. Pada penelitian ini memperkenalkan strategi desain *fault tolerant control*. *Fault tolerant control* (FTC) adalah suatu metode kontrol yang dapat menoleransi kesalahan sehingga performa sistem tetap terjaga. Pada penelitian ini, metode *backstepping control* digunakan sebagai kontroler untuk gerak translasi dan rotasi, sedangkan untuk menjaga stabilisasi saat terjadi kesalahan pada aktuator (rotor) digunakan *modified least square* untuk mengestimasi kesalahan, kemudian hasil estimasi tersebut digunakan untuk mengkompensasi kesalahan pada aktuator. Hasil simulasi menunjukkan bahwa metode yang digunakan mampu membawa quadrotor mengikuti sinyal referensi dengan *tracking error* menjadi lebih kecil dan kontroler mampu mengatasi kesalahan hingga 65% dari total gaya yang dihasilkan oleh motor.

Kata Kunci : Quadrotor, *tracking error*, *fault tolerant control*, *backstepping control*, *modified least square*.

(Halaman ini sengaja dikosongkan)

COMPENSATION OF QUADROTOR ACTUATOR FAULT BY USING MODIFIED LEAST SQUARE IN CONTROL ASSEMBLER

By : Anisa Ulya Darajat

Student Identity Number : 2214202002

Supervisor : Dr. Trihastuti Agustinah, S.T., M.T.

ABSTRACT

Quadrotor mostly discuss the implementation of the current controller normal flight conditions. For quadrotor when an error occurs on one or more actuators may cause quadrotor can not follow the reference signal and cause tracking error increases. In this research introduces the design of fault tolerant control strategies. Fault tolerant control (FTC) is a control method that can tolerate errors so that system performance is maintained. In this research, backstepping control method is used as a controller for translational and rotational motion, while the stabilization when an error occurs in the actuator (rotor) used modified least square to estimate the error, then the results of these estimates are used to compensate for errors in the actuator. The simulation results showed that the method used is capable of carrying quadrotor follow the reference signal with the tracking error becomes smaller and the controller is able to cope with an error of up to 65% of the total force generated by the motor.

Keywords: Quadrotor, tracking error, fault tolerant control, backstepping control, modified least square.

(Halaman ini sengaja dikosongkan)

DAFTAR ISI

DAFTAR TABEL

(Halaman ini sengaja dikosongkan)

DAFTAR GAMBAR

BAB I
PENDAHULUAN

1.1 Latar Belakang

Unmanned aerial vehicle (UAV) atau wahana udara tak berawak merupakan wahana yang saat ini banyak dikembangkan dikarenakan memiliki potensi yang besar, baik untuk keperluan militer maupun sipil. Ada dua jenis wahana udara tak berawak, yaitu *fix wing* yang merupakan wahana yang terbang dengan menggunakan sayap seperti pesawat. Dan jenis lainnya adalah *rotary wing*, yang terbang hanya menggunakan motor dan baling-baling yaitu helikopter dan quadrotor. Pengembangan wahana udara tak berawak tidak hanya fokus pada fungsi pesawat, tetapi juga pada kontrol pesawat. Saat ini banyak dikembangkan metode kontrol otomatis wahana udara tak berawak. Pada penelitian ini akan dikembangkan metode kontrol otomatis pada wahana quadrotor.

Quadrotor adalah kendaraan yang mampu terbang secara vertikal dan landing meskipun pada ruang yang sempit. Quadrotor sebagaimana fungsinya, harus memiliki keseimbangan yang baik saat terbang, terutama pada gerak rotasi dan translasi yang sangat mempengaruhi terbang quadrotor. Quadrotor memiliki 4 buah motor yang dipasang simetris pada ujung-ujung kerangka utama. Motor depan dan belakang berputar searah jarum jam (*clockwise*), sedangkan motor kanan dan motor kiri berputar berlawanan arah jarum jam (*counter clockwise*) [1].

Beberapa tahun terakhir, berbagai macam metode kontrol telah diaplikasikan untuk pengendalian posisi dan orientasi quadrotor. Metode kontrol seperti PID banyak digunakan untuk pengendalian gerak quadrotor dengan pendekatan model linear, namun efek nonlinearitas dari quadrotor tidak dapat dikompensasi dengan baik oleh kontroler PID [2]. Sementara dengan menggunakan metode *Gain Scheduling LQR* juga tidak memberikan hasil yang lebih baik dari pada mengunakan kontroler PID. Kontroler tidak mampu mengikuti *trajectory* [3].

Ketidakmampuan metode kontrol linear dalam menangani permasalahan *stability* dan *height position control* pada quadrotor, dikarenakan model dinamika quadrotor yang *nonlinear*, *multiple-input multiple output*, serta adanya faktor

uncertainty sehingga mendorong untuk menggunakan metode kontrol *nonlinear* yang lebih efektif dan mampu mengatasi permasalahan ini. *Trajectory tracking* menggunakan metode *backstepping*, quadrotor mampu mengikuti *trajectory* namun terjadi pergeseran posisi quadrotor dari posisi referensi [1].

Pengendalian pada quadrotor memiliki tantangan sendiri agar quadrotor dapat terbang secara otomatis, karena quadrotor memiliki nonlinearitas yang tinggi, tingkat kestabilan yang rendah, *multiple input multiple output* (MIMO), rentan terhadap gangguan dari luar dan dapat terjadi kesalahan pada sistem [4]. Banyak penelitian yang telah dilakukan pada wahana quadrotor, diantaranya adalah penelitian tentang kontrol kestabilan, *tracking* posisi yang meliputi kontrol gerak translasi dan rotasi. Selain itu, terdapat penelitian tentang *fault tolerant control*.

Fault tolerant control dibagi menjadi dua jenis yaitu *passive fault tolerant control* dan *active fault tolerant control*. *Passive fault tolerant control* adalah sistem kontrol yang mampu mentoleransi kesalahan pada sistem yang telah diketahui dimana letak kesalahan tapi memiliki batas kesalahan yang bisa ditangani, sedangkan *active fault tolerant control* adalah sistem kontrol yang mampu mendiagnosis dan mendeteksi kesalahan secara *online* sehingga dapat meningkatkan stabilitas dan performa keseluruhan sistem saat terjadi kesalahan [5]. Dalam kasus aktif FTC, skema keseluruhan aktif FTC harus menyediakan estimasi kesalahan menggunakan beberapa pengukuran dan pengetahuan tentang model sistem. Selanjutnya, deteksi kesalahan dan isolasi (FDI) bisa ditambahkan untuk mengurangi jumlah kesalahan, memungkinkan peningkatan kinerja [10].

Terjadinya kesalahan pada komponen dalam sistem, salah satunya yaitu aktuator akan berdampak terjadinya penurunan performa dan stabilitas quadrotor tidak terjamin. Penelitian yang dilakukan untuk mengatasi hal tersebut adalah dengan menggunakan metode kontrol *nonlinear* seperti *sliding mode controller.* Metode *sliding mode controller* merupakan metode kontrol *nonlinear* yang dapat membawa *state* menuju permukaan luncur dan dipertahankan untuk selalu berada di tempat tersebut. Respon sistem menunjukkan kontroler mampu menjaga stabilitas dan mengikuti *trajectory* namun masih terdapat fenomena *chattering* [4].

Kontrol *backstepping* sering digunakan untuk pengendalian pada quadrotor. Namun, metode kontrol *backstepping* yang umum, tidak mampu menangani

kesalahan. Berdasarkan respon sistem dari hasil simulasi untuk kasus kesalahan pada aktuator didapatkan sistem mampu mengikuti *trajectory* namun masih terdapat deviasi [5]. Selanjutnya, penelitian mengenai metode estimasi yang banyak digunakan untuk parameter estimasi yaitu *least square*. Metode *least square* pada sistem *nonlinear* dilakukan agar mendapatkan parameter model sistem *nonlinear* menggunakan algoritma kuadrat. Dari hasil simulasi didapatkan estimasi parameter yang konsisten konvergen ke nilai rekursif algoritma kuadrat yang diusulkan [6], sedangkan pada [7] menggunakan *input-output feedback control* (IOFC) dan metode RLS, menunjukkan bahwa *error* asymtotik konvergen ke nol dan sistem drive stabil terhadap ketidakpastian parameter dan gangguan eksternal sehingga RLS baik digunakan untuk memprediksi prilaku sistem dan estimasi parameter.

Oleh karena itu, dari beberapa penelitian tersebut maka ide dari penelitian ini adalah merancang algoritma kontrol toleransi kesalahan berbasis estimasi dan kompensasi. Estimasi dan kompensasi kesalahan pada quadrotor menggunakan *modified least square* dengan estimasi parameter *gain* η yang digunakan untuk kompensasi pada kontrol *assembler* sehingga quadrotor mampu mengikuti sinyal referensi meskipun terjadi kesalahan aktuator (rotor) pada quadrotor.

1.2 Rumusan Masalah

Rumusan masalah dari penelitian ini adalah bagaimana merancang algoritma kontrol toleransi kesalahan berbasis estimasi dan kompensasi kesalahan aktuator (rotor) pada quadrotor yang menyebabkan quadrotor tidak dapat mengikuti sinyal referensi.

1.3 Tujuan

Tujuan dari penelitian ini adalah menghasilkan algoritma kontrol toleransi kesalahan yang dapat mengkompensasi kesalahan aktuator (rotor) agar quadrotor mampu mengikuti sinyal referensi meskipun terjadi kesalahan aktuator pada quadrotor.

1.4 Batasan Masalah

Batasan masalah dalam perancangan dan penelitian ini adalah sebagai berikut:

1. Tidak membahas masalah pengendalian orientasi dari quadrotor.
2. Tidak membahas pengujian pada kesalahan aditif.
3. Batas maksimum input motor yang digunakan adalah 0 - 0.1.
4. Parameter yang digunakan hanya parameter Qball X4.

1.5 Kontribusi

Kontribusi dari penelitian ini adalah menghasilkan algoritma kontrol toleransi kesalahan menggunakan *modified least square* agar dapat mengkompensasi kesalahan aktuator pada quadrotor yang berupa berkurangnya gaya angkat (*thrust*).

1.6 Metodology Penelitian

Pada penelitian ini dilakukan beberapa proses secara bertahap dan berurutan agar tercapai tujuan akhir dari penelitian.

1. Studi literatur

   Tahap pertama yang dilakukan dalam penelitian ini adalah melakukan studi literatur. Materi yang diperlukan meliputi konsep tentang dinamika dan kinematika quadrotor, pemrograman Matlab dan Simulink, kontrol *backstepping*, dan kontrol *least square*.

2. Pemodelan sistem

   Model matematika dari quadrotor didapat dari pemodelan sistem fisika analisis gaya yang terdapat pada quadrotor dan konstanta yang terdapat pada pemodelan diperoleh dari parameter sistem.

3. Desain sistem kontrol berdasarkan parameter-parameter pada quadrotor, akan dirancang sistem kontrol dengan menggunakan *modified least square* pada Matlab – Simulink.

4. Pengujian dan Analisis

   Pada tahap ini, akan dilakukan pengujian terhadap sistem kontrol hasil desain. Untuk beberapa kondisi pengujian, yaitu pada kondisi nominal (*fault free case*) dan kondisi dengan kesalahan (*faulty case*). Hasil pengujian tersebut kemudian dianalisis untuk diperoleh deskripsi terhadap performa sistem secara keseluruhan.

5. Kesimpulan

   Pada tahap ini, kesimpulan diperoleh sesuai dengan hasil pengujian dan analisis yang dilakukan.

6. Penulisan Laporan Tesis

   Penulisan laporan tesis dilakukan sebagai dokumentasi dari hasil penelitian yang dilakukan.

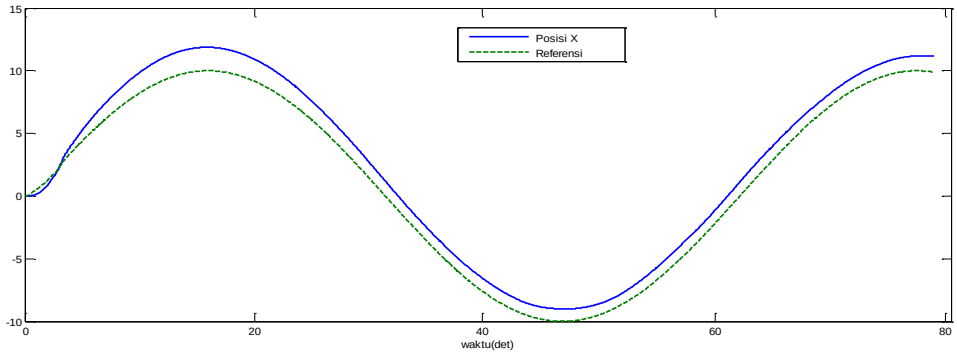
(Halaman ini sengaja dikosongkan)

BAB II
KAJIAN PUSTAKA DAN TEORI DASAR

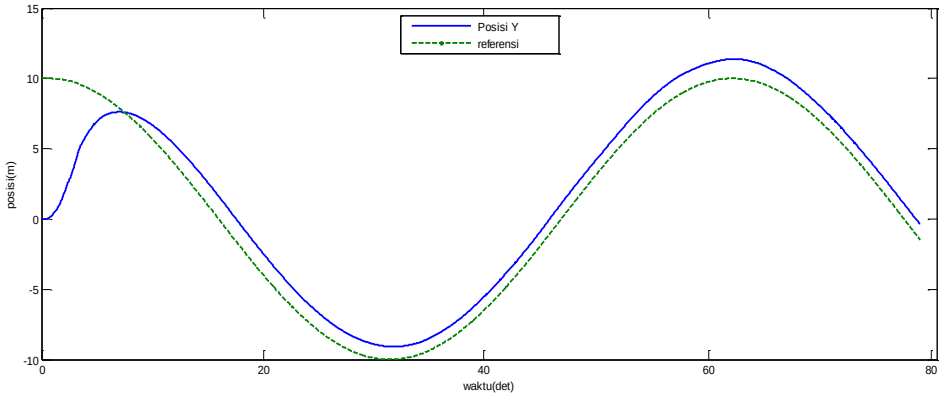
2.1 Kajian Penelitian Terkait

Beberapa persoalan kontrol pada quadrotor telah dibahas seperti gerak *hover*, kestabilan, *trajectory tracking*, dan *fault tolerant control* sebagaimana dipaparkan pada [1] – [11].

Trajectory tracking dengan menggunakan metode *backstepping control* pada quadrotor ditunjukkan pada Gambar 2.1 dan Gambar 2.2, hasil dari respon sistem didapatkan kontroler mampu mengikuti *trajectory* namun terjadi pergeseran posisi quadrotor dari posisi referensi dikarenakan quadrotor tidak mampu mengatasi gangguan pada *state* percepatan [1].

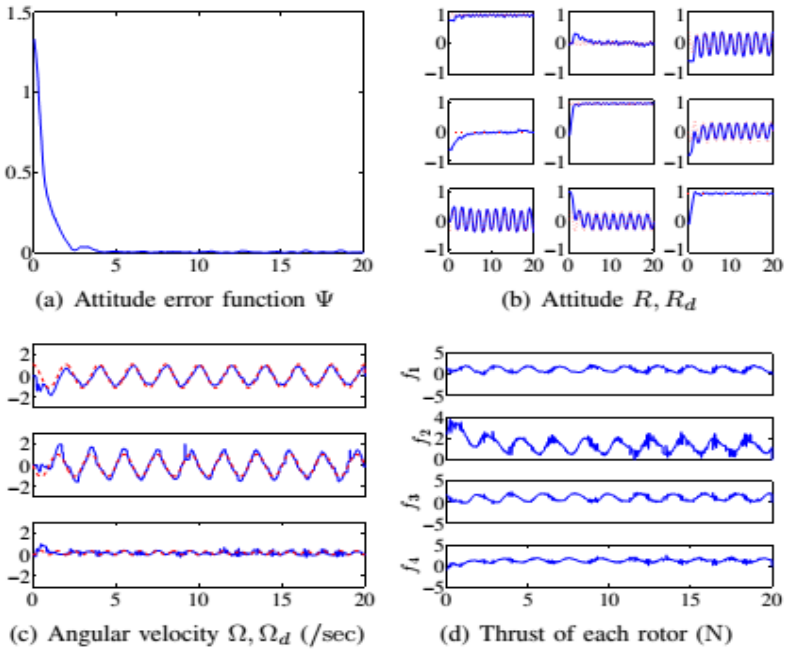


Gambar 2.1. Pergerakan quadrotor pada sumbu *X* dengan *backstepping* [1]



Gambar 2.2. Pergerakan quadrotor pada sumbu *Y* dengan *backstepping* [1]

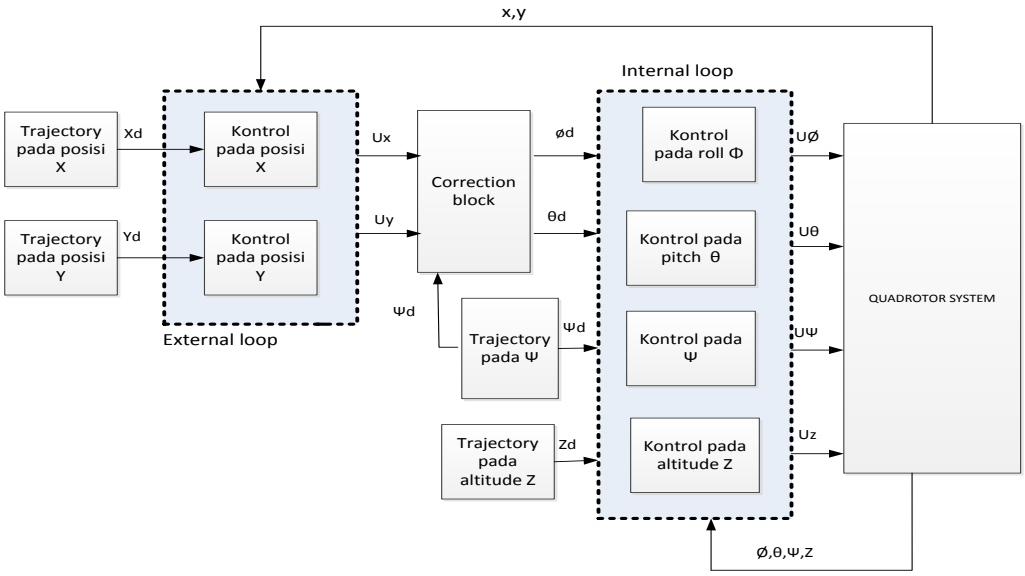
Penggunaan kontroler PID untuk *tracking attitude* dan *tracking* posisi pada quadrotor digunakan untuk menghindari hasil yang kacau dari keempat rotor karena adanya gangguan [2]. Berdasarkan respon sistem yang ditunjukkan pada Gambar 2.3, diketahui bahwa performa *tracking* ketinggian dengan kontroler PID menunjukkan performa yang baik dengan nilai *tracking error* yang kecil terhadap referensi yang diberikan namun respon sistem memiliki deviasi yang besar saat sistem pertama kali menuju titik yang ditentukan. Mekanisme pengendalian *gain* PID dilakukan secara *online* sehingga memerlukan waktu eksekusi yang lama. Sementara dengan menggunakan metode *gain scheduling LQR*, kontroler tidak mampu mengikuti *trajectory* [3], sehingga tidak memberikan hasil yang lebih baik dari pada mengunakan kontroler PID.



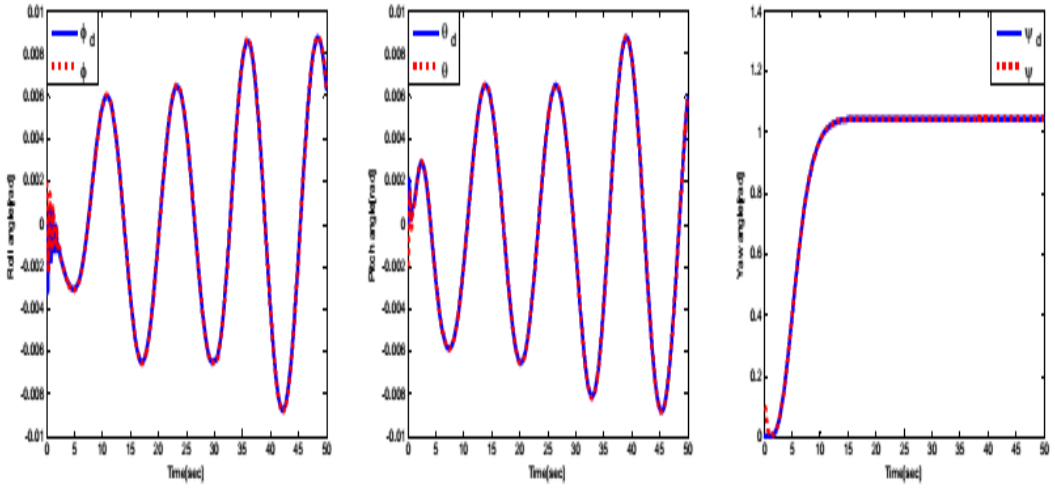
Gambar 2.3. *Tracking attitude* menggunakan kontroler PID [2]

Pada [4] mengajukan metode kontrol *sliding mode control* dengan pendekatan *backstepping* untuk mengatasi permasalan *tracking*. Berdasarkan Gambar 2.4, kontrol menggunakan pendekatan *backstepping*, algoritma rekursif digunakan untuk mensintesis sinyal kontrol sehingga memaksa sistem mengikuti lintasan yang diinginkan meskipun terjadi kesalahan aktuator dengan menyederhanakan semua tahap perhitungan mengenai kesalahan pelacakan dan fungsi Lyapunov, namun kesalahan yang terjadi pada aktuator tidak terfokus langsung pada

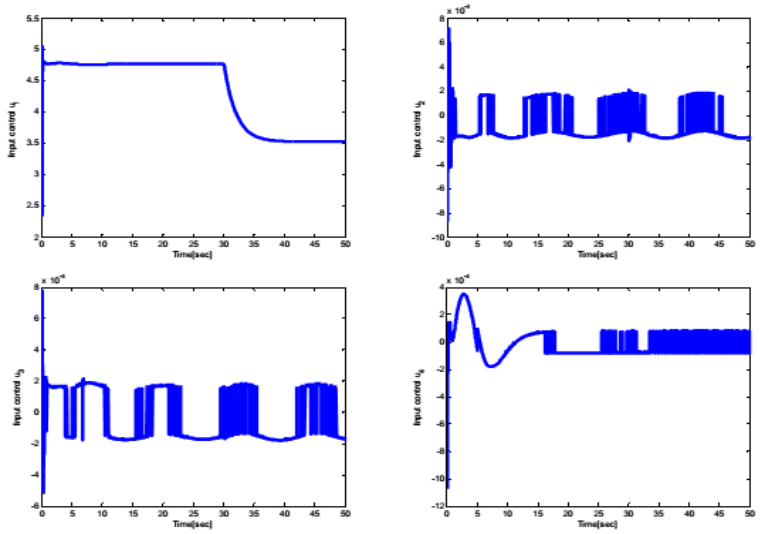
permasalahannya. Hasil respon sistem ditunjukkan pada Gambar 2.5, kontroler mampu menjaga stabilitas dan mengikuti *trajectory* meskipun terjadi kesalahan pada aktuator, tetapi masih terdapat fenomena *chattering*. Fenomena *chattering* terdapat pula pada kontrol input yang ditunjukkan pada Gambar 2.6, yang diakibatkan oleh kontrol *switching* yang akan mengakibatkan penggunaan sumber daya yang lebih besar.



Gambar 2.4. Blok diagram kesalahan pada aktuator quadrotor [4]

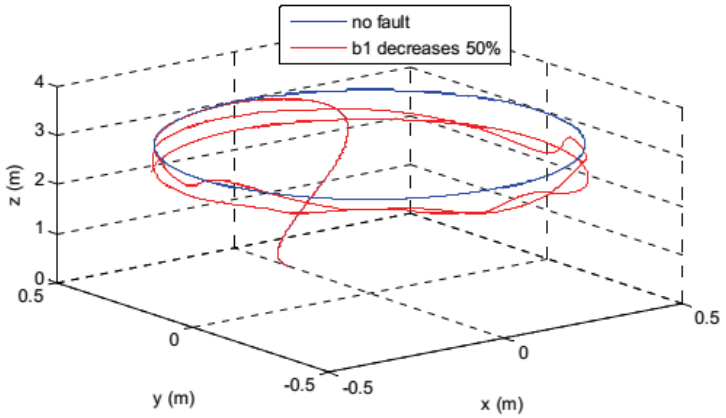


Gambar 2.5. *Tracking* pada *trajectory roll*, *pitch*, *yaw* [4]



Gambar 2.6. U_1, U_2, U_3, dan U_4 [4]

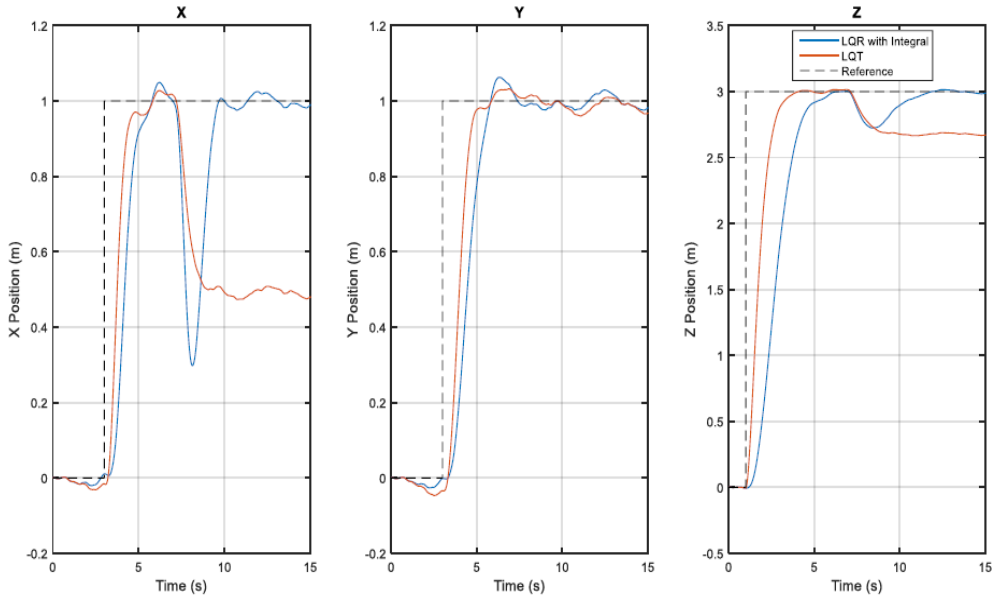
Sama halnya dengan penelitian [1] membahas mengenai metode *backstepping*, menggunakan strategi desain baru dengan kesalahan pada aktuator yang dapat ditoleransi. Dengan menggunakan metode *passive fault tolerant control* dihasilkan sistem kontrol yang mampu mengikuti *trajectory* namun masih terdapat deviasi seperti yang ditunjukkan pada Gambar 2.7. Hal ini dikarenakan *backstepping control* memiliki kemampuan mengatasi gangguan eksternal dan mengakomodasi kesalahan yang terbatas meskipun besarnya *gain* sinyal kontrol telah disesuaikan dengan besarnya kesalahan yang terjadi [5].



Gambar 2.7. Kondisi ketika terjadi kesalahan aktuator [5]

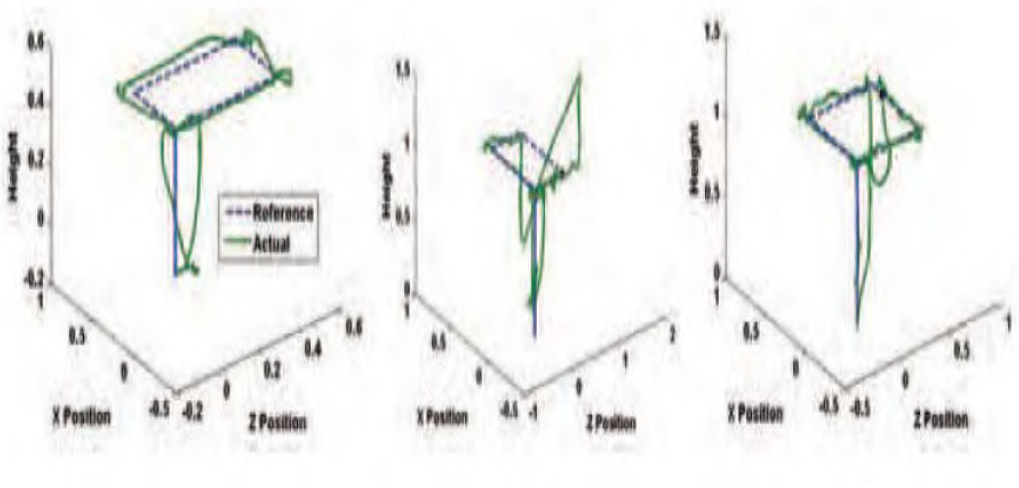
Model linear lebih disukai untuk digunakan dalam simulasi. Dua kontroler, LQT dan LQR dirancang untuk menstabilkan quadrotor. Setelah *tunning*

kontroler, respon *X*, *Y* dan *Z* dari quadrotor ditunjukkan seperti pada Gambar 2.8. Dapat dilihat dari Gambar 2.8, respon LQT lebih cepat dari LQR, tetapi dalam kasus 10% *loss throttle* saat 7 detik waktu simulasi, quadrotor *track* kontroler LQT pada respon X dan Z tidak dapat mengkompensasi kesalahan *steady-state*. Sebaliknya, LQR dengan integral mengkompensasi kesalahan *steady-state* dengan cepat [8]. *State* Y tidak terpengaruh oleh kesalahan karena model matematika yang digunakan dalam simulasi adalah linear, tidak ada *coupling* sehingga lebih baik untuk menggunakan LQT bukan LQR dengan integral dalam kondisi operasi normal karena respon yang cepat.



Gambar 2.8. Respon X, Y and Z pada LQT dan LQR dengan Integral dengan 10% kesalahan pada motor 1 [8]

Penelitian [9] menggunakan metode GS-PID terjadi 18% kesalahan *thrust*, quadrotor diminta untuk *tracking* satu meter pada lintasan persegi. Kasus *fault-free* ditampilkan di sebelah kiri plot Gambar 2.9 sedangkan gambar yang tengah dengan metode GS-PID menunjukkan penyimpangan dari *trajectory* yang diinginkan setelah terjadinya kesalahan 0,5 detik. Kinerja *tracking* yang lebih baik dapat dicapai dengan waktu *switching* pendek yang memerlukan deteksi kesalahan cepat dan benar. Plot kanan pada Gambar 2.9 menunjukkan kinerja yang lebih baik ketika *switching* dilakukan tanpa *time delay*.



Gambar 2.9. *Fault-free case* dan 18 % kesalahan *thrust* [9]

Untuk kasus aktif FTC, skema keseluruhan aktif FTC harus menyediakan estimasi kesalahan menggunakan beberapa pengukuran dan pengetahuan tentang model sistem. Selanjutnya, deteksi kesalahan dan isolasi (FDI) bisa ditambahkan untuk mengurangi jumlah kesalahan, memungkinkan peningkatan kinerja, seperti yang ditunjukkan oleh [10]. FTC yang efisien memperhitungkan informasi dari estimator kesalahan, menunjukkan bahwa FTC lebih kokoh dengan menggunakan aktif FTC [11].

Algoritma estimasi parameter yang biasa digunakan adalah *recursive least square* (RSL). Metode RSL digunakan untuk memprediksi perilaku sistem [7]. Pada [6] dengan menggunakan metode *input-output feedback control* (IOFC) dan metode *recursive least square* (RLS), kontroler *nonlinear* adaptif dirancang untuk *surface permanent magnet synchronous motor* (SPMSM) dan RLS estimator secara *online* mendeteksi parameter elektromekanis motor termasuk kerugian resistansi motor. Selain itu, *Sliding-Mode* (SM) observer secara *online* mendeteksi kecepatan rotor dan posisi rotor. Skema kontrol ini, sinyal referensi torsi diperoleh dengan PI konvensional. Hasil simulasi diperoleh, menunjukkan bahwa *error* asymtotik konvergen ke nol dan sistem drive stabil terhadap ketidakpastian parameter dan gangguan eksternal.

2.2 Teori Dasar

2.2.1 Pergerakan Dasar Qball X4 Quadrotor

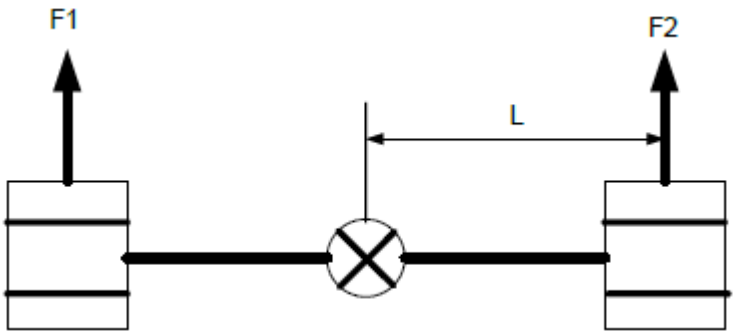
Quadrotor merupakan helikopter tak berawak dengan empat rotor yang saling dikombinasikan sesuai skema berbentuk silang yang simetris, setiap rotor terdiri dari baling-baling yang diterapkan pada motor DC. Rotor tersebut merupakan penggerak baling-baling yang digunakan untuk menghasilkan gaya angkat. Quadrotor dapat terbang dengan syarat gaya angkat pada quadrotor lebih besar daripada gaya gravitasi. Pada kondisi titik berat yang seimbang dan karakteristik motor yang sama, kondisi melayang (*hover*) tercapai saat semua motor memiliki kecepatan yang besarnya sama [12]. Secara garis besar, gerakan quadrotor terbagi menjadi empat pergerakan, yaitu *Throttle* (U_1), *Roll* (U_2), *Pitch* (U_3) dan *Yaw* (U_4).

Gerakan *Throttle* ($\boldsymbol{U_1}$) [N] yaitu pergerakan ini dilakukan dengan cara memberikan kecepatan motor 1, motor 2, motor 3 dan motor 4 dengan besaran yang sama pada kecepatan nominal. Ketika kecepatan motor 1, 2, 3, dan 4 melebihi kecepatan nominal, maka quadrotor akan bergerak naik, dan sebaliknya ketika kecepatan motor 1, 2, 3, dan 4 lebih kecil dari kecepatan nominal maka ketinggian quadrotor akan menurun. Untuk setiap sikap yang berubah, dapat mengubah kecepatan sudut motor tetapi total dorong (*thrust*) dari empat motor adalah konstan untuk mempertahankan ketinggian.

Gerakan *Roll* ($\boldsymbol{U_2}$) [Nm] yaitu pergerakan ini bergerak dengan acuan pada sumbu *Y*, dilakukan dengan cara menambahkan atau mengurangi kecepatan putar motor 2 pada quadrotor dan bersamaan dengan itu, menurunkan atau menaikkan kecepatan motor 4.

Gerakan *Pitch* ($\boldsymbol{U_3}$) [Nm] *yaitu* pergerakan ini bergerak dengan acuan pada sumbu *X*. Pergerakan *pitch* dilakukan dengan cara menambahkan atau mengurangi kecepatan putar motor 1 pada quadrotor dan bersamaan dengan itu, menurunkan atau menaikkan kecepatan motor 3. Sedangkan gerakan *Yaw* ($\boldsymbol{U_4}$) [Nm] yaitu berputar dengan acuan pada sumbu Z. Pergerakan ini dilakukan dengan cara meningkatkan atau menurunkan kecepatan putar motor 2 dan 4 pada quadrotor dan bersamaan dengan itu, menurunkan atau menaikkan kecepatan motor 1 dan 3.

Motor Qball-X4 tidak persis terletak di ujung batang aluminium, tapi 6 inci dari titik akhir agar tidak menyentuh *fiber carbon cage* baling-baling dan $\boldsymbol{l}$ adalah panjang antara sumbu rotasi masing-masing motor/rotor dari pusat gravitasi dari Qball-X4, seperti ditunjukkan pada Gambar 2.10.



Gambar 2.10. Model sumbu *roll*/*pitch* [9]

2.2.2 Pemodelan Quadrotor

Perhitungan untuk memperoleh pemodelan fisis yang tepat dapat dilakukan dengan memakai pendekatan mengenai kondisi fisik dari quadrotor. Hal ini bertujuan untuk menyederhanakan kompleksitas sistem yang dihasilkan dari dinamika quadrotor. Dalam pemodelan quadrotor terdapat beberapa asumsi yang harus dipahami [12]:

1. Percepatan gravitasi konstan dan tegak lurus terhadap permukaan bumi.
2. Desain quadrotor yang dibuat, dianggap simetris.
3. Struktur *body* dan baling-baling dari quadrotor merupakan benda kaku (*rigid)* sehingga pemodelan dapat menggunakan pendekatan Newton Euler.
4. Struktur *frame body* dari *yaw* quadrotor bersifat simetris sepanjang sumbu x dan y.
5. Qball quadrotor merupakan quadrotor yang digunakan dalam ruangan sehingga kecepatan dianggap rendah, gesekan udara diabaikan, efek gyroscopic dan torsi aerodinamis dapat diabaikan.

Tabel 2.1 Variabel pada Pergerakan Quadrotor [12]

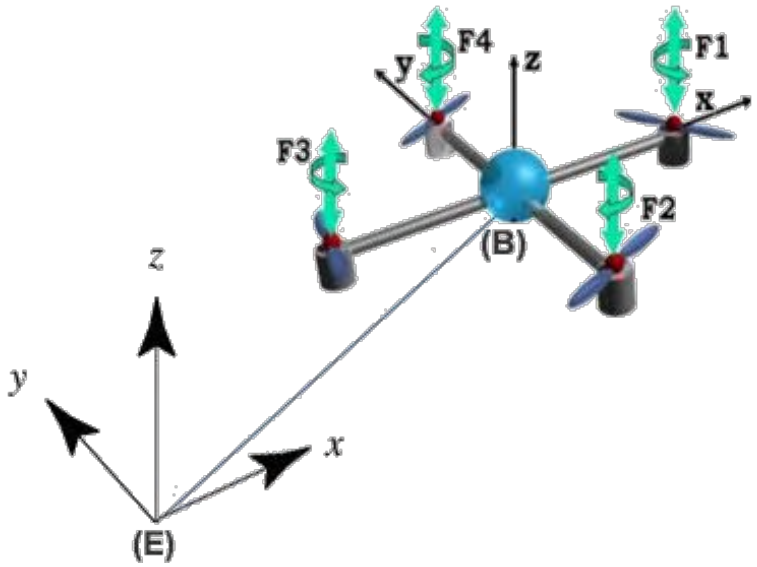
Variabel	Keterangan
x	Posisi *quadrotor* terhadap sumbu X_u
y	Posisi *quadrotor* terhadap sumbu Y_u
z	Posisi *quadrotor* terhadap sumbu Z_u
u	Kecepatan *quadrotor* yang diukur pada sumbu X_b
v	Kecepatan *quadrotor* yang diukur pada sumbu Y_b
w	Kecepatan *quadrotor* yang diukur pada sumbu Z_b
ϕ	Sudut *roll* terhadap sumbu X_u
θ	Sudut *pitch* terhadap sumbu Y_u
ψ	Sudut *yaw* terhadap sumbu Z_u
p	Kecepatan sudut *roll* yang diukur pada sumbu X_b
q	Kecepatan sudut *pitch* yang diukur pada sumbu Y_b
r	Kecepatan sudut *yaw* yang diukur pada sumbu Z_b
$\dot{u}$	Percepatan *quadrotor* yang diukur pada sumbu X_b
$\dot{v}$	Percepatan *quadrotor* yang diukur pada sumbu Y_b
$\dot{w}$	Percepatan *quadrotor* yang diukur pada sumbu Z_b
$\dot{p}$	Percepatan sudut *roll* yang diukur pada sumbu X_b
$\dot{q}$	Percepatan sudut *pitch* yang diukur pada sumbu Y_b
$\dot{r}$	Percepatan sudut *yaw* yang diukur pada sumbu Z_b

Dari tabel tersebut diketahui bahwa quadrotor memiliki 6 *Degree of Freedom* (DOF) yang menghasilkan 18 keluaran. Keluaran tersebut merupakan gerakan yang mempresentasikan sikap (*attitude*) dari quadrotor yaitu gerakan translasi pada sumbu *x, y, z* dan gerakan rotasi ϕ, θ, ψ yaitu *roll*, *pitch*, dan *yaw*. Sumbu tersebut merupakan sumbu yang biasa dipakai untuk menentukan gerakan quadrotor.

2.2.3 Sistem Koordinat Quadrotor

Sistem koordinat merupakan bagian dasar yang akan menjadi acuan setiap pergerakan quadrotor. Sistem koodinat pada quadrotor terdiri dari (X_E, Y_E, Z_E) sebagai *frame* bumi dan (X_B, Y_B, Z_B) sebagai *frame body*. *Frame* bumi merupakan *rigid* atau kaku sedangkan *frame body* bergerak baik rotasi maupun translasi. *Frame body* memiliki dua baling-baling yang berlawanan arah jarum jam sepanjang sumbu *X*, dan dua baling-baling searah jarum jam sepanjang sumbu *Y*

dan sumbu *Z* memiliki arah ke atas, seperti kaidah tangan kanan. Pada Gambar 2.11 ditunjukan arah koordinat *frame body* relatif terhadap *frame* bumi.



Gambar 2.11. Pemodelan *frame* bumi dan *frame body* [2]

2.2.4 Kinematika Quadrotor

Untuk dapat menganalisa kinematika maupun dinamika quadrotor, diperlukan pengetahuan mengenai matriks transformasi terlebih dahulu. Analisis digunakan menggunakan diagram cartesius tiga dimensi yaitu *X*, *Y* dan *Z*. Untuk mengubah vektor *state* dari *frame* bumi (E) ke *frame body* (B) digunakan matriks transformasi. Matriks ini berfungsi untuk konversi rotasi yang pertama terhadap sumbu *x*, kemudian terhadap sumbu *y*, dan yang terakhir terhadap sumbu *z*. Matriks tersebut dapat dicari menggunakan Persamaan (2.1).

$$R_{EB} = R_{x,\emptyset} * R_{y,\theta} * R_{z,\Psi} \quad (2.1)$$

dengan $R_{x,\phi}, R_{y,\theta}$ dan $R_{z,\Psi}$ merupakan matriks rotasi pada setiap sumbunya, seperti yang dijabarkan pada Persamaan (2.2), (2.3) dan (2.4)

- Rotasi terhadap sumbu *x*

$$R_{x,\emptyset} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\emptyset & -\sin\emptyset \\ 0 & \sin\emptyset & \cos\emptyset \end{bmatrix} \quad (2.2)$$

dengan sudut *roll* $\phi \in \left(-\frac{\pi}{2}, \frac{\pi}{2}\right)$

- Rotasi terhadap sumbu ***y***

$$R_{y,\theta} = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \quad (2.3)$$

dengan sudut *pitch* $\theta \in \left(-\frac{\pi}{2}, \frac{\pi}{2}\right)$

- Rotasi terhadap sumbu *z*

$$R_{z,\Psi} = \begin{bmatrix} \cos\Psi & -\sin\Psi & 0 \\ \sin\Psi & \cos\Psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (2.4)$$

dengan sudut *yaw* $\psi \in \left(-\frac{\pi}{2}, \frac{\pi}{2}\right)$

Jadi, matriks tranformasi dari *frame* bumi (E) ke *frame body* (B) adalah:

$$R_{EB} = R_{x,\emptyset} * R_{y,\theta} * R_{z,\Psi}$$

$$= \begin{bmatrix} \cos\theta\cos\Psi & -\cos\theta\sin\Psi & \sin\theta \\ \sin\emptyset\sin\theta\cos\Psi + \cos\emptyset\sin\Psi & -\sin\emptyset\sin\theta\sin\Psi + \cos\emptyset\cos\Psi & -\sin\emptyset\cos\theta \\ -\cos\emptyset\sin\theta\cos\Psi + \sin\emptyset\sin\Psi & \cos\emptyset\sin\theta\sin\Psi + \sin\emptyset\cos\Psi & \cos\emptyset\cos\theta \end{bmatrix} \quad (2.5)$$

Sedangkan matriks transformasi dari *frame body* (B) ke *frame bumi* (E) seperti pada Persamaan 2.6.

$$R_{BE} = R_{z,\Psi} * R_{y,\theta} * R_{x,\emptyset}$$

$$= \begin{bmatrix} \cos\theta\cos\Psi & -\cos\emptyset\sin\Psi + \sin\emptyset\sin\theta\cos\Psi & \sin\emptyset\sin\Psi + \cos\emptyset\sin\theta\cos\Psi \\ \cos\theta\sin\Psi & \sin\emptyset\sin\theta\sin\Psi + \cos\emptyset\cos\Psi & -\sin\emptyset\cos\Psi + \cos\emptyset\sin\theta\cos\Psi \\ -\sin\theta & \sin\emptyset\cos\theta & \cos\emptyset\cos\theta \end{bmatrix} \quad (2.6)$$

Gaya pada Quadrotor

Gaya pada quadrotor dihasilkan oleh keempat motor penggerak yang dipasang baling-baling. Sebelum menganalisa model dinamika dari quadrotor, akan dijelaskan mengenai gaya-gaya yang bekerja pada quadrotor. Gaya dan momen aerodinamis pada quadrotor bernilai kecil sehingga dapat diabaikan dalam proses pemodelan. Persamaan berikut menunjukkan gaya-gaya yang terjadi [13]:

$$U_1 = F_{T1} + F_{T2} + F_{T3} + F_{T4} \quad (2.7)$$

$$U_2 = l(F_{T4} - F_{T2}) \quad (2.8)$$

$$U_3 = l(F_{T3} - F_{T1}) \quad (2.9)$$

$$U_4 = (F_{T2} + F_{T4} - F_{T1} - F_{T3})d \quad (2.10)$$

dengan d adalah konstanta gaya drag dan F_{Ti} adalah gaya angkat (*thrust*) yang dihasilkan oleh tiap baling-baling motor yang didefinisikan sebagai berikut:

$$F_{Ti} = K\frac{\omega}{s+\omega}u_i \quad (2.11)$$

dengan K adalah konstanta gaya dorong, ω adalah lebar *bandwidth* motor dan u adalah sinyal kontrol dari *inner loop* kontroler ke motor. Untuk mendapatkan nilai konstanta gaya dorong K dilakukan dengan cara menerbangkan quadrotor pada posisi *hover* sehingga diketahui besar total gaya yang dihasilkan oleh keempat motor adalah sama besar dengan gaya gravitasi.

2.2.5 Model Dinamika Translasi Quadrotor

Model dinamika quadrotor direpresentasikan dengan beberapa asumsi untuk penyederhanaan model. Ketika quadrotor bergerak secara perlahan, efek dari momentum badan quadrotor pada gerakan translasi dapat diabaikan. *Frame* yang digunakan pada quadrotor dianggap *rigid* dan simetris.

Untuk mendapatkan dinamika gerak translasi pada quadrotor dengan menggunakan hukum Newton II, yaitu:

$$\sum F = m\ddot{v}$$

$$F_f + F_g = m\ddot{v} \quad (2.12)$$

dengan dan m adalah massa quadrotor, $v = [x \quad y \quad z]^T$ merupakan posisi dari pusat massa yang berada pada *frame* bumi quadrotor, resultan gaya yang terdapat pada masing-masing sumbu X, Y dan Z yang dihasilkan oleh keempat motor dinyatakan dengan $F_f = [0 \quad 0 \quad U_1]^T$ dan $F_g = [0 \quad 0 \quad -mg]^T$ merupakan gaya gravitasi.

Pemodelan sumbu translasi quadrotor terletak pada koordinat bumi, untuk transformasi dari *frame body* ke *frame* bumi maka perlu menggunakan matriks transformasi dari *frame body* ke *frame* bumi yang dinyatakan pada Persamaan (2.6):

$$\begin{bmatrix} F\ddot{X}_E \\ F\ddot{Y}_E \\ F\ddot{Z}_E \end{bmatrix} = \begin{bmatrix} \cos\theta\cos\Psi & -cos\emptyset\ sin\Psi + sin\emptyset\ sin\theta\ cos\Psi & sin\emptyset sin\Psi + cos\emptyset\ sin\theta\ cos\Psi \\ cos\emptyset\ sin\Psi & sin\emptyset\ sin\theta\ sin\Psi + cos\emptyset\ cos\Psi & -sin\emptyset cos\Psi + cos\emptyset\ sin\theta cos\Psi \\ -\sin\theta & sin\ \emptyset\ cos\ \theta & cos\emptyset\ cos\theta \end{bmatrix} \begin{bmatrix} Fx_B \\ Fy_B \\ Fz_B \end{bmatrix}$$

(2.13)

Dikarenakan gaya pada quadrotor hanya terjadi pada sumbu z_b (pada kondisi *hover*), sehingga gaya pada masing-masing sumbu translasi dapat dinyatakan dengan:

$$F_{X_E} = U_1(sin\ \emptyset\ sin\ \Psi + cos\emptyset\ sin\ \theta\ cos\ \Psi)$$

$$F_{Y_E} = U_1(-sin\ \emptyset\ cos\ \Psi + cos\emptyset\ sin\ \theta\ sin\ \Psi) \quad (2.14)$$

$$F_{Z_E} = U_1(cos\emptyset\ cos\ \theta\ )$$

Persamaan dinamika gerak translasi dapat diperoleh dengan mensubstitusikan Persamaan (2.14) pada Persamaan (2.12) didapatkan

$$\begin{bmatrix} U_1 sin\ \emptyset\ sin\ \Psi + cos\emptyset\ sin\ \theta\ cos\ \Psi \\ U_1 - sin\ \emptyset\ cos\ \Psi + cos\emptyset\ sin\ \theta\ sin\ \Psi \\ U_1 cos\emptyset\ cos\ \theta \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} = m \begin{bmatrix} \ddot{X} \\ \ddot{Y} \\ \ddot{Z} \end{bmatrix} \quad (2.15)$$

Dengan mengacu pada Persamaan (2.14) dimana $F_f = U_1$ maka didapatkan persamaan dinamika gerak translasi yang dinyatakan pada Persamaan (2.16), (2.17) dan (2.18).

$$\ddot{X} = \frac{U_1}{m}(\ sin\ \emptyset\ sin\ \Psi + \cos\emptyset \sin\theta\ \cos\Psi\ ) \quad (2.16)$$

$$\ddot{Y} = \frac{U_1}{m}(-\ sin\ \emptyset\ cos\ \Psi + \cos\emptyset \sin\theta\ \sin\Psi\ ) \quad (2.17)$$

$$\ddot{Z} = \frac{U_1}{m}\ cos\ \emptyset \cos\theta - g \quad (2.18)$$

2.2.6 Model Dinamika Rotasi Quadrotor

Untuk mendapatkan dinamika gerak rotasi, dengan menggunakan persamaan Newton Euler, maka akan didapatkan persamaan berikut ini:

$$\tau_f = \Omega J\ x\ \Omega + J\dot{\Omega}$$

$$J\dot{\Omega} = -\Omega J\ x\ \Omega + \tau_f \quad (2.19)$$

dimana $J \in R^{3x3}$ adalah matriks inersia pada *frame body* quadrotor yang didefinisikan sebagai:

$$J = \begin{bmatrix} J_{xx} & 0 & 0 \\ 0 & J_{yy} & 0 \\ 0 & 0 & J_{zz} \end{bmatrix} \quad (2.20)$$

dan Ω merupakan kecepatan sudut quadrotor yang didefinisikan dengan:

$$\Omega = \begin{bmatrix} \Omega_x \\ \Omega_y \\ \Omega_z \end{bmatrix} = \begin{bmatrix} \dot{\emptyset} \\ \dot{\theta} \\ \dot{\Psi} \end{bmatrix} \quad (2.21)$$

Nilai $-\Omega J\ x\ \Omega$ dapat dihitung dengan substitusi persamaan (2.20) dan (2.21) seperti yang dinyatakan pada persamaan berikut ini:

$$-\Omega \mathrm{x} J\,\Omega = -\begin{bmatrix} \dot{\emptyset} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} x \begin{bmatrix} J_{xx} & 0 & 0 \\ 0 & J_{yy} & 0 \\ 0 & 0 & J_{zz} \end{bmatrix} \begin{bmatrix} \dot{\emptyset} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \quad (2.22)$$

$$-\Omega \mathrm{x} J\,\Omega = -\begin{bmatrix} \dot{\emptyset} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} x \begin{bmatrix} J_{xx}\dot{\emptyset} \\ J_{yy}\dot{\theta} \\ J_{zz}\dot{\psi} \end{bmatrix} \quad (2.23)$$

Dengan menggunakan perkalian *cross*, maka vektor dari Persamaan 2.23 menjadi:

$$= -\left(J_{zz}\,\dot{\theta}\dot{\psi} - J_{yy}\,\dot{\theta}\dot{\psi}\right)i - \left(J_{xx}\,\dot{\phi}\dot{\psi} - J_{zz}\,\dot{\phi}\dot{\psi}\right)j - \left(J_{yy}\,\dot{\theta}\dot{\phi} - J_{xx}\,\dot{\theta}\dot{\phi}\right)k$$

$$= \left(J_{yy}\,\dot{\theta}\dot{\psi} - J_{zz}\,\dot{\theta}\dot{\psi}\right)i + \left(J_{zz}\,\dot{\phi}\dot{\psi} - J_{xx}\,\dot{\phi}\dot{\psi}\right)j + \left(J_{xx}\,\dot{\theta}\dot{\phi} - J_{yy}\,\dot{\theta}\dot{\phi}\right)k$$

$$= \dot{\theta}\dot{\psi}\,(J_{yy} - J_{zz})i + \dot{\phi}\dot{\psi}\,(J_{zz} - J_{xx})j + \dot{\theta}\dot{\phi}\,(J_{xx} - J_{yy})k \quad (2.24)$$

Atau dapat ditulis menjadi

$$-\Omega \mathrm{x} J\,\Omega = \begin{bmatrix} \dot{\theta}\dot{\psi}\,(J_{yy} - J_{zz}) \\ \dot{\phi}\dot{\psi}\,(J_{zz} - J_{xx}) \\ \dot{\theta}\dot{\phi}\,(J_{xx} - J_{yy}) \end{bmatrix} \quad (2.25)$$

Sedangkan τ_f pada Persamaan 2.19 merupakan resultan momen gaya dorong yang dihasilkan oleh masing-masing rotor. Nilai momen pada sumbu *x*, *y* dan *z* dapat dinyatakan dalam Persamaan 2.26.

$$\begin{aligned} \tau_{fx} &= l(F_{T4} - F_{T2)} \\ \tau_{fy} &= l(F_{T3} - F_{T1)} \\ \tau_{fz} &= (F_{T2} + F_{T4} - F_{T1} - F_{T3})d \end{aligned} \quad (2.26)$$

dimana l adalah jarak antara pusat massa quadrotor dengan sumbu rotasi propeller

dan *d* adalah koefisien gaya *drag*.

Dinamika gerak rotasi quadrotor diperoleh dengan substitusi nilai $-\Omega J\ x\ \Omega$ dan momen gaya τ_f ke persamaan hukum Newton Euler yang dinyatakan dalam Persamaan (2.27).

$$J\dot{\Omega} = -\Omega \text{ x} \text{J}\,\Omega + \tau_f$$

$$\begin{bmatrix} J_{xx}\ddot{\phi} \\ J_{yy}\ddot{\theta} \\ J_{zz}\ddot{\psi} \end{bmatrix} = \begin{bmatrix} \dot{\theta}\dot{\psi}\,(J_{yy} - J_{zz}) \\ \dot{\phi}\dot{\psi}\,(J_{zz} - J_{xx}) \\ \dot{\theta}\dot{\phi}\,(J_{xx} - J_{yy}) \end{bmatrix} + \begin{bmatrix} l(F_{T4} - F_{T2)} \\ l(F_{T3} - F_{T1)} \\ (F_{T2} + F_{T4} - F_{T1} - F_{T3})d \end{bmatrix} \quad (2.27)$$

Maka diperoleh dinamika rotasi quadrotor yang dinyatakan pada persamaan (2.28).

$$\ddot{\emptyset} = \dot{p} = \frac{U_2}{J_{xx}} + \frac{qr}{J_{xx}}(J_{yy} - J_{zz})$$

$$\ddot{\theta} = \dot{q} = \frac{U_3}{J_{yy}} + \frac{pr}{J_{yy}}(J_{zz} - J_{xx}) \quad (2.28)$$

$$\ddot{\psi} = \dot{r} = \frac{U_4}{J_{zz}} + \frac{pq}{J_{zz}}\left(J_{xx} - J_{yy}\right)$$

Nilai inersia J_{xx} merupakan inersia quadrotor terhadap sumbu x, J_{yy} merupakan inersia quadrotor terhadap sumbu y, dan J_{zz} merupakan inersia quadrotor terhadap sumbu z , dengan $p = \dot{\emptyset}$, $q = \dot{\theta}$ dan $r = \dot{\psi}$. Mengambil persamaan gerak translasi pada Persamaan (2.16)-(2.18) dan gerak rotasi pada persamaan (2.28), maka model dinamika keseluruhan quadrotor dinyatakan sebagai berikut:

$$\ddot{X} = \frac{U_1}{m}(\ sin\,\emptyset\ sin\,\Psi + \cos\emptyset \sin\theta\ \cos\Psi\ )$$

$$\ddot{Y} = \frac{U_1}{m}(-\ sin\,\emptyset\ cos\ \Psi + \cos\emptyset \sin\theta\ \sin\Psi\ )$$

$$\ddot{Z} = \frac{U_1}{m}\ cos\,\emptyset \cos\theta - g \quad (2.29)$$

$$\ddot{\emptyset} = \dot{p} = \frac{U_2}{J_{xx}} + \frac{qr}{J_{xx}}(J_{yy} - J_{zz})$$

$$\ddot{\theta} = \dot{q} = \frac{U_3}{J_{yy}} + \frac{pr}{J_{yy}}(J_{zz} - J_{xx})$$

$$\ddot{\Psi} = \dot{r} = \frac{U_4}{J_{zz}} + \frac{pq}{J_{zz}}\left(J_{xx} - J_{yy}\right)$$

Persamaan 2.29 dapat dituliskan kedalam persamaan sistem nonlinear orde 2 berikut ini.

$$\ddot{x} = f(x, u) \tag{2.30}$$

Dengan memperkenalkan *state vector* $x = [\, x_1\ x_2 \ldots x_{12}]^T \in R^{12}$ dan jika dipilih

$$\begin{aligned} x_1 &= X & x_7 &= \emptyset \\ x_2 &= \dot{X} & x_8 &= \dot{\emptyset} = p \\ x_3 &= Y & x_9 &= \theta \\ x_4 &= \dot{Y} & x_{10} &= \dot{\theta} = q \\ x_5 &= Z & x_{11} &= \Psi \\ x_6 &= \dot{Z} & x_{12} &= \dot{\Psi} = r \end{aligned} \tag{2.31}$$

Maka Persamaan (2.30) dapat ditulis menjadi:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \\ \dot{x}_5 \\ \dot{x}_6 \\ \dot{x}_7 \\ \dot{x}_8 \\ \dot{x}_9 \\ \dot{x}_{10} \\ \dot{x}_{11} \\ \dot{x}_{12} \end{bmatrix} = \begin{bmatrix} x_2 \\ \frac{U_1}{m} u_x \\ x_4 \\ \frac{U_1}{m} u_y \\ x_6 \\ \frac{U_1}{m}\ cos\ x_7 \cos x_9 - g \\ x_8 \\ \frac{U_2}{J_{xx}} - \frac{x_{10}x_{12}}{J_{xx}}\ (J_{xx} - J_{yy}) \\ x_{10} \\ \frac{U_3}{J_{yy}} - \frac{x_8 x_{12}}{J_{yy}}\ (J_{xx} - J_{zz}) \\ x_{12} \\ \frac{U_4}{J_{zz}} - \frac{x_8 x_{10}}{j_{zz}}\ (J_{yy} - J_{zz}) \end{bmatrix} \tag{2.32}$$

dengan:

$$u_x = (\sin x_7 \sin x_{11} + \cos x_7 \sin x_9\ \cos x_{11}) \tag{2.33}$$

$$u_y = (-\ sin\ x_7\ cos\ x_{11} + \cos x_7\ \sin x_9\ \sin x_{11}) \tag{2.34}$$

$$U_1 = F_{T1} + F_{T2} + F_{T3} + F_{T4} \tag{2.35}$$

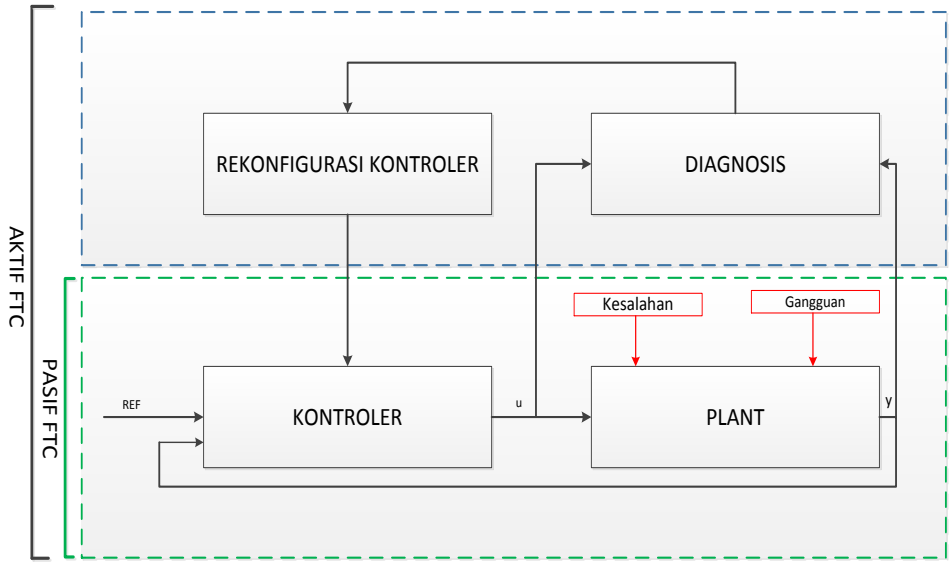
$$U_2 = l(F_{T4} - F_{T2}) \tag{2.36}$$

$$U_3 = l(F_{T3} - F_{T1}) \tag{2.37}$$

$$U_4 = (F_{T2} + F_{T4} - F_{T1} - F_{T3})d \tag{2.38}$$

2.2.7 *Fault Tolerant Control* (FTC)

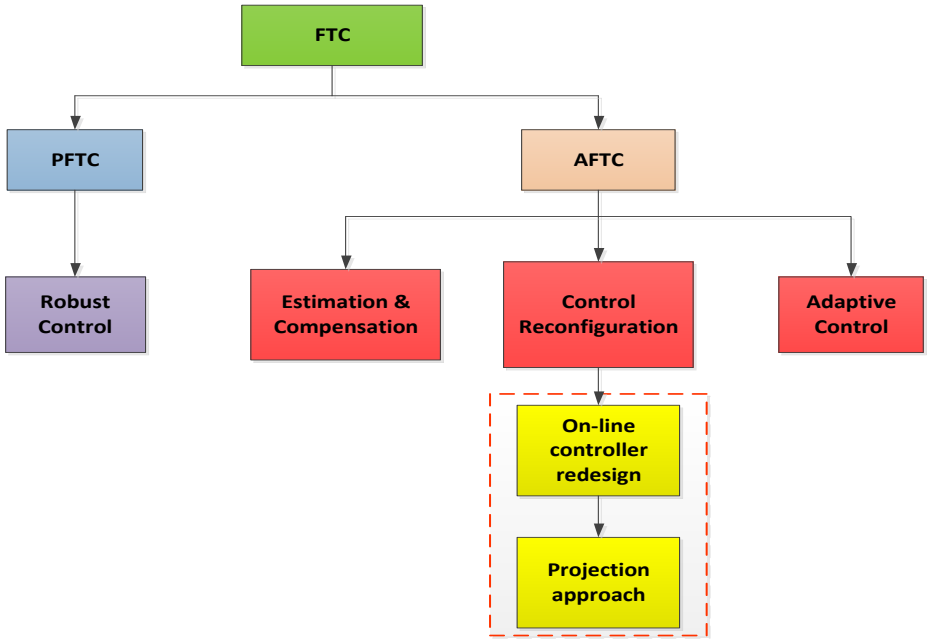
Suatu metode kontrol yang dapat mempertahankan kinerja sistem sesuai dengan yang diinginkan meskipun terjadi kesalahan [14]. FTC dapat mengkompensasi dampak penurunan performa yang diakibatkan adanya kesalahan. Pada umumnya kontroler nominal dapat memberikan kestabilan dan performa *closed-loop* yang diinginkan pada kondisi operasi nominal (tanpa adanya kesalahan). Akan tetapi kondisi tidak dapat berjalan seperti itu disaat terjadi kesalahan pada sensor, aktuator, atau pada sistemnya sendiri. Agar kontroler mempunyai kemampuan dalam menoleransi kesalahan tersebut, maka dibutuhkan beberapa blok tambahan dalam struktur sistem *closed-loop* sebagai contoh seperti pada Gambar 2.12.



Gambar 2.12. Skema *Fault Tolerant Control* (FTC) secara umum [14]

Fault tolerant control (FTC) dibedakan dalam dua pendekatan yaitu *active* FTC (AFTC) dan *passive* FTC (PFTC). Pada pendekatan secara PFTC, kontroler didesain secara tetap dan menjadi *robust* terhadap perkiraan-perkiraan kesalahan. Jika dilihat secara sekilas tidak ada perbedaan dengan sistem kontrol nominal dalam hal skema blok diagramnya. Hal ini dikarenakan pada pendekatan ini tidak diperlukan skema *fault detection* dan diagnosis (FDD) ataupun rekonfigurasi kontroler. Berbeda dengan PFTC, pada pendekatan AFTC memberikan aksi

kontrol secara aktif terhadap kesalahan yang terjadi pada sistem dengan cara rekonfigurasi aksi control sehingga terdapat perubahan struktur sistem kontrol. Hasilnya sistem menjadi lebih stabil dan performa dapat diterima secara penuh. Penerapan kedua tipe FTC ini biasanya tergantung tingkat keparahan kesalahan yang terjadi. Klasifikasi umum dari metode FTC pada Gambar 2.13.



Gambar 2.13. Klasifikasi Metode FTC [14]

a. Estimasi dan Kompensasi

Secara umum, FTC membutuhkan aktivitas kontroler nominal dan unit FDD [15]. Rekonfigurasi kontroler atau kompensasi kesalahan dilakukan berdasarkan evaluasi dan parameter identifikasi oleh unit estimator. Namun, tantangan utama metodologi FTC ini terletak di persyaratan untuk unit FDD yang robust dan cepat. Sebagai konsep FDD ini estimasi dan kompensasi FTC didasarkan pada perhitungan estimasi kesalahan dan mekanisme untuk mengkompensasi efek kesalahan ini dengan penambahan kompensasi baru sinyal kontrol ke nominal.

Metode estimasi dan kompensasi pertama kali diusulkan pada [16] untuk mengkompensasi efek aktuator aditif atau multiplikatif pada kesalahan sistem. Dalam [17] estimasi dan kompensasi telah digunakan untuk mengkompensasi

efek kesalahan aktuator pada *winding machine*. Kinerja sistem pasca-kesalahan menggunakan pendekatan ini sangat dipengaruhi oleh estimasi dari sinyal kesalahan.

b. Kesalahan Aktuator

Kesalahan yang terjadi pada aktuator akan mengakibatkan tidak sesuainya sinyal kontrol yang menjadi masukan dari plant. Oleh karena itu, sinyal kontrol yang terkena kesalahan ini akan mengakibatkan ketidakstabilan pada plant tersebut. Terdapat beberapa kondisi kesalahan aktuator yang memberi dampak pada performa sistem. Sinyal kontrol yang terkena kesalahan dapat dimodelkan sebagai berikut [18]:

$$U_f = \Gamma U + U_{f\theta} \tag{2.39}$$

dengan U adalah sinyal kontrol keluaran kontroler, U_f adalah sinyal kontrol yang sudah terkena *actuator fault* dan menjadi masukan dari plant, Γ adalah sinyal yang memberi pengaruh *mutiplicative actuator fault*, dan $U_{f\theta}$ sinyal yang memberikan pengaruh *additive actuator fault*. Dengan penjelasan tersebut, maka kesalahan aktuator dapat diklasifikasikan menjadi beberapa tipe berdasarkan pada dampak yang dihasilkan pada sistem seperti pada Tabel 2.2.

Tabel 2.2. Jenis-jenis Kesalahan Aktuator

Kondisi	$\boldsymbol{U_{f\theta} = 0}$	$\boldsymbol{U_{f\theta} \neq 0}$
$\boldsymbol{\Gamma = 1}$	*Fault free case*	*Bias*
$\boldsymbol{0 < \Gamma < 1}$	*Loss of effectiveness*	*Loss of effectiveness*
$\boldsymbol{\Gamma = 0}$	*Out of order*	*Actuator blocked*

Kondisi nominal (*fault free case*) adalah kondisi saat tidak terjadi kesalahan pada aktuator, yang artinya sistem bekerja sesuai kontrol nominal. Kondisi *bias* adalah kondisi saat hanya terdapat kesalahan aktuator yang bersifat *additive*. Hal ini mengakibatkan sinyal kontrol yang menjadi masukan dari plant mempunyai nilai menyimpang dari sinyal kontrol yang dihasilkan oleh kontroler. Pada kondisi *loss of effectiveness*, aktuator mengalami kesalahan yang sifatnya *multiplicative*

terhadap sistem. Oleh karena nilai pengalinya adalah antara 0 sampai 1, maka besarnya sinyal kontrol yang masuk ke plant akan menjadi lebih kecil sesuai dengan skala tertentu jika dibandingkan dengan sinyal kontrol yang dihasilkan oleh kontroler. Atau dengan kata lain, aktuator tersebut telah kehilangan keefektifannya.

Aktuator pada kondisi *out of order* akan mengakibatkan sinyal kontrol yang masuk ke plant selalu bernilai 0. Akibatnya, berapapun besarnya sinyal kontrol yang dihasilkan oleh kontroler tidak akan mempengaruhi sistem. Sistem pada kondisi ini tidak akan bekerja lagi. Sedangkan kondisi *actuator blocked* adalah kondisi saat kontroler tidak mampu mengendalikan plant dari sistem, karena sinyal kontrol yang masuk ke plant murni dari sinyal kesalahan bukan dari sinyal yang dihasilkan oleh kontroler. Oleh karena itu, saat terjadi kondisi seperti ini maka aktuator harus diganti agar tidak terjadi kesalahan pada sistem.

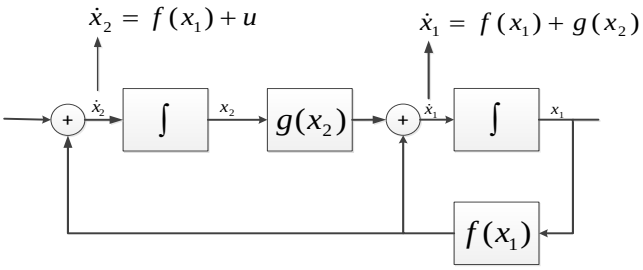
2.2.8 *Backstepping Control*

Pertimbangkan sebuah sistem nonlinear yang dinyatakan pada Persamaan (2.40) dan (2.41):

$$\dot{x}_1 = x_2 \tag{2.40}$$

$$\dot{x}_2 = f(x_1) + g(x)u \tag{2.41}$$

dengan $x = [x_1 \quad x_2]^T$ merupakan *state*, u adalah input kontrol, f dan g merupakan *smooth function* $(x_1 = 0 \rightarrow f(0) = 0)$. Diagram blok sistem ditunjukkan pada Gambar 2.14.



Gambar 2.14. Diagram Blok Sistem pada Persamaan (2.40-2.41) [19]

Sebuah *feedback control law* didesain untuk menstabilkan *origin* ($x_1 = 0$ dan $x_2 = 0$) melalui prosedur desain berikut ini.

Langkah pertama adalah dengan mendefinisikan *state* sebagai

$$x_1 = x$$

$$x_2 = \dot{x}_1 = \dot{x}$$

$$z_1 = e = x_r - x_1 \tag{2.42}$$

Kemudian untuk melakukan *backstepping* maka definisikan variabel yang disebut variabel kontrol virtual. Dengan x_2 adalah variabel kontrol virtual dan $a(z_1)$ adalah fungsi yang membawa $z_1 \to 0$ kemudian definisikan z_2 sebagai pada persamaan berikut ini

$$z_2 = a(z_1) - x_2 \tag{2.43}$$

dengan: $a(z_1) = \dot{x}_r + k_1 z_1$

Kemudian diferensialkan z_1, diperoleh Persamaan 2.44.

$$\dot{z}_1 = \dot{x}_r - \dot{x}_1 = \dot{x}_r - x_2 = \dot{x}_r + z_2 - a(z_1) \tag{2.44}$$

Dengan mendefinisikan $a(z_1) = \dot{x}_r + k_1 z_1$ dimana $k_1 > 0$ dan mendefinisikan fungsi Lyapunov sebagai

$$V_1 = \frac{1}{2} z_1^2 \tag{2.45}$$

Kemudian turunkan V_1 terhadap waktu, maka diperoleh

$$\dot{V}_1 = z_1 \dot{z}_1 = z_1\big(\dot{x}_r + z_2 - a(z_1)\big) = -k_1 z_1^2 + z_1 z_2 \tag{2.46}$$

Dari persamaan 2.46 dapat dilihat jika $z_2 = 0$ maka $\dot{V}_1 = -k_1 z_1^2$ dan z_1 akan menuju 0 secara asymtotis. Kemudian definisikan fungsi Lyapunov untuk membuat $z_2 \to 0$

Untuk melakukan *backstepping* digunakan variabel peubah sebagai berikut:

$$V_2 = V_1 + \frac{1}{2} z_2^2 \tag{2.47}$$

Turunkan terhadap fungsi waktu didapatkan

$$\dot{V}_2 = \dot{V}_1 + z_2 \dot{z}_2 = -k_1 z_1^2 + z_2(\ddot{x}_r + z_1 + k_1 \dot{z}_1 - \dot{x}_2) \tag{2.48}$$

Untuk membuat $\dot{V}_2 < 0$ maka dipilih fungsi

$$\dot{x}_2 = \ddot{x}_r + (1 - k_1^2)z_1 + (k_1 + k_2)z_2 \quad (2.49)$$

Sinyal kontrol *backsteping* untuk plant 2.41 yaitu sebagai berikut:

$$\dot{x}_2 = f(x_1) + g(x)u$$
$$u = g(x)^{-1}\left(\dot{x}_2 + f(x_1)\right)$$
$$u = g(x)^{-1}\left(\ddot{x}_r + (1 - k_1^2)z_1 + k_1 z_2 + f(x_1)\right)$$

2.2.9 *Least Square Parameter Estimation*

Least Square Parameter Estimation merupakan metode estimasi yang banyak digunakan untuk estimasi parameter [20]. Dalam penelitian ini, metode LSPE digunakan untuk estimasi parameter *gain* η. Untuk melakukan estimasi parameter, langkah awal yang harus dilakukan adalah menentukan target yang harus dicapai [21]. Pertimbangkan sebuah sistem:
Target yang harus dicapai adalah estimasi parameter θ untuk meminimalkan fungsi biaya pada persamaan berikut.

$$J = \frac{1}{2}(\hat{\theta}(t) - \hat{\theta}(t-1))^T \left(\hat{\theta}(t-1)\right) + \alpha(y(t) - \varphi^T(t)\hat{\theta}(t)) \quad (2.50)$$

Algoritma $\theta(t)$ dipilih sehingga $||\theta(t) - \theta(t) - 1||$ meminimalkan subject ke *constraint* $y(t) = \varphi^T(t)\hat{\theta}(t)$. Dengan α adalah *lagrange multiplier*, derivatif terhadap $\theta(t)$ dan α. Sehingga didapatkan parameter estimasi $\theta(t)$.

$$\hat{\theta}(t) = \hat{\theta}(t-1) + \frac{\varphi(t)}{\varphi^T(t)\varphi(t)}(y(t) - \varphi^T(t)\hat{\theta}(t-1)) \quad (2.51)$$

Untuk menyesuaikan parameter dan untuk menghindari penyebut bernilai 0 pada Persamaan (2.51), maka dilakukan modifikasi estimasi sebagai berikut:

$$\hat{\theta}(t) = \hat{\theta}(t-1) + \frac{\gamma\varphi(t)}{\lambda + \varphi^T(t)\varphi(t)}(y(t) - \varphi^T(t)\hat{\theta}(t-1)) \quad (2.52)$$

dengan:

$$\lambda > 0 \quad 0 < \gamma < 2$$

Formulasi *update parameter* $\frac{\gamma\varphi(t)}{\lambda + \varphi^T(t)\varphi(t)}$, disebut sebagai Algoritma Kaczmarz.

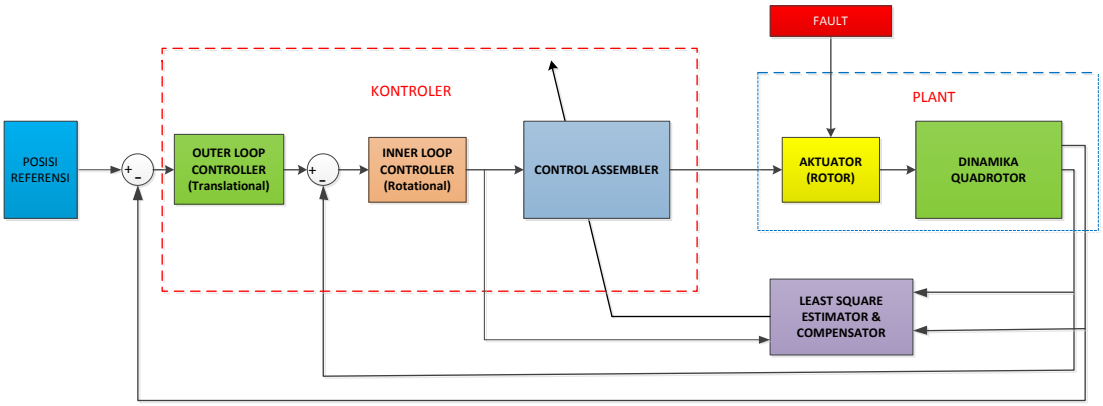
BAB III
PERANCANGAN SISTEM

3.1 Perancangan Kontroler

Kontroler dirancang untuk mengatasi masalah kestabilan quadrotor, pengendali posisi quadrotor dan kompensasi *fault* pada aktuator. Skema *Fault tolerant control* (FTC) yang dirancang terdiri dari kontroler nominal dan estimator serta kompensator kesalahan. Kontrol nominal yang akan dirancang adalah *backstepping control* sebagai kontroler untuk gerak translasi dan rotasi, sedangkan estimator kesalahan menggunakan *Modified Least Square*.

Strategi kontrol yang akan diterapkan pada quadrotor ditunjukkan pada Gambar 3.1 yang terdiri *outer loop, inner loop,* dan *control assembler. Outer loop* merupakan kontroler *tracking* posisi (x, y dan z), *inner loop* merupakan kontroler stabilisasi sudut (ϕ, θ dan ψ), *control assembler* berfungsi mengubah sinyal kontrol U_{1c}, U_{2c}, U_{3c} dan U_{4c} menjadi sinyal masukan motor u_1, u_2, u_3 dan u_4. Sinyal referensi untuk sistem kontrol quadrotor ini terdiri atas referensi untuk posisi *x, y, z* dan sudut ψ. Sinyal referensi posisi akan dibandingkan dengan sinyal aktualnya pada *outer loop* menggunakan kontroler *backstepping*. Output dari kontroler posisi *x* dan *y* akan dikonversi untuk menjadi sudut referensi pada *inner loop*. Sedangkan output dari kontroler posisi *z* akan menjadi input kontrol U_{1p}. Sedangkan kontroler stabilisasi sudut (ϕ, θ dan ψ) membandingkan sinyal sudut referensi dengan sinyal aktualnya sehingga menghasilkan output U_2, U_3 dan U_4.

Pada penelitian ini, kesalahan terjadi pada aktuator (rotor) yang merupakan gaya angkat $F_{T1p}, F_{T2p}, F_{T3p}$ dan F_{T4p} (*thrust*). Metode *Modified Least Square* digunakan untuk estimasi dan kompensasi kontroler sehingga mampu menyesuaikan besarnya sinyal kontrol yang diberikan dengan besarnya kesalahan yang terjadi dengan mengestimasi nilai parameter *gain* η sebagai *gain* kompensator.



Gambar 3.1 Blok diagram kesalahan pada aktuator quadrotor

Pada penelitian ini parameter quadrotor dinyatakan pada Tabel 3.1 yang akan digunakan sebagai parameter untuk simulasi.

Tabel 3.1 Nilai parameter quadrotor [14]

Parameter	Simbol	Nilai	Unit
Massa	m	1.4	kg
Jarak rotor dari pusat massa	l	0.2	m
Momen inertia pada sumbu x	J_{xx}	0.03	$N.m.s^2$
Momen inertia pada sumbu y	J_{yy}	0.03	$N.m.s^2$
Momen inertia pada sumbu z	J_{zz}	0.04	$N.m.s^2$
Gravitasi	g	9.8060	m/s^2
Koefisien gaya angkat	b	2.9842×10^{-5}	$N.s^2/rad^2$
Koefisien gaya *drag*	d	3.230×10^{-7}	$N.m.s^2/rad^2$
Bandwidth aktuator	ω	15	rad/sec
Konstanta gaya dorong	K	120	N
Input pada motor dan propeller		0.1	

3.1.1 Perancangan *Backstepping Control* untuk Gerak Translasi (*Outer Loop*)

Blok *outer loop* digunakan untuk mengatur gerak quadorotor pada sumbu translasi yaitu *X*, *Y* dan *Z*. Langkah awal untuk melakukan desain kontrol sumbu translasi adalah menuliskan kembali persamaan *state space* untuk subsistem translasi yang dinyatakan pada Persamaan 3.1.

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\\ \dot{x}_3\\ \dot{x}_4\\ \dot{x}_5\\ \dot{x}_6\end{bmatrix} = \begin{bmatrix} x_2 \\ \frac{U_1}{m}u_x \\ x_4 \\ \frac{U_1}{m}u_y \\ x_6 \\ \frac{U_1}{m}cosx_7 cosx_9 - g \end{bmatrix} \quad (3.1)$$

Kemudian adalah mendefinisikan *tracking error* e antara nilai aktual x dengan referensi x_r:

$$e = x_r - x \quad (3.2)$$

Langkah selanjutnya adalah dengan mendefinisikan *state*:

$$x_1 = x$$

$$x_2 = \dot{x}_1 = \dot{x}$$

$$z_1 = e = x_r - x_1 \quad (3.3)$$

Untuk mendapatkan persamaan kontrol *backstepping*, maka diperlukan variabel *virtual control*. Asumsikan x_2 adalah variabel *virtual control* dan $a(z_1)$ adalah fungsi yang membawa $z_1 \rightarrow 0$ dengan mendefinisikan $\bar{z_1}$:

$$\bar{z_1} = a(z_1) - x_2 \quad (3.4)$$

Differensialkan terhadap waktu:

$$\dot{z_1} = \dot{x}_r - \dot{x}_1 = \dot{x}_r - x_2 = \dot{x}_r + \bar{z}_1 - a(z_1) \quad (3.5)$$

Dengan mendefinisikan $a(z_1) = \dot{x}_r + k_1 z_1$ dimana $k_1 > 0$ dan mendefinisikan fungsi Lyapunov sebagai:

$$V_1 = \frac{1}{2} z_1^2 \quad (3.6)$$

Turunkan terhadap fungsi waktu, maka:

$$\dot{V}_1 = z_1 \dot{z}_1 = z_1\left(\dot{x}_r + \bar{z_1} - a(z_1)\right) = z_1(\bar{z_1} - k_1 z_1) = -k_1 z_1^2 + z_1 \bar{z_1} \quad (3.7)$$

Persamaan 3.7, jika $\bar{z_1} = 0$ maka $\dot{V}_1 = -k_1 z_1^2$ dan z_1 akan menuju 0 secara asymtotis. Kemudian definisikan fungsi Lyapunov untuk membuat $\bar{z_1} \rightarrow 0$

Untuk melakukan *backstepping* digunakan variabel peubah:

$$V_2 = V_1 + \frac{1}{2}\bar{z_1}^2 \quad (3.8)$$

Turunkan terhadap fungsi waktu didapatkan:

$$\dot{V}_2 = \dot{V}_1 + \bar{z_1}\dot{\bar{z}_1} = -k_1 z_1^2 + \bar{z_1}(\ddot{x}_r + z_1 + k_1\dot{z}_1 - \dot{x}_2) \quad (3.9)$$

Untuk membuat $\dot{V}_2 < 0$, maka digunakan variable $\dot{x}_2$ sebagai sinyal kontrol. Dapat dicari dengan cara berikut ini:

1. Persamaan $-k_1 z_1^2$ dengan $k_1 > 0$ memiliki nilai lebih kecil dari 0 sehingga $\dot{x}_2 = 0$
2. Untuk menghilangkan persamaan $\bar{z_1}\ddot{x}_r$ maka $\dot{x}_2 = \ddot{x}_r$
3. Untuk menghilangkan $z_1\bar{z_1}$ maka $\dot{x}_2 = \ddot{x}_r + z_1$
4. Untuk menghilangkan $k_1\dot{z}_1\bar{z_1}$ maka ubah $\dot{z}_1 = \bar{z_1} - k_1 z_1$ maka persamaan menjadi $k_1(\bar{z_1} - k_1 z_1)\bar{z_1}$ sehingga $\dot{x}_2 = \ddot{x}_r + z_1 - k_1^2 z_1 + k_1\bar{z_1}$

Dapat disederhanakan menjadi Persamaan 3.10.

$$\dot{x}_2 = \ddot{x}_r + (1 - k_1^2)z_1 + (k_1 + \bar{k}_1)\bar{z_1} \quad (3.10)$$

Nilai $\bar{k}_1$ merupakan variabel untuk memperbesar dan memperkecil nilai $\bar{z_1}$ dengan syarat $k_1 > 0$ dan $\bar{k}_1 > 0$, $z_1 = x^r - x_1$ dan $\bar{z}_1 = \dot{x_r} + k_1 z_1 - x_2$

Subsitusikan Persamaan 3.1 ke Persamaan 3.10 didapatkan:

$$\dot{x}_2 = \frac{U_{1c}}{m}u_x$$

$$u_x = \frac{m}{U_{1c}}\dot{x}_2 \quad (3.11)$$

Sehingga didapatkan:

$$u_x = \frac{m}{U_{1c}}(\ddot{x}_r + \left(1 - k_1{}^2\right)z_1 + (k_1 + \bar{k}_1)\bar{z}_1 \quad (3.12)$$

Subsitusikan Persamaan 3.1 ke Persamaan 3.10 didapatkan:

$$\dot{x}_4 = \frac{U_{1c}}{m}u_y \quad (3.13)$$

$$u_y = \frac{m}{U_{1c}}\dot{x}_4 \quad (3.14)$$

Sehingga diperoleh:

$$u_y = \frac{m}{U_{1c}}\left(\ddot{x}_{3r} + \left(1 - {k_3}^2\right)z_3 + (k_3 + \bar{k}_3)\bar{z}_3\right) \quad (3.15)$$

Subsitusikan Persamaan 3.1 ke Persamaan 3.10 didapatkan:

$$\dot{x}_6 = \frac{U_{1c}}{m}\ cos\ x_7 \cos x_9 - g \quad (3.16)$$

$$U_{1c} = u_z = \frac{m}{\cos\emptyset\cos\theta}\dot{x}_6 + g \quad (3.17)$$

Sehingga didapatkan:

$$U_{1c} = u_z = \frac{m}{\cos\emptyset\cos\theta}(\ddot{x}_{5r} + g + \left(1 - {k_5}^2\right)z_5 + (k_5 + \bar{k}_5)\bar{z}_5) \quad (3.18)$$

dengan:

U_{1c} : resultan vektor gaya *thrust* yang diperlukan untuk mendapatkan pergerakan linear yang diharapkan.

u_x :sinyal kontrol yang bertanggung jawab untuk menentukan pergerakan quadrotor pada sumbu x atau dengan kata lain merupakan referensi sumbu *pitch* yang diperlukan.

u_y: sinyal kontrol yang bertanggung jawab untuk menentukan pergerakan quadrotor pada sumbu y atau dengan kata lain merupakan sumbu *roll* yang diperlukan.

Maka diperoleh persamaan kontroler pergerakan translasi sebagaimana dinyatakan dalam Persamaan 3.19.

$$u_x = \frac{m}{U_{1c}}(\ddot{x}_r + \left(1 - {k_1}^2\right)z_1 + (k_1 + \bar{k}_1)\bar{z}_1)$$

$$u_y = \frac{m}{U_{1c}}\left(\ddot{x}_{3r} + \left(1 - {k_3}^2\right)z_3 + (k_3 + \bar{k}_3)\bar{z}_3\right)$$

$$U_{1c} = u_z = \frac{m}{\cos\emptyset\cos\theta}(\ddot{x}_{5r} + g + \left(1 - {k_5}^2\right)z_5 + (k_5 + \bar{k}_5)\bar{z}_5) \quad (3.19)$$

3.1.2 Perancangan *Backstepping Control* untuk Gerak Rotasi (*Inner Loop*)

Blok ini digunakan untuk mengatur gerak rotasi quadrotor yaitu *pitch*, *roll* dan *yaw*. *Pitch* dan *roll* mendapatkan sinyal referensi dari keluaran sumbu *X* dan *Y*. Persamaan *state space* untuk subsistem rotasi yang dinyatakan pada Persamaan 3.20.

$$\begin{bmatrix} \dot{x}_7 \\ \dot{x}_8 \\ \dot{x}_9 \\ \dot{x}_{10} \\ \dot{x}_{11} \\ \dot{x}_{12} \end{bmatrix} = \begin{bmatrix} x_8 \\ \frac{U_2}{J_{xx}} - \frac{x_{10}x_{12}}{J_{xx}}(J_{xx} - J_{yy}) \\ x_{10} \\ \frac{U_3}{J_{yy}} - \frac{x_8x_{12}}{J_{yy}}(J_{yy} - J_{zz}) \\ x_{12} \\ \frac{U_4}{J_{zz}} - \frac{x_8x_{10}}{J_{zz}}(J_{yy} - J_{zz}) \end{bmatrix} \quad (3.20)$$

Cara yang digunakan untuk mendapatkan persamaan sinyal kontrol sumbu rotasi quadrotor sama dengan cara untuk mendapatkan persamaan kontrol sumbu translasi yang dinyatakan pada Persamaan 3.2 – 3.10, sehingga persamaan sinyal kontrol untuk sumbu rotasi (*inner loop*) yaitu sebagai berikut:

$$U_{2c} = J_{xx}(\frac{x_{10}x_{12}}{J_{xx}}(J_{zz} - J_{yy}) + (1 - {k_7}^2)z_7 + (k_7 + \bar{k}_7)\bar{z}_7)$$

$$U_{3c} = J_{yy}(\frac{x_8x_{12}}{J_{yy}}(J_{xx} - J_{zz}) + (1 - {k_9}^2)z_9 + (k_9 + \bar{k}_9)\bar{z}_9)$$

$$U_{4c} = J_{zz}(\frac{x_8x_{10}}{J_{zz}}(J_{xx} - J_{xx}) + (1 - {k_{11}}^2)z_{11} + (k_{11} + \bar{k}_{11})\bar{z}_{11}) \quad (3.21)$$

3.1.3 Pembuktian Stabilitas *Backstepping Control*

Pada dasarnya sistem dapat dikatakan stabil dikarenakan penurunan sistem kontrol diperoleh dengan fungsi Lyapunov $\dot{V}_1 < 0$ dan $\dot{V}_2 < 0$. Sebagai pembuktianya adalah sebagai berikut:

Dengan mengambil contoh persamaan sinyal kontrol:

$$U_{1c} = \frac{m}{\cos\emptyset\cos\theta}(\ddot{x}_{5r} + g + (1 - {k_5}^2)z_5 + (k_5 + \bar{k}_5)\bar{z}_5) \quad (3.22)$$

Merupakan masukan sinyal kendali sumbu *Z*, dengan model dinamika sumbu *Z* adalah:

$$\dot{x}_5 = \frac{U_{1c}}{m}\ cos\ \emptyset\cos\theta - g \quad (3.23)$$

Substitusi U_1 Persamaan (3.22) ke Persamaan (3.23) maka:

$$\dot{x}_5 = \frac{m\,(\ddot{x}_{5r} + g + (1 - k_5^2)z_5 + (k_5 + \bar{k}_5)\bar{z}_5)}{m\ \cos\emptyset\cos\theta}\ cos\ \emptyset\cos\theta - g \quad (3.24)$$

Dapat disederhanakan menjadi:

$$\dot{x}_5 = (\ddot{x}_{5r} + (1 - k_5^2)z_5 + (k_5 + \bar{k}_5)\bar{z}_5) \quad (3.25)$$

dengan:

$z_5 = e = x_r - x_5$

$\bar{z_5} = \dot{x}_r + k_5 z_5 - x_2 = \dot{z}_1 + k_5 z_5$

Dengan mengambil nilai $0 \leq k_5 \leq 1$ dan $\bar{k}_5 > 0$ maka:

$$\dot{x}_5 = \left(1 + 2k_5^2 + k_5\bar{k}_5\right)z_5 + \left(k_5 + \bar{k}_5\right)\dot{\bar{z}_5} \quad (3.26)$$

Untuk mendapatkan keluaran berupa *state* posisi dan jika dinyatakan dalam fungsi transfer domain laplace adalah sebagai berikut:

$$\dot{x}_5 = \left(1 + 2k_5^2 + k_5\bar{k}_5\right)(x_r - x_5) + \left(k_5 + \bar{k}_5\right)(\dot{x_r} - \dot{x_5})$$

$$s^2\, x_5 = K_c\, (x_r - x_5)$$

$$x_5 = \frac{K_c\, (x_r - x_5)}{s^2} \quad (3.27)$$

dengan:

$K_c = \left(1 + 2k_5^2 + k_5\bar{k}_5\right) + \left(k_5 + \bar{k}_5\right)s$

Fungsi transfer dari keseluruhan sistem kendali pada sumbu *Z* dapat di peroleh dengan:

$x_5 = s^{-2}K_c(x_r - x_5)$

$x_5 = s^{-2}K_c x_r - s^{-2}K_c x_5$

$x_5 + s^{-2}K_c x_5 = s^{-2}K_c x_r =$

$x_5(1 + s^{-2}K_c) = s^{-2}K_c x_r$

$$\frac{x_5}{x_r} = H(s) = \frac{s^{-2}K_c}{1 + s^2K_c} = \frac{K_c}{s^2 + K_c} \quad (3.28)$$

Substitusi nilai K_c maka:

$$H(s) = \frac{(k_1 + k_2)s + \left(1 + 2k_1^2 + k_1k_2\right)}{s^2 + (k_1 + k_2)s + \left(1 + 2k_1^2 + k_1k_2\right)} \quad (3.29)$$

$$H(t) = Kd * e^{\frac{-Kd*t)}{2}} Cosh\left(t * \left(\frac{Kd^2}{4} - Kp\right)^{0.5}\right) - \frac{\left(Sinh\left(t*\left(\frac{Kd^2}{4}-Kp\right)^{0.5}\right)\left(\frac{Kd}{2}-\frac{Kp}{Kd}\right)\right)}{\left(\frac{K_d^2}{4}-Kp\right)^{0.5}} \quad (3.30)$$

Dari Persamaan 3.30, dapat dilihat ketika waktu t = 0 maka sistem akan memiliki keluaran = 0, sedangkan ketika $t \rightarrow \infty$ maka keluaran sistem menuju nilai referensi yaitu 1. Hal ini menunjukkan bahwa sistem akan stabil dimana $e \rightarrow 0$ ketika $t \rightarrow \infty$.

3.1.4 Perancangan Blok Peubah u_x u_y ke $\emptyset$ dan θ

Blok ini berfungsi untuk mengubah sinyal kontrol sumbu translasi *X* dan *Y* yaitu u_x dan u_y ke besaran sudut $\emptyset_{ref}$ dan θ_{ref} yang merupakan sinyal referensi untuk kontroler sumbu *roll* dan *pitch*.

Dengan mendefinisikan Persamaan 2.33 dan 2.34 menjadi sebuah matriks yaitu:

$$\begin{bmatrix} u_x \\ u_y \end{bmatrix} = \begin{bmatrix} \cos\Psi & \sin\Psi \\ \sin\Psi & -\cos\Psi \end{bmatrix} \begin{bmatrix} \cos\emptyset \sin\theta \\ \sin\emptyset \end{bmatrix}$$

$$\begin{bmatrix} \cos\emptyset \sin\theta \\ \sin\emptyset \end{bmatrix} = \begin{bmatrix} \cos\Psi & \sin\Psi \\ \sin\Psi & -\cos\Psi \end{bmatrix}^{-1} \begin{bmatrix} u_x \\ u_y \end{bmatrix}$$

dengan $\cos^2\Psi + \sin^2\Psi$ =1 maka:

$$\begin{bmatrix} \cos\emptyset \sin\theta \\ \sin\emptyset \end{bmatrix} = \begin{bmatrix} \cos\Psi & \sin\Psi \\ \sin\Psi & -\cos\Psi \end{bmatrix} \begin{bmatrix} u_x \\ u_y \end{bmatrix}$$

Sehingga persamaan blok peubah sinyal kontrol u_x u_y ke $\emptyset_{ref}$ dan θ_{ref} adalah

$$\theta_{ref} = \sin^{-1}\left(\frac{u_x \cos\Psi + u_y \sin\Psi}{\cos\emptyset}\right)$$

$$\emptyset_{ref} = \sin^{-1}(u_x \sin\Psi - u_y \cos\Psi) \quad (3.31)$$

3.1.5 Perancangan Blok *Control Assembler*

Blok *control assembler* merupakan kombinasi sinyal kontrol yang akan diubah kedalam sinyal kontrol motor. Langkah- langkah perancangan blok *control assembler* yaitu sebagai berikut:

Tulis persamaan gaya yang terdapat pada dinamika quadrotor yang terdapat pada Persamaan 2.7 – 2.11 yaitu:

$$U_1 = (F_{T1} + F_{T2} + F_{T3} + F_{T4})$$

$$U_2 = l(F_{T4} - F_{T2})$$

$$U_3 = l(F_{T3} - F_{T1})$$

$$U_4 = (F_{T2} + F_{T4} - F_{T1} - F_{T3})d \tag{3.32}$$

Persamaan 3.32 dapat dinyatakan dalam matriks berikut ini:

$$\begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & -l & 0 & l \\ -l & 0 & l & 0 \\ -d & d & -d & d \end{bmatrix} \begin{bmatrix} F_{T1} \\ F_{T2} \\ F_{T3} \\ F_{T4} \end{bmatrix} \tag{3.33}$$

Dengan nilai U_{1c}, U_{2c}, U_{3c} dan U_{4c} yang telah diperoleh dari kontroler translasi dan rotasi maka dengan melakukan invers Persamaan 3.33 akan diperoleh nilai F_{T1c} pada *control assembler* yang merupakan referensi nilai gaya yang harus dihasilkan oleh rotor, Sehingga *Invers* Persamaan 3.33 adalah sebagai berikut:

$$\begin{bmatrix} F_{T1c} \\ F_{T2c} \\ F_{T3c} \\ F_{T4c} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & -l & 0 & l \\ -l & 0 & l & 0 \\ -d & d & -d & d \end{bmatrix}^{-1} \begin{bmatrix} U_{1c} \\ U_{2c} \\ U_{3c} \\ U_{4c} \end{bmatrix} \tag{3.34}$$

Dengan diketahui pada Qball-X4: $F_{Ti} = K \frac{\omega}{s+\omega} u_i$

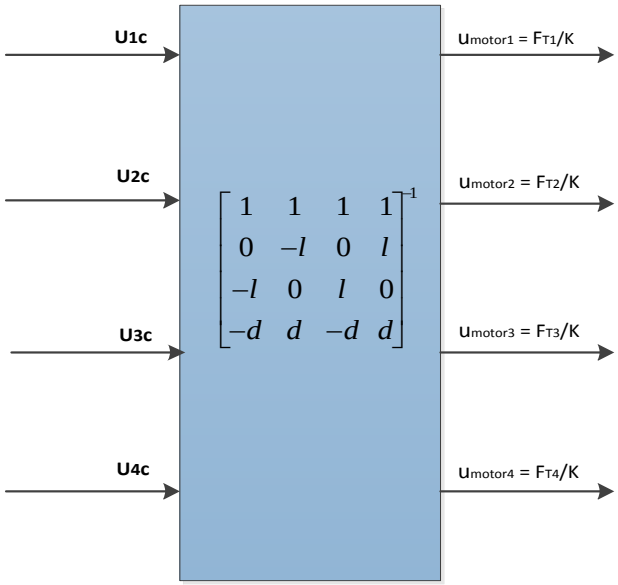
dengan: F_{Ti} adalah gaya angkat (*thrust*), K adalah konstanta gaya dorong, $\frac{\omega}{s+\omega}$ adalah dinamika motor dan u_i adalah sinyal kontrol dari kontroler ke motor.

Dari hubungan F_{Ti}, K, dan u_i yang berkaitan maka untuk mendapatkan sinyal masukan motor (u_i) yaitu sebagai berikut:

$u_{ic} = \frac{F_{Tic}}{K}$ dimana i = 1,2,3,4

Atau dapat ditulis kedalam persamaan berikut ini:

$$\begin{bmatrix} u_{1c} \\ u_{2c} \\ u_{3c} \\ u_{4c} \end{bmatrix} = \begin{bmatrix} F_{T1}/K \\ F_{T2}/K \\ F_{T3}/K \\ F_{T4}/K \end{bmatrix} \tag{3.35}$$



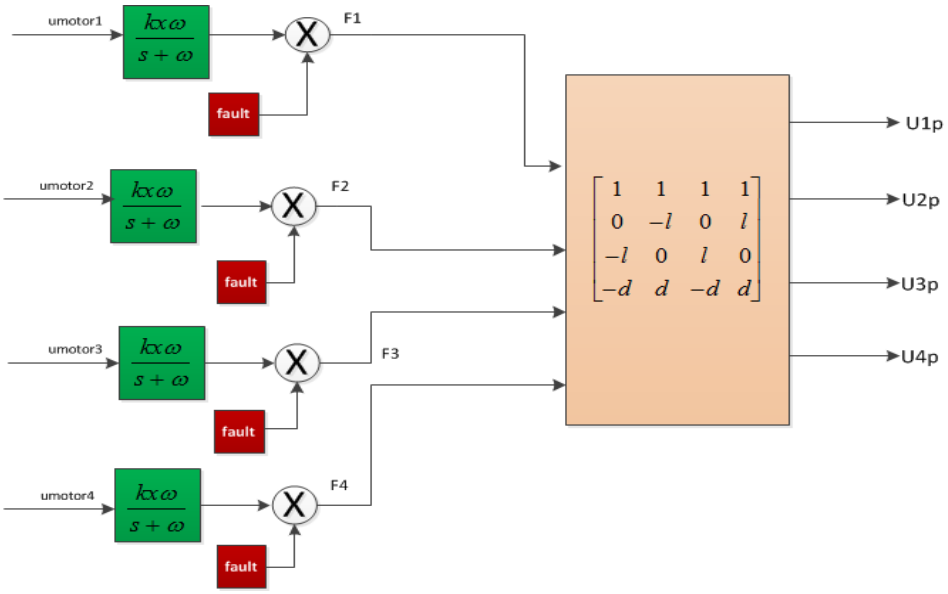
Gambar 3.2 Blok Diagram *Control Assembler*

3.1.6 Perancangan Kesalahan pada Aktuator (Rotor) Quadrotor

Kesalahan yang terjadi pada aktuator akan mengakibatkan tidak sesuainya sinyal kontrol yang menjadi masukan plant. Oleh karena itu, sinyal kontrol yang terkena kesalahan ini akan mengakibatkan ketidakstabilan pada plant tersebut. Pada penelitian ini, aktuator mengalami kesalahan yang bersifat *multiplicative* terhadap gaya yang dihasilkan oleh rotor (kondisi *loss of effectiveness*). Oleh karena nilai pengalinya adalah 0 sampai 1 (0-100%) maka besarnya sinyal kontrol yang masuk ke plant akan menjadi lebih kecil sesuai dengan skala tertentu jika dibandingkan dengan sinyal kontrol yang dihasilkan oleh kontroler. Kesalahan aktuator pada quadrotor terjadi pada rotor terlihat pada Gambar 3.3 yang merupakan gaya angkat/*thrust* (F_{Ti}) pada motor dan propeller sebagai berikut:

$$F_{Ti_m} = f_x \, K \frac{\omega}{s+\omega} u_{i_m} \tag{3.36}$$

dengan F_{Ti_m} adalah gaya angkat yang dihasilkan oleh motor dan f_x adalah besar kesalahan dengan rentang nilai 0 yang menyatakan terjadi *fault* sebesar 100% dan 1 yang menyatakan tidak terjadi *fault* (0%).



Gambar 3.3. Kesalahan pada Aktuator (Rotor)

Pada Q-Ball X4 diketahui tiap motor menghasilkan gaya angkat (F_{Ti}) maksimal sebesar 12N. Sedangkan untuk terbang melayang pada ketinggian tertentu (*hover*) dengan berat Qball-X4 Quadrotor adalah m = 1.4 kg, maka gaya angkat yang harus dihasilkan oleh keempat rotor adalah sebesar:

$$F_{total} = U_1 = m.g = 13.72 \text{ N}$$

dengan m: massa quadrotor (1.4kg) dan g= percepatan gravitasi (9.8m/s^2)

Sedangkan gaya yang harus dihasilkan oleh tiap-tiap motor untuk tetap terbang pada ketinggian tertentu adalah dengan menggunakan Persamaan dinamika berikut

$$U_1 = F_1 + F_2 + F_3 + F_4$$

Ketika quadrotor melakukan *hover* atau terbang pada ketinggian tertentu tanpa melakukan pergerakan kearah sumbu X,Y maka keempat rotor menghasilkan gaya yang relative sama besar, maka $F_1 = F_2 = F_3 = F_4 = F$. Sehingga Persamaan diatas dapat diubah menjadi:

$U_1 = 13.72 \text{ N} = 4F$

Sehingga diperoleh gaya yang diperlukan untuk *hover* oleh tiap-tiap motor adalah

$F = \frac{13.72}{4} = 3.43\ N$

Nilai gaya yang digunakan untuk *hover* ini jika dipresentasekan dengan maksimum gaya (12N) yang dapat dihasilkan adalah sebagai berikut:

% F = $\frac{3.43\ N}{12\ N} x\ 100\% = 29\%$

Dari nilai 29% ini, menunjukan bahwa tiap motor masih dapat ditingkatkan kemampuan mendekati 70% tanpa melakukan pergerakan naik ketika terjadi kesalahan (*fault*), karena untuk melakukan pergerakan ke atas akan membutuhkan gaya lebih besar dari 13.72N atau lebih besar dari 30% pada tiap-tiap motor.

3.1.7 Perancangan *Fault Estimator* dan *Compensator*

Hal pertama yang dilakukan adalah estimasi kesalahan yang terjadi di rotor. Untuk melakukan estimasi yaitu dengan menggunakan formulasi *least square*. Langkah- langkah untuk mendapatkan formulasi *Fault Estimator* dan *Compensator* adalah sebagai berikut:

– Diketahui pada Persamaan (3.36), *fault* terletak pada gaya yang dihasilkan oleh motor dan propeller (rotor) yaitu:

$$F_{Tip}(\boldsymbol{t}) = f_x K\ \frac{\omega}{s+\omega} u_{ip}(\boldsymbol{t}) \qquad (3.37)$$

Dari Persamaan 3.37 terdapat permasalahan yaitu tidak adanya sensor yang dapat membaca nilai f_x secara langsung, maupun membaca nilai gaya F_{Tip} yang dihasilkan oleh tiap motor.

– Dari Persamaan (3.37) diketahui variable gaya (F_{Tip}) rotor mempengaruhi nilai variable U yang ditunjukkan pada persamaan berikut ini:

$$\begin{bmatrix} U_{1p} \\ U_{2p} \\ U_{3p} \\ U_{4p} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & -l & 0 & l \\ -l & 0 & l & 0 \\ -d & d & -d & d \end{bmatrix} \begin{bmatrix} F_{T1p} \\ F_{T2p} \\ F_{T3p} \\ F_{T4p} \end{bmatrix} \tag{3.38}$$

Permasalahan yang sama seperti Persamaan (3.37) yaitu tidak ada sensor yang dapat digunakan untuk membaca nilai U_{ip}.

– Dari Persamaan (3.38) diketahui nilai $U_{1P}, U_{2P}, U_{3P}, U_{4P}$ mempengaruhi dinamika quadrotor yang dinyatakan dalam persamaan:

$$\begin{bmatrix} \ddot{Z} = \frac{U_1 p}{m} \cos x_7 \cos x_9 - g \\ \ddot{\emptyset} = \frac{U_{2p} l}{J_{xx}} \\ \ddot{\theta} = \frac{U_{3p} l}{J_{yy}} \\ \ddot{\psi} = \frac{U_{4p}}{J_{zz}} \end{bmatrix} \tag{3.39}$$

Dari Persamaan 3.39, terdapat sensor yang mampu membaca *state* $\ddot{Z}, \ddot{\emptyset}, \ddot{\theta}, \ddot{\psi}$ sehingga dengan menggunakan *state* persamaan ini akan diperoleh nilai *fault* (f_x) melalui persamaan berikut:

a. Untuk mendapatkan nilai $\widehat{U}_{1P}, \widehat{U}_{2P}, \widehat{U}_{3P}, \widehat{U}_{4P}$ dengan melakukan *Invers* Persamaan (3.39) maka dapat diperoleh:

$$\widehat{U}_{1p} = \frac{m(\ddot{z} + g)}{cos\emptyset \cos\theta}$$

$$\widehat{U}_{2p} = \frac{J_{xx}\ddot{\emptyset}}{l}$$

$$\widehat{U}_{3p} = \frac{J_{yy}\ddot{\theta}}{l}$$

$$\widehat{U}_{4p} = J_{zz}\ddot{\psi} \tag{3.40}$$

b. Dengan memperoleh nilai $\widehat{U}_{1p}, \widehat{U}_{2p}, \widehat{U}_{3p}, \widehat{U}_{4p}$ maka gaya motor (F_{Tip}) dapat diestimasi dengan persamaan berikut ini:

$$\begin{bmatrix} \hat{F}_{T1p} \\ \hat{F}_{T2p} \\ \hat{F}_{T3p} \\ \hat{F}_{T4p} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & -l & 0 & l \\ -l & 0 & l & 0 \\ -d & d & -d & d \end{bmatrix}^{-1} \begin{bmatrix} \widehat{U}_{1p} \\ \widehat{U}_{2p} \\ \widehat{U}_{3p} \\ \widehat{U}_{4p} \end{bmatrix} \tag{3.41}$$

Dengan membandingkan antara F_{plant} dengan F_{input} maka diperoleh nilai *fault* (f_x):

$$f_x = \frac{\hat{F}_{plant}}{Finput} \tag{3.42}$$

Namun Persamaan 3.42, tidak dapat dilakukan ketika nilai $F_{input} = 0$, karena akan menghasilkan perhitungan tak terdefinisi dan dengan rumus tersebut akan mengakibatkan nilai f_x yang tidak stabil karena sangat dipengaruhi oleh perubahan *state* [22]. Untuk itu diperlukan suatu metode yang mampu mengatasi perubahan *state* tersebut, pada penelitian ini metode *modified least square* digunakan agar mampu mengatasi permasalahan tersebut. Untuk mendapatkan estimasi f_x dengan menggunakan metode *least square* yaitu dengan memisalkan nilai $F_{output} = F_{Tp}$, $F_{input} = F_{Tc}$ dan nilai f_x $^{-1}$= ɲi sehingga dapat digunakan untuk mengkompensasi besar nilai gaya yang harus diberikan pada rotor. Sehingga persamaan *control assembler* dapat ditulis menjadi:

$$\begin{bmatrix} F_{T1c}(\boldsymbol{t}) \\ F_{T2c}(\boldsymbol{t}) \\ F_{T3c}(\boldsymbol{t}) \\ F_{T4c}(\boldsymbol{t}) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & -l & 0 & l \\ -l & 0 & l & 0 \\ -d & d & -d & d \end{bmatrix}^{-1} \begin{bmatrix} U_{c1}(\boldsymbol{t}) \\ U_{c2}(\boldsymbol{t}) \\ U_{c3}(\boldsymbol{t}) \\ U_{c4}(\boldsymbol{t}) \end{bmatrix} \circ \begin{bmatrix} \eta_1(\boldsymbol{t}) \\ \eta_2(\boldsymbol{t}) \\ \eta_3(\boldsymbol{t}) \\ \eta_4(\boldsymbol{t}) \end{bmatrix} \tag{3.43}$$

dimana $\eta_i > 0$, $i = 1,2,3,4$ dan operator adalah operator perkalian Haddamar, yaitu perkalian element matriks pada baris dan kolom yang sama, sehingga nilai parameter *gain* matriks η dapat diperoleh dengan menggunakan metode *Modified Least Square* yang diperoleh sebagai berikut:

1) Tentukan fungsi objektif yaitu meminimisasi *gain* $\left(\hat{\eta}(t) - \hat{\eta}(t-1)\right)^{\circ 2}$

   Sehingga:

   $$J = \left[\left(\hat{\eta}(t) - \hat{\eta}(t-1)\right)^{\circ 2} \right]$$

   Dengan *constrain* atau batasan yaitu $a \circ \left(\hat{\eta}(t) \circ F_{Tp}(\boldsymbol{t}) - F_{Tc}(\boldsymbol{t})\right)$ sehingga dapat disusun persamaan lagrangian yaitu:

   $$L = \frac{1}{2}\left[\left(\hat{\eta}(t) - \hat{\eta}(t-1)\right)^{\circ 2} + a \circ \left(\hat{\eta}(t) \circ F_{Tp}(\boldsymbol{t}) - F_{Tc}(\boldsymbol{t})\right) \right] \quad (3.44)$$

   dengan a adalah *lagrange multiplier*

2) Uraikan Persamaan (3.44)

   $$L = \frac{1}{2}\left[\hat{\eta}^{\circ 2}(t) - 2\hat{\eta}(t) \circ \hat{\eta}(t-1) + \hat{\eta}^{\circ 2}(t-1) + a \circ \hat{\eta}(t) \circ F_{Tp} - a \circ F_{Tc} \right] (3.45)$$

3) Turunkan L terhadap η

   $$\hat{\eta}(t) - \hat{\eta}(t-1) + a \circ F_{Tp}(\boldsymbol{t}) = 0$$
   $$\hat{\eta}(t) = \hat{\eta}(t-1) + a \circ F_{Tp}(\boldsymbol{t}) \quad (3.46)$$

4) Turunkan L terhadap a

   $$\frac{dL}{d\hat{\eta}} = 0 = \hat{\eta}(t) \circ F_{Tp(\boldsymbol{t})} - F_{Tc(\boldsymbol{t})} = 0$$
   $$F_{Tc}(\boldsymbol{t}) = \hat{\eta}(t) \circ F_{Tp}(\boldsymbol{t}) \quad (3.47)$$

5) Substitusikan Persamaan (3.46) ke Persamaan (3.47)

   $$F_{Tc} = (\hat{\eta}(t-1) + a \circ F_{Tp}(\boldsymbol{t})) \circ F_{Tp}(\boldsymbol{t})$$

   $$F_{Tc} = \hat{\eta}(t-1) \circ F_{p}(\boldsymbol{t}) + a \circ F_{Tp}(\boldsymbol{t}) \circ F_{Tp}(\boldsymbol{t})$$

$$a = [F_{Tc(\boldsymbol{t})} - \hat{\eta}(t-1) \circ F_{Tp}(\boldsymbol{t})] \circ [F_{Tp}(\boldsymbol{t}) \circ F_{Tp}(\boldsymbol{t})]^{\circ -1} \quad (3.48)$$

6) Substitusikan Persamaan (3.48) ke Persamaan (3.46) maka diperoleh:

$$\hat{\eta}(t) = \hat{\eta}(t-1) + ([F_{Tc(\boldsymbol{t})} - \hat{\eta}(t-1) \circ F_{Tp}(\boldsymbol{t})] \circ [F_{Tp}(\boldsymbol{t}) \circ F_{Tp}(\boldsymbol{t})]^{\circ -1}) F_{Tp}(\boldsymbol{t})$$

Dapat ditulis menjadi:

$$\hat{\eta}(t) = \hat{\eta}(t-1) - [F_{Tp}(\boldsymbol{t})] \circ [F_{Tp}(\boldsymbol{t}) \circ F_{Tp(\boldsymbol{t})}]^{\circ -1} (\hat{\eta}(t-1) \circ F_p(\boldsymbol{t}) - F_{Tc}(\boldsymbol{t})) \quad (3.49)$$

Untuk mengatasi pembagian nol pada Persamaan 3.49, maka digunakan persamaan berikut ini:

$$[\lambda F_{Tp(\boldsymbol{t})}] \circ [1 + F_{Tp(\boldsymbol{t})} \circ F_{Tp}(\boldsymbol{t})]^{\circ -1} \text{ dengan } \lambda > 0$$

Maka Persamaan 3.49 menjadi:`

$$\hat{\eta}(t) = \hat{\eta}(t-1) - [\lambda F_{Tp(\boldsymbol{t})}] \circ [1 + F_{Tp(\boldsymbol{t})} \circ F_{Tp}(\boldsymbol{t})]^{\circ -1} \left(\hat{\eta}(t-1) \circ F_p(\boldsymbol{t}) - F_{Tc}(\boldsymbol{t})\right) \quad (3.50)$$

Pembuktian:

Diketahui untuk meminimalkan fungsi Objektif *J* yaitu:

$$J = \left(\hat{\eta}(t) - \hat{\eta}(t-1)\right)^2$$

Untuk pembuktiannya, yang perlu dilakukan adalah analisis $\hat{\eta}(t) - \hat{\eta}(t-1)$.

Dari hasil perhitungan yang diperoleh pada Persamaan (3.50), yaitu:

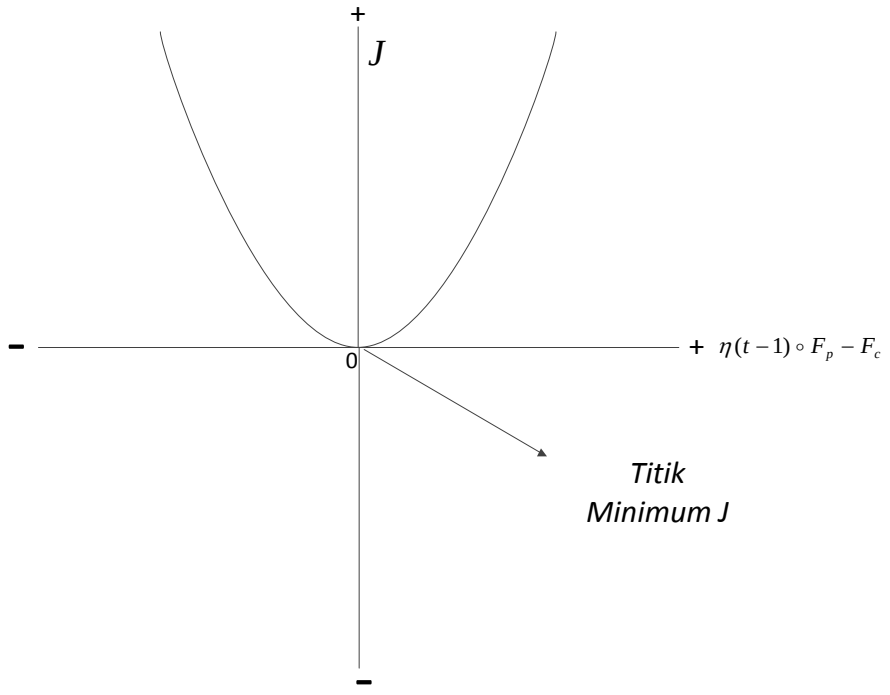
$$\hat{\eta}(t) - \hat{\eta}(t-1) = -[\lambda F_{Tp(\boldsymbol{t})}] \circ [1 + F_{Tp(\boldsymbol{t})} \circ F_{Tp}(\boldsymbol{t})]^{\circ -1} (\hat{\eta}(t-1) \circ F_p(\boldsymbol{t}) - F_{Tc}(\boldsymbol{t}))$$

Diketahui gaya yang menjadi referensi dan gaya yang dihasilkan selalu bernilai positif yang dikarenakan putaran rotor tidak dapat berputar arah, maka persamaan diatas dapat diketahui kondisi pergerakannya dimana:

1) Ketika $\hat{\eta}(t-1) \circ F_p(\boldsymbol{t}) < F_{Tc}(\boldsymbol{t})$ maka $\hat{\eta}(t) - \hat{\eta}(t-1) > 0$ sehingga $J > 0$
2) Ketika $\hat{\eta}(t-1) \circ F_p(\boldsymbol{t}) > F_{Tc}(\boldsymbol{t})$ maka $\hat{\eta}(t) - \hat{\eta}(t-1) < 0$ sehingga $J > 0$

3) Ketika $\hat{\eta}(t-1) \circ F_p(\boldsymbol{t}) = F_{Tc}(\mathbf{t})$ maka $\hat{\eta}(t) - \hat{\eta}(t-1) = 0$ sehingga $J = 0$

Dari ketiga kondisi diatas dapat dibuat grafik sebagai berikut:



Gambar 3.4 Konvergensi Fungsi Objective *J* terhadap $\hat{\eta}(t-1) \circ F_p(\boldsymbol{t}) - F_{Tc}(\boldsymbol{t}))$

Dari Gambar 3.4 terlihat bahwa fungsi objektif *J* akan diminimisasi dengan titik paling minimum 0, yaitu pada keadaan ketika gaya yang dihasilkan (F_p) sama dengan gaya yang direferensikan ke motor (F_c) sebelum dikalikan dengan $\hat{\eta}(t)$ meskipun terjadi kesalahan (*fault*).

Dari langkah-langkah diatas dapat disusun algoritma perancangan kontrol toleransi kesalahan dengan estimasi dan kompensasi menggunakan *Modified Least Square* yang dinyatakan pada Tabel 3.2

Tabel 3.2 Algoritma Kontrol Toleransi Kesalahan *Modified Normalized Least Square*

1. ***State F* (Gaya Rotor) Output *Estimator***

$$\begin{bmatrix} \widehat{F}_{T1p}(t) \\ \widehat{F}_{T2p}(t) \\ \widehat{F}_{T3p}(t) \\ \widehat{F}_{T4p}(t) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & -l & 0 & l \\ -l & 0 & l & 0 \\ -d & d & -d & d \end{bmatrix}^{-1} \begin{bmatrix} \widehat{U}_{1p}(t) \\ \widehat{U}_{2p}(t) \\ \widehat{U}_{3p}(t) \\ \widehat{U}_{4p}(t) \end{bmatrix}$$

dengan:

$$\widehat{U}_{1p}(t) = m(\ddot{z}(t) + g)/cos\emptyset \cos\theta$$

$$\widehat{U}_{2p}(t) = \frac{J_{xx}\ddot{\emptyset}(t)}{l}$$

$$\widehat{U}_{3p}(t) = \frac{J_{yy}\ddot{\theta}(t)}{l}$$

$$\widehat{U}_{4p}(t) = J_{zz}\ddot{\psi}(t)$$

2. ***Invers Fault Estimator* dengan *Modified Normalized Least Square***

$$\hat{\eta}(t) = \hat{\eta}(t-1) + \left[\lambda F_{Tp(t)}\right] \circ [1 + F_{Tp(t)} \circ F_{Tp}(t)]^{\circ -1} (F_{Tc}(t) - \hat{\eta}(t-1) \circ F_{Tp}(t))$$

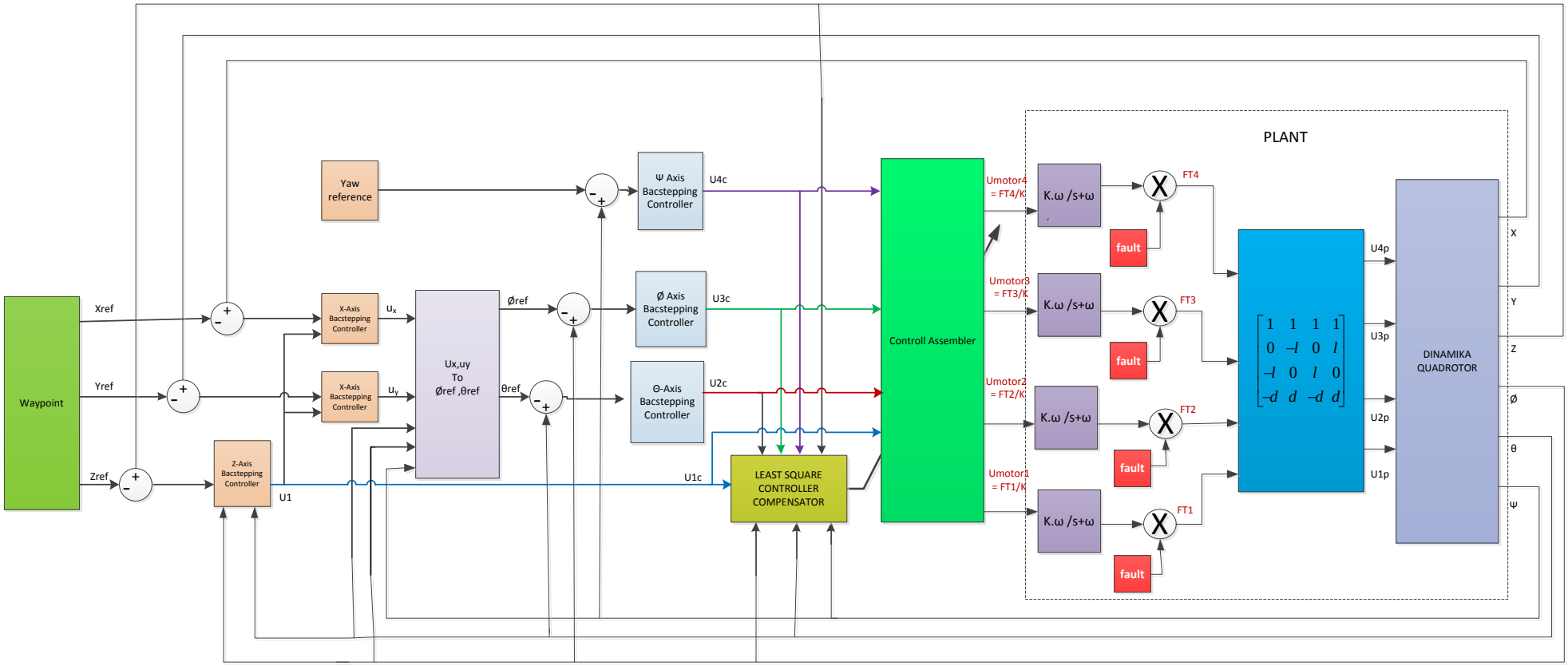
dengan: $\lambda > 0$

3. **Gunakan *Gain* $\hat{\eta}(t)$ sebagai kompensator pada *Control Assembler***

$$\begin{bmatrix} F_{T1c}(t) \\ F_{T2c}(t) \\ F_{T3c}(t) \\ F_{T4c}(t) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & -l & 0 & l \\ -l & 0 & l & 0 \\ -d & d & -d & d \end{bmatrix}^{-1} \begin{bmatrix} U_{c1}(t) \\ U_{c2}(t) \\ U_{c3}(t) \\ U_{c4}(t) \end{bmatrix} \circ \begin{bmatrix} \hat{\eta}_1(t) \\ \hat{\eta}_2(t) \\ \hat{\eta}_3(t) \\ \hat{\eta}_4(t) \end{bmatrix}$$

3.2 Blok Diagram

Pada Gambar 3.5 merupakan blok diagram sistem kontrol FTC quadrotor dengan menggunakan kontrol *Backstepping* sebagai kontroler untuk gerak translasi dan rotasi, sedangkan metode *Modified Least Square* digunakan untuk estimasi dan kompensasi kontroler sehingga mampu menyesuaikan besarnya sinyal kontrol yang diberikan dengan besarnya kesalahan yang terjadi dengan estimasi nilai parameter *gain* η sebagai *gain* kompensator.



Gambar 3.5 Blok Diagram Sistem Kontrol FTC Quadrotor

BAB IV
HASIL DAN PEMBAHASAN

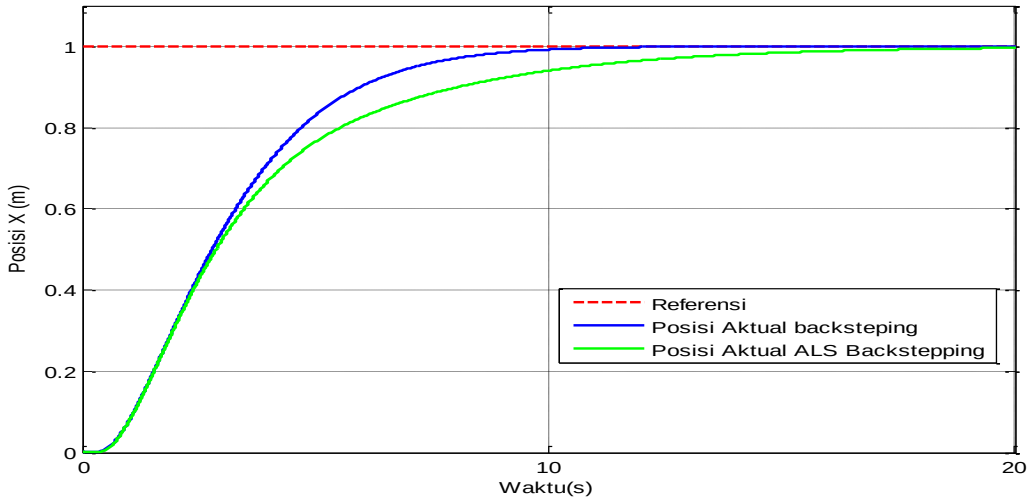
Bab ini membahas mengenai pengujian hasil perancangan sinyal kontrol pada quadrotor seperti yang telah dijelaskan pada Bab III. Pengujian dilakukan dengan menggunakan Simulink pada MATLAB. Pada pembahasan pengujian sistem akan ditampilkan saat keadaan nominal (*fault free case*) dan kondisi dengan kesalahan pada aktuator (*faulty case*).

4.1 Pengujian *Fault Free Case*

Pada *fault free case* diasumsikan sistem tidak mengalami kesalahan pada aktuator. Pengujian dilakukan dengan memberikan referensi sinyal *step* dan *waypoint*, hal ini bertujuan untuk mengetahui performa sistem dengan kontroler dengan berbagai metode kontrol.

4.1.1 Hasil Pengujian *Fault Free Case*

Pada tahap pengujian kondisi *fault free case*, pengujian dilakukan dengan memberikan nilai referensi sumbu X, Y dan Z tetap dari awal simulasi hingga akhir simulasi. Pengujian bertujuan untuk melihat perbandingan respon sistem quadrotor yang di kendalikan menggunaan metode *backstepping* konvensional dan *modified least square*.



Gambar 4.1 Posisi quadrotor pada sumbu X saat *fault free case*

Perbandingan pergerakan sistem quadrotor pada sumbu X dengan menggunakan metode *backstepping* konvensional dan *modified least square* ditunjukkan pada Gambar 4.1. Pada gambar tersebut terlihat bahwa pergerakan menggunakan kontroler *backstepping* memiliki respon lebih cepat, dengan waktu *rise time* dan *setling time* seperti yang ditunjukkan pada Tabel 4.1 dan Tabel 4.2, dimana pada tabel tersebut telihat perbedaan respon *step* dengan menggunakan *backstepping control* dan *modified least square*.

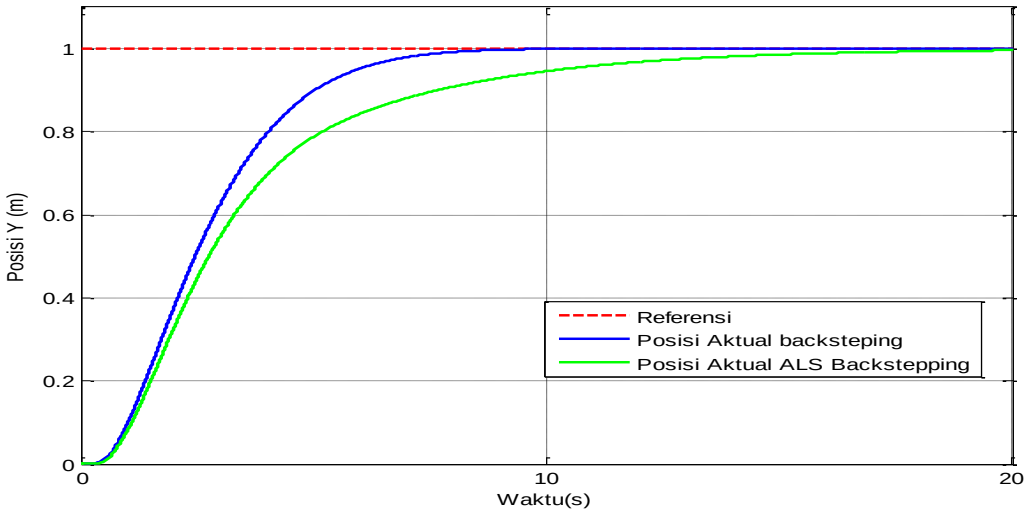
Tabel 4.1 Respon *step* menggunakan *backstepping control*

Axis	***Rise Time***	***Setling time***
X	6.3192	8.6317
Y	5.1630	7.0265
Z	7.2574	9.9136

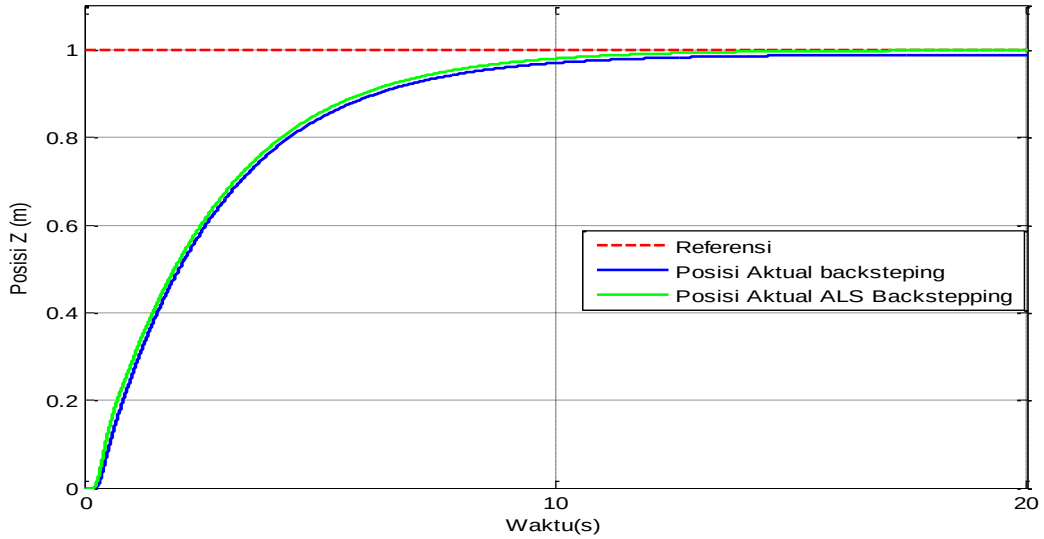
Tabel 4.2 Respon *step* menggunakan *modified least square*

Axis	***Rise Time***	***Setling time***
X	9.5596	13.4067
Y	9.5590	13.4063
Z	7.2550	9.8418

Hasil dari respon pergerakan quadrotor pada sumbu Y menggunakan metode *modified least square* memiliki kemampuan lebih lambat dibandingkan dengan *backstepping* konvensional terlihat pada Gambar 4.2. Respon *step* sumbu X pada *backstepping* memiliki performa lebih cepat dibandingkan dengan respon *step* pada sumbu X seperti yang ditunjukkan pada Tabel 4.1 dan 4.2.

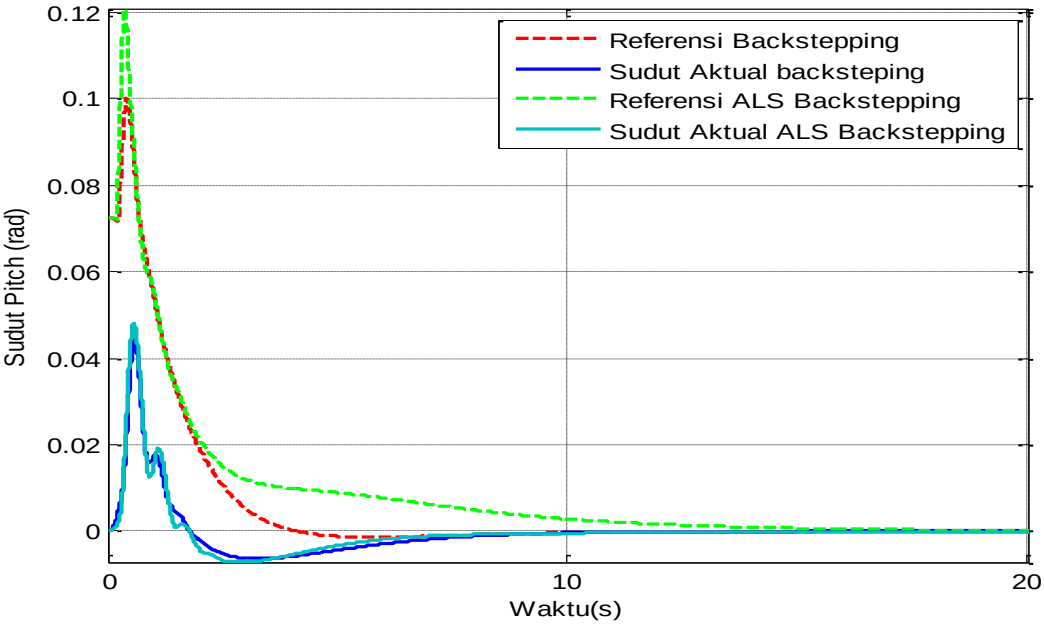


Gambar 4.2 Posisi quadrotor pada sumbu Y saat *fault free case*



Gambar 4.3 Posisi quadrotor pada sumbu Z saat *fault free case*

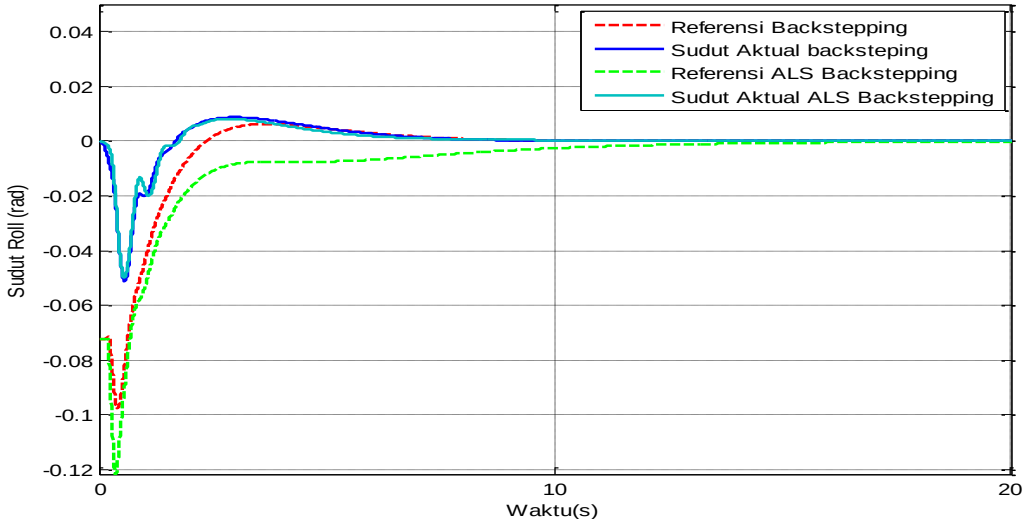
Performa sumbu Z atau sumbu vertikal dari kedua metode memiliki respon *step* yang hampir sama, dimana terlihat pada Gambar 4.3 bahwa kedua kontroler bergerak menuju *setpoint* dengan respon yang mendekati sama, namun pergerakan dengan metode yang diusulkan sedikit lebih cepat, seperti yang ditunjukan pada Tabel 4.1 dan 4.2.



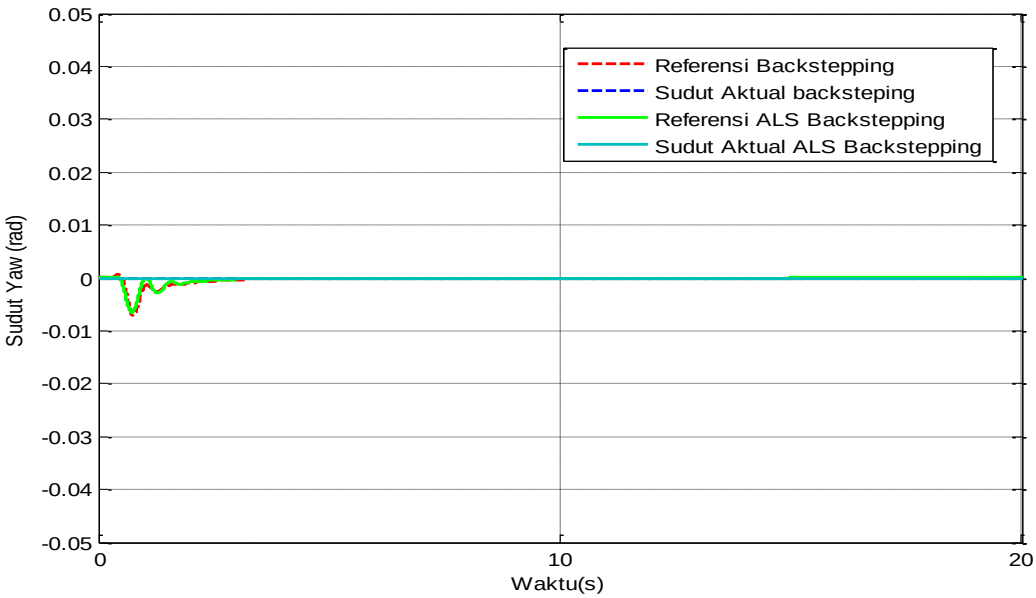
Gambar 4.4 Sudut *pitch* saat *fault free case*

Pergerakan sudut *pitch* sedikit lebih cepat dengan menggunakan metode *modified least square* daripada menggunakan metode *backstepping* konvensional, terlihat pada Gambar 4.4. Hal yang sama berlaku pada pergerakan sudut *roll*.

Ketika awal pergerakan quadrotor, respon mengalami osilasi dikarenakan sistem sedang berusaha mencari nilai *gain η* terbaik sehingga terjadi dinamika *gain η* dengan nilai besar dan kecil seperti yang ditunjukan pada Gambar 4.5. Tidak terjadi perubahan yang besar pada sumbu *yaw*, perubahan tersebut maksimal hanya sebesar 0.02 radian atau sebesar 1.15^{o} yang ditunjukkan pada Gambar 4.6.



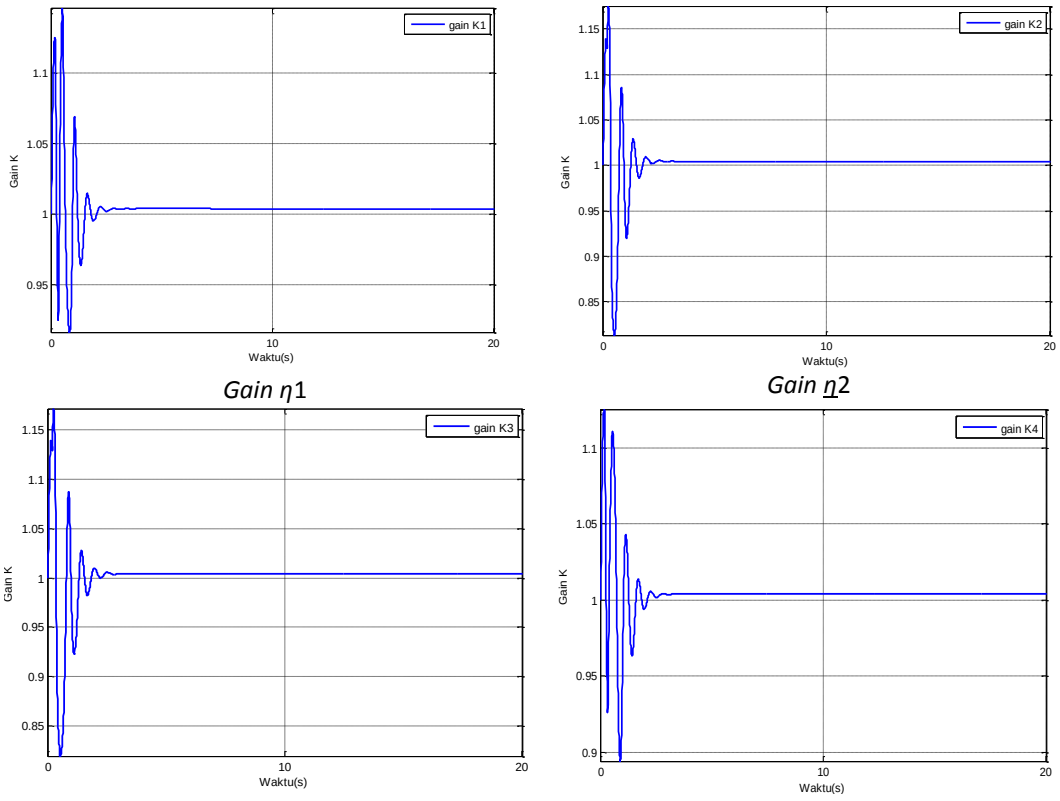
Gambar 4.5 Sudut *roll* saat *fault free case*



Gambar 4.6 Sudut *yaw/heading* saat *fault free case*

Besar *gain η* quadrotor pada awal dimulainya simulasi memiliki nilai yang tidak stabil atau mengalami osilasi seperti yang ditunjukkan pada Gambar 4.7, hal ini dikarenakan ketika sistem mulai berjalan perbedaan antara sinyal kontrol dan

output sangat besar. Sehingga sistem kontrol *least square* akan berusaha untuk mengatasi perbedaan tersebut.



Gain η1 *Gain η2*

Gain η3 *Gain η4*

Gambar 4.7 *Gain* saat *fault free case* dengan metode *modified least square*

4.1.2 Pengujian *Waypoint* dengan *Backstepping Control*

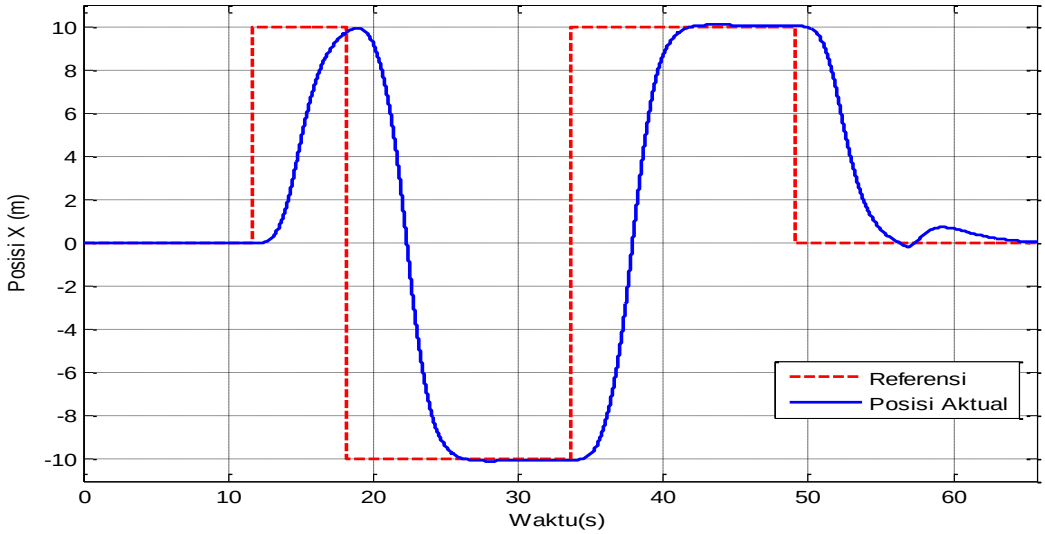
Quadrotor ditugaskan untuk menjejaki titik referensi yang berupa *waypoint*. Terdapat 3 *waypoint* yang terdiri dari titik pada sumbu X, Y, dan Z (ketinggian) yang harus dijejaki oleh quadrotor. Quadrotor dapat bergerak ke *waypoint* berikutnya jika quadrotor telah berada pada radius 0.2 meter dari ketiga sumbu X,Y dan Z. Jika quadrotor sudah mencapai titik referensi pada satu sumbu sedangkan titik referensi pada sumbu lain belum tercapai, maka quadrotor harus menjaga posisi pada salah satu sumbu tersebut hingga ketiga titik referensi pada sumbu terpenuhi. Titik referensi *waypoint* yang diberikan pada penelitian ini ditunjukan pada Tabel 4.3. Hal inilah yang menyebabkan tidak dapat dilakukan perbandingan pergerakan quadrotor dengan menggunakan *backstepping* dan

modified least square pada 1 grafik, karena akan terjadi perbedaan titik referensi yang disebabkan oleh salah satu sistem telah mencapai titik referensi terlebih dahulu maka quadrotor tersebut akan melanjutkan ke tititk referensi selanjutnya sehingga metode yang lainnya akan tertinggal untuk tetap melakukan *tracking* pada referensi sebelumnya.

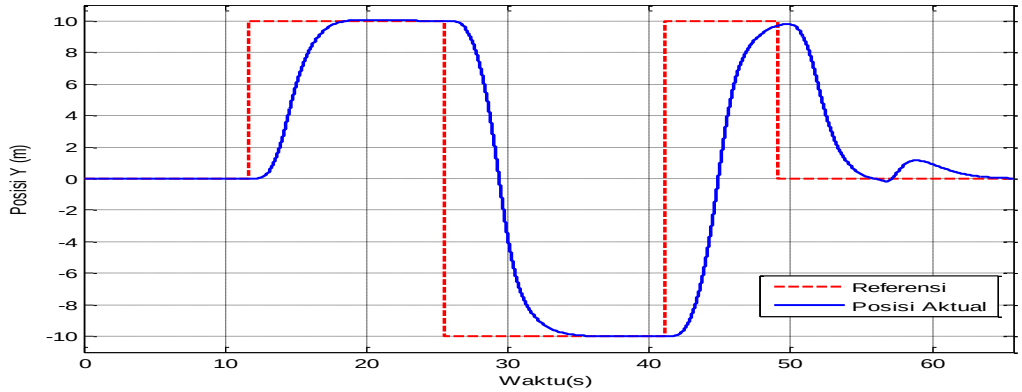
Tabel 4.3 Titik referensi *waypoint* yang harus dijejaki quadrotor

Waypoint	**Sumbu X(m)**	**Sumbu Y(m)**	**Sumbu Z(m)**
#1	0	0	30
#2	10	10	30
#3	-10	10	30
#4	-10	-10	30
#5	10	-10	30
#6	10	10	30
#7	0	0	30
#8	0	0	0

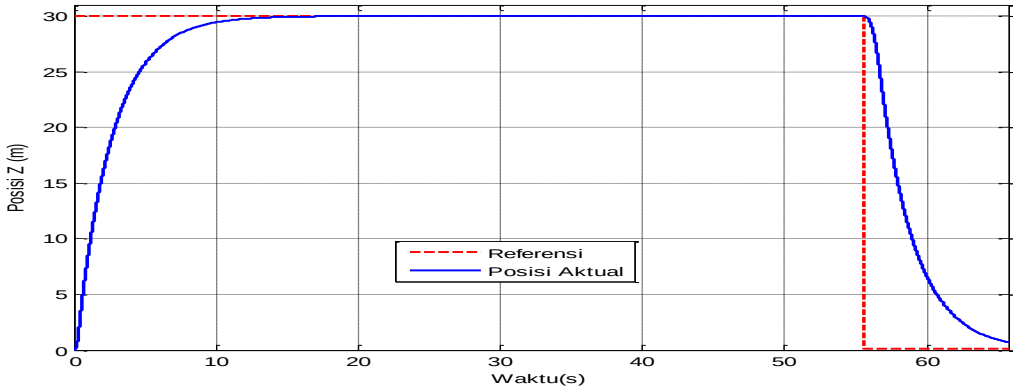
Hasil pengujian *waypoint* dengan *backstepping control* ditunjukan pada Gambar 4.8, Gambar 4.9 dan Gambar 4.10. Quadrotor mampu melakukan *waypoint tracking* dengan baik. Sehingga dengan menggunakan metode kontrol *backstepping*, quadrotor mampu bergerak dengan baik pada sumbu X, sumbu Y dan sumbu Z menuju *waypoint* yang diberikan ketika keadaan tidak terjadi *fault* pada aktuator.



Gambar 4.8 Posisi quadrotor pada sumbu X saat *waypoint tracking* saat *fault free*

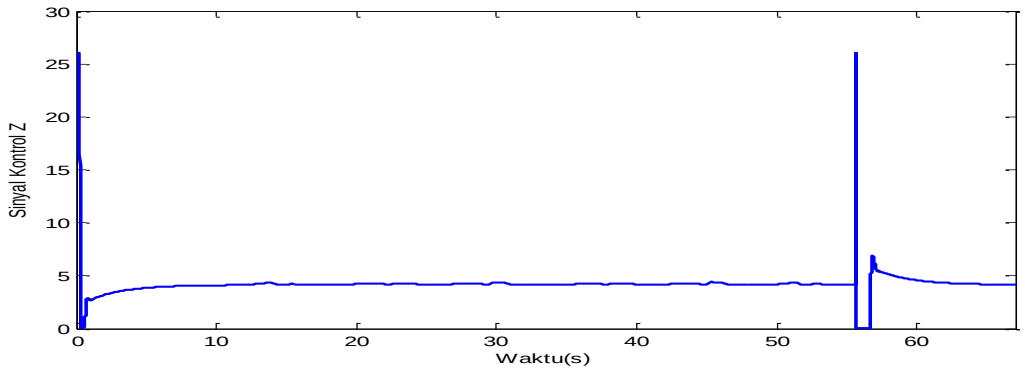


Gambar 4.9 Posisi quadrotor pada sumbu Y saat *waypoint tracking* saat *fault free*



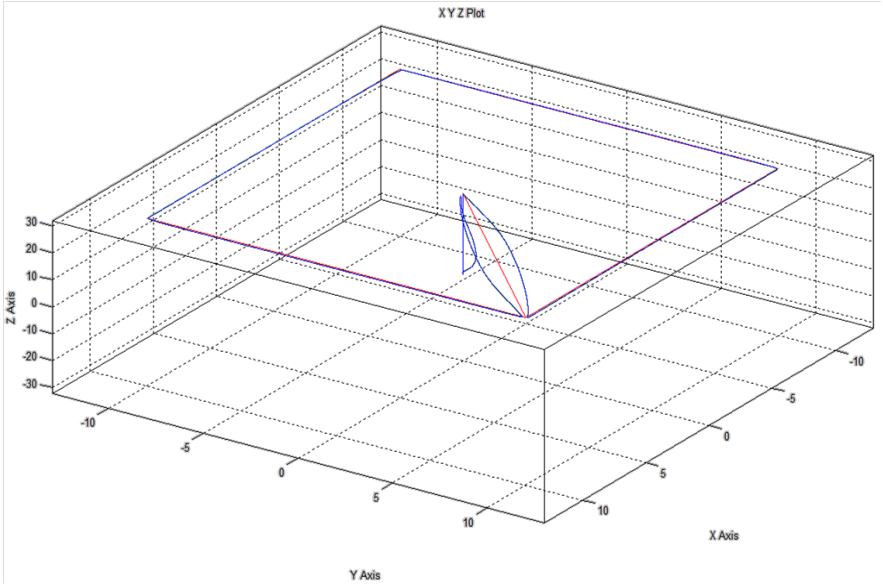
Gambar 4.10 Posisi quadrotor pada sumbu Z saat *waypoint tracking* saat *fault free*

Dari gambar pergerakan X dan Y, menunjukan bahwa sistem pergerakan quadrotor memiliki keterkaitan satu dengan yang lainnya, seperti yang ditunjukan pada Gambar 4.8 dan Gambar 4.9 pada detik ke 57-67, dimana pada rentang waktu tersebut quadrotor mengalami osilasi kemudian kembali ke titik 0. Hal ini dikarenakan oleh pergerakan sumbu Z untuk melakukan pergerakan turun ke bawah, yang dalam hal tersebut quadrotor harus menurunkan sinyal kontrolnya beberapa detik untuk turun, sinyal kontrol sumbu Z ditunjukan pada Gambar 4.11



Gambar 4.11 Sinyal kontrol Sumbu Z

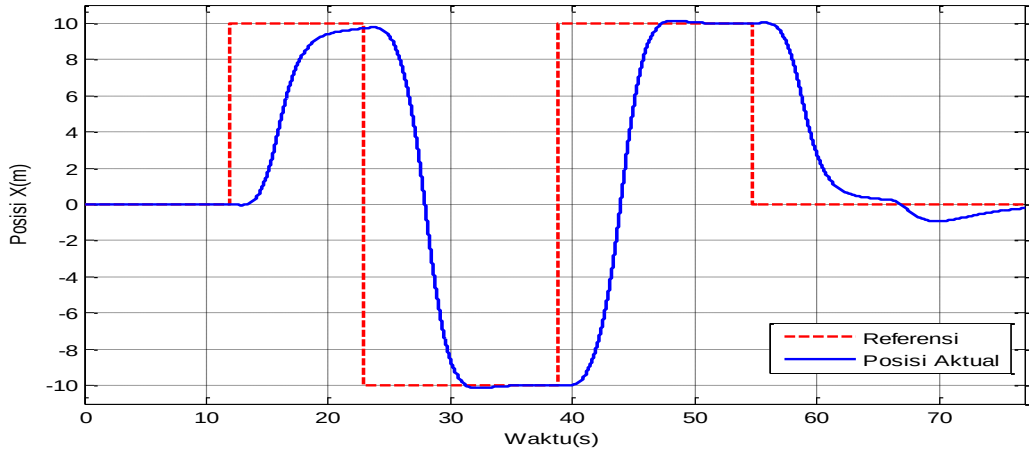
Grafik 3 dimensi pada Gambar 4.12 digunakan untuk melihat pergerakan X, Y dan Z quadrotor dalam melakukan *waypoint tracking*. Dari grafik tersebut terlihat pergerakan quadrotor sesuai dengan *waypoint*, hanya saja terlihat cembung pada sisi diagonal, dikarenakan oleh lebih cepatnya pergerakan pada satu sumbu dibanding sumbu lainnya. Pada *waypoint tracking*, penjejakan tidak harus melalui lintasan, yang terpenting adalah quadrotor telah mencapai *waypoint* atau tidak.



Gambar 4.12 Pergerakan quadrotor pada sumbu X,Y,Z

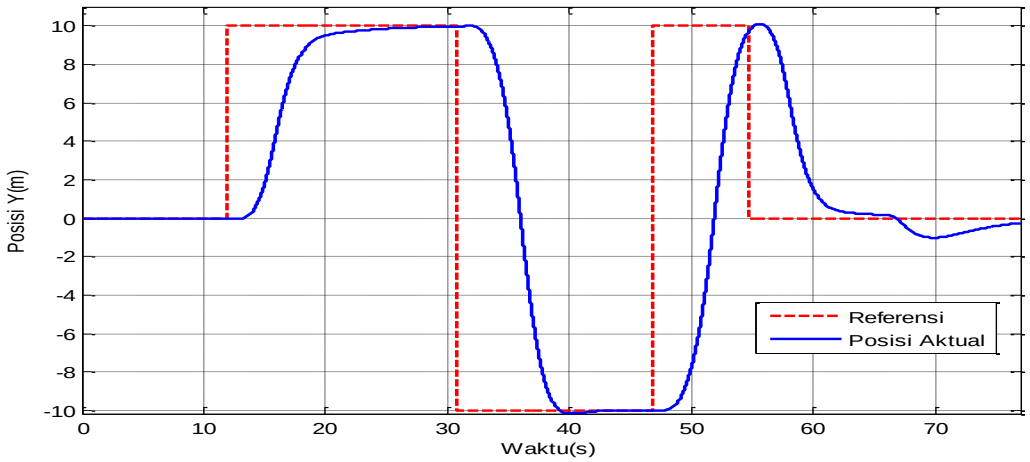
4.1.3 Pengujian *Waypoint* dengan *Modified Least Square*

Pengujian dilakukan untuk melihat performansi quadrotor menggunakan metode yang diajukan untuk melihat apakah quadrotor dapat terbang dengan baik ketika ditugaskan untuk *waypoint tracking* yang ditunjukan pada Tabel 4.3 dengan menggunakan metode kontrol yang diusulkan.

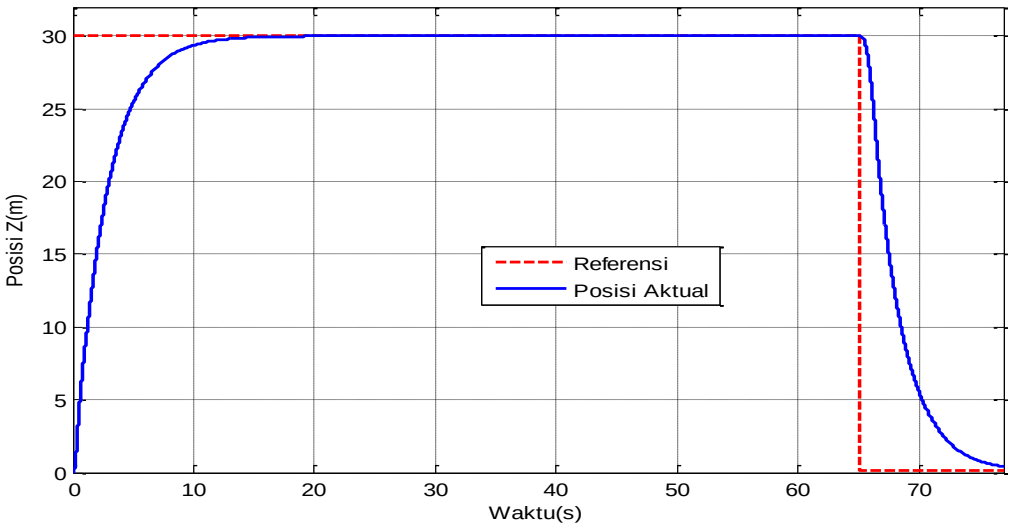


Gambar 4.13 Posisi quadrotor pada sumbu X saat *waypoint tracking* (*fault free)*

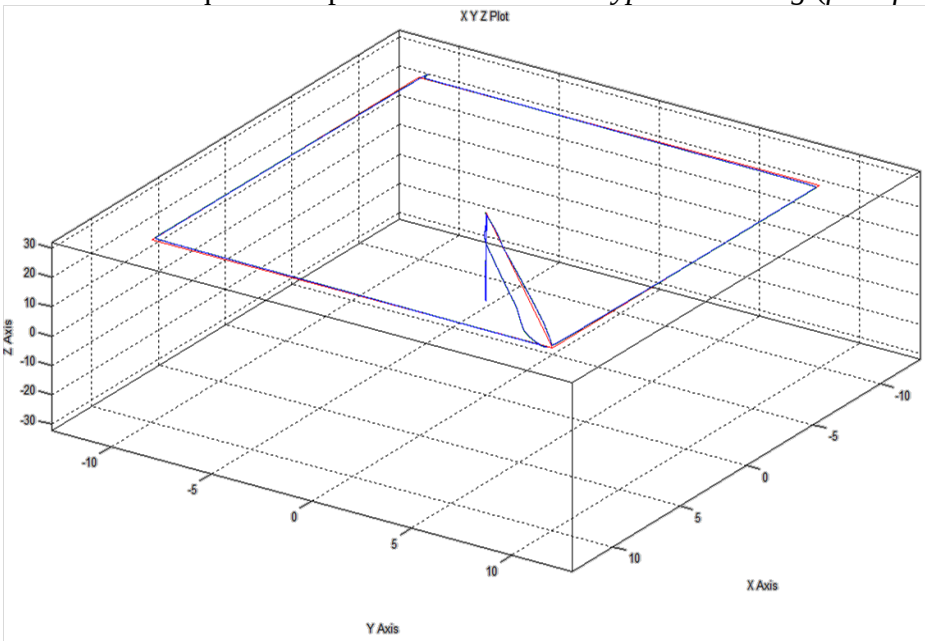
Pergerakan quadrotor ketika melakukan *waypoint tracking* keadaan tanpa terjadi kesalahan aktuator (*fault free case*) ditunjukkan pada Gambar 4.13-4.15. Terlihat quadrotor dapat melakukan *tracking* dengan baik. Namun dari ketiga gambar tersebut dapat dilihat bahwa pergerakan sumbu X, Y dan Z saling mempengaruhi. Sebagai contoh, ketika detik 67-77, quadrotor sedang melakukan *tracking* sumbu Z, sedangkan posisi pada sumbu X dan Y terjadi osilasi yang kemudian kembali lagi ketitik referensinya.Hal ini menunjukan bahwa pergerakan quadrotor pada tiap-tiap sumbu mempengaruhi pergerakan sumbu lainnya yang diakibatkan oleh efek *couple* atau saling terkaitnya dinamika sistem quadrotor. Hal ini menunjukan bahwa metode yang diusulkan belum mampu menghilangkan efek *couple* antar sistem.



Gambar 4.14 Posisi quadrotor pada sumbu Y saat *waypoint tracking* (*fault free)*

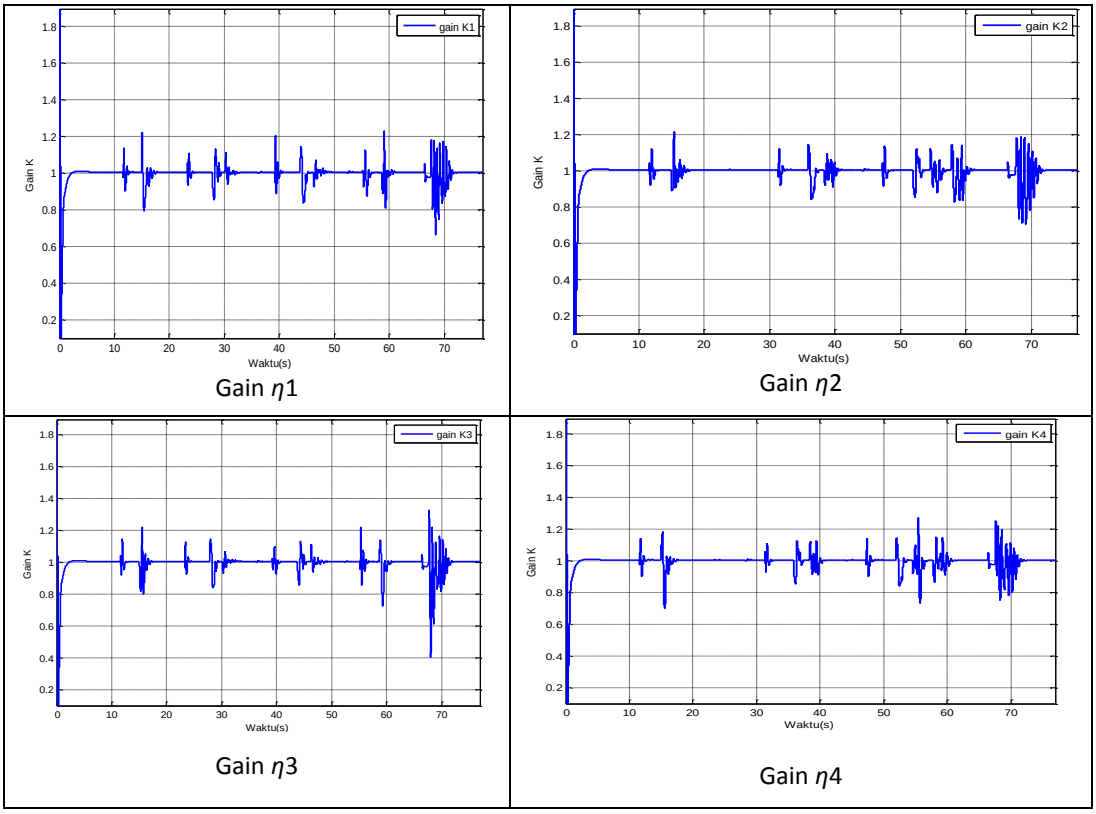


Gambar 4.15 Posisi quadrotor pada sumbu Z saat *waypoint tracking* (*fault free)*



Gambar 4.16 Pergerakan Quadrotor pada sumbu X,Y,Z

Nilai parameter *gain η* yang digunakan pada kontrol *assembler* terlihat pada Gambar 4.17 mengalami perubahan. *Gain* berosilasi dengan cepat dengan titik pusat osilasi adalah titik 1. Namun jika dilihat dari pergerakan X, Y dan Z pada Gambar 4.13-4.15 menunjukan bahwa sistem tetap bergerak dengan baik. Hal ini menunjukan bahwa parameter *gain η* yang diatur dengan metode *modified least square* tetap membuat sistem melakukan *tracking waypoint* dengan baik.



Gambar 4.17 Perubahan nilai *Gain η* saat *waypoint tracking* (*fault free case*)

4.2 Pengujian Kesalahan pada Aktuator (*Faulty Case*)

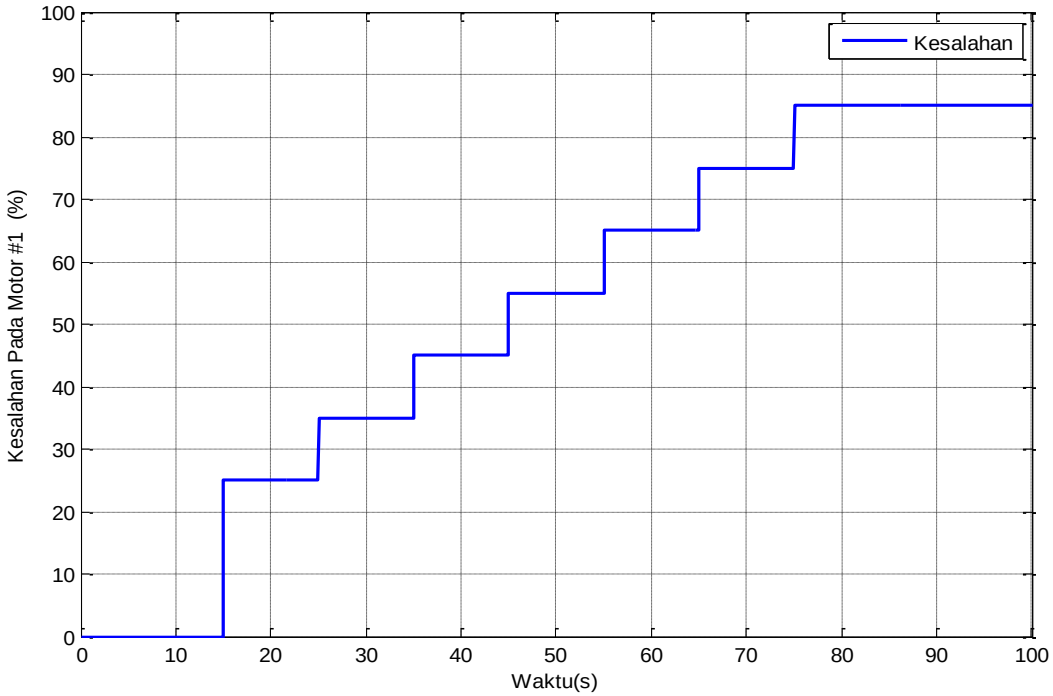
Pengujian ini bertujuan untuk melihat respon kestabilan quadrotor ketika terjadi kesalahan pada aktuator motor. Pengujian dilakukan dengan memberikan sinyal referensi *step* dan titik referensi *waypoint*.

4.2.1 Pengujian Kesalahan Aktuator

Pengujian ini dilakukan dengan menggunakan metode *backstepping control* dan *modified least square* untuk mengendalikan quadrotor mengikuti titik referensi berupa sinyal *step* kemudian *hover* pada titik tersebut. Hal ini bertujuan untuk melihat respon sistem ketika keadaan stabil pada saat kondisi *steady state* kemudian terjadi kesalahan pada salah satu motor dengan *fault* yang berubah. Sehingga dapat mengetahui berapa batas maksimum kesalahan yang mampu diatasi oleh salah satu metode.

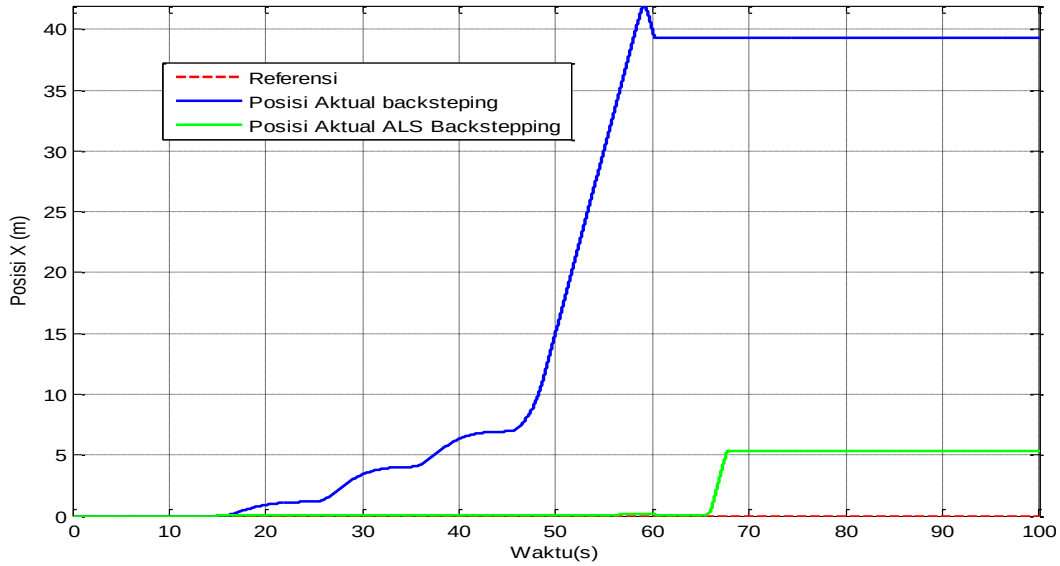
Tabel 4.4 Skenario kesalahan pengujian sinyal *step* pada motor #1

No	Besar Kesalahan (*Fault*)	Waktu mulai
1	25 %	detik ke 25
2	35 %	detik ke 35
3	45 %	detik ke 45
4	55%	detik ke 55
5	65%	detik ke 65
6	75%	detik ke75
7	85%	detik ke 85

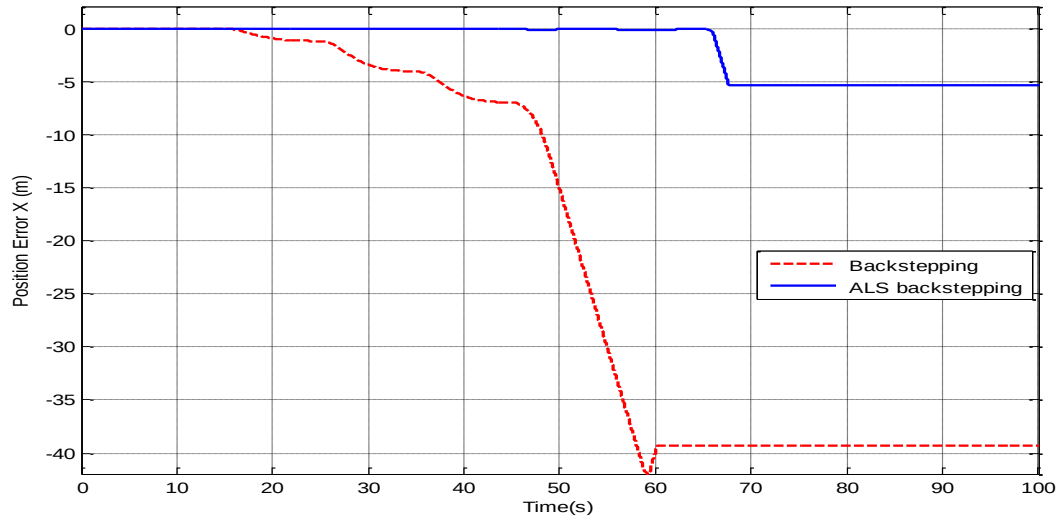


Gambar 4.18 Skenario kesalahan pada motor #1

Pada Gambar 4.18 terlihat skenario perubahan kesalahan (*fault)* yang terjadi pada aktuator #1, dimana pada penelitian ini kesalahan adalah berkurangnya besar gaya dorong (*thrust*) dalam persentase atau disebut juga *multiplicative fault.* Pada gambar diatas terlihat kesalahan dimulai dari 25% pada detik ke 15-25, kemudian kesalahan meningkat menjadi 35% pada detik ke 25-35 dan nilai kesalahan meningkat setiap 10 detik, kesalahan maksimum pada pengujian ini adalah 85%. Pengujian ini berfungsi untuk melihat seberapa besar kesalahan yang mampu diatasi oleh kontroler yang digunakan.



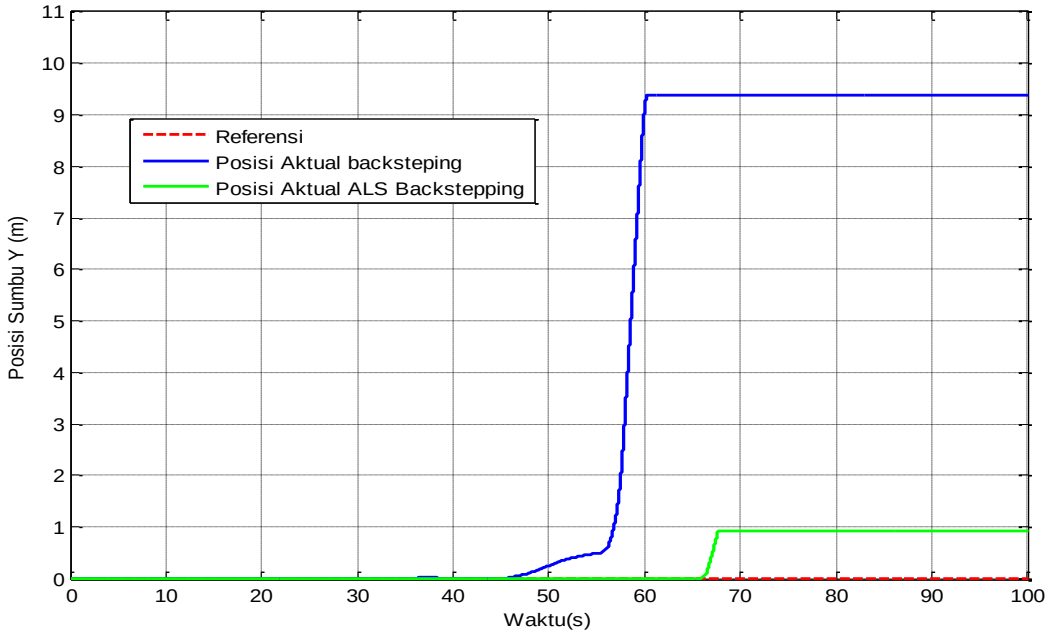
Gambar 4.19 Pergerakan Sumbu X saat terjadi *fault*



Gambar 4.20 *Error* posisi Sumbu X saat terjadi *fault*

Perbedaan yang signifikan antara metode *backstepping* konvensional dengan metode *modified least square* ditunjukkan pada Gambar 4.19. Ketika terjadi kesalahan sebesar 25% pada detik ke 15, pada kontroler *backstepping* terjadi pergeseran sebesar 2 meter dari referensi, pergeseran ini terus meningkat ketika nilai kesalahan meningkat, hingga pada detik ke 45, dimana nilai kesalahan bernilai 55% quadrotor bergerak semakin menjauhi referensi hingga 40 meter, sedangkan pada kontroler *modified least square* quadrotor masih mampu mengikuti referensi yang diberikan dengan baik, hingga pada detik ke 55 dengan

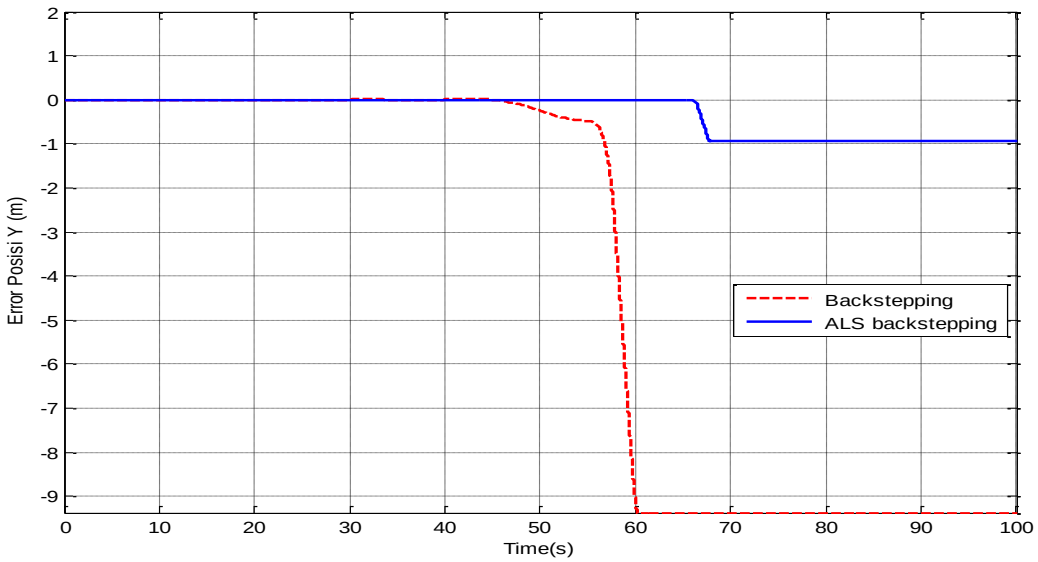
nilai kesalahan sebesar 65% quadrotor terjadi deviasi sebesar 5 meter dari titik referensi dan berhenti dikarenakan quadrotor terjatuh. Nilai *error* ditunjukan pada Gambar 4.20, pada gambar tersebut terlihat selisih antara posisi referensi dengan posisi aktual pada masing-masing kontroler, dimana kontroler *backstepping* menunjukan selisih 2 meter dan meningkat dimulai dari 25% dan kontroler *modified least square* dimulai dari 65%.



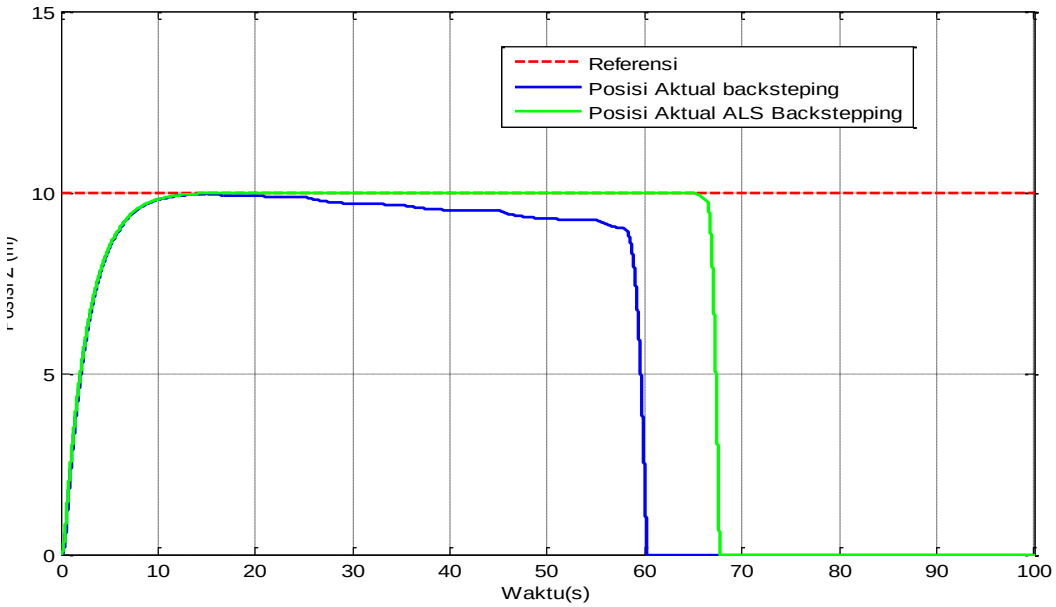
Gambar 4.21 Pergerakan Sumbu Y saat terjadi *fault*

Pada Gambar 4.21 terlihat quadrotor dengan menggunakan metode *backstepping* konvensional masih mampu tetap berada pada titik referensi dan tidak terjadi pergeseran, hal ini dapat dilihat pada gambar *error* dimana error menunjukan nilai 0 pada detik 0-45 seperti yang ditunjukan pada Gambar 4.22 namun kemudian terjadi pergeseran sebesar 0.5 meter dan meningkat hingga 10 meter dan berhenti yang diakibatkan oleh jatuh, dapat dijelaskan yaitu ketika kesalahan terjadi pada aktuator motor #1 atau aktuator motor depan, maka pergerakan akibat efek kesalahan tersebut akan berpengaruh besar pada pergerakan sumbu X, namun hal ini akan mempengaruhi pergerakan pada sumbu Y karena pergerakan *heading* berubah seperti yang ditunjukan pada Gambar 4.23. Oleh karena itu, sumbu X dan Y akan terpengaruhi. Dengan menggunakan metode

modified least square pergerakan quadrotor dapat kembali ketitik referensinya atau *error* menuju 0 kecuali pada detik ke 65 atau ketika terjadi kesalahan sebesar 65% dimana kontroler sudah tidak dapat mengatasi *fault*.



Gambar 4.22 *Error* posisi Sumbu Y saat terjadi *fault*

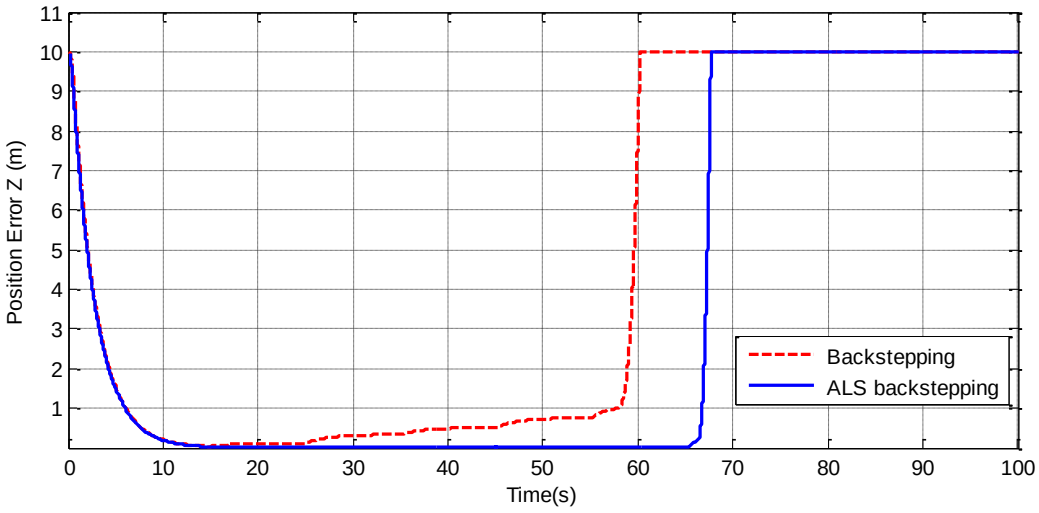


Gambar 4.23 Pergerakan Sumbu Z saat terjadi *fault*

Pada pergerakan sumbu Z menggunakan metode *bacstepping* dan *modified least square* yang ditunjukkan pada Gambar 4.23. Ketika menggunakan kontroler *backstepping*, quadrotor bergerak turun secara perlahan ketika terjadi kesalahan sebesar 25% pada detik ke 15 hingga kemudian jatuh bebas pada detik ke 55 yaitu

ketika kesalahan sebesar 55%. Sedangkan pada kontroler *modified least square*, tidak terjadi pergerakan turun yang diakibatkan oleh penambahan nilai kesalahan, kecuali pada detik ke 65 yaitu ketika kesalahan bernilai 65%.

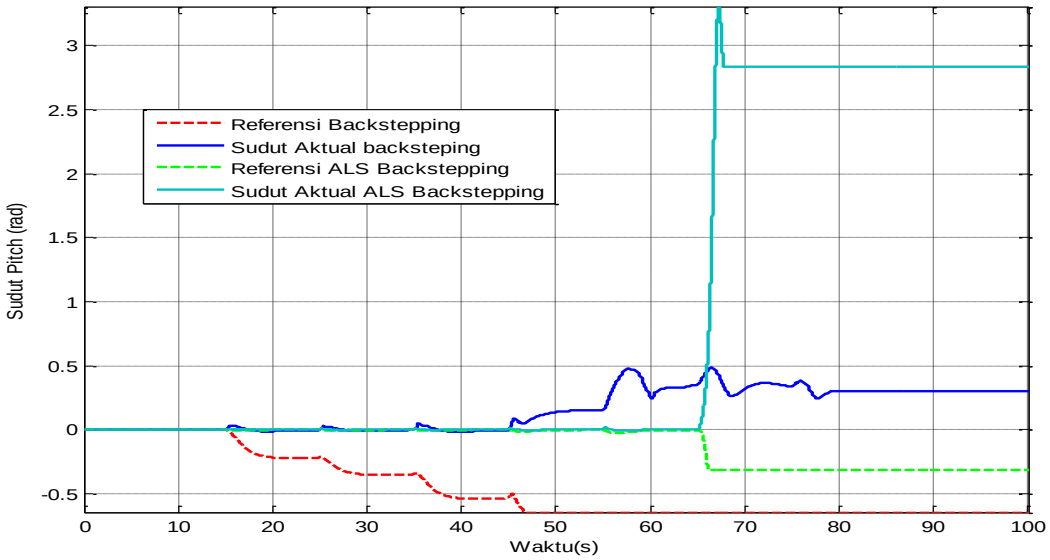
Dari ketiga pergerakan translasi diatas diperoleh bahwa ketika terjadi *fault* dibawah 55%, dengan menggunakan kontroler *backstepping* quadrotor masih tetap mampu terbang pada ketinggian tertentu, namun pergerakan quadrotor pada sumbu X dan Y tidak dapat dikendalikan untuk mengikuti referensi. Solusi untuk permasalahan ini ialah ketika terjadi kesalahan dibawah 55% sebaiknya quadrotor segera di turunkan, karena quadrotor tidak dapat melakukan misi dengan baik. Berbeda dengan menggunakan kontroler *modified least square* yaitu ketika terjadi kesalahan dibawah 65%, quadrotor masih dapat tetap terbang dan melakukan *waypoint tracking* dengan baik, hal ini ditunjukan pada Gambar 4.24 dimana pada gambar tersebut menunjukan selisih error antara referensi dengan posisi aktual. Namun ketika terjadi *fault* sebesar 65% quadrotor akan terjatuh bebas dikarenakan kontroler tidak mampu mengatasi *fault* karena keterbatasan kemampuan motor.



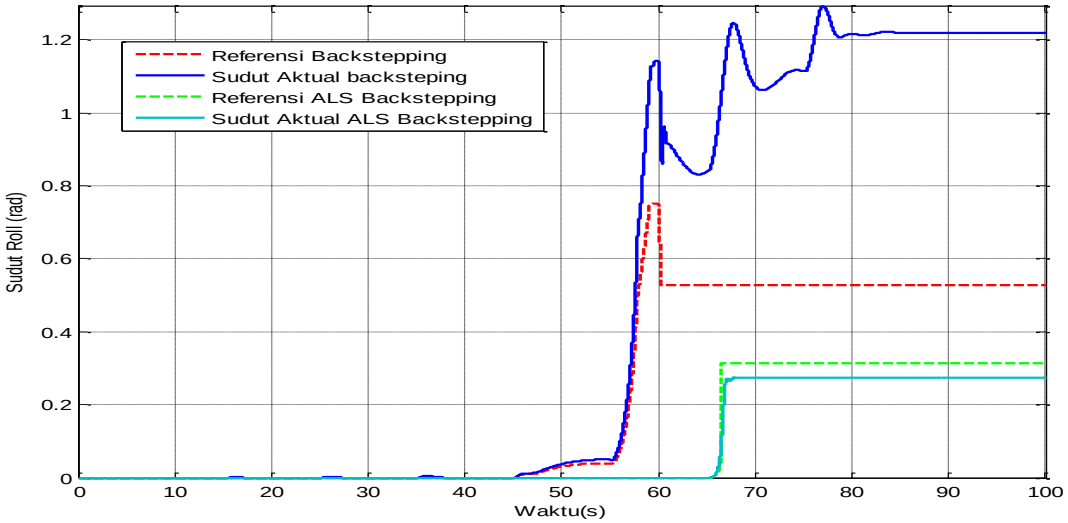
Gambar 4.24 Error Posisi Sumbu Z saat terjadi *fault*

Respon pergerakan pada sumbu *pitch* quadrotor terjadi perubahan menjauh dari *setpoint* pada detik ke 65 dengan menggunakan *modified least square* yang ditunjukkan pada Gambar 4.25. Dari gambar pergerakan sumbu *roll* menunjukan banyak perbedaan antara *backstepping* konvensional dan *modified least square*. Terlihat pada Gambar 4.26, sinyal referensi sudut *roll* yang merupakan keluaran

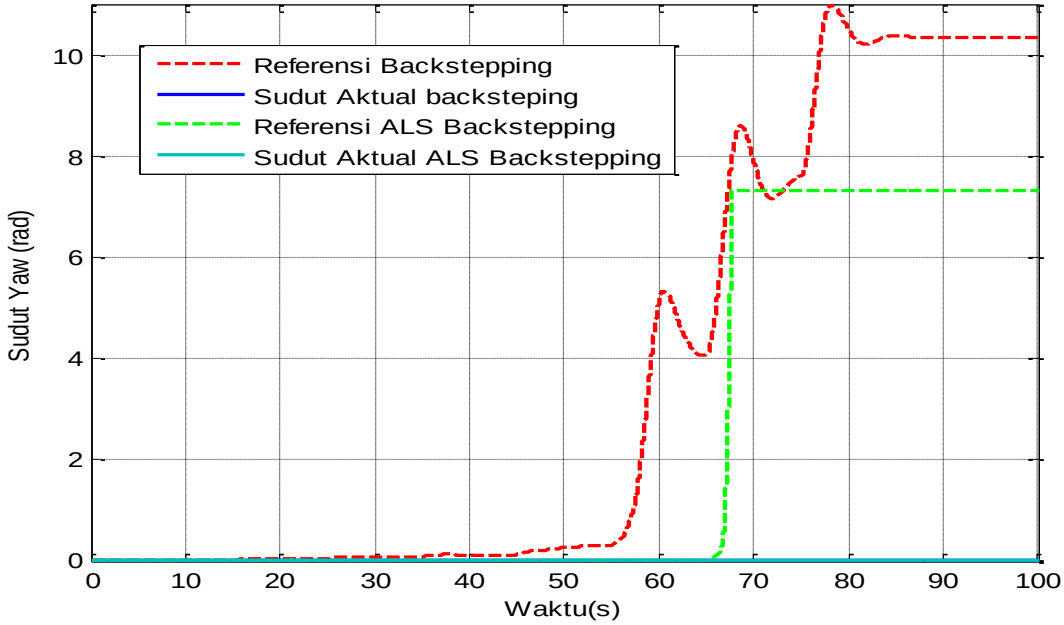
kontroler Y *backstepping* tidak dapat diikuti dengan baik oleh kontroler *roll backstepping* sehingga mengakibatkan keterlambatan pada sumbu Y dengan menggunakan metode *backstepping*. Sedangkan pada metode *modified least square* tidak terjadi deviasi yang ditunjukkan pada Gambar 4.27. Sehingga menunjukan kedua metode tetap mampu menjaga sumbu *yaw* tetap pada referensi meskipun terjadi *fault*.



Gambar 4.25 Pergerakan Sumbu *Pitch* saat terjadi *fault*

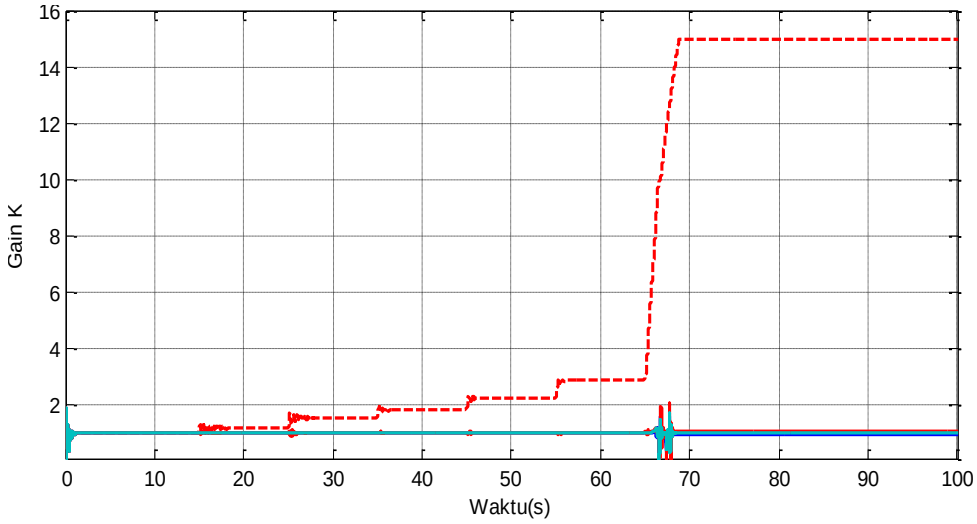


Gambar 4.26 Pergerakan Sumbu *Roll* saat terjadi *fault*



Gambar 4.27 Pergerakan Sumbu *Yaw* saat terjadi *fault*

Perubahan nilai *gain η* yang diatur oleh *modified least square* saat *tracking* sinyal *step* ditunjukan pada Gambar 4.28. Pada gambar tersebut terlihat *gain η* yang merupakan *gain η* #1 memiliki nilai yang berubah sesuai dengan keadaan *fault*. Perlahan *gain η* berubah dan hasil yang diperoleh quadrotor dapat tetap melakukan *tracking* dengan baik, namun ketika terjadi kesalahan diluar kemampuan aktuator *gain η* berubah tinggi, namun hal ini tetap tidak dapat mengatasi kesalahan yang terjadi.



Gambar 4.28 Perubahan Parameter *gain η modified least square* saat terjadi *fault*

4.2.2 Pengujian *Waypoint Tracking* dengan Kesalahan Aktuator menggunakan Metode *Backstepping Control*

Pada Subbab ini membahas tentang pengujian saat kesalahan atau *fault* pada aktuator terjadi. Masing-masing kesalahan aktuator akan dimodelkan ke dalam bentuk persamaan matematika sebagai *multiplicative fault*. Kemudian kesalahan ini dikalikan pada keluaran aktuator motor dan propeller (rotor) yang merupakan gaya *thrust* atau gaya dorong sebagai representasi *fault* aktuator. Hasil estimasi *fault* aktuator digunakan untuk mengatasi *fault* aktuator yang terjadi. Untuk menguji sistem kontrol yang diusulkan, pada penelitian ini akan diuji dengan skenario pada dua motor (depan dan kanan). Skema posisi tiap-tiap motor dapat dilihat pada Bab 2 Gambar 2.11.

Untuk menguji sistem kontrol *backstepping control* konvensional, metode ini akan diuji dengan skenario kesalahan pada motor yang ditentukan pada Tabel 4.5, berdasarkan pengujian sinyal *step* diperoleh maksimum kesalahan yang dapat diatasi oleh kontroler *backstepping* adalah 55%, sehingga pada pengujian pada kasus *waypoint tracking* ini, kesalahan maksimum adalah 55% dan ketika sudah mencapai kesalahan maksimum, quadrotor diharuskan turun pada ketinggian Z=0.

Tabel 4.5 Skenario kesalahan aktuator pada pengujian *waypoint tracking* dengan kontroler *backstepping*

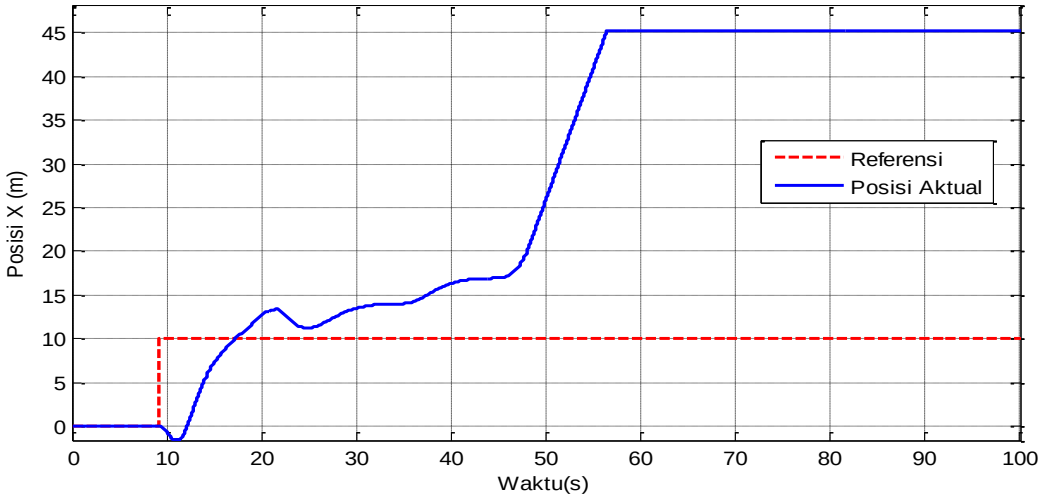
No	Besar Kesalahan (*Fault*)	Waktu mulai
1	25 %	detik ke 15
2	35 %	detik ke 25
3	45 %	detik ke 35
4	55 %	detik ke 45

Tabel 4.6 Daftar *waypoint* yang harus di jejaki quadrotor

Waypoint	Sumbu X(m)	Sumbu Y(m)	Sumbu Z(m)
#1	0	0	10
#2	10	10	10
#3	-10	10	10
#4	-10	-10	10
#5	10	-10	10
#6	10	10	10
#7	0	0	10
#8	0	0	0

Tabel 4.6 merupakan daftar *waypoint* yang harus dijejaki quadrotor, namun keadaan tersebut akan beralih ketika quadrotor mencapai kesalahan maksimum pada detik ke 45, sehingga secara darurat quadrotor harus mendarat. Hal ini bertujuan untuk melihat performa quadrotor ketika mendarat dalam keadaan terjadi kesalahan pada aktuator.

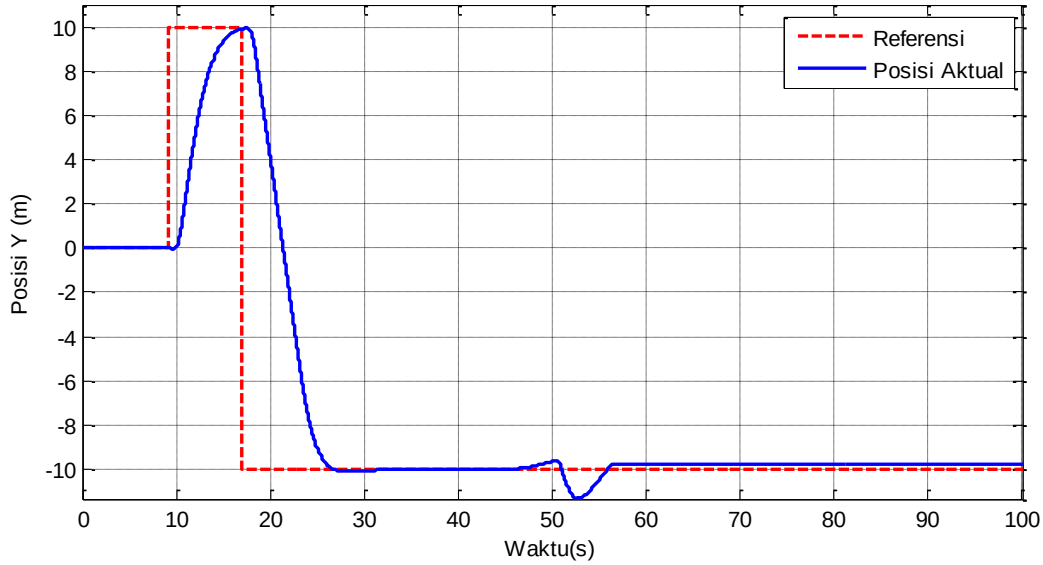
Setelah dilakukan simulasi, diperoleh hasil pergerakan quadrotor seperti yang ditunjukkan pada Gambar 4.29 – 4.34.



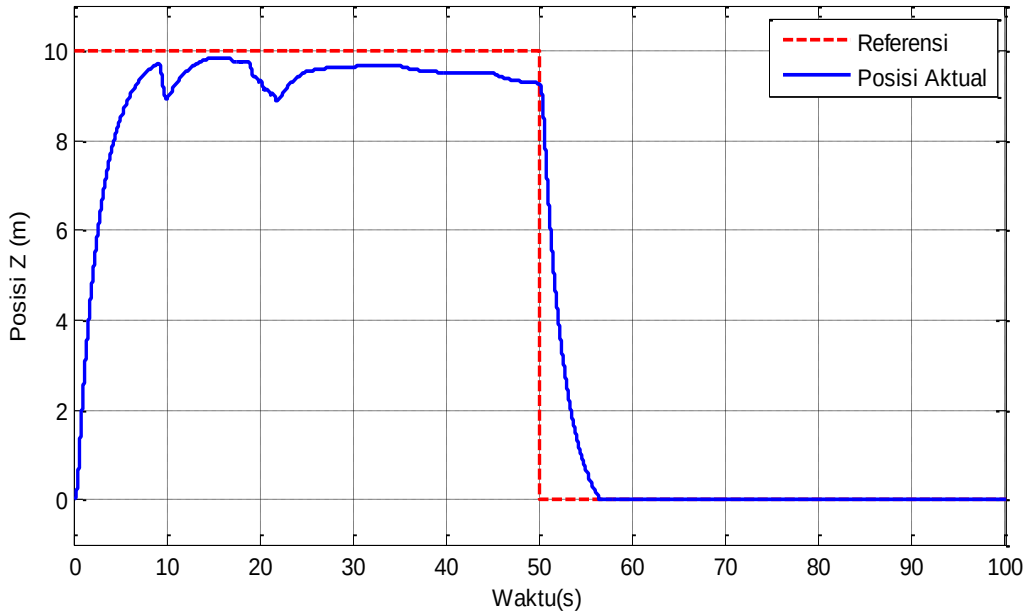
Gambar 4.29 Pergerakan Sumbu X *waypoint tracking* saat *faulty case*

Pada detik ke 20-50 pada Gambar 4.29 terjadi *fault* pada aktuator sebesar 25% terjadi osilasi dengan radius 2 meter untuk pergerakan quadrotor pada sumbu X dengan menggunakan metode *backstepping* konvensional dan ketika *fault* aktuator semakin besar yaitu 45% pada detik 50 yang terjadi adalah quadrotor bergerak semakin menjauhi titik referensi sejauh 35 meter. Hal ini menunjukan bahwa sistem kontrol tersebut sudah tidak mampu menjaga quadrotor tetap stabil. Sedangkan pergerakan quadrotor pada sumbu Y yang ditunjukkan pada Gambar 4.30 menunjukan quadrotor tetap mampu mengikuti titik referensi, hal ini karena kesalahan aktuator #1 yaitu motor depan tidak mempengaruhi dinamika sumbu Y. Pada pergerakan vertikal yang ditunjukan pada Gambar 4.31 menunjukan quadrotor bergerak turun secara perlahan sesuai berkurangnya daya angkat yang diakibatkan oleh kesalahan pada aktuator #1. Kemudian pendaratan darurat dilakukan dan hasil yang diperoleh adalah quadrotor bergerak turun dengan cepat

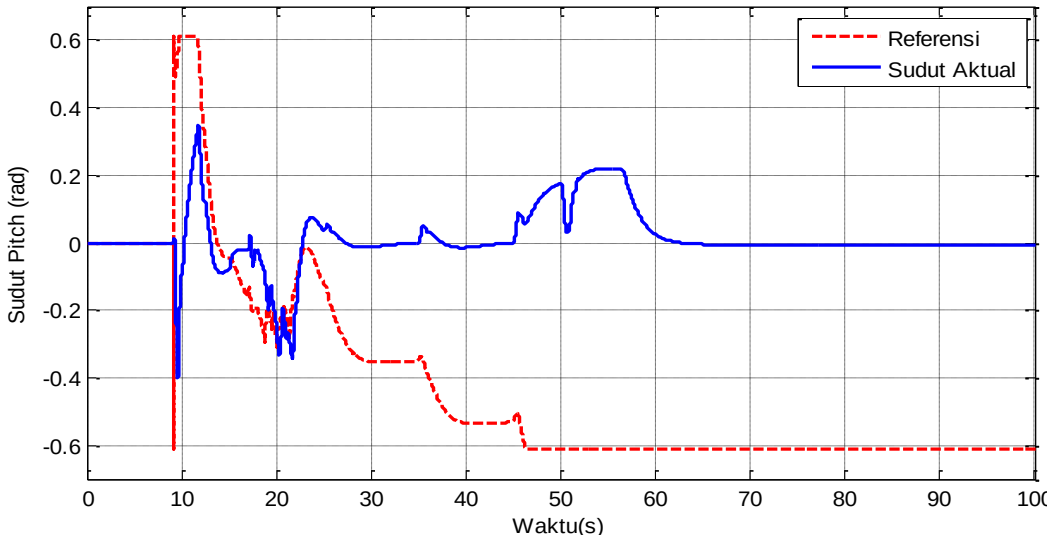
dan terjadi sedikit pendaratan yang keras, hal ini dapat dilihat pada grafik dimana grafik warna biru terpotong ketika menyentuh titik nol atau tanah.



Gambar 4.30 Pergerakan Sumbu Y *waypoint tracking* saat *faulty case*

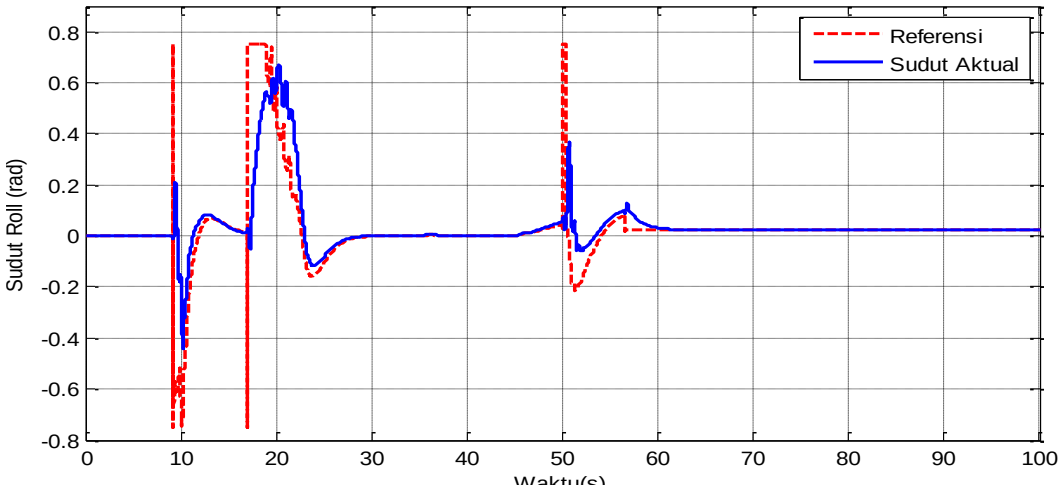


Gambar 4.31 Pergerakan Sumbu Z *waypoint tracking* saat *faulty case*



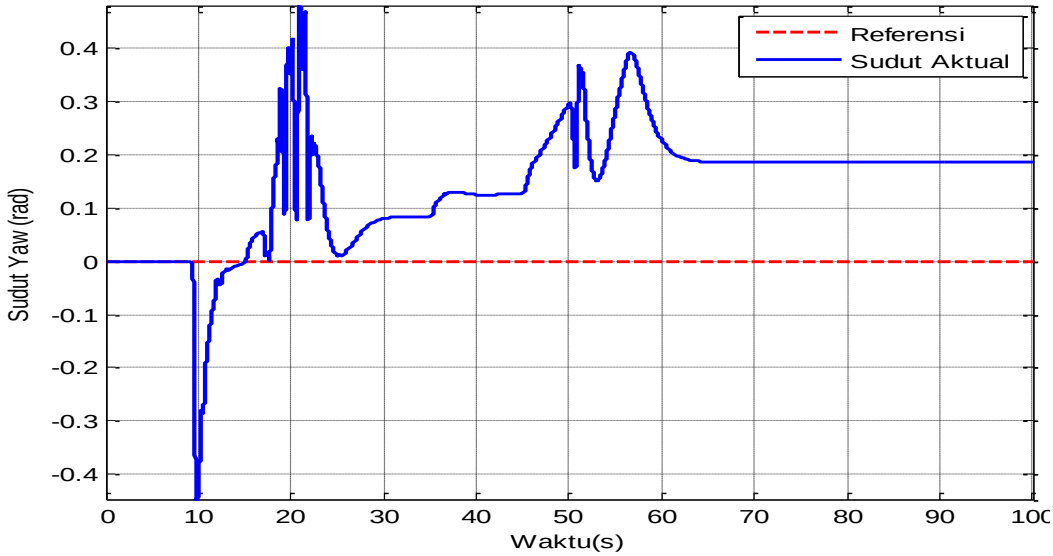
Gambar 4.32 Pergerakan sudut *pitch* saat *faulty case*

Terlihat pada Gambar 4.32 bahwa hingga pada detik kurang dari 25 saat terjadi *fault* pada aktuator sebesar 25%, sistem kontrol rotasi masih bisa mengikuti referensi yang diberikan oleh kontroler sumbu X dan Y, namun ketika terjadi *fault* pada detik ke 25 - 100 sebesar 35% - 45 % sistem kontrol rotasi hanya mampu menjaga sudut rotasi pada 0 radian. Hal ini karena sistem kendali quadrotor tidak dapat melawan efek yang diakibatkan oleh kesalahan pada aktuator, sehingga quadrotor terus bergerak hingga akhirnya jatuh.



Gambar 4.33 Pergerakan sudut rotasi *roll* saat *faulty case*

Pergerakan sudut *roll* dan *yaw* yang ditunjukan pada Gambar 4.33 dan Gambar 4.34. Pada sudut *roll* pergerakan quadrotor tidak terdapat masalah, dikarenakan aktuator #1 tidak mempengaruhi dinamika quadrotor pada sumbu Y, namun pada pergerakan sumbu *Yaw* terlihat sedikit pengaruh sebesar 0.2 radian atau sekitar 11°, hal ini menunjukan bahwa kesalahan motor #1 mempengaruhi pergerakan sumbu *yaw*.



Gambar 4.34 Pergerakan sudut *yaw* saat *faulty*

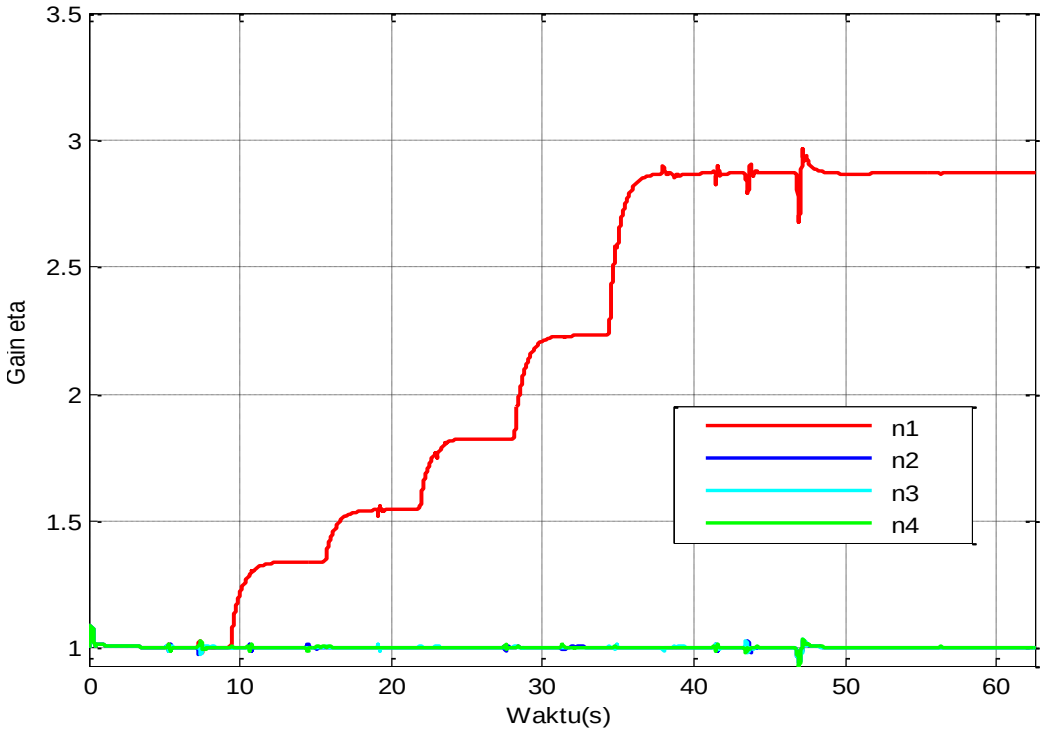
4.2.3 Pengujian *Waypoint Tracking* dengan Kesalahan Aktuator menggunakan metode *Modified Least Square*

Untuk menguji keefektifan metode kontrol yang diusulkan, pada penelitian ini dilakukan pengujian terjadi kesalahan aktuator yang dinyatakan pada Tabel 4.6. Sedangkan *waypoint* yang harus dilalui quadrotor terdapat pada Tabel 4.7.

Tabel 4.7 Skenario kesalahan aktuator pada pengujian *waypoint tracking* dengan kontroler *Modified Least Square*

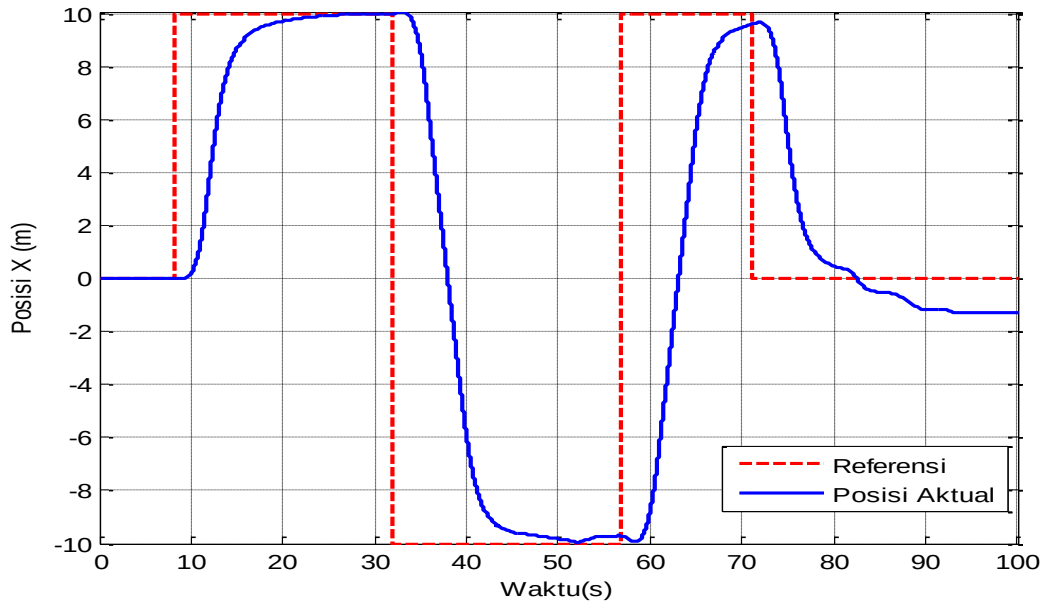
No	Besar Kesalahan (*Fault*)	Waktu mulai
1	25 %	detik ke 15
2	35 %	detik ke 25
3	45 %	detik ke 35
4	55 %	detik ke 45
5	65 %	detik ke 55

Grafik perubahan *gain η*, grafik tersebut dapat dilihat pada Gambar 4.35, terlihat perubahan seiring dengan terjadinya *fault* namun ketika terjadi *fault* terdapat perbedaan. Dari gambar tersebut terlihat hanya *gain η*1 yang memiliki perubahan yang signifikan, yaitu perubahan yang sesuai dengan kesalahan yang terjadi yaitu semakin meningkat. Pada gambar terlihat grafik hingga 3. Hal ini dikarenakan oleh total gaya dorong yang digunakan berbeda ketika melakukan pendaratan.



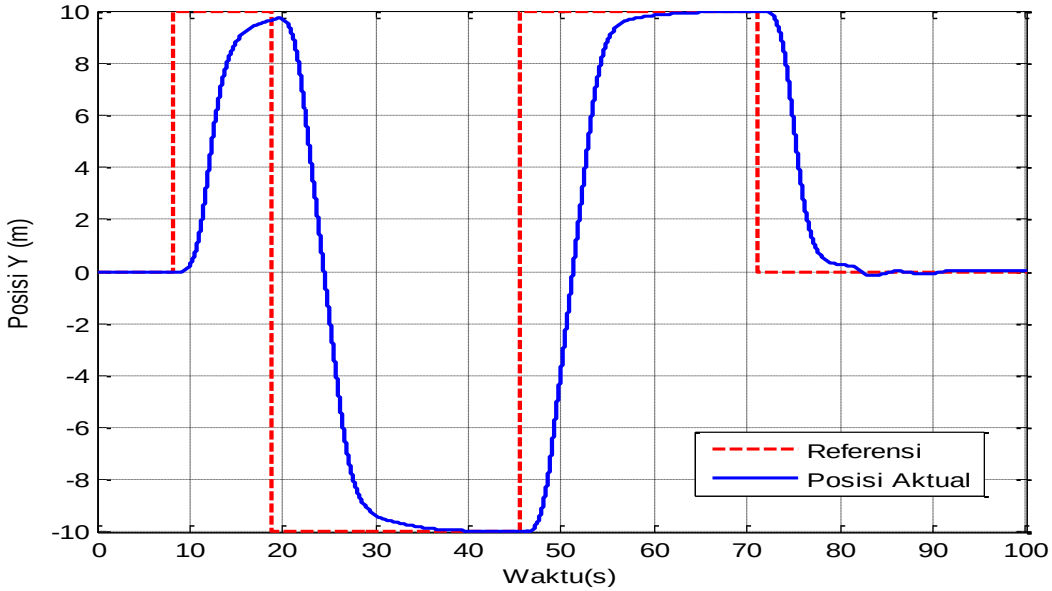
Gambar 4.35 Perubahan *Gain η Faulty case*

Pergerakan quadrotor pada Sumbu X ditunjukkan pada Gambar 4.36, pada gambar tersebut menunjukan quadrotor tidak mengalami pergeseran yang diakibatkan oleh kesalahan atau *fault* pada detik ke 20-80 dengan terjadi kesalahan aktuator hingga 65%. Dengan kesalahan aktuator yang bertahap dari 25%-65%, terlihat quadrotor dapat melakukan *waypoint tracking* dengan baik. Hal ini menunjukan bahwa kontroler *modified least square* tetap mampu melakukan *waypoint tracking* hingga 65%. Namun pada detik ke 80 quadrotor menunjukan pergerakan yang tak terkendali ketika melakukan pendaratan, hal ini diakibatkan oleh efek *couple* atau saling terkait nya dinamika pada quadrotor.



Gambar 4.36 Pergerakan Sumbu X saat *Faulty case*

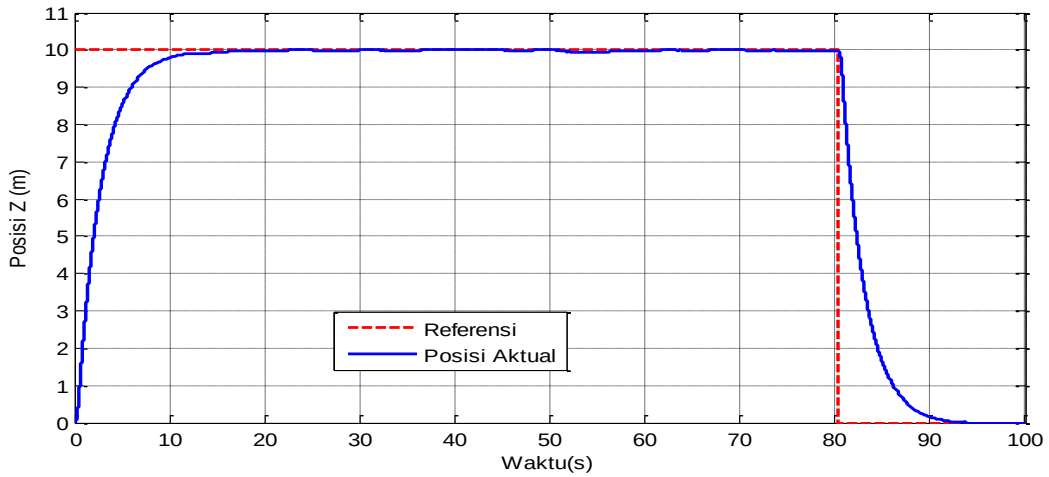
Pada pergerakan sumbu Y pada Gambar 4.37 tidak terlihat efek yang diakibatkan oleh kesalahan aktuator pada detik ke 20 -100. Dari gambar tersebut terlihat sesaat quadrotor mengalami pergeseran dari referensinya, namun kemudian quadrotor dapat tetap mencapai referensi kembali meskipun kesalahan yang terjadi sebesar 65%.



Gambar 4.37 Pergerakan Sumbu Y saat *Faulty case*

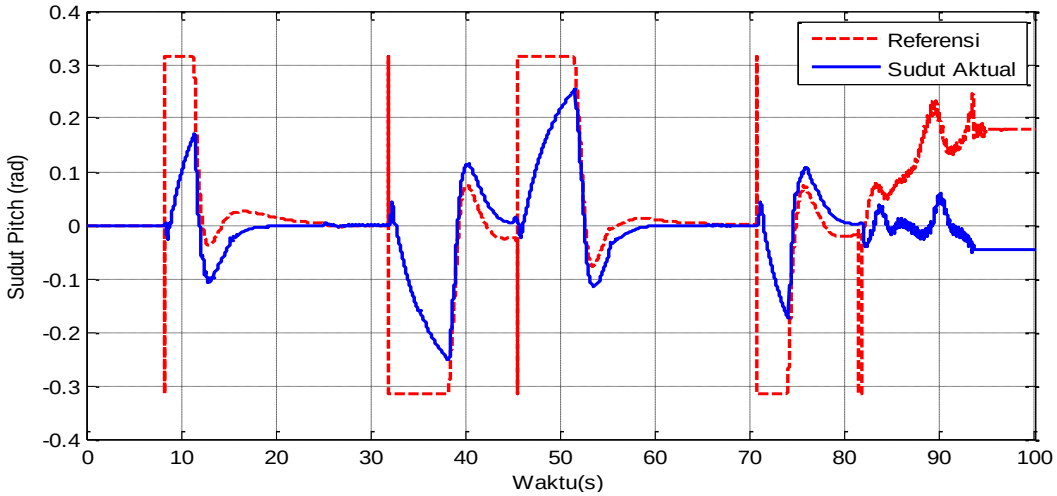
Sedangkan pergerakan quadrotor pada sumbu Z atau vertikal yang ditunjukan pada Gambar 4.38 terlihat bahwa quadrotor hanya terjadi pergeseran

sedikit saat detik ke 50 dimana terjadi besar kesalahan aktuator hingga 60% hingga quadrotor mendarat dengan baik pada detik ke 90. Hal ini menunjukan bahwa dengan metode *modified least square* quadrotor tetap mampu terbang dan melakukan pendaratan dengan baik meskipun terjadi *fault* 65% pada aktuator.

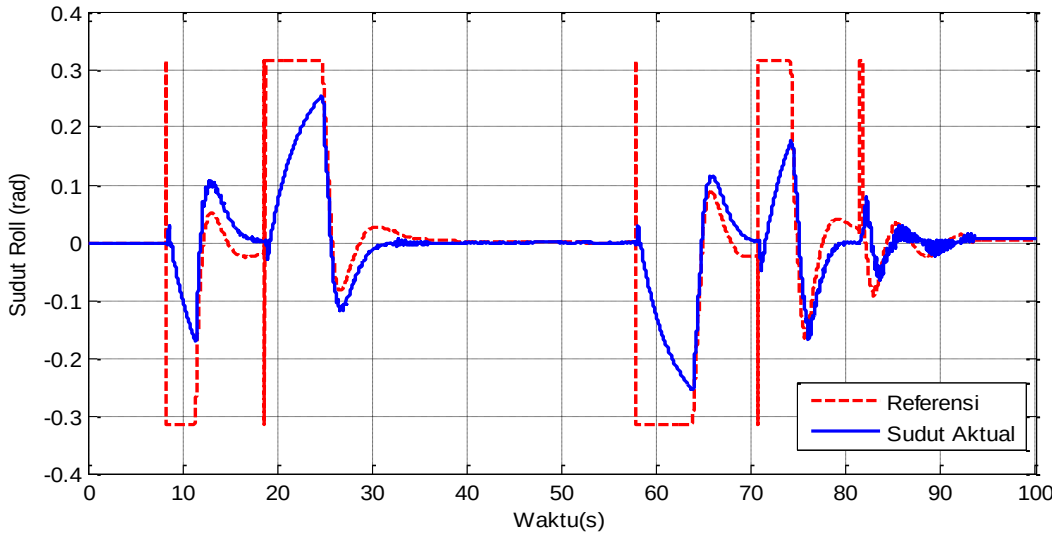


Gambar 4.38 Pergerakan Sumbu Z saat *faulty case*

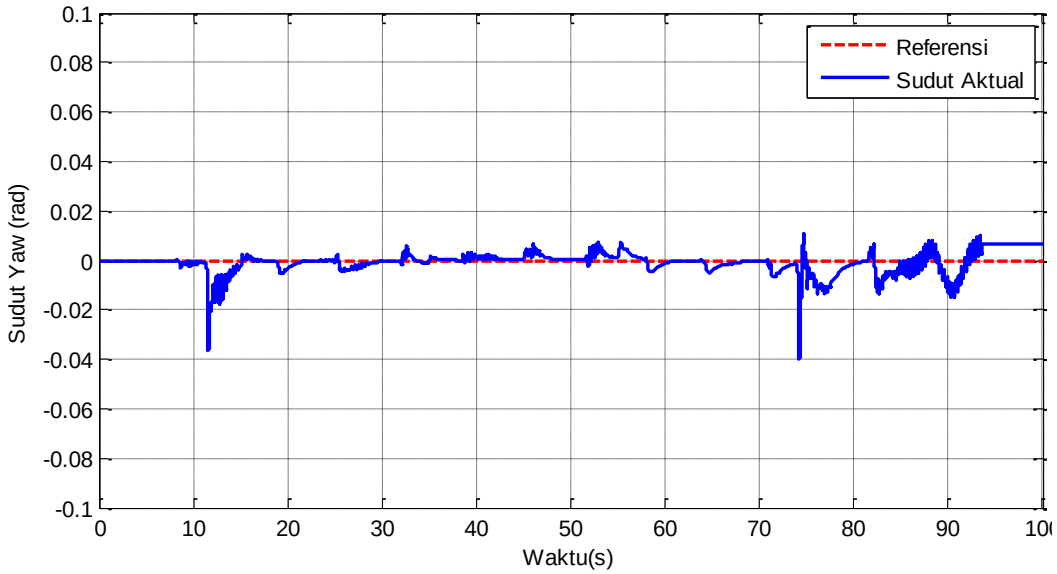
Pergerakan rotasi *pitch* dan *roll* dengan menggunakan metode *modified least square* dapat dilakukan dengan baik seperti yang ditunjukan pada Gambar 4.39 dan Gambar 4.40. Pada gambar tersebut terlihat quadrotor mampu melakukan *tracking* sinyal referensi yang masing-masing dihasilkan oleh kontroler X dan Y. Besar maksimum kemiringan sudut rotasi adalah 0.3 radian atau sebesar 15º meskipun terjadi kesalahan sebesar 65%.



Gambar 4.39 Pergerakan sudut *Pitch* saat *faulty case*



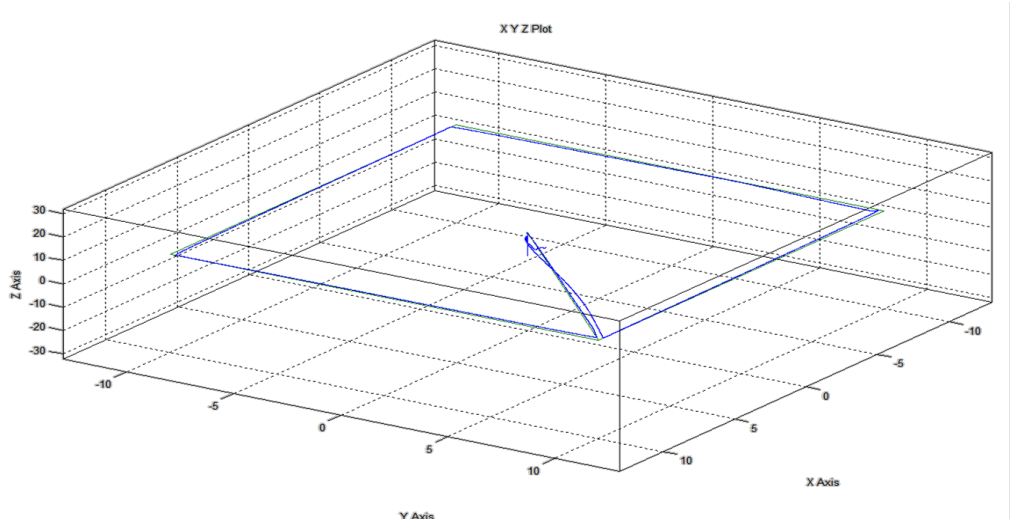
Gambar 4.40 Pergerakan sudut *Roll* saat *faulty case*



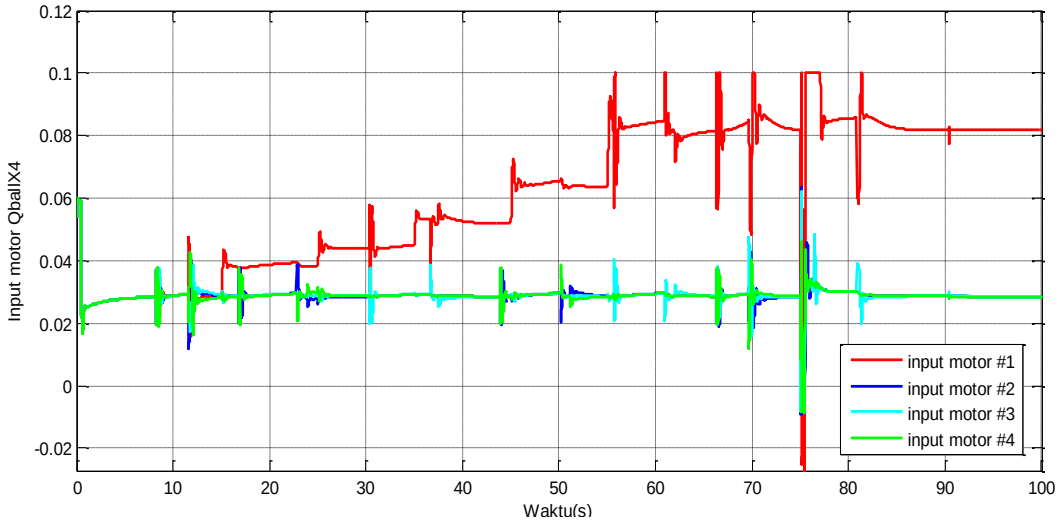
Gambar 4.41 Pergerakan sudut *Yaw* saat *Faulty case*

Pada sudut *yaw* atau *heading* quadrotor yang ditunjukkan pada Gambar 4.41 terlihat bahwa *heading* quadrotor tidak jauh dari referensi yaitu 0º. Dengan prilaku pergerakan rotasi dengan terjadinya kesalahan aktuator ini menunjukan bahwa dengan metode *modified least square*, quadrotor mampu melakukan *waypoint tracking* dengan baik dan quadrotor tetap aman untuk terbang meskipun terjadi kesalahan sebesar 65%.

Pergerakan quadrotor pada sumbu 3 dimensi X, Y dan Z ditunjukkan pada Gambar 4.42, terlihat quadrotor mampu menuju *waypoint* referensi dengan baik, hanya saja pada sisi diagonal, terlihat garis warna biru tidak mengikuti garis warna merah dengan baik. Hal ini dikarenakan pada *waypoint tracking*, quadrotor tidak diharuskan mengikuti lintasan yang berwarna biru. Tetapi yang terpenting adalah quadrotor harus mampu mencapai titik *waypoint* yang diberikan dengan tepat. Input motor Qball-X4 terlihat pada Gambar 4.43 dimana pada input motor#1 memperlihatkan batas maksimum input motor adalah 0.1 dengan kesalahan aktuator yang terjadi pada motor 1.



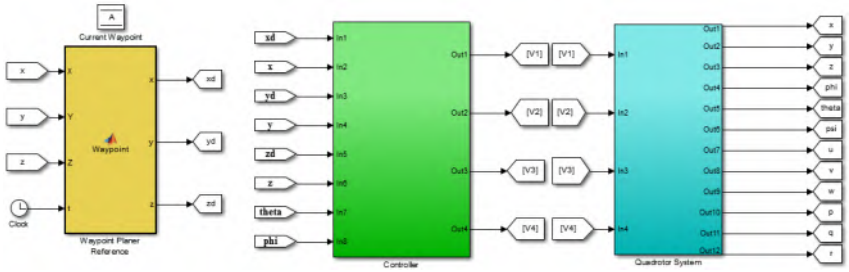
Gambar 4.42 Tampilan 3D *Waypoint Tracking* Quadrotor saat *Faulty Case*



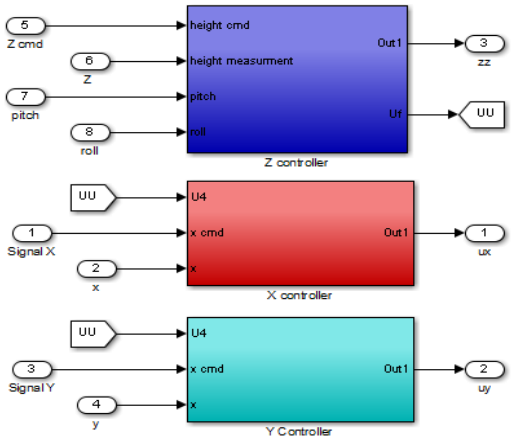
Gambar 4.43. Persamaan pada Qball X4 Manual $F = K\ \frac{\omega}{s+\omega}$

LAMPIRAN

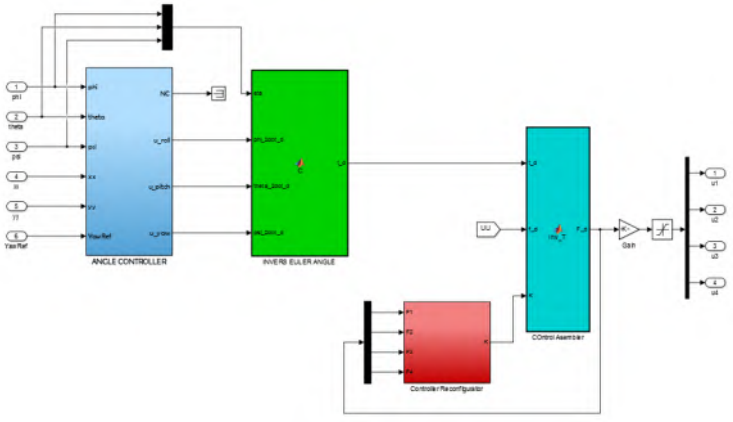
Lampiran A : Blok Diagram Simulink



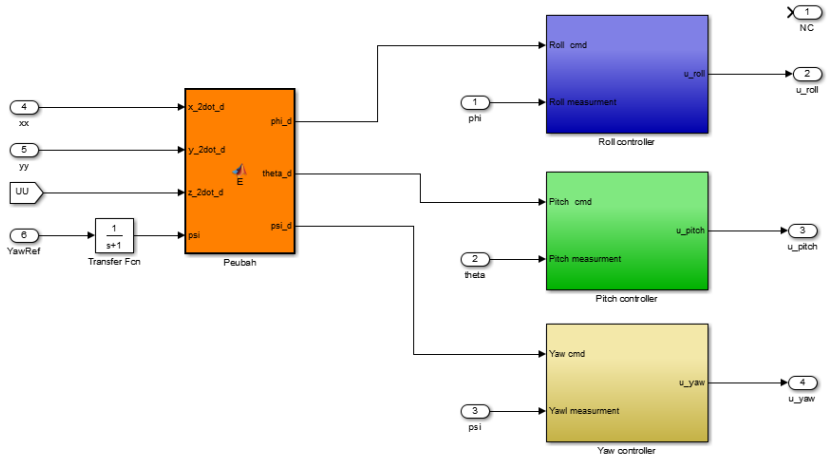
Gambar 1. Blok Simulink Sistem Keseluruhan



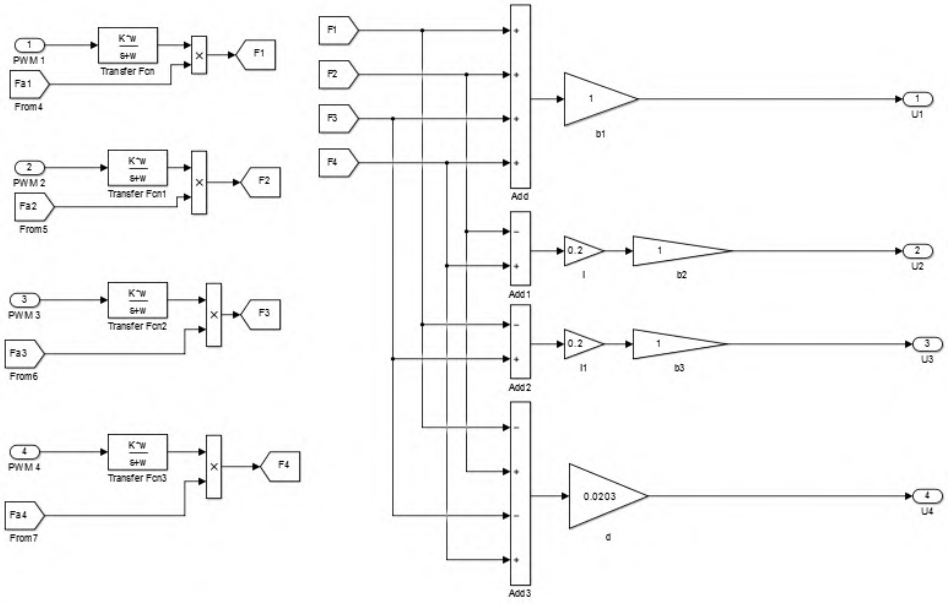
Gambar 2. Blok Simulink *OuterLoop Controller* (Sumbu Translasi)



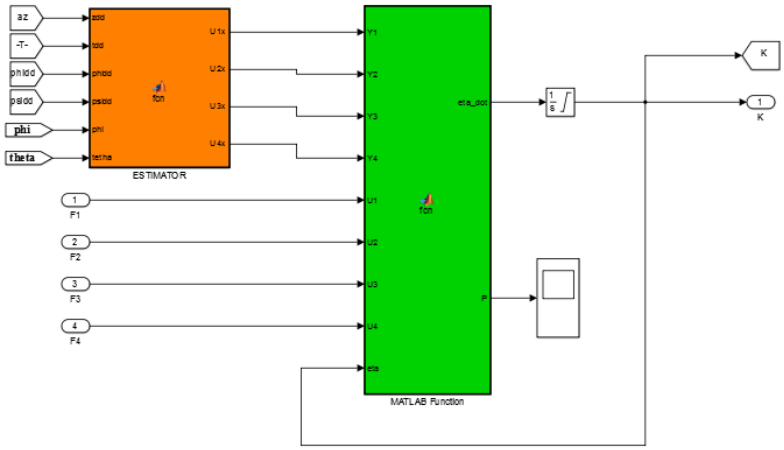
Gambar 3. Blok Simulink *Inner Loop Controller* (Sumbu Rotasi)



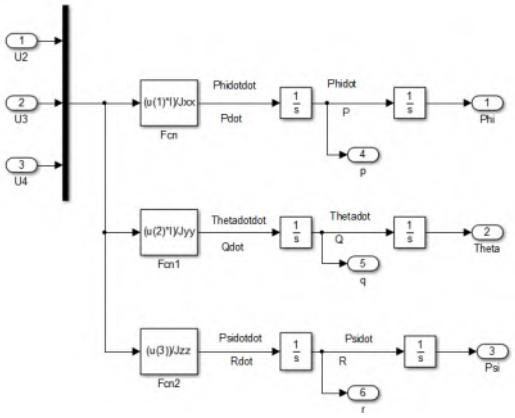
Gambar 4. Blok Simulink *Backstepping Control*



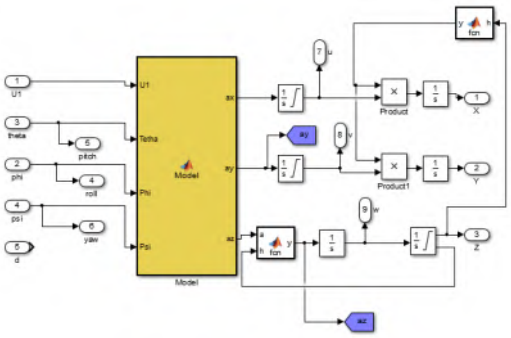
Gambar 5. Blok Simulink Kesalahan pada Aktuator (Rotor)



Gambar 6. Blok Simulink *Controller* Estimator *dan* Kompensator



Gambar 7. Blok Simulink *Anguler Velocity*



Gambar 8. Blok Simulink *Velocity*

Lampiran B: Source Code

B1. Listing Blok *Waypoint Planner Reference*

```
function [x,y,z]    = Waypoint( WP,X,Y,Z,t)
%# Waypoint  [x1 y1 z1; x2 y2 z2; dst ]
global A;
cur_wp = A;

[m,n] = size(WP);
Jumlah_wp=m;
toleransi = 0.4;

if Z <= (WP(int16(cur_wp),3)+toleransi) &&  Z >=
(WP(int16(cur_wp),3)-toleransi) && X <=
(WP(int16(cur_wp),1)+toleransi) &&  X >= (WP(int16(cur_wp),1)-
toleransi)&& Y <= (WP(int16(cur_wp),2)+toleransi) &&  Y >=
(WP(int16(cur_wp),2)-toleransi)

            cur_wp=cur_wp+1;
            if cur_wp>Jumlah_wp
               cur_wp=1;
            end
end
```

```
if(t<5)
    cur_wp=1;
end
A = cur_wp;
ref = WP(cur_wp,1:3);
x=ref(1);
y=ref(2);
z=ref(3);
```

B2. Listing *Euler Angle*

```
function [phi_d,theta_d,psi_d] = E(x_2dot_d,y_2dot_d,z_2dot_d,psi)
psi_d = psi;
d = sqrt(x_2dot_d*x_2dot_d + y_2dot_d*y_2dot_d +
z_2dot_d*z_2dot_d);
xx =1/( cos(psi)^2 + sin(psi)^2);
if -0.001 < d && d < 0.001
    phi_d = 0;
else
    phi_d = asin((x_2dot_d*sin(psi_d) - y_2dot_d*cos(psi_d)));
    %phi_d =  xx*(sin(psi)*x_2dot_d + cos(psi)*y_2dot_d);
end

theta_d = atan2(x_2dot_d*cos(psi_d) + y_2dot_d*sin(psi_d),1);
%theta_d = xx*(cos(psi)*x_2dot_d-sin(psi)*y_2dot_d);

min_ang = -pi/10;
max_ang =  pi/10;

if phi_d < min_ang
    phi_d = min_ang;
end
if phi_d > max_ang
    phi_d = max_ang;
end
if theta_d < min_ang
    theta_d = min_ang;
end
if theta_d > max_ang
    theta_d = max_ang;
end
%eta_d= [phi_d; theta_d; psi_d];
```

B3. Listing Invers *Euler Angle*

```
function t_d = C(eta,phi_2dot_d,theta_2dot_d,psi_2dot_d,I)

tr_d = [phi_2dot_d;theta_2dot_d;psi_2dot_d];
sphi = sin(eta(1));
cphi = cos(eta(1));
stht = sin(eta(2));
ctht = cos(eta(2));
```

```
C = [1   0     -stht;
     0   cphi  sphi*ctht;
     0   -sphi cphi*ctht];

%C = [1   0     0;
%     0   1     0;
%     0   0    1];
I=eye(3);
t_d = I*C*tr_d;
```

B4. Listing *Control Assembler*

```
function F_d = Inv_T(t_d, f_d,K)
d = 0.02;
l = 0.24;
K = diag(K);
T = [ 1    1     1      1;
      0    -l    0        l;
     -l    0     l   0;
     -d    d    -d     d];
u = [f_d; t_d(1); t_d(2); t_d(3)];
F_d =K*inv(T)*u;
```

B5. Listing Estimator

```
function [U1x,U2x,U3x,U4x] = fcn(zdd,tdd,phidd,psidd,phi,tetha)
Jxx = 0.03;
Jyy = 0.03;
Jzz = 0.04;
l = 0.2;
g = 9.8;
m=1.4;

U1x = m*(zdd+g)/(cos (phi)*cos(tetha));
U2x = Jxx*phidd/l;
U3x = Jyy*tdd/l;
U4x = Jzz*psidd;
End
```

B6. Listing Kompensator

```
function [eta_dot,P] = fcn(Y1,Y2,Y3,Y4,U1,U2,U3,U4,eta)
b = 5.42e-1;
d = 1.1e-2;
l=  .24;
T = [ 1    1     1     1;
      0    -l*1  0     l*1;
   -l*1    0     l*1   0;
     -d/b    d/b    -d/b     d/b];
Y = [Y1 Y2 Y3 Y4]';
U = [U1 U2 U3 U4]';
invT = inv(T);
Fp = ((invT*Y));
Fc = U;
P= (1*Fp).*(1+Fp.*Fp).^-1;
eta_dot     = P.*(Fc-eta.*Fp);
```

(Halaman ini sengaja dikosongkan)

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Penggunaan kontroler *modified least square* menunjukkan performansi kontrol yang lebih baik dimana quadrotor mampu mengatasi kesalahan pada aktuator (rotor) hingga 65% dari total gaya yang dihasilkan oleh motor pada kasus *waypoint tracking* sehingga quadrotor tetap mampu mengikuti referensi. Sedangkan pengujian kesalahan aktuator (rotor) quadrotor menggunakan *backstepping control*, quadrotor tidak mampu kembali pada sinyal referensi atau titik equilibrium saat 35%, ketika terjadi kesalahan sebesar 45% quadrotor semakin menjauhi referensi dan ketika *fault* 55% quadrotor terjatuh.

Dalam mengatasi perubahan parameter quadrotor saat melakukan *waypoint tracking*, metode *modified least square* mampu melakukan perubahan parameter *gain* η dengan cepat.

5.2 Saran

Saran untuk penelitian selanjutnya adalah diharapkan dapat dilakukan implementasi untuk menguji kualitas metode kontrol *modified least square* pada quadrotor.

(Halaman ini sengaja dikosongkan)

DAFTAR PUSTAKA

[1]. Madani, Tarek, dkk,. (2006).“ *Control of a Quadrotor Mini-Helicopter via Full State Backstepping Technique”*, IEEE Conference on Decision & Control. San Diego, USA.

[2]. Goodarzi, Farhad, dkk,. (2013), *“Geometric Nonlinear PID Control of a Quadrotor UAV on SE”*, European Control Conference (ECC), Switzerland.

[3]. Reyes, Elias, dkk,. (2013), *“LQR Control for a Quadrotor using Unit Quaternions: Modeling and Simulation.* Electronics Department INAOE .Puebla, Mexico.

[4]. Khebbache, Hicham, dkk,. (2012), *“Modeling and Stabilizing Control Laws Design Taking Into Account the Actuator Faults for an UAV Type-Quadrotor*. J. Automation & Systems Engineering. Algeria.

[5]. Xiaobing, Zhang, dkk,. (2010), *“Fault Tolerant Control for Quadrotor via Backstepping Approach*”, American Institute of Aeronautics and Astronautics. Irlando, Florida.

[6]. Abjadi, N.R, dkk,. (2006), *“Adaptive Control of Doubly Fed Field-Orinted Induction Machine Based on Recursive Least Squares Method Taking The Iron Loss Into Account”*, International Power Electronics and Motion Control Conference, China.

[7]. Abjadi, N.R, dkk,. (2006), *“A Nonlinear Adaptive Controller for Speed Sensorless PMSM Taking the Iron Loss Resistance Into Account”*, International Power Electronics and Motion Control Conference, China.

[8]. Ermey, Ahmet, dkk,. (2015), *“Active Fault Tolerant Control Againts Actuator Faults In a Quadrotor Using Unknown Input Observer Approach”*, International Aerospace Conference, Angkara.

[9]. Sadeghzadeh, Iman, dkk,. (2011), *“Fault Tolerant Trajectory Control of a Quadrotor Helicopter Using Gain-Scheduled PID and Model Reference Adaptive Control”*, Conference of the Prognestics and Health Management Society, North.

[10]. Theilliol, D., Noura, H. & Sauter, D. (1998), " *Fault-tolerant control method for actuator and component faults"*, Proceedings of the 37th IEEE Conference on Decision and Control, Tampa, Florida USA, 604-609.

[11]. Noura, H., Sauter, D., Hamelin, F. & Theilliol, D. (2000)," *Fault-tolerant control in dynamicsystems: application to a winding machine*", *IEEE Control Systems.*

[12]. Tim Quanser, (2010). *"Quanser Qball X-4 User Manual"*, Quanser Inovate Educate.

[13]. Cen, Zhohui., (2013)." *A Composite Fault Tolerant Control based on Fault Estimation for quadrotor UAVs"*, Conference Paper, Qatar.

[14]. Shaker, M. S., (2012)." *Active Fault Tolerant Control of Nonlinear Systems with Wind Turbine Application"*, M.Sc Electronics, University of Hull, Baghdad.

[15]. Patton, R. J., (1997a)," *Fault tolerant control: The 1997 situation*", IFAC Safeprocess '97, Hull. United Kingdom.

[16]. Rotondo, D., Nejjari, F., Torren, A. and Puig, V,. (2013c)," *Fault tolerant control design for polytopic uncertain LPV systems: Application to a quadrotor"*, Proceedings of the 2nd Conference on Control and Fault.

[17]. Zhang, Y. M., Chamseddine, A., , (2013)," *Development of advanced FDD and FTC techniques with application to an unmanned quadrotor helicopter testbed"*, Journal of the Franklin Institute : 2396–2422.

[18]. Noura, H., Sauter,. dkk,. (2000), "*Fault Tolerant Control in dynamic systems*: *Application to a winding machine* . Electronics Department INAOE IEEE Control Systems Magazines 20,pp.33-49.

[19]. Khalil, H.K. (2000), Nonlinear System, International Edition, Prentice-Hall, New Jersey.

[20]. Iaonou, P.A. dan J. Sun., (2008), "*Robust Adaptive Contro*l".

[21]. Amstrong, K.J, dkk,. (1995), "*Adaptive Control: Second Edition"*, International Edition, Prentice-Hall, New Jersey.

[22].Theillliol, Dididier, dkk,. (1998), "*Fault Tolerant Control Method for Actuator and Component Faults"*, Conference on Decision and Control, Tampa, Florida USA.

RIWAYAT PENULIS

Anisa Ulya Darajat dilahirkan di Bandar Lampung, Lampung pada Tanggal 10 Juni 1991. Putri kedua dari Bapak Mundzir dan Ibu Zuwilinar Dani. Penulis menempuh pendidikan tingginya di Jurusan Teknik Elektro, Universitas Lampung. Pada bulan Desember 2012, penulis berhasil menyelesaikan pendidikan sarjananya dan mendapatkan gelar Sarjana Teknik. Pada tahun 2014, penulis melanjutkan pendidikan Magister di Jurusan Teknik Elektro, Fakultas Teknologi Industri Institut Teknologi Sepuluh Nopember (ITS) Surabaya dengan bidang keahlian Teknik Sistem Pengaturan.

(Halaman ini sengaja dikosongkan)