TUGAS AKHIR TF091381

SIMULASI PENGENDALIAN *ATTITUDE* UAV *(UNMANNED AERIAL VEHICLE) QUADROTOR* BERBASIS *ARTIFICAL NEURAL NETWORK*

YUHARA RAHMANTYA
NRP. 2410 100 102

Dosen Pembimbing
Dr. Ir. Purwadi Agus Darwito, MSc.

JURUSAN TEKNIK FISIKA
Fakultas Teknologi Industri
Institut Teknologi Sepuluh Nopember
Surabaya 2014

FINAL PROJECT TF091381

ATTITUDE CONTROL SIMULATION OF UAV (UNMANNED AERIAL VEHICLE) QUADROTOR BASED ON ARTIFICIAL NEURAL NETWORK

YUHARA RAHMANTYA
NRP. 2410 100 102

Supervisor
Dr. Ir. Purwadi Agus Darwito, MSc.

DEPARTMENT OF ENGINEERING PHYSICS
Faculty of Industrial Technology
Sepuluh Nopember Institute of Technology
Surabaya 2014

LEMBAR PENGESAHAN

SIMULASI PENGENDALIAN *ATTITUDE* UAV *(UNMANNED AERIAL VEHICLE) QUADROTOR* BERBASIS *ARTIFICIAL NEURAL NETWORK*

TUGAS AKHIR
Oleh:

YUHARA RAHMANTYA
NRP : 2410 100 102

Surabaya, Agustus 2014
Mengetahui/Menyetujui

Pembimbing

Dr. Ir Purwadi Agus Darwito, MSc
NIP : 19620822 198803 1 001

Ketua Jurusan
Teknik Fisika FTI-ITS

KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN INSTITUT TEKNOLOGI SEPULUH NOPEMBER FAKULTAS TEKNOLOGI INDUSTRI JURUSAN TEKNIK FISIKA

Dr. Ir. Totok Soehartanto, DEA
NIP : 19650309 199002 1 001

LEMBAR PENGESAHAN

SIMULASI PENGENDALIAN *ATTITUDE* UAV (*UNMANNED AERIAL VEHICLE*) *QUADROTOR* BERBASIS *ARTIFICIAL NEURAL NETWORK*

TUGAS AKHIR

Diajukan Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Teknik
pada
Bidang Studi Rekayasa Instrumentasi
Program Studi S-1 Jurusan Teknik Fisika
Fakultas Teknologi Industri
Institut Teknologi Sepuluh Nopember

Oleh:

YUHARA RAHMANTYA
NRP 2410 100 102

Disetujui oleh Tim Penguji Tugas Akhir:

1. Dr. Ir. Purwadi Agus Darwito, MSc (Pembimbing)
2. Ir. Ya'umar, MT. (Penguji I)
3. Ir. Tutug Dhanardono, MT. (Penguji II)
4. Arief Abdurrakhman, ST, MT. (Penguji III)

SIMULASI PENGENDALIAN *ATTITUDE* UAV (*UNMANNED AERIAL VEHICLE*) *QUADROTOR* BERBASIS *ARTIFICIAL NEURAL NETWORK*

Nama Mahasiswa	: YUHARA RAHMANTYA
NRP	: 2410 100 102
Jurusan	: Teknik Fisika
Dosen Pembimbing	: Dr. Ir. Purwadi Agus Darwito, MSc

Abstrak

Pengamatan yang dilakukan pada daerah yang sulit dijangkau atau daerah yang berbahaya misalnya daerah pegunungan atau dearah bencana alam seringkali menyulitkan observasi pada daerah tersebut. Solusinya adalah dengan pengamatan dari langit atau dari atas. Untuk itu diperlukan sarana yang dapat dikendalikan secara *unmanned* (tanpa awak). Salah satu sarana yang dapat melakukan hal tersebut adalah *quadrotor*. *Quadrotor* adalah suatu kendaraan yang terdiri dari empat motor dan empat *propeller*. Salah satu manuver dari quadrotor adalah *Hovering.* Kondisi *Hovering* adalah sebuah keadaan di mana *quadrotor* tersebut melayang diam di udara. *Attitude* adalah keadaan atau posisi dari quadrotor saat melakukan manuver. Pada penelitian ini dilakukan simulasi mengenai sistem kendali *Attitude quadrotor* Berdasarkan pemodelan yang telah dilakukan sebelumnya. jenis metode kontrol yang digunakan adalah *Neural Network* (Jaringan Syaraf tiruan). Pengujian sistem dilakukan terlebih dahulu dengan mensimulasikan gangguan pada *quadrotor.* Didapatkan respon steady state error pada altitude, roll, pitch berturut turut sebesar 9.1%, 0.65%, dan 1%.

Kata Kunci : *Attitude Control, Hovering, Neural Network, Quadrotor*

Halaman ini sengaja dikosongkan

ATTITUDE CONTROL SIMULATION OF UAV (UNMANNED AERIAL VEHICLE) QUADROTOR BASED ON ARTIFICIAL NEURAL NETWORK

Student Name	: YUHARA RAHMANTYA
NRP	: 2410 100 102
Department	: Teknik Fisika
Supervisor	: Dr. Ir. Purwadi Agus Darwito, MSc

Abstract

Observation on remote or dangerous place such as mountain, and natural disaster area often complicate an surveillance on that area. The solution is by observation from above. Therefor, some unmanned vehicle is needed that can be controlled from a long distance. One of the possible solution is by using quadrotor. Quadrotor is a vehicle consist of four motors and four propellers. One of the quadrotor maneurver is hovering Hovering is a state where quadrotor floating steadly in sky. Attitude is a state or position of quadrotor when doing some maneuver. In this research, a simulation about quadrotor Attitude control system has been done based in quadrotor modeling. A Control method that used is Neural Network. System test is done by simulating disturbance on quadrotor. altitude, roll, pitch steady state error response consecutively is 9.1%, 0.65%, and 1%.

Keywords : *Attitude Control, Hovering, Neural Network, Quadrotor*

Halaman ini sengaja dikosongkan

KATA PENGANTAR

Puji syukur kehadirat Allah SWT yang senantiasa melimpahkan rahmat dan hidayah-Nya serta shalawat dan salam kepada Nabi Muhammad SAW, hingga terselesaikannya Tugas Akhir beserta Laporan Tugas Akhir yang berjudul SIMULASI PENGENDALIAN ATTITUDE UAV (UNMANNED AERIAL VEHICLE) QUADROTOR BERBASIS ARTIFICIAL NEURAL NETWORK.

Penulis telah banyak memperoleh bantuan dari berbagai pihak dalam penyelesaian Tugas Akhir dan Laporan Tugas Akhir ini. Penulis mengucapkan terima kasih kepada :

1. Allah SWT yang telah melimpahkan rahmad dan hidayah-Nya, sehingga penulis dapat menyelesaikan Tugas Akhir dan Laporan Tugas Akhir ini.
2. Nabi Muhammad SAW yang telah menjadi panutan bagi seluruh alam sehingga penulis dapat mengerti bagaimana hidup yang baik dan benar di dunia dan di hadapan Allah.
3. Almarhum Ayah dan Ibu tercinta di rumah yang telah membesarkan dan mendidik saya dari saya kecil sampai menyelesaikan tugas akhir ini, dengan kasih saying beliau saya mengerti bagaimana menyayangi dan mencintai keluarga dan sesama.
4. Bapak Dr. Ir. Purwadi Agus Darwito, MSc. selaku pembimbing yang telah ikhlas dan penuh kesabaran dalam menghadapi, mengarahkan dan membimbing selama saya di Teknik Fisika.
5. Bapak Ustadz Andi Rahmadiansyah atas ilmu agamanya dan pencerahannya di tengah-tengah gelap dan horrornya labkom. Serta mengijinkan untuk menyelesaikan tugas akhir.
6. Keluarga besar e205, angkatan 2010 dan rekan lain yaitu rendra, syamsul, nindyan, karim, hastin, atma, rois, Chandra, dimas, angkik, ninin, rhio, purwanto, icang, dhikri, masbi, yusnia, aas, anggi, rio, ahara, nico, icha, novia, yurid, Edwin, berlian, faith, shinta, risky, ndok, akbar. yang menemani

tidur, makan, ngopi rame-rame selama di Teknik Fisika, dan bercandaannya yang rame membuat suasana hari-hari di Teknik Fisika semakin asyik dan tidak terasa sudah 3 tahun telah bersama-sama.

Penulis menyadari bahwa penulisan pada tugas akhir ini tidak sempurna, masih banyak yang kurang dan tidak penting. Namun penulis berharap semoga tulisan Tugas Akhir yang telah diselesaikan memiliki manfaat yang besar untuk diri sendiri terutama, untuk Jurusan Teknik Fisika FTI-ITS, dan insya'allah untuk bangsa dan negara tercinta. Dan semoga generasi penerus Tugas Akhir dengan materi yang sama dapat menjadikan Tugas Akhir ini salah satu referensi yang dapat dipertanggungjawabkan.

Surabaya, Agustus 2014

Penulis

DAFTAR ISI

DAFTAR GAMBAR

Halaman ini sengaja dikosongkan

DAFTAR TABEL

Halaman ini sengaja dikosongkan

BAB I
PENDAHULUAN

1.1 Latar Belakang

Daerah yang sulit di jangkau manusia misalnya pegunungan, tebing, maupun daerah bencana sering kali menyulitkan proses identifikasi korban bencana, pemetaan kontur, serta pengamatan daerah setempat. Hal tersebut dapat dengan mudah di lakukan pada ketinggian tertentu, maka dari itu diperlukan suatu sarana yang dapat melakukan maneuver di udara. Salah satu solusi permasalahan itu adalah dengan menggunakan helikopter. namun helikopter sendiri memiliki kekurangan yaitu tingginya biaya bahan bakar, serta membutuhkan lahan yang relatif luas. *Quadrotor* atau juga bisa di sebut *quadrotor-helicopter* adalah helikopter yang mempunyai empat motor dan empat propeler. Karena Quadrotor sangat berguna pada lingkungan atau area yang tidak terjangkau atau sulit dijangkau, berbahaya, tidak memungkinkan bahkan oleh pilot yang cukup handal, sebuah UAV (*Unmanned Aerial Vehicle*) atau kendaraan tanpa awak mempunyai banyak keuntungan[1]. *Quadrotor* pun dapat disebut sebagai model UAV yang cukup populer karena konstruksinya yang cukup sederhana dan *payload* atau muatan yang dibawa cukup besar jika dibandingkan dengan helikopter konvensional atau helikopter pada umumnya[2]. Konsep *quadrotor* sendiri bukanlah hal baru, pada tahun 1922, Georges de Bothezat membuat rangka menyilang berbentuk "x"dan di ujungnya diberi propeler atau baling-baling. Pada umumnya *quadrotor* modern bersifat *unmanned* yang artinya tidak memiliki awak. Karena dapat menggunakan *high-speed brushless motor*, dan baterai Li-polymer, *quadrotor* kini dapat di rancang dan di produksi, namun untuk sistem pengendalian dari *quadrotor* sendiri masih merupakan hal yang cukup sulit [3]. Keuntungan dari penggunaan *quadrotor* adalah kemudahan nya bermanuver sehingga dalam hal ini pengambilan gambar secara FPV (*first-person view*). Beberapa permasalahan yang terkadang di hadapi dalam perancangan UAV adalah pada gangguan yang terjadi di saat terbang misalkan

angin, yang mana mengganggu kestabilan dari *quadrotor* tersebut.

Beberapa metode pengendalian kestabilan atau *attitude* telah banyak di kembangkan dalam berbagai penelitian. Metode Pengendalian yang dapat di gunakan bermacam – macam misalnya menggunakan Tuning PID[4] serta Fuzzy Logic[5]. Dan salah satu metode pengendalian yang dapat digunakan adalah Jaringan Syaraf Tiruan atau ANN (*Artificial Neural Network)*.

NN (Neural Network) Adalah suatu ilmu mengenai perhitungan matematis yang merepresentasikan arsitektur dari *neural network* (jaringan syaraf) yang sampai saat ini belum dapat dipastikan secara akurat dalam menggambarkan system syaraf manusia. *Artificial Neural Network* atau ANN pada dasarnya adalah sebuah logikaberfikir yang di pengaruhi oleh input yang telah direncanakan. JST mempunyai alogaritma pelatihan yang memungkinkan sistem tersebut belajar untuk penyesuaian terbaik terhadap output yang dihasilkan. Berdasarkan hal diatas mengenai kebutuhan sistem pengendalian yang handal pada *attitude quadrotor*, diperlukan sebuah sistem pengendalian yang digunakan untuk mengendalikan *attiude* pada *quadrotor*.

1.2 Rumusan Masalah

Attitude merupakan hal yang sangat penting dalam bidang penerbangan maupun *aeromodeling*. Dalam hal ini *quadrotor* yang di gunakan adalah jenis (+) plus. Maka dari itu diperlukan sebuah simulasi mengenai pengendalian *attitude* dari *quadrotor* dan sistem pengendalian *attitude quadrotor* yang digunakan berbasis *Artificial Neural Network.*

1.3 Tujuan

Tujuan utama dari penelitian ini adalah mensimulasikan perancangan sistem pengendalian *Attitude* dari *quadrotor*. Agar pada saat dilakukan manuver secara tanpa awak (*unmanned*) pada kondisi *hovering*, *quadrotor* tetap stabil.

1.4 Batasan masalah

Adapun Beberapa batasan masalah pada penelitian ini adalah:

1. *Quadrotor* yang digunakan adalah tipe plus (+)
2. Sistem pengendalian yang dirancang hanyalah untuk *pitch, roll,* dan ketinggian (*altitude*) *quadrotor.*
3. Ganguan pada saat uji kestabilan pada simulasi dilakukan secara sengaja.

1.5 Sistematika laporan

Secara sistematis, penyusunan laporan tugas akhir ini tersusun dalam lima bab dengan penjelasan sebagai berikut:

BAB I Pendahuluan

Bab ini berisi latar belakang, perumusan masalah, tujuan penelitian, batasan masalah, dan sistematika laporan.

BAB II Tinjauan Pustaka

Bab ini berisi mengenai teori-teori penunjang yang terkait dalam penulisan tugas akhir.

BAB III Metodologi Penelitian

Bab ini akan dijelaskan mengenai langkah-langkah yang telah dilakukan dalam penelitian.

BAB IV Analisis Data dan Pembahasan

Bab ini akan ditampilkan data dan analisa hasil simulasi beserta pembahasannya.

BAB V Kesimpulan dan Saran

Bab ini berisi tentang kesimpulan pokok dari seluruh rangakaian penelitian yang telah dilakukan dan saran yang dapat dijadikan sebagai pengembangan penelitian selanjutnya.

Halaman ini sengaja dikosongkan

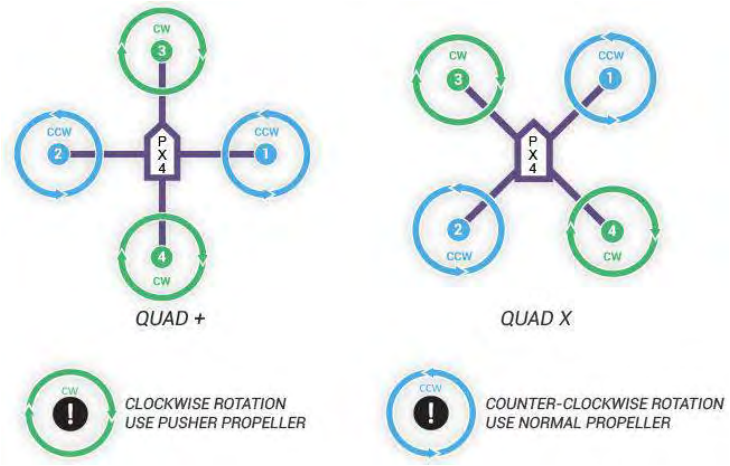
BAB II
TINJAUAN PUSTAKA

2.1 *Quadrotor*

Quadrotor adalah sebuah kendaraan *multi-rotor* yang menggunakan prinsip *vertical take-off landing* (VTOL)[1], artinya adalah cara *take-off* atau lepas landas yaitu secara vertikal atau dan pada saat *quadrotor* mendarat pun bisa dilakukan pada keadaan vertikal tanpa memerlukan landasan pacu (*runway*) seperti pada kendaraan udara jenis *fixed-wing* misalnya pesawat terbang tipe CN-235. Daya angkat dari *Quadrotor* berasal dari ke-empat motor atau rotornya. *Quadrotor* termasuk dalam jenis *rotorcraft*[3] atau *rotary-wing*[6], berbeda dengan pesawat jenis *fixed-wing*, karena gaya angkat pada *Quadrotor* pada 4 motornya sedangkan pada pesawat *fixed-wing* terletak pada sayap atau perbedaan tekanan udara yang menggunakan prinsip bernoulli.[7]

Gambar 2.1 Contoh pesawat *Fixed-wing* CN-235

Ada dua tipe konfigurasi *Quadrotor* yaitu jenis plus (+) dan jenis cross (x).



Gambar 2.2 Tipe konfigurasi *Quadrotor*

Pada konfigurasi *Quadrotor* tersebut, keduanya mempunyai dua motor dan *propeller* yang berputar searah jarum jam dan dua motor yang lain berputar berlawanan arah dengan jarum jam dengan posisi saling berhadapan. Hal ini dilakukan agar momen dari rotasi kedua pasang *propeller* saling meniadakan sehingga *Quadrotor Attitude* atau orientasi *yaw* dari *Quadrotor* stabil pada saat mengudara[8]**.**

Gambar 2.3 Contoh *Mini-Quadrotor*

Quadrotor menggunakan sistem kontrol elektronik, serta sensor yang digunakan untuk mengetahui *attitude* dari *Quadrotor*. Karena bentuknya yang relatif kecil dan dapat melakukan manuver yang beragam, *Quadrotor* dapat mengudara pada ruangan tertutup maupun ruangan terbuka.

Gambar 2.4 Kondisi *Hovering Quadrotor*

Sebuah *Quadrotor* terdiri dari beberapa komponen penting yang menunjang agar *Quadrotor* tersebut dapat terbang. Beberapa komponen tersebut adalah :

2.1.1 Mikrokontroller

Mikrokontroller adalah suatu *integrated circuit* (IC) yang pada umumnya digunakan untuk mengontrol alat tertentu misalnya sistem kontrol mobil atau sistem permesinan serta lain sebagainya. Berbeda dengan mikroprosesor yang pada umumnya di gunakan pada PC (*Personal Computer*) yang hanya digunakan untuk memproses suatu data input yang di berikan sesuai dengan namanya yaitu mikroprosesor. Mikrokontroller dirancang untuk digunakan pada suatu alat, sistem elektronik maupun mekanik, dan telah mempunyai memori, *clock*, *I/O port*, dan lain sebagainya dalam satu sistem yaitu MinSis (*Minimum System*) berbeda dengan mikroprosesor yang membutuhkan tambahan komponen lain agar suatu sistem komputer tersebut dapat beroperasi sesuai dengan yang kita inginkan.

Gambar 2.5 Contoh gambar Mikrokontroller (ATMega 32)

Pada *Quadrotor* suatu kontroller dibutuhkan guna mengendalikan kecepatan putar dari ke-empat motor sedemikian rupa yang berakibat pada orientasi *roll*, *pitch* dan *yaw* serta ketinggian *Quadrotor* tersebut stabil.

2.1.2 *Accelerometer* dan *Gyroscpe*

Accelerometer adalah suatu alat yang digunakan untuk mengukur percepatan. Percepatan disini tidak harus perubahan kecepatan, namum di sini percepatan yang dialami oleh beban dari benda yang ada pada *accelerometer* tersebut. Misalnya disini adalah apabila *accelerometer* diletakan pada permukaan subjek, maka accelerometer akan mendeteksi percepatan grafitasi yang di alami benda tersebut yaitu sebesar G (9,81 m/s^2) dengan arah ke atas, karena gaya beban mengarah ke bawah. Jenis dari accelerometer yaitu *single-axis* dan multi-axis dan dapat mengukur nilai serta arah dari percepatan tersebut.

Gambar 2.6 Sensor *Accelerometer*

Sensor *Gyroscope* adalah suatu alat untuk mengukur orientasi *quadrotor*, berdasarkan momen anguler dari benda tersebut. Kedua sensor tersebut yang akan menunjukan

kemiringan sudut dari *quadrotor* tersebut. Kedua sensor itu saling mendukung satu dengan yang lain

Gambar 2.7 Sensor *Gyroscope*

2.1.3 *Ultrasonic Sensor*

Sensor *Ultrasonic* adalah sebuah yang digunakan untuk mengukur ketinggian *Quadrotor*. Pada dasarnya Sensor *ultrasonic* adalah berupa *transceivers* dimana sensor *ultrasonic* ini mengirim dan menerima sinyal berupa gelombang suara *ultrasonic* yang di kirim dan menerima pantulannya sebagai *input* untuk mengukur jarak dari pengamat. Cara kerjanya tidak berbeda jauh dengan sonar yang digunakan untuk mengukur kedalaman laut pada kapal selam.

Gambar 2.8 Sensor Ultrasonik

2.1.4 *Brushless DC Motor*

Quadrotor menggunakan motor DC *brushless* sebagai aktuatornya. Kecepatan Motor DC diatur agar orientasi dari Quadrotor stabil dan *horizontal* atau tegak lurus dengan tanah. Motor DC adalah sebuah aktuator yang berfungsi untuk mengubah energi tegangan menjadi energi mekanik. Motor DC

terdiri dari dua rangkaian elektromagnetik berupa *rotor* dan *stator*. Pada rotor, beberapa kumparan tersambung secara seri dan pada *stator* beberapa magnet permanen menghasilkan medan magnet yang mempengaruhi *rotor* agar berputar dengan cara mengalirkan arus DC menuju kumparan, *rotor* berputar karena adanya interaksi antara kedua medan magnet pada *rotor* dan *stator* yang saling tolak menolak sehingga mengasilkan perputaran motor[9].

Gambar 2.9 *Brushless DC Motor* pada Quadrotor

2.2 *Neural Network*

Neural Network (NN) atau jaringan syaraf tiruan adalah suatu pendekatan matematis yang didasari dengan pengamatan terhadap sistem kerja dari otak serta (NN) merupakan salah satu representasi buatan dari otak yang manusia yang mana untuk mensimulasikan proses pembelajaran pada otak manusia. Otak manusia mempunyai jutaan sel syaraf yang berfungsi untuk memproses informasi, tiap sel syaraf atau neuron memiliki satu inti sel, inti sel inilah yang akan bertugas untuk memproses informasi. Bila di gunakan dalam implementasinya, Alogaritma dari (NN) di masukkan pada mikrokontroller. Plantnya adalah *Quadrotor*, sedangkan set point nya adalah orientasi (*attitude*) dari *Quadrotor*, yaitu nilai *pitch, roll, yaw*, serta ketinggian *Quadrotor* yang mana nilai tersebut didapat dari sensor pada *Quadrotor*.[10]

2.2.1 Model Pembelajaran

Neural Network (NN) adalah suatu alogaritma yang meniru cara berfikir otak manusia. Maka dari itu sistem dari NN sendiri

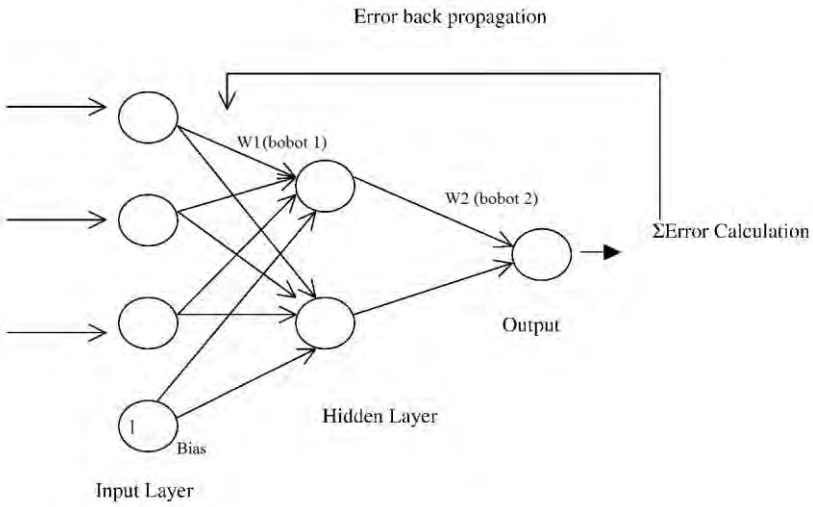
meniru cara belajar atau logika berfikir manusia dalam mencari solusi dari permasalahan yang di hadapi berdasarkan pengalaman yang ia alami. Adapun 2 bentuk dari metode atau model pembelajaran NN sendiri yaitu :

- *Supervised Learning* :

Pada metode *supervised learning* ini, suatu sistem NN memerlukan data acuan sebagai dasar untuk pembelajaran sistem. Umumnya di gunakan akar kuadrat dari rata-rata *error* agar mengurangi atau meminimalisir nilai *error*. Jadi pada *supervised learning* ini terdapat arah belajar yang jelas terhadap sistem agar menghasilkan *output* yang kita inginkan.

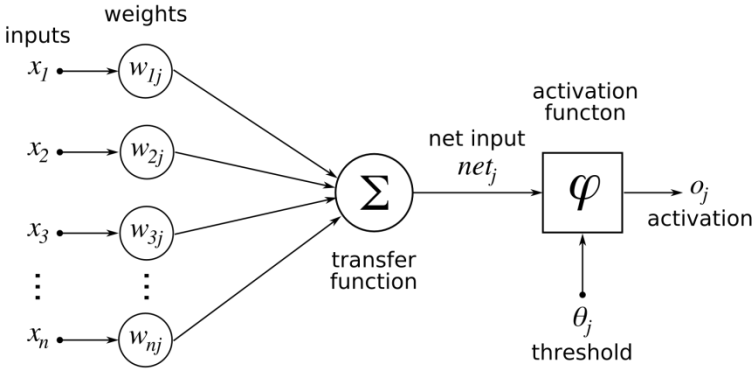
- *Unsupervised Learning* :

Untuk metode *Unsupervised Learning* hanya memerlukan suatu data *input* misalkan “a” lalu *output* dari *network* berupa “b”. jadi pada dasarnya sistem tersebut belajar secara “mandiri” dan menghasilkan *output* berdasarkan pengalaman yang telah sistem tersebut alami selama ini.



Gambar 2.10 Contoh Arsitektur *Backpropagation*

Backpropagation adalah salah satu Jenis alogaritma pembelajaran *Neural Network*. Metode *backpropagation* mempunyai satu atau lebih *Hidden Layer*. Pada metode ini digunakan suatu data yang dibutuhkan untuk melakukan training agar menghasilkan *weight* atau bobot yang akan menjadi faktor pengali pada NN. Adapun beberapa parameter yang digunakan pada metode ini yaitu *epoch*, jumlah *neuron* pada *hidden layer*, dan *learning rate*. *Epoch* adalah jumlah *training* yang dibutuhkan agar menghasilkan *weight* agar menghasilkan error seminimal mungkin terhadap target yang menjadi acuan. *Hidden layer* adalah suatu koneksi atau hubungan yang terdiri dari perhitungan matematis sebelum menuju *output*. Dan *learning rate* adalah laju pembelajaran dari training NN, semakin tinggi *learning rate* maka semakin cepat sistem NN belajar, namun hal tersebut dapat menimbulkan error yang relatif lebih besar.



Gambar 2.11 Contoh Fungsi Aktifasi pada *hidden layer*

Adapun fungsi aktivasi yang digunakan pada penelitian ini adalah tipe tangen hiperbolik pada *hidden layer* dan tipe linear (*purelin*) pada *output layer*.

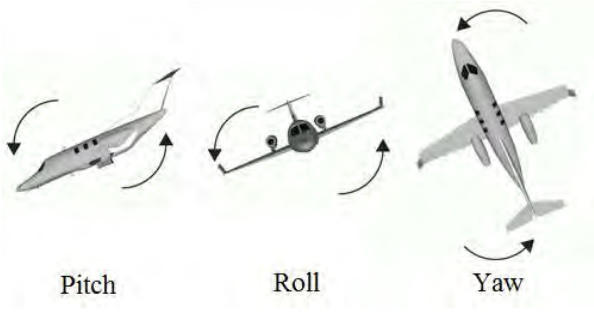
2.5 MATLAB

Simulasi dilakukan dengan menggunakan *software* MATLAB. MATLAB adalah sebuah *software* yang dapat membantu untuk melakukan perhitungan matematis, pemodelan

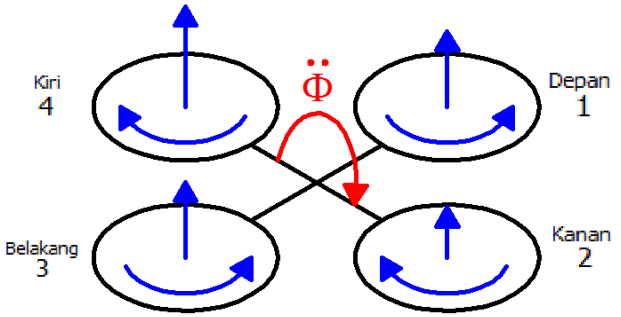
sistem, dan beberapa fitur lain. Namun untuk penelitian ini digunakan fitur Simulink untuk membantu pemodelan sistem *quadrotor.*

2.4 *Flight Dynamic*

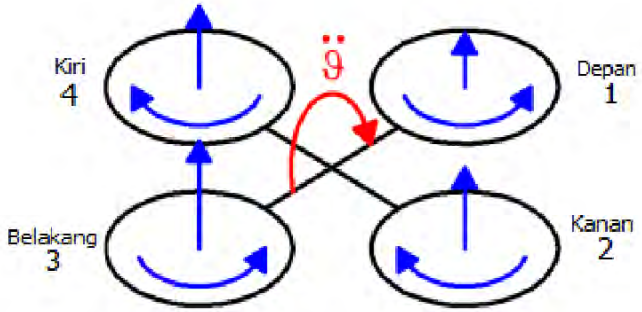
Dalam penerbangan, suatu *attitude* (orientasi dari pesawat baik *fixed-wing* atau *rotary-wing*) sangatlah penting. Karena suatu *attitude* dapat mempengaruhi lancar tidaknya penerbangan. Adapun beberapa komponen atau faktor penting dalam *attitude* penerbangan, antara lain *pitch, roll,* dan *yaw*. Pada penelitian ini pengendalian dari ketiga elemen ini sangat penting dan berpengaruh nantinya terhadap posisi dari *quadrotor*, misalkan bila *quadrotor* tersebut miring 5^o saja akan mengakibatkan *quadrotor* tersebut bergerak dan tidak stabil pada kondisi *hovering* (melayang) atau dengan kata lain *attitude* dari *quadrotor* tersebut buruk. Dan pada penelitian ini yang perlu diperhatikan juga adalah *altitude* atau ketinggian dari *quadrotor.* Berikut adalah ilustrasi orientasi *pitch, roll,* dan *yaw* tersebut.



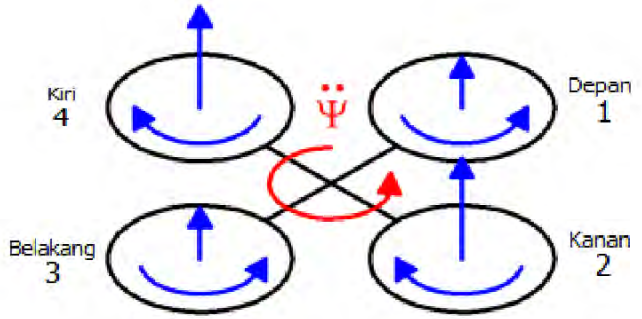
Gambar 2.12 Pitch, Roll, dan Yaw pada pesawat



Gambar 2.13 Perubahan Roll pada *Quadrotor*



Gambar 2.14 Perubahan Pitch pada *Quadrotor*



Gambar 2.15 Perubahan Yaw pada *Quadrotor*

Beberapa gambar diatas menunjukan ketiga perubahan orientasi quadrotor, sehingga quadrotor dapat stabil atau orientasinya tetap pada keadaan *hovering*.

2.5 Pemodelan Euler – Newton

Tujuan utama dari metode *Euler newton* ini adalah untuk memodelkan suatu benda dalam hal ini yaitu *quadrotor* yang terdiri dari 6 *Degree of Freedom* dalam suatu persamaan. Untuk mengetahui pusat massa dari *quadrotor* sendiri yang mana diasumsikan simetris maka menggunakan persamaan (A.7) pada lampiran.

$$\begin{bmatrix} m\,\boldsymbol{I}_{3z3} & \boldsymbol{0}_{3x3} \\ \boldsymbol{0}_{3x3} & \boldsymbol{I} \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{V}}^B \\ \dot{\boldsymbol{\omega}}^B \end{bmatrix} + \begin{bmatrix} \omega^B & X & (m\,\boldsymbol{V}^B) \\ \omega^B & X & (\boldsymbol{I}\,\boldsymbol{\omega}^B) \end{bmatrix} = \begin{bmatrix} \boldsymbol{F}^B \\ \boldsymbol{\tau}^B \end{bmatrix} \qquad (2.1)$$

Halaman ini sengaja dikosongkan

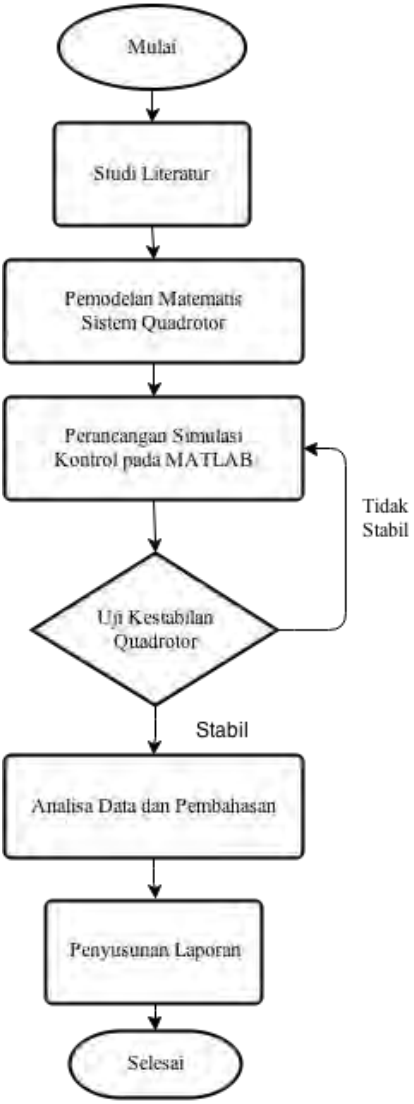
BAB III
METODOLOGI PENELITIAN

3.1 Alur Penelitian

Pada penelitian ini dilakukan simulasi mengenai sistem pengendalian *Quadrotor* menggunakan neural network pada software MATLAB. Parameter *Quadrotor* yaitu berupa orientasi sudut *pitch*, *roll*, dan serta *altitude* (ketinggian) quadcopter yang menjadi input untuk pengendalian. Dan output yang dihasilkan yaitu kecepatan dari ke empat motor pada quadcopter agar menghasilkan kestabilan orienstasi dan posisi *Quadrotor*.

Pada penelitian ini ke empat variabel input berpengaruh terhadap 4 motor dan variabel input ini tergantung pada input yang lain. Misalnya pada keadaan pitch miring dengan sudut tertentu, motor akan mengatur kecepatan putar agar *attitude Quadrotor* kembali horizontal dan stabil, apabila dari ke empat motor tidak saling menyesuaikan dengan kata lain tidak stabil, maka *attitude Quadrotor* dapat dikatakan buruk, karena kondisi saat terbang (*hovering*) tidak stabil.

Agar mendapatkan hasil yang optimal, maka sebelum penelitian ini dimulai, dilakukan terlebih dahulu studi literatur mengenai *Quadrotor* dan *neural network* sebagai sistem pengendaliannya. Setelah itu mencari persamaan model matematis *Quadrotor* sebagai acuan perancangan pengendalian *Quadrotor*. Lalu merancang sistem kontrol *neural network* berdasarkan pemodelan matematis yang telah di dapatkan sebelumnya. Uji kinerja sistem pengendalian *Quadrotor* diperlukan untuk mengetahui apakah sistem pengendalian yang telah dirancang telah berhasil agar *Quadrotor* dapat melakukan manuver *hovering*. Untuk alur selengkapnya dapat di lihat pada diagram-alir pada Gambar 3.1 :

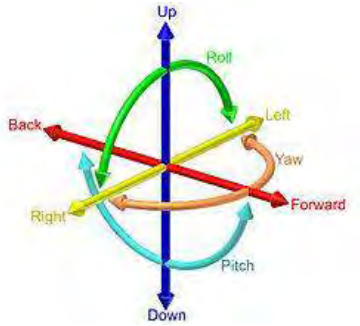


Gambar 3.1 Diagram alir penelitian

3.2 Pemodelan sistem *Quadrotor*

Quadrotor adalah helikopter yang terdiri dari 4 motor dan propeller yang terdapat pada ujung *frame* yang menyilang. 2 motor berputar searah jarum jam dan 2 motor yang lain berputar berlawanan arah jarum jam. Quadcopter yang digunakan pada simulasi adalah jenis plus (+). Untuk memodelkan 6 DOF (*Degree of Freedom*) *Quadrotor* di perlukan beberapa asumsi sebagai berikut[4][9] :

- Struktur *Quadrotor* simetris dan kaku
- Struktur Propeller kaku
- *Ground Effect* Diabaikan



Gambar 3.2 Penjabaran 6 DOF (*Degree of Freedom*)

Perlunya beberapa asumsi di atas digunakan untuk memudahkan pemodelan *Quadrotor 6 DOF*. Kinematika untuk pemodelan *Quadrotor* memerlukan dua refernsi *frame*[9] yaitu sebagai berikut:

- Referensi Inersia pada Bumi (E-*frame*)
- Referensi Body-fixed (B-*frame*)

Pemodelan *Quadrotor* dilakukan dengan metode pemodelan persamaan *Newton-Euler*, dan penurunan rumus selengkapnya dapat dilihat pada Lampiran A.

$$\dot{\xi} = J_\theta \, v \tag{3.1}$$

$\dot{\boldsymbol{\xi}}$ [+] adalah vektor kecepatan dari E-*frame*, v [+] adalah vektor kecepatan pada B-*frame*, J_θ [-] adalah matriks invers generalisasi.

$$\boldsymbol{\xi} = [\boldsymbol{\Gamma}^E \; \boldsymbol{\Theta}^E]^T = [X \; Y \; Z \; \phi \; \theta \; \psi]^T \tag{3.2}$$

ξ [+] terdiri dari Γ^E [m] dan Θ^E [rad] adalah masing-masing vektor posisi linear dan posisi anguler E-*frame*.

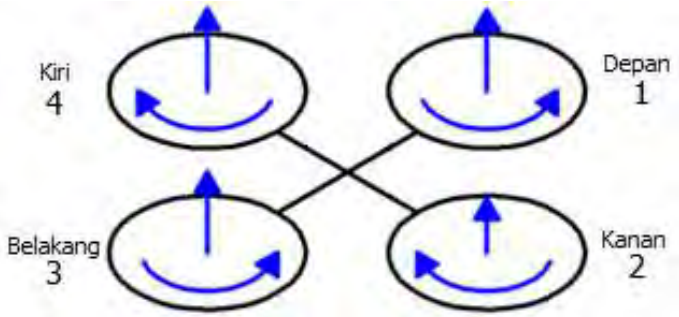
$$\boldsymbol{v} = [\boldsymbol{V}^B \; \boldsymbol{\omega}^B]^T = [u \; v \; w \; p \; q \; r]^T \tag{3.3}$$

$\boldsymbol{v}$ [+] juga terdiri dari $\boldsymbol{V}^B$[m s^{-1}] dan $\boldsymbol{\omega}^B$[rad s^{-1}] yang masing masing adalah vektor kecepatan linear dan kecepatan anguler pada B-*frame*.

Dengan mempertimbangakan Masa dan Inersia dari *Quadrotor*, Maka pemodelan *newton-euler* nya adalah sebagai berikut.

$$\begin{bmatrix} m\,\boldsymbol{I}_{3z3} & \boldsymbol{0}_{3x3} \\ \boldsymbol{0}_{3x3} & \boldsymbol{I} \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{V}}^B \\ \dot{\boldsymbol{\omega}}^B \end{bmatrix} + \begin{bmatrix} \boldsymbol{\omega}^B & X & (m\,\boldsymbol{V}^B) \\ \boldsymbol{\omega}^B & X & (\boldsymbol{I}\,\boldsymbol{\omega}^B) \end{bmatrix} = \begin{bmatrix} \boldsymbol{F}^B \\ \boldsymbol{\tau}^B \end{bmatrix} \tag{3.4}$$

Dimana $\boldsymbol{I}_{3x3}$ adalah matriks identitas dengan dimensi 3x3. $\boldsymbol{V}^B[m\,s^{-2}]$ adalah vektor akselarasi linear B-*frame Quadrotor* dan $\boldsymbol{\omega}^B[rad\,s^{-2}]$ adalah vektor percepatan anguler B-*frame Quadrotor*. $\boldsymbol{F}^B$ $[N]$ adalah vektor gaya *Quadrotor* dan $\boldsymbol{\tau}^B$ $[N\,m]$ adalah vektor torsi dari *Quadrotor* pada B-*frame*. Berdasarkan perhitungan yang terdapat pada lampiran A maka didaptkan persamaan untuk kecepatan anguler total propeller Ω [rad s^{-1}], dan vektor kecepatan anguler total propeller $\boldsymbol{\Omega}$ [rad s^{-1}].



Gambar 3.3 Penomoran propeller *Quadrotor*

$$\Omega = -\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4 \qquad \boldsymbol{\Omega} = \begin{bmatrix} \Omega_1 \\ \Omega_2 \\ \Omega_3 \\ \Omega_4 \end{bmatrix} \qquad (3.5)$$

Dimana Ω_1 [rad s^{-1}] adalah kecepatan propeller depan, Ω_2 [rad s^{-1}] adalah kecepatan propeller kanan, Ω_3 [rad s^{-1}] adalah kecepatan propeller belakang, dan Ω_4 [rad s^{-1}] adalah kecepatan propeller kiri. Faktor selanjutnya adalah gaya dan torsi yang dihasilkan oleh putaran propeller.

$$\boldsymbol{U_B(\Omega)} = \boldsymbol{E_B \Omega^2} = \begin{bmatrix} 0 \\ 0 \\ U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ b\left({\Omega_1}^2 + {\Omega_2}^2 + {\Omega_3}^2 + {\Omega_4}^2\right) \\ b\, l\left({\Omega_4}^2 - {\Omega_2}^2\right) \\ b\, l\left({\Omega_3}^2 - {\Omega_1}^2\right) \\ d\left({\Omega_2}^2 + {\Omega_4}^2 - {\Omega_1}^2 - {\Omega_3}^2\right) \end{bmatrix} \qquad (3.6)$$

Menurut prinsip Aerodinamika, bahwa gaya dan torsi sebanding dengan kecepatan propeller kuadrat. Maka E_B, matrik perpindahan posisi pada E-*frame* dikalikan dengan Ω^2 untuk mendapatkan matrik perpindahan posisi pada B-*frame* $U_B(\Omega)$ [+]. Sedangkan b [N s^2] adalah *thrust factor* dan d [N m s^2] adalah *drag factor*, l [m] adalah jarak antara titik tengah *Quadrotor* dan titik tengah propeller. U_1 [N] adalah *throtle* atau *thrust* ke atas *Quadrotor*, U_2 [N m] torsi *roll* terhadap *Quadrotor*, U_3 [N m] torsi *pitch* terhadap *Quadrotor*, U_4 [N m] torsi *yaw* terhadap *Quadrotor*.

Lalu dengan memindahkan seluruh persamaan diruas kiri kecuali $\dot{\boldsymbol{v}}$ pada persamaan (3.17) pada lampiran maka didapatkan persamaan sebagai berikut:

$$\dot{\boldsymbol{v}} = \boldsymbol{M_B}^{-1}(-\boldsymbol{C_B(v)}\,\boldsymbol{v} + G_B(\boldsymbol{\xi}) + \boldsymbol{O_B(v)}\,\boldsymbol{\Omega} + E_B\Omega^2) \quad (3.7)$$

Dari persamaan (3.7) yang masih berbentuk persamaan matriks dapat pula ditulis dalam persamaan matematis yang dapat dilihat pada persamaan (3.8) sampai persamaan (3.11), serta untuk persamaan propeller nya dapat dilihat pada persamaan (3.12) sampai persamaan (3.15)

$$\ddot{w} = (u\ g - v\ \dot{p}) - g\ Cos_{\phi} + \frac{U_1}{m} \quad (3.8)$$

$$\ddot{p} = \frac{I_{YY} - I_{ZZ}}{I_{XX}} \dot{q}\ \dot{r} - \frac{J_{TP}}{I_{XX}} \dot{q}\ \Omega + \frac{U_2}{I_{XX}} \quad (3.9)$$

$$\ddot{q} = \frac{I_{ZZ} - I_{XX}}{I_{YY}} \dot{p}\ \dot{r} - \frac{J_{TP}}{I_{YY}} \dot{p}\ \Omega + \frac{U_2}{I_{YY}} \quad (3.10)$$

$$\ddot{r} = \frac{I_{XX} - I_{YY}}{I_{ZZ}} \dot{p}\ \dot{q} + \frac{U_4}{I_{ZZ}} \quad (3.11)$$

$$U_1 = b\left(\Omega_1{}^2 + \Omega_2{}^2 + \Omega_3{}^2 + \Omega_4{}^2\right) \quad (3.12)$$

$$U_2 = b\ l\ \left(\Omega_4{}^2 - \Omega_2{}^2\right) \quad (3.13)$$

$$U_3 = b\ l\ \left(\Omega_3{}^2 - \Omega_1{}^2\right) \quad (3.14)$$

$$U_4 = d\ \left(\Omega_2{}^2 + \Omega_4{}^2 - \Omega_1{}^2 - \Omega_3{}^2\right) \quad (3.15)$$

Persamaan (3.8) – (3.15) yang digunakan pada Pemodelan 6 DOF *Quadrotor*. Namun pada pemodelan selanjutnya akan digunakan referensi inersia baru dari kombinasi antara persamaan translasi E-*frame* dan persamaan rotasi B-*frame* yang dinamakan H-*frame* (*Hybrid*), hal ini dilakukan untuk memudahkan memodelkan dengan pemodelan sistem pengendalian.

$$\zeta = [\dot{\Gamma}^E\ \omega^B]^T = [\dot{X}\ \dot{Y}\ \dot{Z}\ p\ q\ r]^T \quad (3.16)$$

persamaan (3.29) menunjukan vektor kecepatan pada H-*frame,* dan dapat ditulis dalam bentruk matrik sebagai berikut:

$$\boldsymbol{M}_H \dot{\boldsymbol{\zeta}} + \boldsymbol{C}_H(\boldsymbol{\zeta})\ \boldsymbol{\zeta} = G_H + \boldsymbol{O}_H(\boldsymbol{\zeta})\ \boldsymbol{\Omega} + \boldsymbol{E}_H(\boldsymbol{\xi})\boldsymbol{\Omega}^2 \quad (3.17)$$

$\dot{\boldsymbol{\zeta}}$ [+] adalah vektor percepatan pada H-frame. Matriks inersia sistem *Quadrotor* pada H-*frame*, bernilai sama dengan yang digunakan pada B-*frame* berdasarkan persamaan (3.7). Namun pada matriks *Coriolis-centripetal* pada H-frame tidak sama seperti yang ada pada B-*frame*.

Efek gyroskopik yang terjadi hanya berpengaruh pada persamaan anguler pada B-frame, maka matriks gyroskopik propeller pada H-frame atau $\boldsymbol{O}_H(\boldsymbol{\zeta})$ [+] didapatkan berdasarkan persamaan (3.13). Matriks gerakan *Quadrotor* pada H-frame juga berbeda terhadap B-frame, karena U_1 mempengaruhi persamaan translasi dan matriks rotasi R_θ.

Dengan memindahkan seluruh persamaan pada ruas kiri dan menyisakan vektor kecepatan pada H-*frame* $\dot{\boldsymbol{\zeta}}$ [+] menuju ruas

kanan pada persamaan (A.36) pada lampiran, maka didapatkan persamaan sebagai berikut.

$$\ddot{Z} = -g + \left(Cos_{\theta} Cos_{\phi}\right) \frac{U_1}{m} \quad (3.18)$$

$$\ddot{p} = \frac{I_{YY} - I_{ZZ}}{I_{XX}} \dot{q}\, \dot{r} - \frac{J_{TP}}{I_{XX}} \dot{q}\, \Omega + \frac{U_2}{I_{XX}} \quad (3.19)$$

$$\ddot{q} = \frac{I_{ZZ} - I_{XX}}{I_{YY}} \dot{p}\, \dot{r} - \frac{J_{TP}}{I_{YY}} \dot{p}\, \Omega + \frac{U_2}{I_{YY}} \quad (3.20)$$

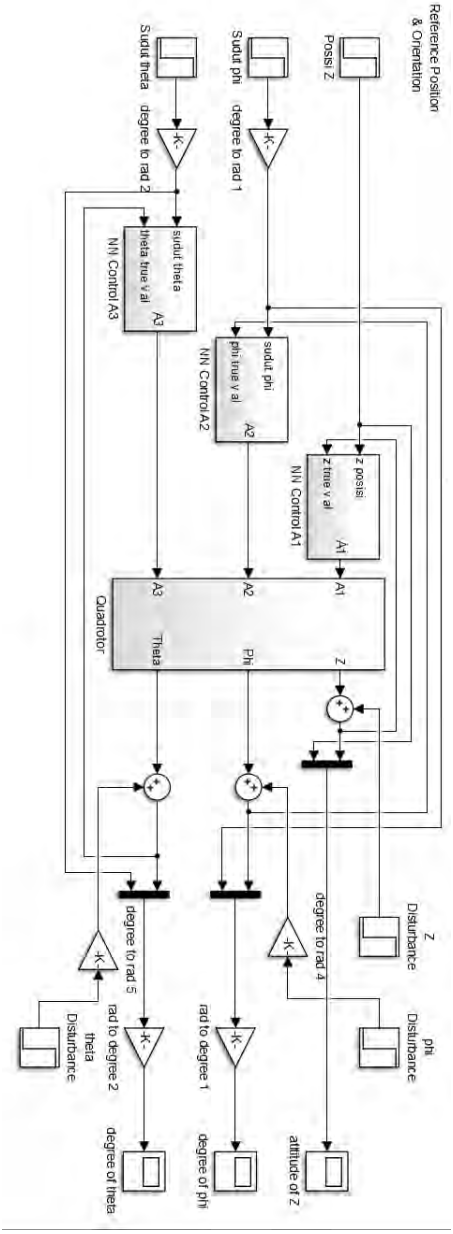
$$\ddot{r} = \frac{I_{XX} - I_{YY}}{I_{ZZ}} \dot{p}\, \dot{q} + \frac{U_4}{I_{ZZ}} \quad (3.21)$$

Untuk persamaan propeller pada H-*frame*, sama dengan persamaan propeller pada B-*frame*. Untuk memudahkan perhitungan maka referensi pada H-frame yang digunakan pada pemodelan *Quadrotor* pada penelitian ini. Karena dalam penelitian ini yang di kendalikan adalah orientasi sudut dan *altitude* (ketinggian) *Quadrotor* dengan kata lain mengendalikan kestabilan pada kondisi Hovering, maka tidak semua persamaan digunakan, hanya orientasi roll, pitch, dan *altitude* (ketinggian) *Quadrotor*, atau hanya menggunakan persamaan (3.12), (3.13), (3.14), (3.15) untuk propeller, dan orientasi posisi *roll*, *pitch* pada persamaan (3.18), (3.19), (3.20).

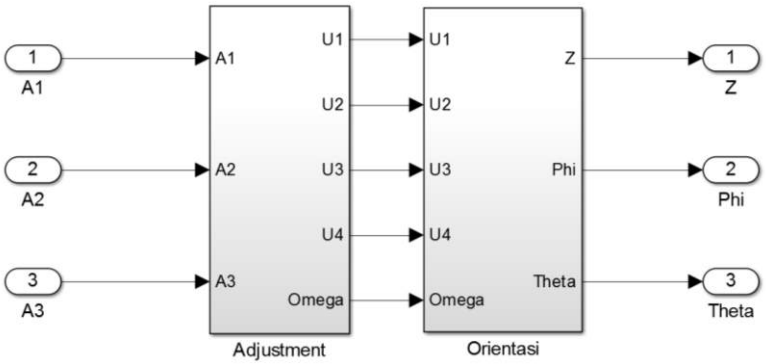
3.3 Simulasi Sistem pada MATLAB

Untuk mengetahui sistem kerja dari pemodelan *Quadrotor*, maka dilakukan simulasi sistem pada simulink berdasarkan persamaan yang telah di dapatkan. Simulasi sistem menggunakan software MATLAB dengan fitur Simulink.

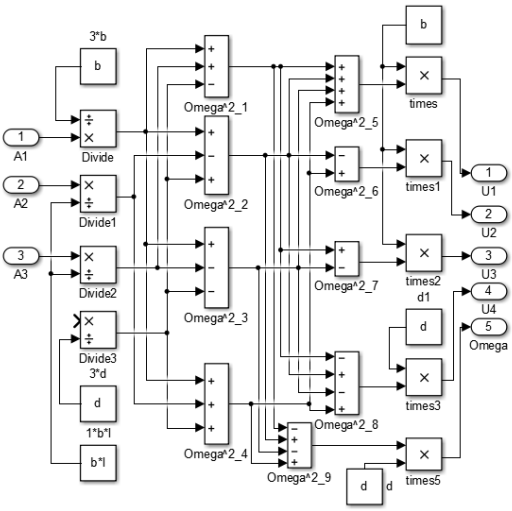
Pada gambar 3.4 dapat di lihat blok sistem *Quadrotor* yang di dalamnya terdiri dari perhitungan omega kuadrat, perhitungan sudut, dan perhitungan posisi, dengan Phi adalah sudut *error roll*, Theta adalah sudut *error pitch*, dan Z adalah *altitude* (ketinggian) *Quadrotor*. Input dari sistem *Quadrotor* tersebut adalah Adjustment yang di gunakan untuk mengatur *attitude Quadrotor* berdasarkan kecepatan motor pada persamaan (3.22) sampai persamaan (3.25)[11][12].



Gambar 3.4 Arsitektur Sistem Pengendalian *Quadrotor*



Gambar 3.5 Sistem *Quadrotor*

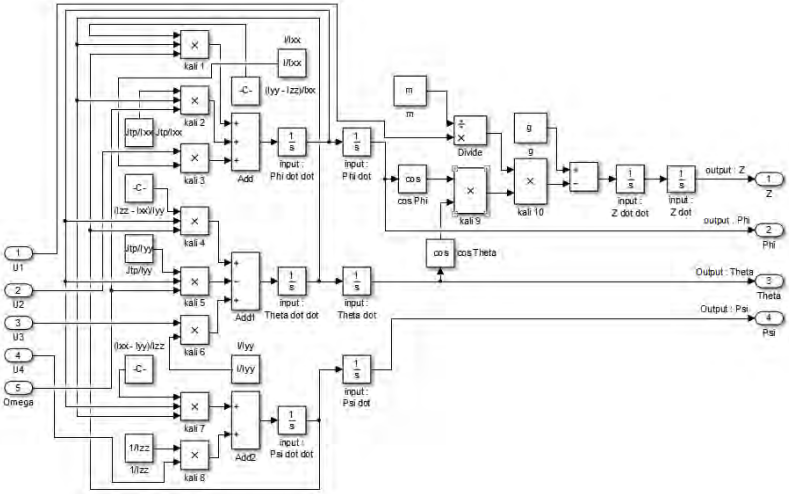


Gambar 3.6 Sistem *Adjustment Parameter Quadrotor*

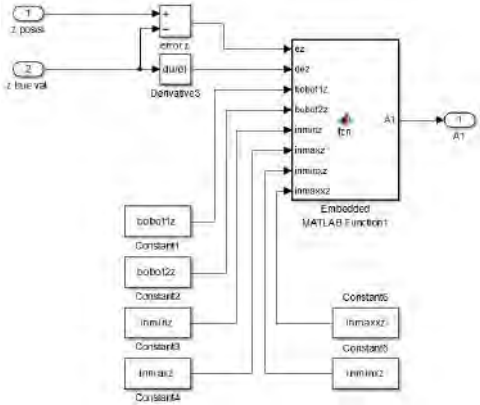
$$\Omega_1 = A_1 - A_3 \quad (3.22)$$
$$\Omega_2 = A_1 - A_2 \quad (3.23)$$
$$\Omega_3 = A_1 + A_3 \quad (3.24)$$
$$\Omega_4 = A_1 + A_3 \quad (3.25)$$

Rancangan pada gambar 3.6 sesuai dengan persamaan (3.12-3.15) yang nantinya akan menghasilkan perhitungan bantuan untuk persamaan (3.18-3.21). A1, A2, A3 adalah masing-masing

Adjustment atau penyesuaian yang berfungsi sebagai input yang nantinya berpengaruh pada kecepatan motor sehingga dapat menstabilkan *attitude Quadrotor*[11]. Untuk posisi dari *Quadrotor* dilakukan di blok yang berbeda untuk mempermudah perhitungan dan dapat dilihat pada gambar 3.7.



Gambar 3.7 Sistem Orientasi dan Posisi *Quadrotor*



Gambar 3.8 Diagram Salah satu kontrol parameter *Quadrotor*

Nilai dari U1, U2, U3, U4, dan Omega yang telah ditentukan akan diolah pada sistem *Quadrotor* pada gambar 3.7 yang akan menghasilkan orientasi dan posisi dari *Quadrotor*.

3.4 Perancangan Sistem Pengendalian *Attitude Quadrotor*

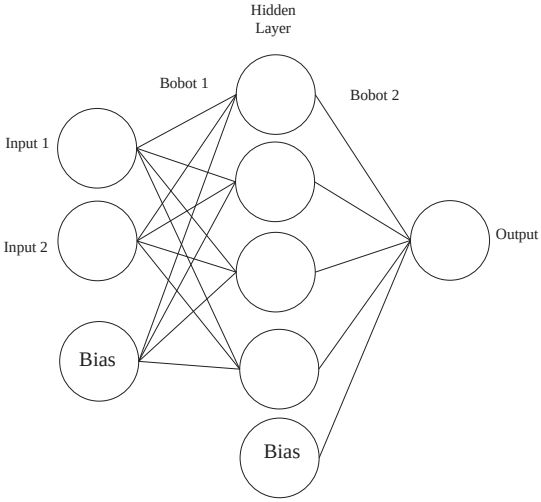
Untuk sistem pengendalian attitude, dilakukan kontrol pada masing-masing parameter *Quadrotor* (*roll, pitch, altitude*). Jenis pengendalian yang digunakan adalah *Neural Network*. Untuk blok pengendalian pada simulink, digunakan library MATLAB *function* yang berisi logika NN. Nilai bobot didapatkan pada training Neural Network. Jenis training yang digunakan yaitu *backproppagation*. Beberapa parameter NN (*Neural Network*) adalah jumlah *neuron* pada *hidden layer*, jumlah *epoch* serta *learning rate.*

3.5 Data Training Neural Network

Berdasarkan pemodelan *Quadrotor* pada bab 3.2 di dapatkan input dan target sebagai data training NN (*Neural Network*). masing masing parameter, diberi kontrol NN agar dapat mencapai posisi dan orientasi yang di inginkan. Sebelum merancang kontrol NN, dilakukan terlebih dahulu training untuk mencari bobot *input* dan *output* dengan menggunakan *Backpropagation Neural Network* (BPNN). Seluruhnya terdapat data training untuk masing masing parameter *Quadrotor* yaitu *altitude*, *roll*, *pitch*. Untuk data selengkapnya dapat dilihat pada Lampiran B. Target dari sistem adalah ke tiga *Adjustment* dan input berupa *error* orientasi dan posisi, serta nilai sebenarnya dari posisi maupun orientasi *Quadrotor*.

Data training diambil dari *project* simulasi *Quadrotor* berdasarkan jurnal dari *Samir Bouabdallah*[13]. Data yang diambil berupa error, selisih error, dan Nilai *Adjusment* hasil pengendalian. Pengambilan data training yaitu dengan cara mensimulasikan selama 10 detik, dan total seluruh data dalam 50 detik adalah sebanyak 144 data, input yang digunakan yaitu fungsi step, dan step time 1 detik, serta final value bernilai 1 untuk ketiga parameter *Quadrotor*. Setelah di plot menjadi sebuah grafik, diambil sebanyak 77 data sebagai data training untuk sistem pengendalian attitude *Quadrotor*.

Untuk Model NN yang di gunakan pada masing-masing parameter *Quadrotor* adalah sebagai berikut.

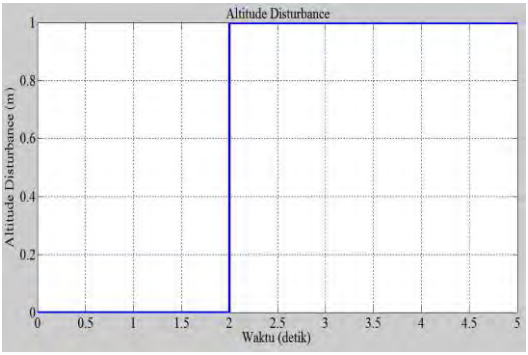


Gambar 3.9 Model *Neural Network*

Gambar 3.9 mengilustrasikan Model NN dari ketiga parameter Quadrotor, namun parameter NN dari masing masing *Altitude, roll*, dan *pitch* nya yang berbeda. Model NN pada *Altitude* terdiri dari dua input dan satu output. Masing masing input adalah Error posisi dan selisih Error posisi, sedangkan outputnya adalah *thrust* (U1 pada persamaan 3,12) dari *quadrotor*. *Hidden Layer* pada *Altitude Neural Network* terdiri dari 75 *neuron* dengan *Learning rate* sebesar 0,5. Sedangkan untuk model NN pada *Roll* terdiri dari input yaitu *Error* orientasi *Roll* dan selisih *Error* orientasi *Roll*, sedangkan outputnya adalah *torsi* pada *roll* (U2 pada persamaan 3.13) memiliki jumlah *neuron* pada *hidden layer* sebanyak 70, dengan *learning rate* sebesar 0,25. lalu pada *pitch* jumlah *neuron* nya sebanyak 80 dengan *learning rate* sebesar 0,1. input yaitu *Error* orientasi *Pitch* dan selisih *Error* orientasi *Pitch*, sedangkan outputnya adalah *torsi* pada *Pitch* (U3 pada persamaan 3.14)

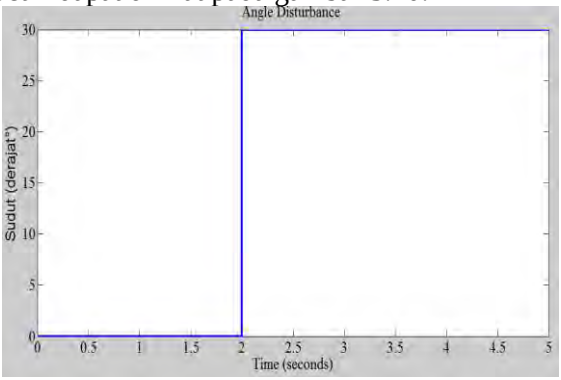
3.6 Disturbance *Quadrotor*

Untuk pengujian kestabilan *Quadrotor* pada saat *hovering*, maka perlu diberi sebuah *disturbance* atau gangguan yang dapat merubah *attitude* dari *Quadrotor*. Untuk penelitian ini gangguan yang diberikan pada *Quadrotor* adalah angin disaat *Quadrotor* tersebut sedang melakukan *hovering*. Untuk merepresentasikan gangguan berupa angin, pada simulink di berikan fungsi step, yang artinya gangguan angin tersebut konstan atau tetap secara terus-menerus.



Gambar 3.10 Fungsi Step Untuk *Disturbance* pada *Altitude*

Gangguan berupa grafik step untuk *altitude* sebesar 1 m pada waktu 2 detik dapat dilihat pada gambar 3.10.



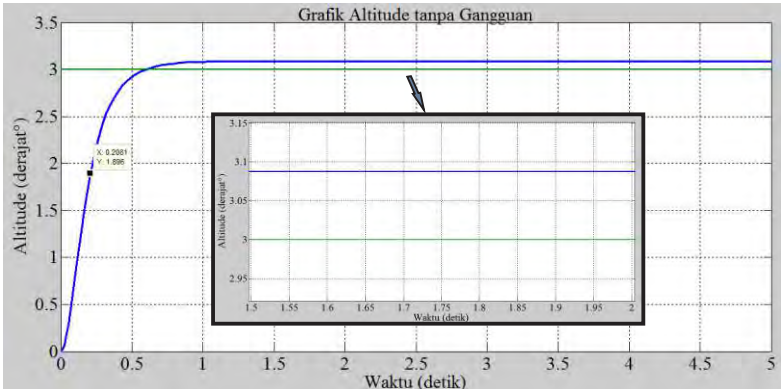
Gambar 3.11 Fungsi Step Untuk *Disturbance* pada *Roll dan Pitch*

Sedangkan untuk gangguan pada sudut roll, dan pitch adalah pada gambar 3.11, yang berupa gangguan sudut dengan fungsi step sebesar 30^o pada detik ke 2.

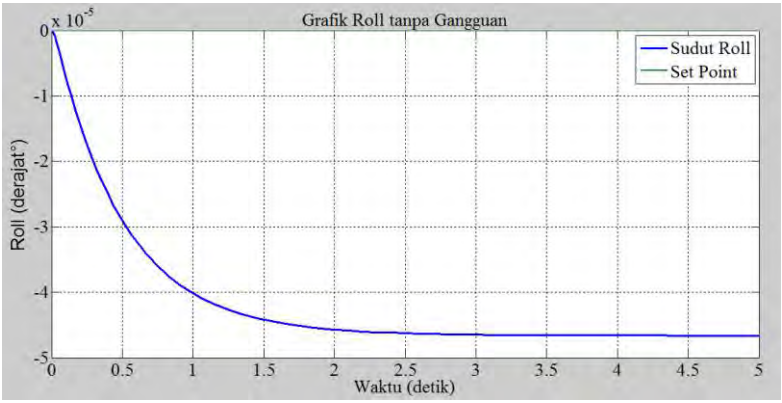
BAB IV
ANALISA DATA DAN PEMBAHASAN

4.1 Uji Sistem *Quadrotor*

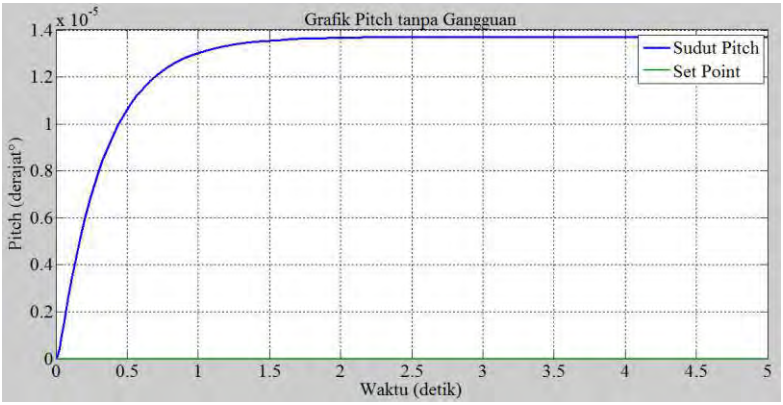
Pemodelan *Quadrotor* yang telah didapatkan, di simulasikan dengan simulink pada MATLAB. Sebelum diberikan gangguan pada tiga parameter quadrotor yaitu *Altitude (ketinggian), Roll*, dan *Pitch*, terlebih dahulu dilakukan uji sistem tanpa gangguan dengan waktu simulasi 5 detik yang dapat dilihat pada gambar 4.1 sampai 4.3.



Gambar 4.1 Uji *Altitude Quadrotor* tanpa gangguan



Gambar 4.2 Uji *Roll Quadrotor* tanpa gangguan

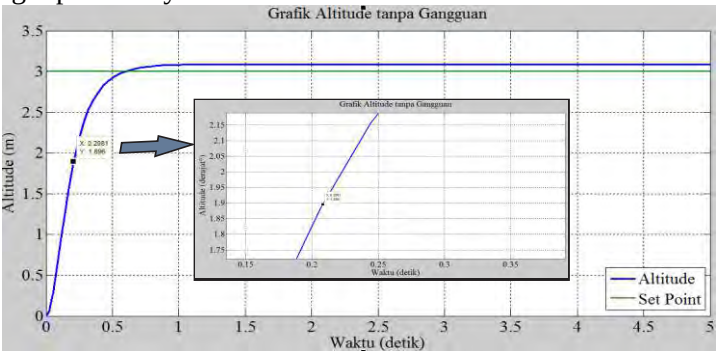


Gambar 4.3 Uji *Pitch Quadrotor* tanpa gangguan

Pada gambar 4.1 menunjukan grafik respon yang *Steady* mendekati set point sebesar 3 m. Sedangkan untuk gambar 4.2 dan gambar 4.3 menunjukan respon sudut *Roll*, dan *Pitch* dari *quadrotor*. Untuk *Error Steady State* dari respon *Altitude* tanpa diberi gangguan adalah sebagai berikut. Karena karakteristik sistem *quadrotor* merupakan orde-1[13] berdasarkan grafik responnya, maka *Time Constant* dari grafik respon yaitu pada 63.2% set point.

$$Set\ Point\ .\ 63{,}2\% = 3\ .\ 63{,}2\%$$
$$= 1{,}896$$

Waktu pada *Altitude* 1,896 m menurut gambar 4.4 dan 4.1 yang diperbesar yaitu 0.2081 detik.

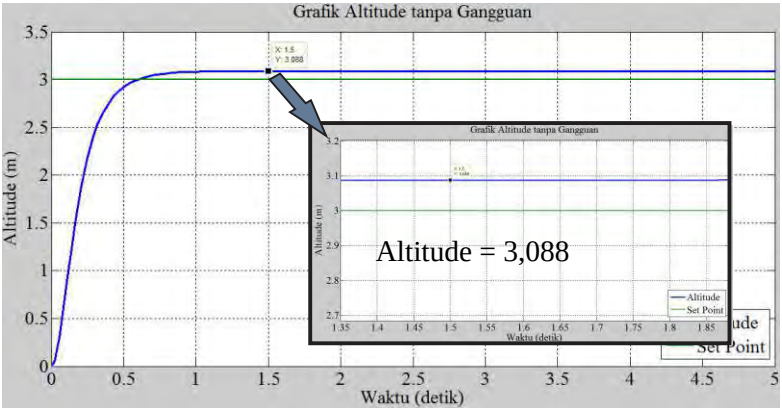


Gambar 4.4 *Time Constant Altitude*

T = 0,2081 detik (4.1)

Waktu mencapai *Steady State* = 4 . T [13] (4.2)

Dimana T adalah *Time Constant*. Sebuah respon orde-1 mencapai keadaan *Steady* pada 4.T [13]. Maka pada *Altitude* dicapai keadaan *Steady* pada waktu 0.8324 detik. Apabila diambil nilai pada saat waktu 1,5 detik yang mana melebihi dari minimal waktu untuk *Steady* maka didapatkan nilai *Steady State* sebesar 3.013 m berdasarkan gambar 4.5.



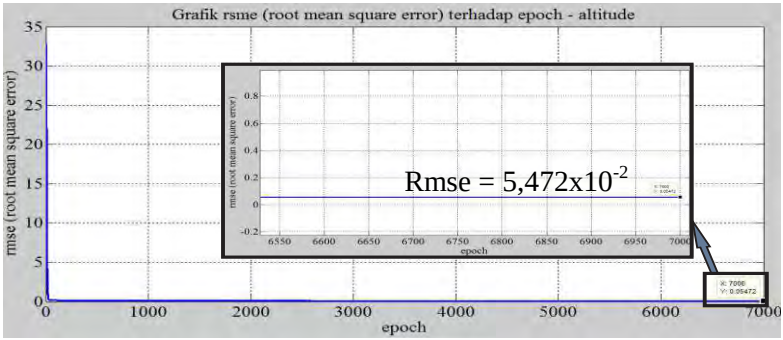
Gambar 4.5 Nilai *Steady State Altitude* tanpa gangguan

Dengan set point sebesar 3 m dan ketinggian *quadrotor* 3,088 m. Maka e_{ss} atau *Error Steady State* nya sebesar 0,088 m.

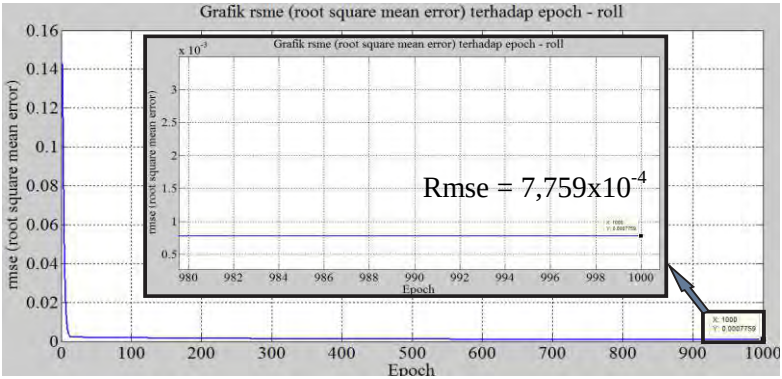
4.2 *Neural Network Training*

Dari data training yang di dapatkan pada tabel yang terdapat pada lampiran B, maka dilakukan pembelajaran *supervised learning, backpropagation* berdasarkan target dan Input pada data training. Dengan pengujian beberapa parameter NN (*Neural Network*) yaitu jumlah neuron pada *hidden layer*, jumlah *epoch* serta *learning rate*, dan di dapat beberapa *Error* maka di pilih *Error* yang relatif kecil yang digunakan pada sistem *Quadrotor*. Untuk penentuan dari ketiga parameter NN tersebut

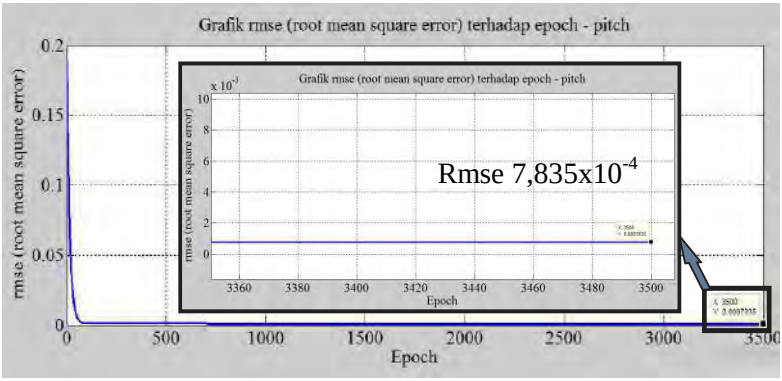
menggunakan trial and *Error*.Pada gambar 4.5 sampai gambar 4.7 menunjukan grafik perbandingan Antara *rmse* atau *root mean square Error* (sumbu y) pembelajaran input terhadap target *Quadrotor* dan *epoch* (sumbu x) pada training NN.



Gambar 4.6 Kurva *Error* bobot pada *Altitude*



Gambar 4.7 Kurva *Error* bobot pada *Roll*



Gambar 4.8 Kurva *Error* bobot pada *Pitch*

parameter NN didapatkan melalui proses *Trial and Error*, dengan kombinasi parameter NN (Jumlah *neuron*, jumlah *epoch*, dan *learning rate*) yang mempunyai nilai *Error* relatif paling kecil dibandingkan dengan *Trial* and *Error* sebelumnya.

4.2 Uji Kestabilan *Quadrotor*

Setelah dimasukan nilai bobot terhadap sistem kontrol *Neural Network*, maka dilakukan pengujian dengan *Disturbance* (gangguan) terhadap kestabilan *Quadrotor*.

Tabel 4.1 Tabel pengujian *Disturbance* pada *Quadrotor*

No.	*Altitude*	*Roll*	*Pitch*
1	**v**		
2		**v**	
3			**v**
4	**v**	**v**	
5	**v**		**v**
6		**v**	**v**
7	**v**	**v**	**v**

Pengujian kestabilan (*Attitude*) *Quadrotor* dilakukan dengan berbagai kombinasi gangguan masing masing parameter. Beberapa ketentuan yang digunakan saat uji gangguan yaitu:

- Simulasi dilakukan dalam waktu 5 detik
- Gangguan *Altitude* yang diberikan sebesar 1 m
- Gangguan Sudut (*Roll* dan *Pitch*) yang diberikan sebesar 30°
- Set point *Altitude Quadrotor* sebesar 3 m
- Set point sudut (*Roll* dan *Pitch*) yang diberikan sebesar 0°
- Gangguan dilakukan selama 0.2 detik pada detik ke-2

Pada tabel 4.1 dilakukan 7 jenis pengujian gangguan terhadap *Attitude Quadrotor*. Nilai **v** adalah keterangan bahwa dilakukan pengujian *Disturbance* dengan parameter quadrotor yaitu pada *Altitude (ketinggian), Roll,* dan *Pitch*.

4.2.1 *Altitude Disturbance*

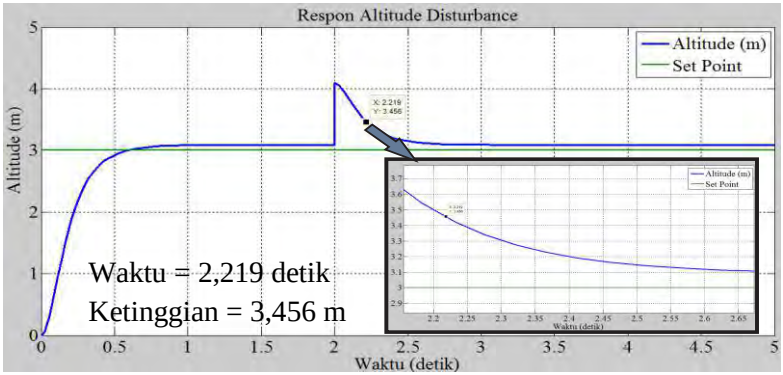
Berdasarkan pengujian pada sub-bab 4.1, dimana posisi *quadrotor* meningkat dari 0 menuju set point yaitu 3 meter, diketahui nilai (*Time Constant*) pada persamaan 4.1, T = 0,1987 detik. Dan mencapai kondisi *Steady State* pada waktu 0.7948 detik maka apabila penambahan gangguan di waktu 2 detik, disaat respon telah *Steady*. *Time Constant* adalah saat 63,2% *Time Constant* pada disturbance adalah :

$$Set\ Point\ .\ 63{,}2\% = 1\ .\ 63{,}2\%$$
$$= 0.632 \qquad (4.3)$$

Karena *disturbance* dimulai pada ketinggian 4,088 m, maka *Time Constant* nya pada saat ketinggian :

$$4{,}088 - 0{,}632 = 3{,}456 \text{ m}$$

Dan waktu pada saat ketinggian bernilai 3,456 m adalah:



Gambar 4.9 hasil uji *disturbance* pada *Altitude*

Karena gangguan dimulai pada detik ke 2 maka

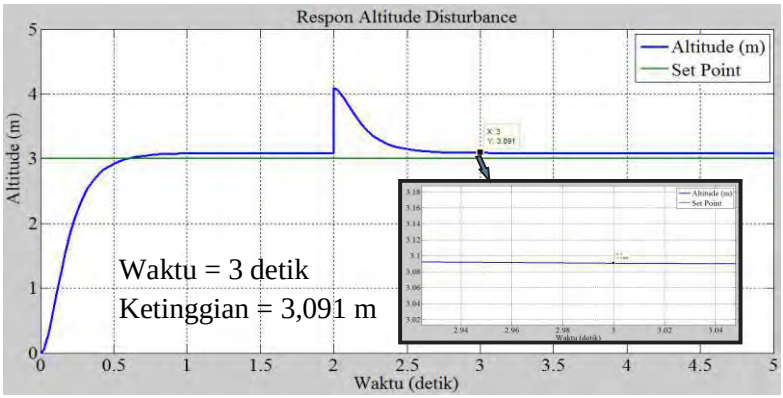
$$2{,}219 - 2 = 0{,}219$$

$$T_A = 0{,}219 \text{ detik} \qquad (4.4)$$

Waktu mencapai *Steady State* = $4\ T_A$

$$= 0{,}876 \text{ detik}$$

Dimana T_A adalah *Time Constant* untuk uji gangguan *Altitude*. Maka respon tersebut kembali stabil pada 2,876 detik, karena di jumlahkan dengan waktu awal *disturbance*. Bila di ambil nilai pada saat detik ke 3 dimana respon telah *Steady*, dapat diketahui nilai e_{ss} (*Error Steady State*).



Gambar 4.10 Nilai *Steady State* pada *Altitude*

Pada gambar 4.10 diketahui nilai *Altitude* pada waktu 3 detik sebesar 3,022 meter dan set point sebesar 3 meter. Maka nilai *Error Steady State* nya adalah :

$$e_{ss\ alt} = 3{,}091 - \text{Set Point}$$

$$e_{ss\ alt} = 0{,}091 \text{ meter} \qquad (4.4)$$

Nilai *Error Steady State* pada pengujian *Altitude* sebesar 0.091 meter, dengan *Time Constant* = 0,219 detik

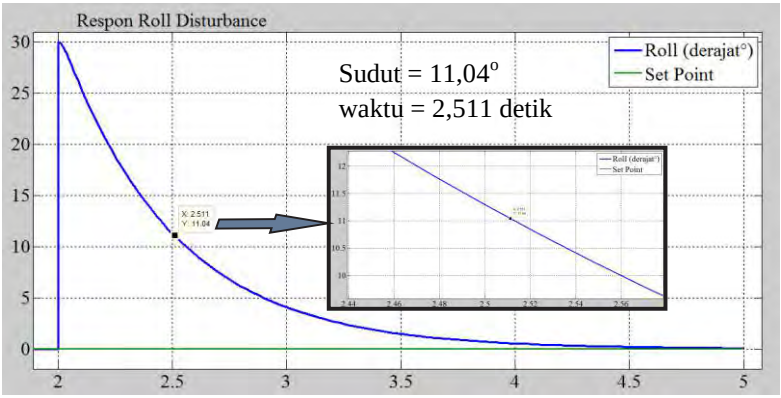
4.2.2 *Roll Disturbance*

Set point sudut *Roll* adalah sebesar 0^o, dan orientasi awal quadrotor yaitu pada sudut 0^o. Lalu diberi gangguan sebesar 30^o, pada waktu 2 detik, Maka *Time Constant* dari respon pada gangguan sudut *Roll* :

$$\text{Sudut} = 30 * 63.2\%$$
$$= 18{,}96$$

Namun karena nilai gangguan mula-mula di angka 30 maka:

$$30^o - 18{,}96^o = 11{,}04^o \qquad (4.5)$$



Gambar 4.11 hasil uji *disturbance* pada Sudut *Roll*

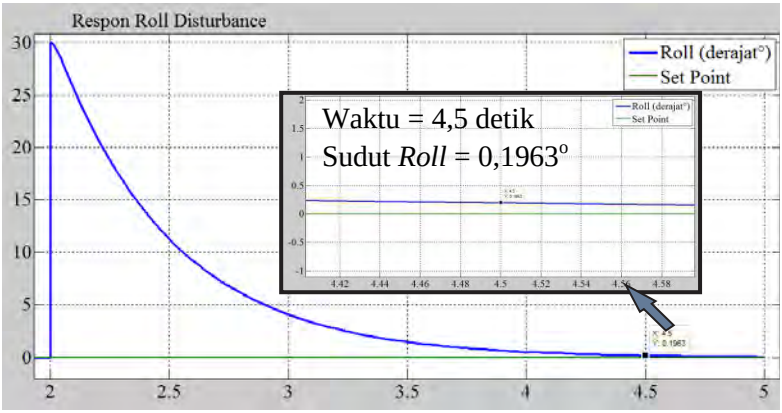
Waktu saat sudut $11{,}04^o$ menurut gambar 4.11 adalah 2,511 detik. Bila dikurangi oleh waktu mulai gangguan maka

$$T_R = 2{,}511 - 2$$
$$T_R = 0{,}511 \text{ detik} \qquad (4.6)$$

Dimana T_R adalah *Time Constant* pada gangguan *Roll*, waktu untuk mencapai *Steady* adalah $4.T_R$, maka waktu *Steady State* adalah

$$4 \, . \, T_R = 2{,}044 \text{ detik} \qquad (4.7)$$

terhitung setelah quadrotor menerima gangguan. Apabila terhitung dari waktu mula-mula quadrotor, maka pada waktu untuk mencapai *Steady State* dijumlah kan dengan waktu saat diberi gangguan yaitu sebesar 4,044 detik untuk mencapai *Steady State*. Bila di ambil nilai sudut saat mencapai waktu 4.5 maka dianggap pada kondisi *Steady State* sama dengan sudut pada waktu 4,016 detik. Nilai sudut pada saat detik 4.5 adalah sebesar $0{,}1963^o$ derajat menurut gambar 4.12.



Gambar 4.12 Nilai sudut *Steady State* pada *Roll disturbance*

dengan set point 0^o derajat maka di dapat *Error Steady State* pada *Roll* atau $e_{ss\ Roll}$ adalah :

$$e_{ss\ Roll} = 0{,}1963^{\,o} - 0^{\,o}$$
$$e_{ss\ Roll} = 0{,}1963^o \qquad (4.8)$$

Nilai *Error Steady State* pada pengujian *Roll* sebesar 0.1963^o, dengan *Time Constant* = 0,511 detik

4.2.3 *Pitch Disturbance*

Set point sudut *Pitch* yaitu sebesar 0^o dan sudut mula-mula sebesar 0^o sedangkan gangguan yang dialami oleh quadrotor pada sudut *Pitch* yaitu sebesar 30^o pada waktu 2 detik.

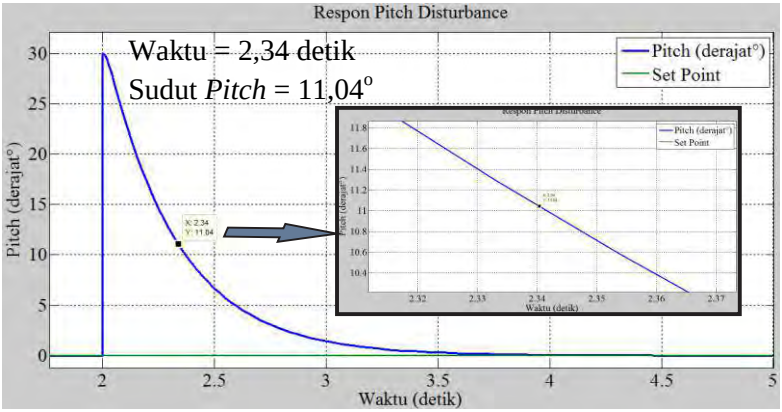
$$Set\ Point\ .\ 63{,}2\% = 30\ .\ 63{,}2\%$$
$$= 18{,}96^o \qquad (4.9)$$

Namun karena respon nya menurun dari nilai 30^o menuju set point yaitu 0^o maka

$$30^o - 18{,}96^o = 11{,}04^o \qquad (4.10)$$

waktu pada sudut $11{,}04^o$ adalah sebesar 2,34 detik. bila di hitung mulai dari waktu gangguan mula-mula yaitu pada detik ke 2. Maka nilai T_P atau *Time Constant* pada gangguan *Pitch* adalah

$$T_P = 2{,}34 - 2$$
$$T_P = 0{,}34 \text{ detik} \qquad (4.11)$$

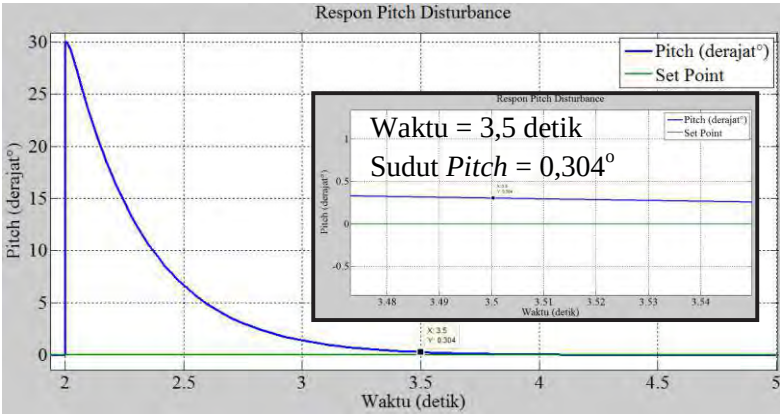


Gambar 4.13 Nilai *Time Constant* pada *Pitch disturbance*

Dengan nilai T_P adalah 0,34 detik maka menurut persamaan (4.2) nilai *Steady State* pada *Pitch disturbance*. Adalah :

$$4\ .\ T_P = 1.36 \text{ detik} \qquad (4.12)$$

Karena T_P dimulai pada saat respon menerima gangguan mula-mula pada detik ke 2 maka waktu keseluruhan untuk mencapai nilai *Steady State* adalah 3,36 detik. bila mengambil nilai pada detik 3,5 detik akan sama halnya dengan pada waktu 3,36 detik Karena telah mencapai *Steady State*.



Gambar 4.14 Nilai *Steady State* pada *Pitch Disturbance*

Karena set point sudut *Pitch* adalah 0^o maka nilai *Error Steady State* atau $e_{ss\ Pitch}$ pada *Pitch* adalah :

$$e_{ss\ Pitch} = 0{,}304^o - 0^o$$

$$e_{ss\ Pitch} = 0{,}304^o \qquad (4.13)$$

Nilai *Error Steady State* pada pengujian *Pitch* sebesar 0.304^o, dengan *Time Constant* = 0,34 detik.

4.2.4 *Altitude* dan *Roll Disturbance*

Pengujian kali ini yaitu memberikan *disturbance* pada 2 parameter sekaligus yaitu *Altitude* (ketinggian) dan sudut *Roll*. *Set Point Altitude* yaitu sebesar 3 m, dan *set point* dari sudut *Roll* adalah 0^o. pengujian pada *Altitude* diberi *disturbance* pada waktu 2 detik sebesar 1 m.

untuk *Time Constant* pada pengujian *Altitude* adalah :

$$\textit{Set Point} * 63.2\% = 1 * 63.2\%$$

$$= 0.632 \qquad (4.14)$$

Karena gangguan dimulai dari 4 meter menuju ke 3 meter maka :

$$4{,}088 - 0{,}632 = 3{,}456 \text{ m} \qquad (4.15)$$

Waktu pada saat ketinggian 3,456 m adalah:

Gambar 4.15 Nilai *Time Constant* dari *Altitude* pada *Altitude* dan *Roll disturbance*

Maka nilai *Time Constant* adalah :

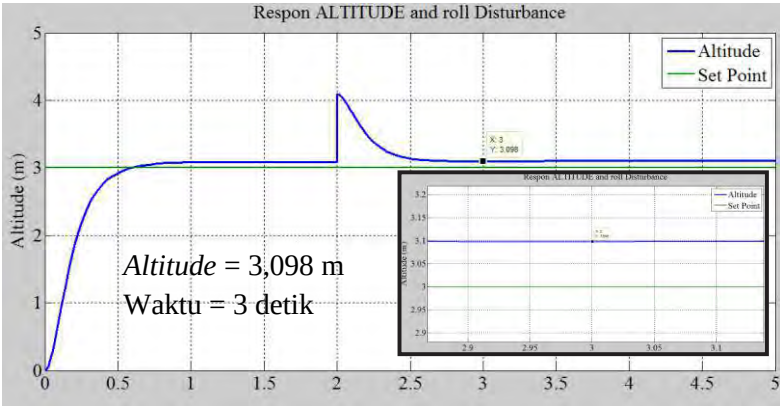
$$T_{A\,2} = 2{,}218 - 2$$

$$T_{A\,2} = 0{,}218 \text{ detik} \qquad (4.16)$$

Dimana $T_{A\ 2}$ adalah nilai *Time Constant Altitude* pada pengujian *disturbance Altitude* dan *Roll*. Nilai *Error Steady State* adalah :

$$4 \,.\, T_{A\,2} = 4 * 0{,}218 = 0.872 \text{ detik} \qquad (4.17)$$

Untuk *respon Steady State* terdapat penambahan waktu 2 detik karena *disturbance* dimulai pada waktu 2 detik. Maka respon pada waktu 2,872 detik telah stabil. Apabila diambil data *Altitude* pada waktu 3 detik akan sama halnya karena dianggap telah stabil.



Gambar 4.16 Nilai *Steady State* dari *Altitude* pada *Altitude* dan *Roll disturbance.*

Nilai Error Steady State untuk *Altitude* atau $e_{ss\ Altitude\ 2}$ adalah selisih dari nilai *Steady* dengan *set point* :

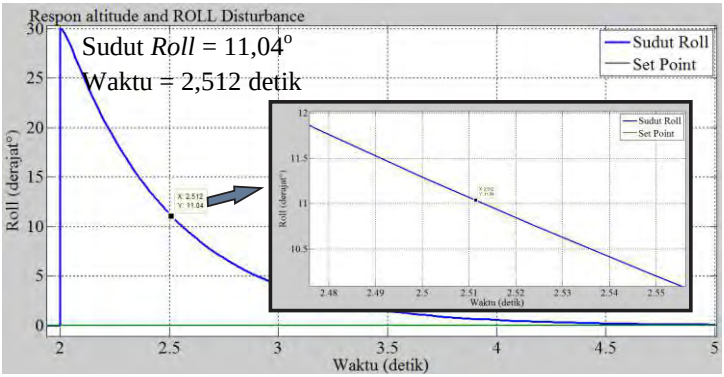
$$e_{ss\ Altitude\ 2} = 3{,}098 - 3$$
$$e_{ss\ Altitude\ 2} = 0{,}098 \text{ meter} \qquad (4.18)$$

Untuk hasil pengujian *Roll* pada Gangguan *Altitude* dan *Roll adalah sebagai* berikut. *Time Constant* pada sudut *Roll* adalah

$$\text{Set point} * 63.2\% = 30 * 63.2\%$$
$$= 18{,}96 \qquad (4.19)$$

Namun karena respon nya menurun dari nilai 30^o menuju set point yaitu 0^o maka

$$30^o - 18{,}96^o = 11{,}04^o \qquad (4.20)$$



Gambar 4.17 Nilai *Time Constant* dari *Roll* pada *Altitude* dan *Roll disturbance*

Maka nilai *Time Constant* adalah :

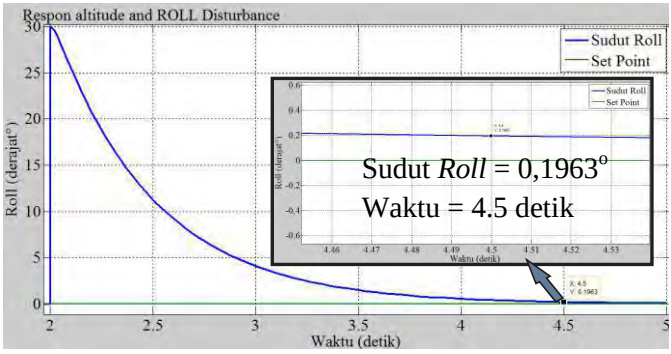
$$T_{R\,2} = 2{,}512 - 2$$

$$T_{R\,2} = 0{,}512 \text{ detik} \qquad (4.16)$$

Untuk respon *Steady State* yaitu :

$$4 * T_{R\,2} = 4 * 0{,}512 = 2{,}048 \text{ detik} \qquad (4.17)$$

Untuk *respon Steady State* terdapat penambahan waktu 2 detik karena gangguan dimulai pada waktu 2 detik. Maka respon *Steady State* pada waktu 4,048 detik telah stabil, Apabila diambil data *sudut Roll* pada waktu 4.5 detik akan sama halnya karena dianggap telah stabil.



Gambar 4.18 Nilai *Steady State* dari *Roll* pada *Altitude* dan *Roll disturbance.*

Nilai Error Steady State untuk *Altitude* atau $e_{ss\ Roll\ 2}$ adalah selisih dari nilai *Steady* dengan *set point* :

$$e_{ss\ Roll\ 2} = 0{,}1963 - 0$$
$$e_{ss\ Roll\ 2} = 0{,}1963^{o} \qquad (4.18)$$

Nilai *Error Steady State Altitude* pada pengujian *Altitude* dan *Roll* sebesar 0.098 meter, dengan *Time Constant* = 0,218 detik. Sedangkan Nilai *Error Steady State Roll* pada pengujian *Altitude* dan *Roll* sebesar 0.1963^{o}, dengan *Time Constant* = 0,512 detik

4.2.5 *Altitude* dan *Pitch Disturbance*

Pengujian kali ini yaitu memberikan *disturbance* pada *Altitude* (ketinggian) dan sudut *Pitch*. *Set Point Altitude* yaitu sebesar 3 m, dan *set point* dari sudut *Roll* adalah 0^{o}. pengujian pada *Altitude* diberi *disturbance* pada waktu 2 detik sebesar 1 m.

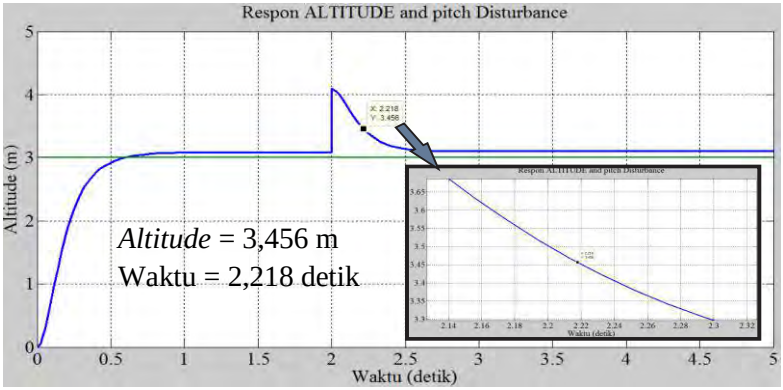
untuk *Time Constant* pada pengujian *Altitude* adalah :

$$\textit{Set Point} * 63.2\% = 1 * 63.2\%$$
$$= 0.632 \qquad (4.14)$$

Karena gangguan dimulai dari 4 meter menuju ke 3 meter maka :

$$4{,}088 - 0{,}632 = 3{,}456 \text{ m} \qquad (4.15)$$

Waktu pada saat ketinggian 3,456 m adalah:



Gambar 4.19 Nilai *Time Constant* dari *Altitude* pada *Altitude* dan *Pitch disturbance*

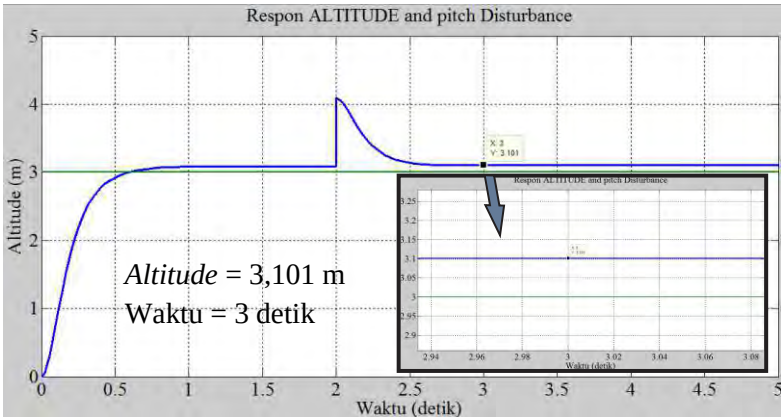
Maka nilai *Time Constant* adalah :

$$T_{A\,3} = 2{,}218 - 2$$
$$T_{A\,3} = 0{,}218 \text{ detik} \qquad (4.16)$$

Dimana $T_{A\ 3}$ adalah nilai *Time Constant Altitude* pada pengujian *disturbance Altitude* dan *Pitch*. Lalu Nilai *Error Steady State* adalah :

$$4 \,.\, T_{A\,3} = 4 * 0{,}21 = 0.872 \text{ detik} \qquad (4.17)$$

Untuk *respon Steady State* terdapat penambahan waktu 2 detik karena *disturbance* dimulai pada waktu 2 detik. Maka respon pada waktu 2,872 detik telah stabil. Apabila diambil data *Altitude* pada waktu 3 detik akan sama halnya karena dianggap telah stabil.



Gambar 4.20 Nilai *Steady State* dari *Altitude* pada *Altitude* dan *Pitch disturbance.*

Nilai Error Steady State untuk *Altitude* atau $e_{ss\ Altitude\ 3}$ adalah selisih dari nilai *Steady* dengan *set point* :

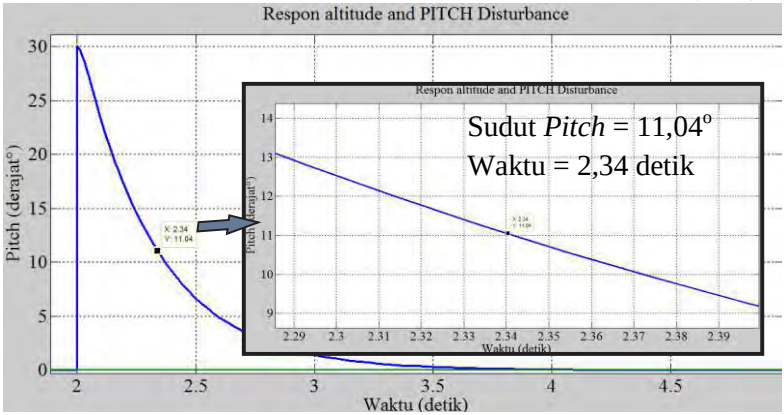
$$e_{ss\ Altitude\ 3} = 3{,}101 - 3$$
$$e_{ss\ Altitude\ 3} = 0{,}101 \text{ meter} \qquad (4.18)$$

Lalu untuk pengujian terhadap sudut *Pitch* adalah sebagai berikut. *Set point* sudut *Pitch* yaitu sebesar 0^o dan sudut mula-mula sebesar 0^o sedangkan gangguan yang dialami oleh quadrotor pada sudut *Pitch* yaitu sebesar 30^o pada waktu 2 detik.

$$Set\ Point\ .\ 63{,}2\% = 30\ .\ 63{,}2\%$$
$$= 18{,}96^{o} \qquad (4.19)$$

Namun karena respon nya turun dari nilai 30^{o} menuju set point yaitu 0^{o} maka

$$30^{o} - 18{,}96^{o} = 11{,}04^{o} \qquad (4.20)$$



Gambar 4.21 Nilai *Time Constant* dari *Pitch* pada *Altitude* dan *Pitch disturbance*

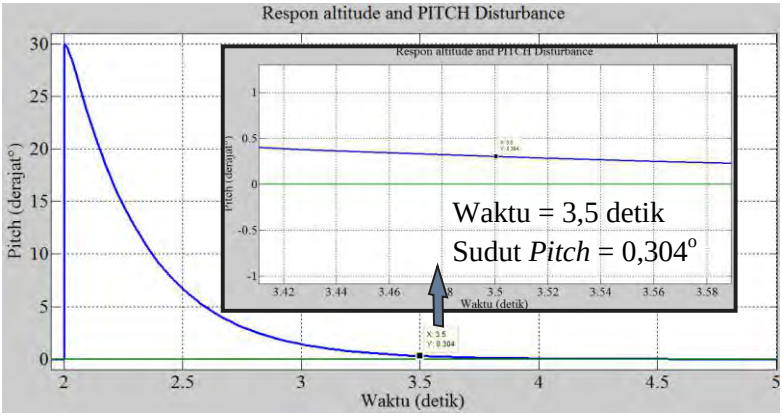
waktu pada sudut $11{,}04^{o}$ adalah sebesar 2,34 detik. bila di hitung mulai dari waktu gangguan mula-mula yaitu pada detik ke 2. Maka nilai $T_{P\ 2}$ atau *Time Constant* dengan gangguan *Pitch* pada disturbance *Altitude* dan *Pitch* adalah

$$T_{P\ 2} = 2{,}34 - 2$$
$$T_{P\ 2} = 0{,}34 \text{ detik} \qquad (4.21)$$

Dengan nilai $T_{P\ 2}$ adalah 0,34 detik, nilai *Steady State* pada *Pitch disturbance*. Adalah :

$$4\ .\ T_{P\ 2} = 1.36 \text{ detik} \qquad (4.22)$$

Karena T_{P2} dimulai pada saat respon menerima gangguan mula-mula pada detik ke 2 maka waktu keseluruhan untuk mencapai nilai *Steady State* adalah 3,36 detik. bila mengambil nilai pada detik 3,5 detik akan sama halnya dengan pada waktu 3,36 detik Karena telah mencapai *Steady State*.



Gambar 4.22 Nilai *Steady State* pada *Pitch* dengan *Altitude dan Pitch Disturbance*

Karena set point sudut *Pitch* adalah 0^o maka nilai *Error Steady State* atau $e_{ss\ Pitch\ 2}$ pada *Pitch* adalah :

$$e_{ss\ Pitch\ 2} = 0{,}304^o - 0^o$$

$$e_{ss\ Pitch\ 2} = 0{,}304^o \qquad (4.23)$$

Nilai *Error Steady State* pada pengujian *Altitude* dengan gangguan *Altitude* dan *Pitch* sebesar 0,101 meter, dengan *Time Constant* = 0,218 detik. Sedangkan Nilai *Error Steady State* pada pengujian *Pitch* dengan gangguan *Altitude* dan *Pitch* sebesar $0{,}304^o$, dengan *Time Constant* = 0,34 detik.

4.2.6 *Roll* dan *Pitch Disturbance*

Pengujian kali ini yaitu memberikan *disturbance* pada 2 parameter *sudut Roll* dan sudut *Pitch*. *Set Point* sudut *Roll* dan sudut *Pitch* yaitu sebesar 0^o. pengujian pada *sudut Roll* dan *Pitch* diberi *disturbance* pada waktu 2 detik sebesar 0^o. untuk *Time Constant* pada pengujian *Roll* adalah :

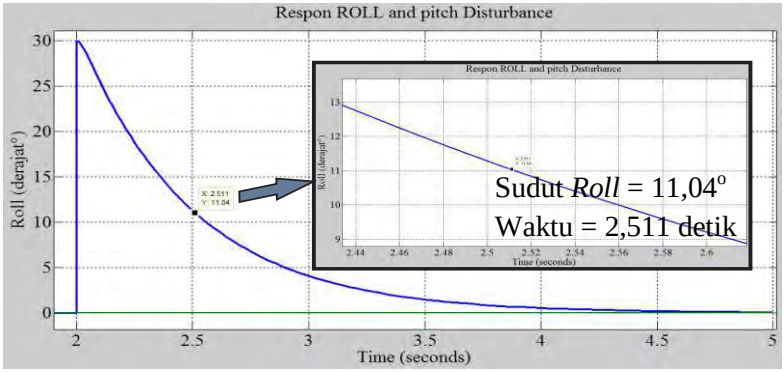
$$\text{Set point} * 63.2\% = 30 * 63.2\%$$

$$= 18{,}96 \qquad (4.24)$$

Namun karena respon nya menurun dari nilai 30^o menuju set point yaitu 0^o maka

$$30^o - 18{,}96^o = 11{,}04^o \qquad (4.25)$$

Waktu pada sudut $11{,}04^o$ adalah:



Gambar 4.23 Nilai *Time Constant* dari *Roll* pada *Roll* dan *Pitch disturbance*

Maka nilai *Time Constant* adalah :

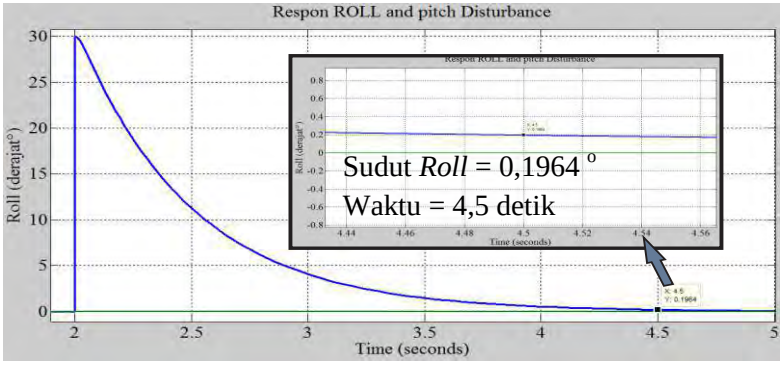
$$T_{R\,3} = 2{,}511 - 2$$

$$T_{R\,3} = 0{,}511 \text{ detik} \qquad (4.26)$$

Dimana $T_{R\,2}$ adalah nilai *Time Constant Roll* pada pengujian *disturbance Roll* dan *Pitch*. Nilai *Error Steady State* adalah :

$$4 \,.\, T_{R\,3} = 4 * 0{,}511 = 2{,}044 \text{ detik} \qquad (4.27)$$

Untuk *respon Steady State* terdapat penambahan waktu 2 detik karena *disturbance* dimulai pada waktu 2 detik. Maka respon pada waktu 4,044 detik telah stabil. Apabila diambil data *Roll* pada waktu 4.5 detik akan sama halnya karena dianggap telah stabil.



Gambar 4.24 Nilai *Steady State* dari *Roll* pada *Roll dan Pitch disturbance.*

Nilai Error Steady State untuk *Roll* atau $e_{ss\ Roll\ 3}$ pada *Roll dan Pitch disturbance* adalah selisih dari nilai *Steady* dengan *set point*:

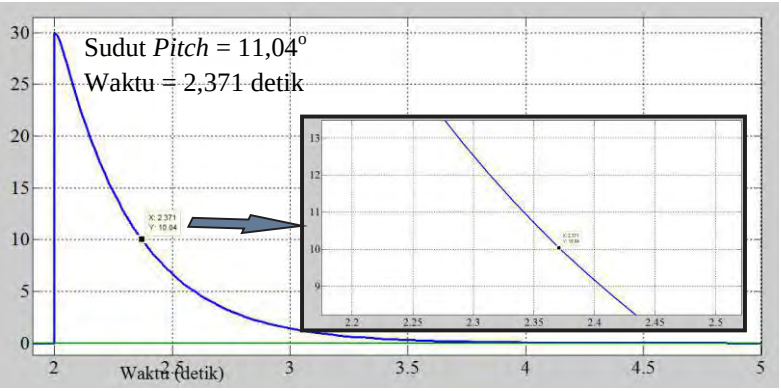
$$e_{ss\ Roll\ 3} = 0{,}1946 - 0$$
$$e_{ss\ Roll\ 3} = 0{,}1946^{o} \quad (4.28)$$

Untuk hasil pengujian *Pitch* pada Gangguan *Roll dan Pitch disturbance* adalah *sebagai* berikut. *Time Constant* pada sudut *Pitch* adalah

$$\text{Set point} * 63.2\% = 30 * 63.2\%$$
$$= 18{,}96 \quad (4.29)$$

Namun karena respon nya menurun dari nilai 30^{o} menuju set point yaitu 0^{o} maka

$$30^{o} - 18{,}96^{o} = 11{,}04^{o} \quad (4.30)$$



Gambar 4.25 Nilai *Time Constant* dari *Pitch* pada *Roll dan Pitch disturbance*

Maka nilai *Time Constant* adalah :

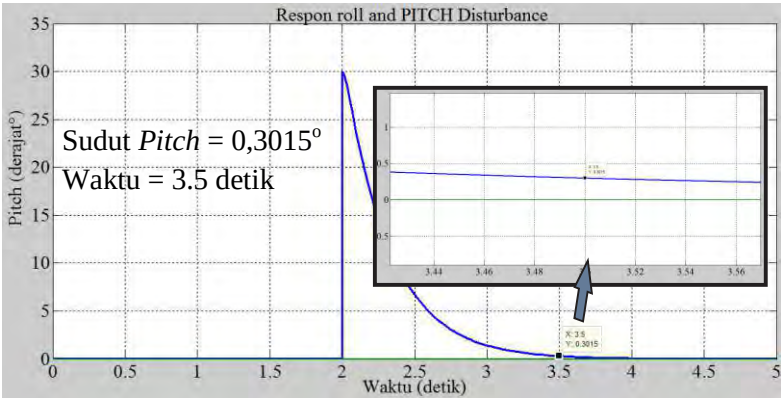
$$T_{P\,3} = 2{,}371 - 2$$
$$T_{P\,3} = 0{,}371 \text{ detik} \quad (4.31)$$

Dimana T_{P3} adalah nilai *Time Constant Pitch* pada pengujian *disturbance Roll* dan *Pitch*. Sedangkan Untuk respon *Steady State* yaitu :

$$4 * T_{P\,3} = 4 * 0{,}371 = 1{,}484 \text{ detik} \quad (4.32)$$

Untuk *respon Steady State* terdapat penambahan waktu 2 detik karena gangguan dimulai pada waktu 2 detik. Maka respon

Steady State pada waktu 3,484 detik telah stabil, Apabila diambil data *sudut Pitch* pada waktu 3.5 detik akan sama halnya karena dianggap telah stabil.



Gambar 4.26 Nilai *Steady State* dari *Pitch* pada *Altitude* dan *Roll disturbance.*

Nilai Error Steady State untuk *Pitch* atau $e_{ss\ Pitch\ 3}$ adalah selisih dari nilai *Steady* dengan *set point* :

$$e_{ss\ pitch\ 3} = 0{,}3015 - 0$$
$$e_{ss\ pitch\ 3} = 0{,}3015^{o} \qquad (4.33)$$

Nilai *Error Steady State Roll* pada pengujian *Roll* dan *Pitch Disturbance* sebesar 0.1946^{o}, dengan *Time Constant* = 0,511 detik. Sedangkan Nilai *Error Steady State Pitch* pada *Roll* dan *Pitch Disturbance* sebesar $0{,}3015^{o}$, dengan *Time Constant* = 0,371 detik

4.2.7 *Altitude, Roll* dan *Pitch Disturbance*

Pengujian kali ini yaitu memberikan *disturbance* pada *Altitude* (ketinggian), Sudut *Roll* dan sudut *Pitch*. *Set Point Altitude* yaitu sebesar 3 m, serta *set point* dari sudut *Roll* dan *Pitch* adalah 0^{o}. pengujian pada *Altitude* diberi *disturbance* pada waktu 2 detik sebesar 1 m. sedangkan untuk sudut *Roll* dan *Pitch* diberikan gangguan sebesar 30^{o} pada waktu 2 detik.

untuk *Time Constant* pada pengujian *Altitude* adalah :

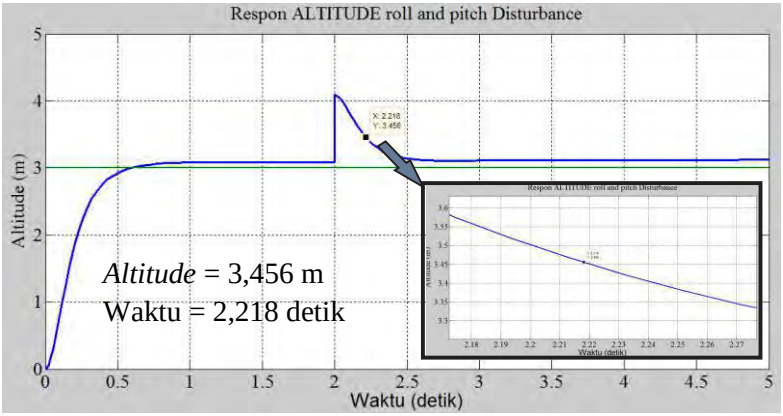
Set Point * 63.2% = 1 * 63.2%

= 0.632 (4.34)

Karena gangguan dimulai dari 4 meter menuju ke 3 meter maka :

4,088 – 0,632 = 3,456 m (4.35)

Waktu pada saat ketinggian 3,456 m adalah:



Gambar 4.27 Nilai *Time Constant* dari *Altitude* pada *Altitude, Roll* dan *Pitch disturbance*

Maka nilai *Time Constant* adalah :

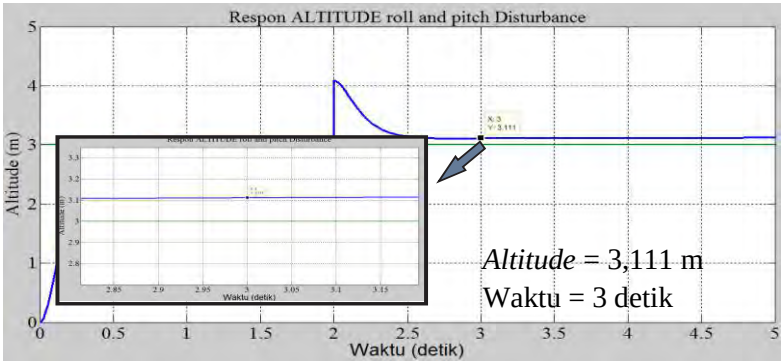
$T_{A\,4} = 2{,}218 - 2$

$T_{A\,4} = 0{,}218$ detik (4.36)

Dimana $T_{A\ 4}$ adalah nilai *Time Constant Altitude* pada pengujian *disturbance Altitude, Roll* dan *Pitch*. Lalu untuk Nilai *Error Steady State* adalah :

$4 \,.\, T_{A\,4} = 4 * 0{,}218 = 0.872$ detik (4.37)

Untuk *respon Steady State* terdapat penambahan waktu 2 detik karena *disturbance* dimulai pada waktu 2 detik. Maka respon pada waktu 2,872 detik telah stabil. Apabila diambil data *Altitude* pada waktu 3 detik akan sama halnya karena dianggap telah stabil.



Gambar 4.28 Nilai *Steady State* dari *Altitude* pada *Altitude* dan *Pitch disturbance.*

Nilai Error Steady State untuk *Altitude* atau $e_{ss\ Altitude\ 3}$ adalah selisih dari nilai *Steady* dengan *set point* :

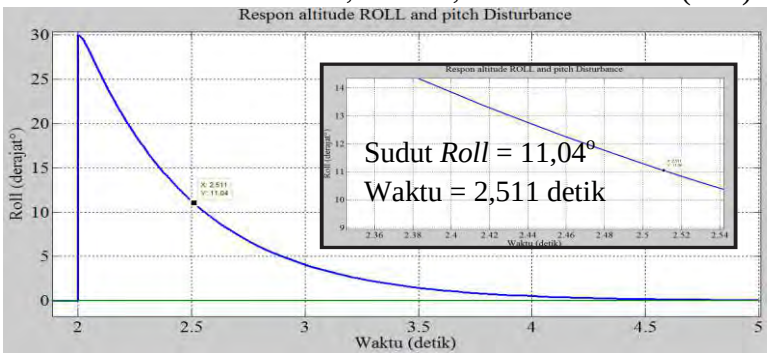
$$e_{ss\ Altitude\ 4} = 3{,}111 - 3$$
$$e_{ss\ Altitude\ 4} = 0{,}111 \text{ meter} \qquad (4.37)$$

Selanjutnya untuk pengujian terhadap sudut *Roll* adalah sebagai berikut.

$$Set\ Point\ .\ 63{,}2\% = 30\ .\ 63{,}2\%$$
$$= 18{,}96^{o} \qquad (4.38)$$

Namun karena respon nya turun dari nilai 30^{o} menuju *set point* yaitu 0^{o} maka

$$30^{o} - 18{,}96^{o} = 11{,}04^{o} \qquad (4.39)$$



Gambar 4.29 Nilai *Time Constant* dari *Roll* pada *Altitude Roll* dan *Pitch disturbance*

waktu pada sudut 11,04^o adalah sebesar 2,511 detik. bila di hitung mulai dari waktu gangguan mula-mula yaitu pada detik ke 2. Maka nilai *Time Constant* dengan gangguan *Roll* pada *disturbance Altitude* dan *Pitch* adalah

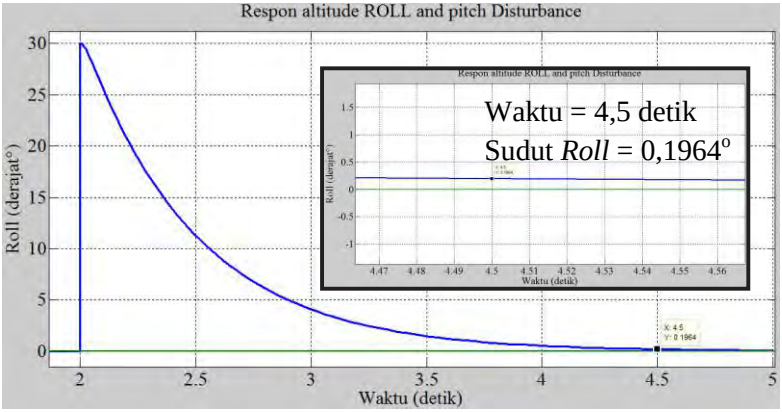
$$T_{R\,4} = 2{,}511 - 2$$

$$T_{R\,4} = 0{,}511 \text{ detik} \qquad (4.40)$$

Dengan nilai $T_{P\,4}$ adalah 0,511 detik, nilai *Steady State* pada *Roll disturbance*. Adalah :

$$4 \,.\, T_{R\,4} = 2.044 \text{ detik} \qquad (4.41)$$

Karena $T_{R\,4}$ dimulai pada saat respon menerima gangguan mula-mula pada detik ke 2 maka waktu keseluruhan untuk mencapai nilai *Steady State* adalah 4,044 detik. bila mengambil nilai pada detik 4,5 detik akan sama halnya dengan pada waktu 4,044 detik Karena telah mencapai *Steady State*.



Gambar 4.30 Nilai *Steady State* pada *Roll* dengan *Altitude Roll dan Pitch Disturbance*

Karena set point sudut *Roll* adalah 0^o maka nilai *Error Steady State* pada *Roll* atau $e_{ss\ Roll\ 4}$ adalah :

$$e_{ss\ Roll\ 4} = 0{,}1964^o - 0^o$$

$$e_{ss\ Roll\ 4} = 0{,}1964^o \qquad (4.42)$$

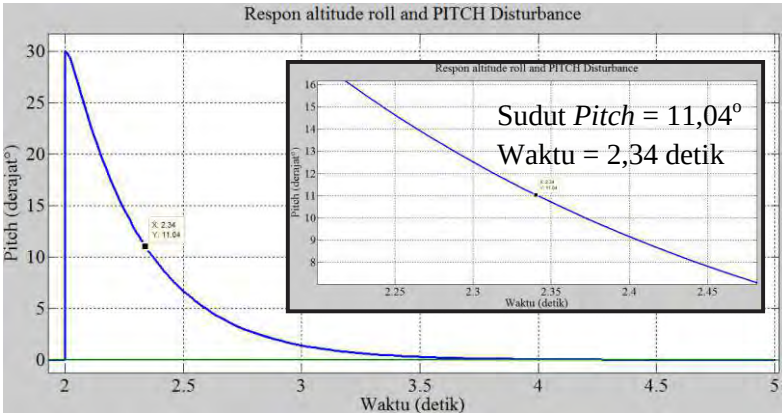
Untuk pengujian terhadap sudut *Pitch* adalah sebagai berikut.

$$Set\ Point\ .\ 63{,}2\% = 30\ .\ 63{,}2\%$$
$$= 18{,}96^{o} \qquad (4.43)$$

Namun karena respon nya turun dari nilai 30^{o} menuju *set point* yaitu 0^{o} maka

$$30^{o} - 18{,}96^{o} = 11{,}04^{o} \qquad (4.44)$$

Dan waktu pada saat sudut pitch sebesar $11{,}04^{o}$ adalah



Gambar 4.31 Nilai *Time Constant* dari *Pitch* pada *Altitude Roll* dan *Pitch disturbance*

waktu pada sudut $11{,}04^{o}$ adalah sebesar 2,34 detik. bila di hitung mulai dari waktu gangguan mula-mula yaitu pada detik ke 2. Maka nilai *Time Constant Pitch* pada *gangguan Altitude Roll* dan *Pitch* adalah

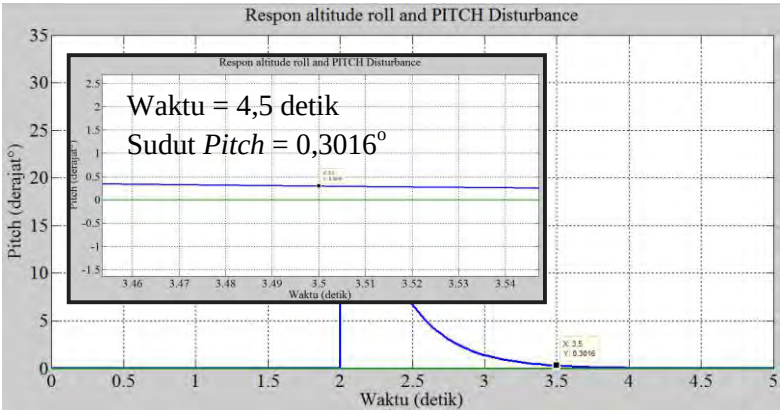
$$T_{P\,4} = 2{,}34 - 2$$
$$T_{P\,4} = 0{,}34 \text{ detik} \qquad (4.40)$$

Dengan nilai $T_{P\,4}$ atau *Time Constant* dari *Pitch* pada *Altitude Roll* dan *Pitch disturbance* adalah 0,34 detik, nilai *Steady State* pada *Roll disturbance*. Adalah :

$$4\ .\ T_{P\,4} = 1{,}36 \text{ detik} \qquad (4.41)$$

Karena $T_{P\ 4}$ dimulai pada saat respon menerima gangguan mula-mula pada detik ke 2 maka waktu keseluruhan untuk mencapai nilai *Steady State* adalah 3,36 detik. bila mengambil

nilai pada detik 3,5 detik akan sama halnya dengan pada waktu 3,36 detik Karena telah mencapai *Steady State*.



Gambar 4.30 Nilai *Steady State* pada *Roll* dengan *Altitude Roll dan Pitch Disturbance*

Karena set point sudut *Pitch* adalah 0^o maka nilai *Error Steady State* pada *Pitch* atau $e_{ss\ Pitch\ 4}$ adalah :

$$e_{ss\ Pitch\ 4} = 0{,}3016^o - 0^o$$

$$e_{ss\ Pitch\ 4} = 0{,}3016^o \qquad (4.42)$$

Nilai *Error Steady State* pengujian *Altitude* dengan *Disturbance altitude*, *Roll* dan *Pitch* sebesar 0,111 meter, dengan *Time Constant* = 0,218 detik. Lalu Nilai *Error Steady State* pada pengujian *Roll* dengan *Disturbance altitude*, *Roll* dan *Pitch* sebesar 0.1964^o dan *Time Constant* 0.511 detik. Lalu Nilai *Error Steady State* pengujian *Pitch* dengan *Disturbance altitude*, *Roll* dan *Pitch* sebesar $0{,}3016^o$ dengan *Time Constant* = 0,34 detik

4.3 Hasil Uji Kestabilan *Quadrotor*

Setelah dilakukan pengujian pada ketiga parameter *quadrotor* (*Altitude*, *Roll*, dan *Pitch*) sesuai dengan Tabel 4.1. Maka didapatkan hasil respon sistem berupa *time constant* dan

error steady state pada masing-masing parameter *quadrotor* yang dapat dilihat pada tabel.4.2.

Tabel 4.2 Tabel *Error Steady State* pada Masing-Masing Pengujian

No.	*Altitude*	*Roll*	*Pitch*
1	0.091 m		
2		0.1963^{o}	
3			0.304^{o}
4	0.098 m	0.1963^{o}	
5	0.101 m		0.304^{o}
6		0.1964^{o}	0.3015^{o}
7	0.111 m	0.1964^{o}	0.3016^{o}

Apabila dibuat nilai persentase dari *Error Steady State* maka dapat dilihat pada tabel 4.3

Tabel 4.3 Tabel Persentase *Error Steady State* pada Masing-Masing Pengujian

No.	*Altitude*	*Roll*	*Pitch*
1	9.1 %		
2		0.65%	
3			1.01%
4	9.8%	0.65%	
5	10.1%		1.01%
6		0.65%	1%
7	11.1%	0.65%	1%

Nilai tersebut di dapatkan dari perbandingan terhadap gangguan yang diberikan pada tiap pengujian. Nilai *Error Steady State* terkecil dari masing masing parameter *quadrotor* (*Altitude, Roll, Pitch*) berturut-turut sebesar 9.1 %, 0.65%, 1%

BAB V
PENUTUP

5.1 Kesimpulan

Dari hasil penelitian yang telah dilakukan telah didapatkan kesimpulan sebagai berikut.

1. Simulasi pengendalian *attitude* UAV (*unmanned aerial vehicle*) *quadrotor* Berbasis *artificial neural network* telah berhasil dilakukan dan menghasilkan respon berupa altitude dan orientasi *quadrotor*.
2. Pengendalian *attitude* UAV (*unmanned aerial vehicle*) *quadrotor* Berbasis *artificial neural network* berhasil mengendalikan ke tiga parameter *quadrotor* sesuai dengan set poin yang diberikan yaitu altitude, roll, pitch dengan error steady state secara berturut turut yaitu : 9.1%, 0.65%, dan 1%.

Halaman ini sengaja dikosongkan

DAFTAR PUSTAKA

[1] Bors Erginer, Erdinc Altug. ”*Modeling and PD Control of a Quadrotor VTOL vehichle”. Proceedings of the 2007 IEEE Intelligent Vehicles Symposium*, Istanbul, Turkey. Juni 2007.

[2] Hana Boudjedir, Fouad Yacef, Omar Bouhali dan Nassim Rizoug. “*Adaptive Neural Network for a Quadrotor Unmanned Aerial Vehicle*”. International Journal in *Foundation of Computer Science and Technology* (IJFCST), Jilel University, Algeria. Vol 2, No 4, Juli 2012

[3] M. Y. Amir and V. U. Abbas, "*Modeling and Neural Control of Quadrotor Helicopter*". Vol. 2, pp. 35-48, 2011.

[4] A. Y. Elruby, M. M. El Khatib, N. H. El Amary, A. I. Hashad. “Dynamic Modeling and Control of Quadrotor Vehicle”

[5] Victoria López Matilde Santos, Franciso Morata. “*Intelligent Fuzzy Controller of a Quadrotor*”. *Intelligent Systems and Knowledge Engineering (ISKE), International Conference* IEEE November 2010.

[6] Aleksandar Rodić, Gyula Mester. “*The Modeling and Simulation of an Autonomous Quad Rotor Microcopter in a Virtual Outdoor Scenario*”. Acta Polytechnica Hungarica, Vol. 8, No. 4, 2011

[7] David F. Anderson, Scott Eberhardt. “*The Newtonian Description of Lift of a Wing*”. *American Journal of Physics*, Maret 2001.

[8] Vedran Sikiric. “*Control of Quadrocopter*”.*Royal Institute of Technology,* Stockholm, Swiss. 2008

[9] Tomasso Bresciani. "*Modelling, Identification and Control of a Quadrotor Helicopter*". *Department of Automatic Control, Lund University*. Oktober 2008.

[10] Krose, Ben. Smagt, Patrick van Der, "*An Introduction to neural Network'. November 1996.*

[11] Jack F.Shepherd, Kagan Tumer. "*Robust Neuro-Control for A Micro Quadrotor*". *Oregon State University. ACM 978-1-4503-0072-8/10/07*

[12] Samir Bouabdallah, "*Design And Control Of Quadrotors With Application To Autonomous Flying*". Aboubekr Belkaid University, 2007

[13] Ogata, Katsuhiko."*Modern Control Engineering*". *Prentice-Hall of India Private Limited*. *New Delhi*, 1984.

[14] Laurene V. Fausett. "*Fundamentals Of Neural Networks*". *Pearson*. *December*, 1993.

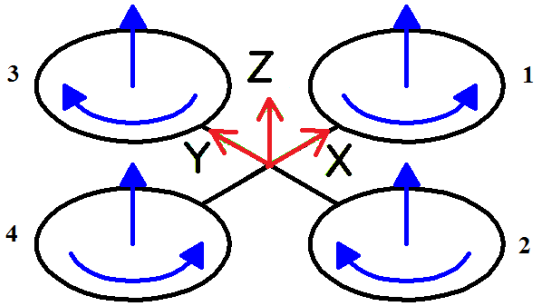
Lampiran A

Quadrotor adalah helikopter yang terdiri dari 4 motor dan propeller yang terdapat pada ujung *frame* yang menyilang. 2 motor berputar searah jarum jam dan 2 motor yang lain berputar berlawanan arah jarum jam. Untuk memodelkan 6 DOF (*Degree of Freedom*) *quadrotor* di perlukan beberapa asumsi sebagai berikut[4][9] :

- Struktur *Quadrotor* simetris dan kaku
- Struktur Propeller kaku
- *Ground Effect* Diabaikan

Perlunya beberapa asumsi di atas digunakan untuk memudahkan pemodelan *quadrotor 6 DOF*. Kinematika untuk pemodelan Quadrotor memerlukan dua refernsi *frame*[9] yaitu sebagai berikut:

- Referensi Inersia pada Bumi (E-*frame*)
- Referensi Body-fixed (B-*frame*)



Gambar A.2 Orientasi *Quadrotor*

Pemodelan quadrotor dilakukan dengan metode pemodelan persamaan *Newton-Euler*[9].

$$\dot{\xi} = J_\theta \, v \tag{A.1}$$

$\dot{\xi}$ [+] adalah vektor kecepatan dari E-*frame*, v [+] adalah vektor kecepatan pada B-*frame*, J_θ [-] adalah matriks invers generalisasi.

$$\boldsymbol{\xi} = [\boldsymbol{\Gamma}^E \; \boldsymbol{\Theta}^E]^T = [X \; Y \; Z \; \phi \; \theta \; \psi]^T \tag{A.2}$$

ξ [+] terdiri dari Γ^E [m] dan Θ^E [rad] adalah masing-masing vektor posisi linear dan posisi anguler E-*frame*.

$$\boldsymbol{v} = [\boldsymbol{V}^B\ \boldsymbol{\omega}^B]^T = [u\ v\ w\ p\ q\ r]^T \tag{A.3}$$

v [+] juga terdiri dari V^B[m s^{-1}] dan ω^B[rad s^{-1}] yang masing masing adalah vektor kecepatan linear dan kecepatan anguler pada B-*frame*. Matriks J_θ terdiri dari 4 sub-matriks yaitu.

$$\boldsymbol{J_\theta} = \begin{bmatrix} \boldsymbol{R_\theta} & \boldsymbol{0}_{3x3} \\ \boldsymbol{0}_{3x3} & \boldsymbol{T_\theta} \end{bmatrix} \tag{A.4}$$

0_{3x3} adalah matriks dengan dimensi 3x3 yang semua nilainya adalah 0. Sementara matriks rotasi (R_θ) dan matriks transfer (T_θ) adalah sebagai berikut.

$$\boldsymbol{R_\theta} = \tag{A.5}$$

$$\begin{bmatrix} Cos_\psi Cos_\theta & -Sin_\psi Cos_\phi + Cos_\psi Sin_\theta Sin_\phi & Sin_\psi Sin_\phi + Cos_\psi Cos_\theta Cos_\phi \\ Sin_\psi Cos_\theta & Cos_\psi Cos_\phi + Cos_\psi Sin_\theta Sin_\phi & -Cos_\psi Sin_\phi + Sin_\psi Sin_\theta Cos_\theta \\ -Sin_\theta & Cos_\theta Sin_\phi & Cos_\theta Cos_\phi \end{bmatrix}$$

$$\boldsymbol{T_\theta} = \begin{bmatrix} 1 & Sin_\phi Tan_\theta & Cos_\phi Tan_\theta \\ 0 & Cos_\phi & -Sin_\phi \\ 0 & Sin_\phi / Cos_\theta & Cos_\phi / Cos_\theta \end{bmatrix} \tag{3.6}$$

Masa dan Inersia dari quadrotor juga digunakan Pemodelan quadrotor[9]. Maka pemodelan *newton-euler* nya adalah sebagai berikut.

$$\begin{bmatrix} m\,\boldsymbol{I}_{3z3} & \boldsymbol{0}_{3x3} \\ \boldsymbol{0}_{3x3} & \boldsymbol{I} \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{V}}^B \\ \dot{\boldsymbol{\omega}}^B \end{bmatrix} + \begin{bmatrix} \omega^B & X & (m\,\boldsymbol{V}^B) \\ \omega^B & X & (\boldsymbol{I}\,\boldsymbol{\omega}^B) \end{bmatrix} = \begin{bmatrix} \boldsymbol{F}^B \\ \boldsymbol{\tau}^B \end{bmatrix} \tag{A.7}$$

$\boldsymbol{I}_{3x3}$ adalah matriks identitas dengan dimensi 3x3. $\boldsymbol{V}^B[m\ s^{-2}]$ adalah vektor akselarasi linear B-*frame quadroto*r dan $\boldsymbol{\omega}^B[rad\ s^{-2}]$ adalah vektor percepatan anguler B-*frame quadrotor*. $\boldsymbol{F}^B$ $[N]$ adalah vektor gaya *quadrotor* dan $\boldsymbol{\tau}^B$ $[N\ m]$ adalah vektor torsi dari *quadrotor* pada B-*frame.* Vektor gaya pada quadrotor dapat dituliskan sebagai berikut[9] **:**

$$\boldsymbol{\Lambda} = [\boldsymbol{F}^B \boldsymbol{\tau}^B] = [F_x\ F_y\ F_z\ \tau_x\ \tau_y\ \tau_z] \tag{A.8}$$

Persamaan A.8 dapat ditulis dalam bentuk matrik sebagai berikut :

$$\boldsymbol{M}_B \dot{\boldsymbol{v}} + \boldsymbol{C}_B(\boldsymbol{v})\, \boldsymbol{v} = \boldsymbol{\Lambda} \tag{A.9}$$

Dimana $\boldsymbol{M}_B$ [+] adalah matriks inersia dari sistem, $\dot{\boldsymbol{v}}$ adalah vektor generalisasi akselarasi, dan $\boldsymbol{C}_B(\boldsymbol{v})$ [+] adalah matriks *Coriolis-centripetal*[9] dan ketiga matriks tersebut pada B-*frame* quadrotor.

$$\boldsymbol{M}_B = \begin{bmatrix} m\,\boldsymbol{I}_{3x3} & \boldsymbol{0}_{3x3} \\ \boldsymbol{0}_{3x3} & \boldsymbol{I} \end{bmatrix} = \begin{bmatrix} m & 0 & 0 & 0 & 0 & 0 \\ 0 & m & 0 & 0 & 0 & 0 \\ 0 & 0 & m & 0 & 0 & 0 \\ 0 & 0 & 0 & I_{XX} & 0 & 0 \\ 0 & 0 & 0 & 0 & I_{YY} & 0 \\ 0 & 0 & 0 & 0 & 0 & I_{ZZ} \end{bmatrix} \tag{A.10}$$

Persamaan A.10 diatas Menunjukan matrik inersia dari sistem *quadrotor*.

$$\boldsymbol{C}_B(\boldsymbol{v}) = \begin{bmatrix} \boldsymbol{0}_{3x3} & -m\,\boldsymbol{S}(\boldsymbol{V}^B) \\ \boldsymbol{0}_{3x3} & -\boldsymbol{S}\,(\boldsymbol{I}\,\boldsymbol{\omega}^B) \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & m\,w & -m\,v \\ 0 & 0 & 0 & -m\,w & 0 & m\,u \\ 0 & 0 & 0 & -m\ v & -m\,u & 0 \\ 0 & 0 & 0 & 0 & I_{ZZ}\,r & -I_{YY}\,q \\ 0 & 0 & 0 & -I_{ZZ}\,r & 0 & I_{XX}p \\ 0 & 0 & 0 & I_{YY}\,q & -I_{XX}\,p & 0 \end{bmatrix} \tag{A.11}$$

Pada persamaan (A.11) terdapat *skew-symetric* matriks yang terdiri dari vektor $\boldsymbol{k}$ [-] yang dijelaskan pada persamaan (A.12)

$$S(k) = -S^T(k) = \begin{bmatrix} 0 & -k_3 & k_1 \\ k_3 & 0 & -k_1 \\ -k_2 & k_1 & 0 \end{bmatrix} \quad k = \begin{bmatrix} k_1 \\ k_2 \\ k_3 \end{bmatrix} \tag{A.12}$$

Salah satu faktor yang mempengaruhi dalam sistem *quadrotor* adalah gravitasi. Gravitasi dipengaruhi oleh percepatan gravitasi, dan gravitasi hanya mempengaruhi persamaan translasi saja, dan tidak mempengaruhi persamaan rotasi.[9] Maka vektor gravitasinya adalah sebagai berikut.

$$G_B(\xi) = \begin{bmatrix} \boldsymbol{F_G^B} \\ \mathbf{0_{3x1}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{R_\Theta^{-1}}\, \boldsymbol{F_G^E} \\ \mathbf{0}_{3x1} \end{bmatrix} =$$

$$\begin{bmatrix} \boldsymbol{R_\Theta^T} \begin{bmatrix} 0 \\ 0 \\ -m\,g \end{bmatrix} \\ \mathbf{0}_{3x1} \end{bmatrix} = \begin{bmatrix} m\,g\,Sin_\theta \\ -m\,g\,Cos_\theta Sin_\phi \\ -m\,g\,Cos_\theta Sin_\phi \\ 0 \\ 0 \\ 0 \end{bmatrix} \qquad \text{(A.13)}$$

$\boldsymbol{F_G^B}$ [N] adalah vektor gaya gravitasi pada B-*frame* dan $\boldsymbol{F_G^E}[N]$ adalah vektor gaya gravitasi pada E-*frame*, $\mathbf{0}_{3x1}$ [-] matriks 3x1 yang semuanya bernilai 0. Karena $\boldsymbol{R_\Theta}$ adalah matrik ortogonal, dan $\boldsymbol{R_\Theta^{-1}}$ adalah invers matrik dari $\boldsymbol{R_\Theta}$, maka transpose matrik dari $\boldsymbol{R_\Theta}$ yaitu $\boldsymbol{R_\Theta^T}$ bernilai sama dengan inversnya atau $\boldsymbol{R_\Theta^{-1}}$. Lalu faktor lain yang berpengaruh adalah efek gyroskopik yang di hasil kan oleh putaran propeller.[9] Karena 2 propeller berputar searah jarum jam dan 2 propeller yang lain berputar berlawanan arah dengan jarum jam, selisih dari kecepatan motor tidak sama dengan 0. Maka matriks gyroskopik propeller adalah sebagai berikut.

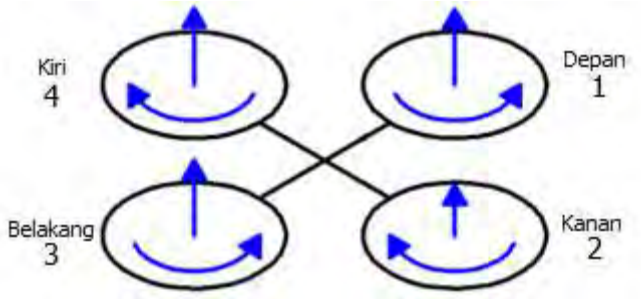
$$\boldsymbol{O}_B(\boldsymbol{v})\,\boldsymbol{\Omega} = \begin{bmatrix} \mathbf{0_{3X1}} \\ \left[-\sum_{k=1}^{4} J_{TP} \left(\omega^B\, x \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \right) (-1)^k \Omega_k \right] \end{bmatrix}$$

$$= \begin{bmatrix} \mathbf{0_{3X1}} \\ J_{TP} \begin{bmatrix} -q \\ p \\ 0 \end{bmatrix} \Omega \end{bmatrix} = J_{TP} \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ q & -q & q & -q \\ -p & p & -p & p \\ 0 & 0 & 0 & 0 \end{bmatrix} \boldsymbol{\Omega} \text{ (A.14)}$$

$\boldsymbol{O}_B(\boldsymbol{v})$ adalah matriks gyrosopik propeller dan J_{TP} [N m s^2] adalah momen inersia rotasi total pada propeller. Dari persamaan A.14 dapat di lihat bahwa efek gyroskopik berpengaruh pada persamaan anguler, tidak pada persamaan translasi. Ω [rad s^{-1}] adalah kecepatan anguler total propeller, Sedangkan $\boldsymbol{\Omega}$ [rad s^{-1}] adalah vektor kecepatan anguler total propeller.

$$\Omega = -\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4 \qquad \boldsymbol{\Omega} = \begin{bmatrix} \Omega_1 \\ \Omega_2 \\ \Omega_3 \\ \Omega_4 \end{bmatrix} \quad \text{(A.15)}$$

Ω_1 [rad s^{-1}] adalah kecepatan propeller depan, Ω_2 [rad s^{-1}] adalah kecepatan propeller kanan, Ω_3 [rad s^{-1}] adalah kecepatan propeller belakang, dan Ω_4 [rad s^{-1}] adalah kecepatan propeller kiri.



Gambar A.2 Penomoran propeller *quadrotor*

Faktor selanjutnya adalah gaya dan torsi yang dihasilkan oleh putaran propeller.

$$\boldsymbol{U}_B(\boldsymbol{\Omega}) = \boldsymbol{E}_B\boldsymbol{\Omega}^2 = \begin{bmatrix} 0 \\ 0 \\ U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ b\left(\Omega_1{}^2 + \Omega_2{}^2 + \Omega_3{}^2 + \Omega_4{}^2\right) \\ b\,l\left(\Omega_4{}^2 - \Omega_2{}^2\right) \\ b\,l\left(\Omega_3{}^2 - \Omega_1{}^2\right) \\ d\left(\Omega_2{}^2 + \Omega_4{}^2 - \Omega_1{}^2 - \Omega_3{}^2\right) \end{bmatrix} \quad \text{(A.16)}$$

Menurut prinsip Aerodinamika, bahwa gaya dan torsi sebanding dengan kecepatan propeller kuadrat[9]. Maka E_B, matrik perpindahan posisi pada E-*frame* dikalikan dengan Ω^2 untuk mendapatkan matrik perpindahan posisi pada B-*frame* $U_B(\Omega)$ [+]. Sedangkan b [N s^2] adalah *thrust factor* dan d [N m s^2] adalah *drag factor*, l [m] adalah jarak antara titik tengah *quadrotor* dan titik tengah propeller. U_1 [N] adalah *throtle* atau *thrust* ke atas *quadrotor*, U_2 [N m] torsi *roll* terhadap *quadrotor* U_3 [N m] torsi *pitch* terhadap *quadrotor*, U_4 [N m] torsi *yaw* terhadap *quadrotor*. Berdasarkan persamaan (A.16) maka untuk mendapatkan matrik E_B adalah sebagai berikut:

$$E_B = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ b & b & b & b \\ 0 & -b\,l & 0 & b\,l \\ -b\,l & 0 & b\,l & 0 \\ -d & d & -d & d \end{bmatrix} \quad \text{(A.17)}$$

Dari persamaan (A.9) dan dari beberapa faktor yang mempengaruhi gerak dari *quadrotor* maka pemodelannya dapat di tuliskan sebagai berikut:

$$\boldsymbol{M_B\dot{v} + C_B(v)\,v = G_B(\xi) + O_B(v)\,\Omega + E_B\Omega^2} \quad \text{(A.18)}$$

Lalu dengan memindahkan seluruh persamaan diruas kiri kecuali $\dot{\boldsymbol{v}}$ maka dapat dituliskan sebagai berikut:

$$\boldsymbol{\dot{v} = {M_B}^{-1}(-C_B(v)\,v + G_B(\xi) + O_B(v)\,\Omega + E_B\Omega^2)} \quad \text{(A.19)}$$

Dari persamaan (A.19) yang masih berbentuk persamaan matriks dapat pula ditulis dalam persamaan matematis:

$$\dot{u} = (v\,r - w\,q) + g\,Sin_\theta \quad \text{(A.20)}$$

$$\dot{v} = (w\,p - u\,r) - g\,Cos_\phi \quad \text{(A.21)}$$

$$\ddot{w} = (u\,g - v\,\dot{p}) - g\,Cos_\phi + \frac{U_1}{m} \quad \text{(A.22)}$$

$$\ddot{p} = \frac{I_{YY} - I_{ZZ}}{I_{XX}}\dot{q}\,\dot{r} - \frac{J_{TP}}{I_{XX}}\dot{q}\,\Omega + \frac{U_2}{I_{XX}} \quad \text{(A.23)}$$

$$\ddot{q} = \frac{I_{ZZ}-I_{XX}}{I_{YY}}\dot{p}\,\dot{r} - \frac{J_{TP}}{I_{YY}}\dot{p}\,\Omega + \frac{U_2}{I_{YY}} \quad \text{(A.24)}$$

$$\ddot{r} = \frac{I_{XX}-I_{YY}}{I_{ZZ}}\dot{p}\,\dot{q} + \frac{U_4}{I_{ZZ}} \quad \text{(A.25)}$$

$\dot{u}$ adalah percepatan linear terhadap sumbu x, $\dot{v}$ adalah percepatan linear terhadap sumbu y, $\dot{w}$ adalah percepatan linear terhadap sumbu z, $\ddot{p}$ adalah percepatan anguler terhadap sumbu x, $\ddot{q}$ adalah percepatan anguler terhadap sumbu z, $\ddot{r}$ adalah percepatan anguler terhadap sumbu z. Dan untuk kecepatan *propeller* adalah sebagai berikut.

$$U_1 = b\left(\Omega_1{}^2 + \Omega_2{}^2 + \Omega_3{}^2 + \Omega_4{}^2\right) \quad \text{(A.26)}$$

$$U_2 = b\,l\left(\Omega_4{}^2 - \Omega_2{}^2\right) \quad \text{(A.27)}$$

$$U_3 = b\,l\left(\Omega_3{}^2 - \Omega_1{}^2\right) \quad \text{(A.28)}$$

$$U_4 = d\left(\Omega_2{}^2 + \Omega_4{}^2 - \Omega_1{}^2 - \Omega_3{}^2\right) \quad \text{(A.29)}$$

Pemodelan pada persamaan (A.20) – (A.25) digunakan pada 6 DOF *quadrotor*. Namun pada pemodelan selanjutnya akan digunakan referensi inersia baru dari kombinasi antara persamaan translasi E-*frame* dan persamaan rotasi B-*frame* yang dinamakan H-*frame* (*Hybrid*), hal ini dilakukan untuk memudahkan memodelkan dengan pemodelan sistem kontrol.[9]

$$\boldsymbol{\zeta} = [\dot{\boldsymbol{\Gamma}}^E\ \boldsymbol{\omega}^B]^T = [\dot{X}\ \dot{Y}\ \dot{Z}\ p\ q\ r]^T \quad \text{(A.30)}$$

persamaan (A.30) menunjukan vektor kecepatan pada H-*frame*, dan dapat ditulis dalam bentruk matrik sebagai berikut:

$$\boldsymbol{M}_H\dot{\boldsymbol{\zeta}} + \boldsymbol{C}_H(\boldsymbol{\zeta})\,\boldsymbol{\zeta} = G_H + \boldsymbol{O}_H(\boldsymbol{\zeta})\,\boldsymbol{\Omega} + \boldsymbol{E}_H(\boldsymbol{\xi})\boldsymbol{\Omega}^2 \quad \text{(A.31)}$$

Dimana $\dot{\boldsymbol{\zeta}}$ [+] adalah vektor percepatan pada H-frame. Matriks inersia sistem *quadrotor* pada H-*frame*, bernilai sama dengan yang digunakan pada B-*frame* berdasarkan persamaan (3.10). Namun pada matriks *Coriolis-centripetal* pada H-frame tidak sama seperti yang ada pada B-*frame*. Untuk mendapatkannya sebagai berikut:

$$\boldsymbol{C}_H(\boldsymbol{\zeta}) = \begin{bmatrix} \mathbf{0}_{3x3} & \mathbf{0}_{3x3} \\ \mathbf{0}_{3x3} & -\boldsymbol{S}\,(\boldsymbol{I}\,\boldsymbol{\omega}^B) \end{bmatrix} =$$

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & I_{ZZ}r & I_{YY}q \\ 0 & 0 & 0 & -I_{ZZ}r & 0 & I_{XX}p \\ 0 & 0 & 0 & I_{YY}q & -I_{xx}p & 0 \end{bmatrix} \quad \text{(A.32)}$$

Sedangkan untuk vektor gravitasi atau $G_H[+]$ pada H-*frame* adalah sebagai berikut:

$$G_H = \begin{bmatrix} \boldsymbol{F}_{\boldsymbol{G}}^{\boldsymbol{E}} \\ \mathbf{0}_{\boldsymbol{3x1}} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -m\,g \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad \text{(A.33)}$$

Efek gyroskopik yang terjadi hanya berpengaruh pada persamaan anguler pada B-frame, maka matriks gyroskopik propeller pada H-frame atau $\boldsymbol{O}_H(\boldsymbol{\zeta})$ [+] didapatkan berdasarkan persamaan (A.13).

$$\boldsymbol{O}_H(\boldsymbol{\zeta})\,\boldsymbol{\Omega} = \boldsymbol{O}_B(\boldsymbol{v})\,\boldsymbol{\Omega} = \begin{bmatrix} \mathbf{0}_{\boldsymbol{3X1}} & \\ J_{TP} & \begin{bmatrix} -q \\ p \\ 0 \end{bmatrix} \quad \Omega \end{bmatrix}$$

$$= J_{TP} \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -q & q & q & -q \\ -p & p & -p & p \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad \text{(A.34)}$$

Matriks gerakan *quadrotor* pada H-frame juga berbeda terhadap B-frame, karena U_1 mempengaruhi persamaan translasi dan matriks rotasi R_θ. Maka penyelesaiannya adalah sebagai berikut:

$$\boldsymbol{E}_H(\boldsymbol{\xi})\boldsymbol{\Omega}^2 = \begin{bmatrix} \boldsymbol{R_\theta} & \boldsymbol{0}_{3x3} \\ \boldsymbol{0}_{3x3} & \boldsymbol{T_\theta} \end{bmatrix} \boldsymbol{E}_B \boldsymbol{\Omega}^2$$

$$= \begin{bmatrix} \left(Sin_\psi Sin_\phi + Cos_\psi Sin_\theta Cos_\phi\right) U_1 \\ \left(-Cos_\psi Sin_\phi + Cos_\psi Sin_\theta Cos_\phi\right) U_1 \\ \left(Cos_\theta Cos_\phi\right) U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} \quad \text{(A.35)}$$

Dengan memindahkan seluruh persamaan pada ruas kiri dan menyisakan vektor kecepatan pada H-*frame* $\dot{\boldsymbol{\zeta}}$ [+] menuju ruas kanan pada persamaan (A.31), maka didapatkan persamaan sebagai berikut.

$$\dot{\boldsymbol{\zeta}} = \boldsymbol{M}_H{}^{-1}(-\boldsymbol{C}_H(\boldsymbol{\zeta})\,\boldsymbol{\zeta} + \boldsymbol{G}_H + \boldsymbol{O}_H(\boldsymbol{\zeta})\,\boldsymbol{\Omega} + \boldsymbol{E}_H(\boldsymbol{\zeta})\boldsymbol{\Omega}^2) \quad \textbf{(A.36)}$$

Dari persamaan (A.36) yang masih berbentuk matriks dibuat menjadi persamaan biasa pada persamaan (A.37) sampai persamaan (A.42).

$$\ddot{X} = \left(Sin_\psi Sin_\phi - Cos_\psi Sin_\theta Cos_\phi\right)\frac{U_1}{m} \quad \text{(A.37)}$$

$$\ddot{Y} = \left(-Cos_\psi Sin_\phi - Cos_\psi Sin_\theta Cos_\phi\right)\frac{U_1}{m} \quad \text{(A.38)}$$

$$\ddot{Z} = -g + \left(Cos_\theta Cos_\phi\right)\frac{U_1}{m} \quad \text{(A.39)}$$

$$\ddot{p} = \frac{I_{YY}-I_{ZZ}}{I_{XX}}\dot{q}\,\dot{r} - \frac{J_{TP}}{I_{XX}}\dot{q}\,\Omega + \frac{U_2}{I_{XX}} \quad \text{(A.40)}$$

$$\ddot{q} = \frac{I_{ZZ}-I_{XX}}{I_{YY}}\dot{p}\,\dot{r} - \frac{J_{TP}}{I_{YY}}\dot{p}\,\Omega + \frac{U_2}{I_{YY}} \quad \text{(A.41)}$$

$$\ddot{r} = \frac{I_{XX}-I_{YY}}{I_{ZZ}}\dot{p}\,\dot{q} + \frac{U_4}{I_{ZZ}} \quad \text{(A.42)}$$

Namun pemodelan yang akan digunakan pada penelitian kali ini adalah pada persamaan A.39 sampai A.42, serta persamaan A.26 sampai A.29.

Lampiran B

Pada lampiran berikut ini adalah data training untuk sistem *Neural Network* dari masing-masing parameter yang di kontrol yaitu z (*altitude*), phi(*roll*), theta(*pitch*), psi(*yaw*).

Tabel B.1 Data Training *altitude quadrotor*

No	Error (m)	True Value (rad)	Adjustment
1	-2.29E-09	0.000100468	1.00201
2	-8.25E-08	0.000702906	1.014066
3	-1.72E-06	0.003352486	1.067222
4	-2.12E-05	0.012340088	1.248924
5	-0.00016902	0.036164687	1.740196
6	-0.00096055	0.087137019	2.838796
7	-0.00292368	0.157912975	4.450628
8	-0.00693676	0.231189626	6.317468
9	-0.01471615	0.296765199	8.406919
10	-0.02529798	0.329957742	10.12895
11	-0.03802189	0.316823389	11.13866
12	-0.05257738	0.260384088	11.46542
13	-0.06310043	0.188192245	11.07389
14	-0.0700806	0.135063202	10.70932
15	-0.07540493	0.097104729	10.48259
16	-0.07930162	0.067599303	10.28215
17	-0.08211892	0.046035095	10.13259
18	-0.08413427	0.030564086	10.02471
19	-0.08554453	0.019688942	9.948232
20	-0.0865065	0.012269971	9.89605
21	-0.08714373	0.007362037	9.861613
22	-0.08755101	0.004225985	9.839621
23	-0.08780044	0.002303632	9.826117
24	-0.087819	0.001540859	9.812717

Tabel B.1(Lanjutan**)** Data Training *altitude quadrotor*

25	0.912181002	0.001953125	-90.179
26	0.912181002	0.000976563	-90.1986
27	0.890197776	1.009675577	-67.8263
28	0.842621546	2.481662033	-33.6289
29	0.750927416	3.351622048	-7.0603
30	0.628870671	3.76449699	13.40287
31	0.482697995	3.615108716	25.03237
32	0.316537525	2.967512223	28.70
33	0.19677899	2.143951902	24.20
34	0.117214792	1.539136707	20.06
35	0.056534455	1.106366869	17.47
36	0.012130844	0.770095656	15.19
37	-0.01997198	0.524372341	13.48
38	-0.04293453	0.348093592	12.26
39	-0.05900063	0.224199232	11.38
40	-0.0699581	0.139692735	10.79
41	-0.07721528	0.083797939	10.40
42	-0.08185278	0.048089776	10.15
43	-0.08469216	0.026206433	9.99
44	-0.08634377	0.013428235	9.90
45	-0.08724604	0.006384406	9.852292
46	-0.08770207	0.002766444	9.825536
47	-0.0879126	0.001072537	9.812711
48	-0.08801387	0.00041901	9.809767
49	-0.0881839	0.000518748	9.828765
50	-0.08787846	-0.00125311	9.762784
51	-0.08855534	0.002776967	9.911073
52	-0.0878868	-0.00327097	9.72326

Tabel B.1(Lanjutan**)** Data Training *altitude quadrotor*

53	-0.0882951	0.00203317	9.870174
54	-0.08818666	-0.00070882	9.80449
55	-0.08809739	-0.00058356	9.798068
56	-0.08818615	0.000614144	9.830898
57	-0.08805391	-0.00092304	9.786931
58	-0.08814723	0.000641618	9.827555
59	-0.08807362	-0.00049287	9.797505
60	-0.0881178	0.000285653	9.817493
61	-0.08809163	-0.0001642	9.805879
62	-0.08810376	7.44E-05	9.811864
63	-0.08809901	-2.88E-05	9.809325
64	-0.08810074	1.05E-05	9.810283
65	-0.08809801	-1.66E-05	9.809468
66	-0.0881042	3.81E-05	9.811182
67	-0.08809351	-6.64E-05	9.808023
68	-0.08810835	9.27E-05	9.812689
69	-0.08809051	-0.00011192	9.806812
70	-0.08810996	0.000122209	9.81344
71	-0.08809011	-0.0001247	9.806517
72	-0.08810952	0.00012182	9.813388
73	-0.08809097	-0.00011622	9.806773
74	-0.08810856	0.000110076	9.813058
75	-0.08809179	-0.00010484	9.807082
76	-0.088108	0.000101227	9.812824
77	-0.08809208	-9.94E-05	9.807221

Tabel B.2 Data Training *roll quadrotor*

No	Error (rad)	True Value (rad)	Adjustment
1	1	0	0.8
2	1	6.94E-13	0.8
3	0.997044158	0.369556072	0.649813
4	0.990622921	0.929163253	0.420833
5	0.98077498	1.278627816	0.273169
6	0.962986884	1.560354137	0.146248
7	0.936823177	1.758368027	0.046111
8	0.903035075	1.831293657	-0.01009
9	8.60E-01	1.803595	-0.03311
10	0.792231981	1.691744704	-0.04291
11	0.734626124	1.554274753	-0.03401
12	0.688022713	1.454935152	-0.03156
13	0.642749398	1.362175519	-0.03067
14	0.599477922	1.270467837	-0.0286
15	0.558903119	1.184403357	-0.02664
16	0.520523557	1.103510589	-0.02499
17	0.484166471	1.026750306	-0.02337
18	0.449832063	0.954179786	-0.02181
19	0.417433828	0.885721521	-0.02034
20	0.386878626	0.82114642	-0.01896
21	0.358093521	0.760291723	-0.01764
22	0.331005194	0.703011418	-0.0164
23	0.305539773	0.649150196	-0.01523
24	0.281626371	0.598558043	-0.01412
25	0.259196251	0.551090991	-0.01308
26	0.238182382	5.07E-01	-0.0121
27	0.21851964	4.65E-01	-0.01117
28	0.200144795	4.26E-01	-0.01031

Tabel B.2 (Lanjutan) Data Training *roll quadrotor*

29	0.182996446	3.90E-01	-0.00949
30	0.167015	3.56E-01	-0.00873
31	0.152142653	0.324312783	-0.00801
32	0.138323364	0.295000814	-0.00734
33	0.125502828	2.68E-01	-0.00672
34	0.113628458	2.43E-01	-0.00613
35	0.102649359	2.19E-01	-0.00559
36	0.092516307	1.98E-01	-0.00508
37	0.083181729	1.78E-01	-0.00461
38	0.074599677	1.60E-01	-0.00418
39	0.066725815	1.43E-01	-0.00378
40	0.059517393	1.28E-01	-0.0034
41	0.052933229	1.14E-01	-0.00306
42	0.04693369	1.01E-01	-0.00274
43	0.041480671	8.91E-02	-0.00245
44	0.036537578	0.07853176	-0.00218
45	0.032069307	6.90E-02	-0.00194
46	0.028042228	6.04E-02	-0.00171
47	0.02442416	5.26E-02	-0.00151
48	0.02118436	4.57E-02	-0.00133
49	0.018293495	3.95E-02	-0.00116
50	0.015723631	0.033959078	-0.001
51	0.013448206	0.029064851	-0.00087
52	0.011442011	0.024743372	-0.00074
53	0.009681174	0.02094423	-0.00063
54	0.008143129	0.017619752	-0.00053
55	0.006806598	0.014724963	-0.00044
56	0.005651567	0.012217535	-0.00037
57	0.004659252	0.010057744	-0.0003

Tabel B.2 (Lanjutan) Data Training *roll quadrotor*

58	0.003812072	0.00820843	-0.00023
59	0.003093601	6.63E-03	-0.00018
60	0.002488514	5.31E-03	-0.00013
61	0.001982482	4.19E-03	-9.00E-05
62	0.00156198	3.26E-03	-5.53E-05
63	0.001213854	2.50E-03	-2.87E-05
64	0.000924326	1.88E-03	-1.37E-05
65	0.000678035	1.39E-03	-1.39E-05
66	0.000492402	8.29E-04	6.23E-05
67	-0.00018492	2.54E-03	-0.00117
68	-4.83E-05	-1.45E-03	0.000539
69	-3.66E-05	-1.59E-04	3.43E-05
70	-4.05E-05	5.27E-05	-5.35E-05
71	-1.78E-05	-3.27E-04	0.000117
72	-2.30E-05	7.21E-05	-4.72E-05
73	-2.04E-05	-3.35E-05	-2.95E-06
74	-9.81E-06	-1.27E-04	4.31E-05
75	-1.99E-05	1.12E-04	-6.06E-05
76	-2.54E-06	-1.79E-04	6.96E-05
77	-1.69E-05	1.42E-04	-7.01E-05

Tabel B.3 Data Training *pitch quadrotor*

No	Error (m)	True Value (rad)	Adjustment
1	1	0	1.2
2	1	1.04E-12	1.2
3	0.995718351	0.544998498	0.976863
4	0.987131564	1.352845168	0.64342
5	0.974764252	1.836641453	0.435061
6	0.952301199	2.23639625	0.248203
7	0.916409226	2.545582936	0.081458
8	0.870326022	2.667926563	-0.02278
9	0.812551273	2.610361935	-0.06908
10	0.72361437	2.391943649	-0.08844
11	0.652569202	2.126682073	-0.06759
12	0.5960894	1.942100187	-0.06153
13	0.541950026	1.772639808	-0.05872
14	0.491571727	1.60765447	-0.05318
15	0.445508655	1.456924858	-0.04816
16	0.402898335	1.318582649	-0.04396
17	0.363487767	1.190256512	-0.03992
18	0.327197437	1.071923897	-0.03613
19	0.29382533	0.963146437	-0.03267
20	0.263181435	0.863218417	-0.02947
21	0.235105913	0.771610429	-0.02652
22	0.209439584	0.687826956	-0.0238
23	0.186027264	0.611363821	-0.02131
24	0.164721857	0.5417432	-0.01903
25	0.145382519	0.478510864	-0.01695
26	0.127874204	0.421230282	-0.01504
27	0.112067892	0.369484058	-0.01331
28	0.097840416	0.322873864	-0.01174

Tabel B.3 (Lanjutan) Data Training *pitch quadrotor*

29	0.085074279	0.281019624	-0.01032
30	0.07365756	0.243559178	-0.00903
31	0.063483788	0.210147939	-0.00788
32	0.054451828	0.180458497	-0.00684
33	0.046465769	0.154180255	-0.00591
34	0.03943481	0.131019079	-0.00509
35	0.033273155	0.110696943	-0.00435
36	0.027899907	0.092951593	-0.0037
37	0.02323896	0.0775362	-0.00313
38	0.019218897	0.064219028	-0.00262
39	0.015772883	0.052783091	-0.00219
40	0.012838559	0.043025811	-0.0018
41	0.010357937	0.034758675	-0.00147
42	0.008277285	0.027806877	-0.00119
43	0.006547018	0.02200896	-0.00095
44	0.005121576	0.017216442	-0.00074
45	0.003959298	0.013293441	-0.00057
46	0.003022283	0.010116307	-0.00042
47	0.002276229	0.007573285	-0.0003
48	0.001690229	0.005564296	-0.0002
49	0.001236458	0.004001024	-0.00012
50	0.000889564	0.002807858	-5.57E-05
51	0.000625214	0.001924819	-1.97E-05
52	0.000416954	0.001307186	-2.25E-05
53	0.000283992	0.000638264	8.55E-05
54	0.000193698	0.000554067	1.08E-05
55	0.000160822	0.000331552	6.04E-05
56	9.66E-05	0.000647272	-0.00014
57	0.000100058	-3.85E-05	0.000135

Tabel B.3 (Lanjutan) Data Training *pitch quadrotor*

58	5.43E-05	0.000527991	-0.00015
59	6.39E-05	-0.00011099	0.000121
60	2.73E-05	0.000416446	-0.00013
61	4.32E-05	-0.00017917	0.000124
62	1.05E-05	0.000363557	-0.00013
63	3.10E-05	-0.00022699	0.000128
64	3.81E-07	0.00033607	-0.00013
65	2.39E-05	-0.00025715	0.000131
66	-5.47E-06	0.000321069	-0.00013
67	1.96E-05	-0.00027477	0.000133
68	-8.81E-06	0.000311646	-0.00014
69	1.70E-05	-0.00028406	0.000134
70	-1.07E-05	0.000305206	-0.00013
71	1.55E-05	-0.0002888	0.000134
72	-1.18E-05	0.000301031	-0.00013
73	1.46E-05	-0.00029139	0.000134
74	-1.25E-05	0.000298556	-0.00013
75	1.41E-05	-0.00029295	0.000134
76	-1.29E-05	0.00029718	-0.00013
77	1.38E-05	-0.00029393	0.000134

Tabel B.4 Data Training *yaw quadrotor*

No	Error (m)	True Value (rad)	Adjustment
1	1	0	1
2	1	4.98E-13	1
3	0.996570716	0.33922295	0.860882
4	0.987303559	0.916702147	0.620623
5	0.969828653	1.37628855	0.419313
6	0.936392565	1.784146251	0.222734
7	0.886611471	2.066190133	0.060135
8	0.822547771	2.151436592	-0.03803
9	0.744410726	2.06507954	-0.08162
10	0.632821627	1.823500651	-0.09658
11	0.552035193	1.553286506	-0.06928
12	0.487799999	1.376795134	-0.06292
13	0.427148502	1.213296615	-0.05817
14	0.373080995	1.05876892	-0.05043
15	0.325231809	0.923236187	-0.04406
16	0.282288882	0.802466053	-0.0387
17	0.243932885	0.694031778	-0.03368
18	0.209874689	0.597642473	-0.02918
19	0.179705995	0.512284936	-0.02521
20	0.153080164	0.436873029	-0.02167
21	0.129687816	0.370552177	-0.01853
22	0.109228378	0.312497287	-0.01577
23	0.091419987	0.261912743	-0.01335
24	0.076000705	0.218064198	-0.01122
25	0.062725477	0.180266288	-0.00938
26	0.051365868	0.147878469	-0.00779
27	0.04170978	0.120305917	-0.00641
28	0.033560746	0.096997525	-0.00524

Tabel B.4 (Lanjutan) Data Training *yaw quadrotor*

29	0.026737383	0.077444147	-0.00424
30	0.021072884	0.061177269	-0.0034
31	0.016414491	0.047767538	-0.00269
32	0.012622977	0.036823311	-0.00211
33	0.00957213	0.027989229	-0.00162
34	0.007148226	0.020944756	-0.00123
35	0.005249502	0.015402712	-0.00091
36	0.003785595	0.011107758	-0.00066
37	0.002676973	0.007834855	-0.00046
38	0.001854301	0.005387714	-0.0003
39	0.001257724	0.003597353	-0.00018
40	0.000835921	0.002321082	-9.25E-05
41	0.000544463	0.001442929	-3.27E-05
42	0.000341857	0.000877941	-9.32E-06
43	0.000183351	0.000559836	-4.06E-05
44	0.000130792	0.000250623	3.05E-05
45	6.47E-05	0.000315195	-6.14E-05
46	0.000184385	-0.00038988	0.00034
47	4.97E-05	0.00070104	-0.00023
48	7.57E-05	-0.00017388	0.000145
49	1.51E-05	0.000406004	-0.00015
50	3.73E-05	-0.00015751	0.0001
51	5.12E-06	0.00023164	-8.75E-05
52	2.33E-05	-0.00013047	7.54E-05
53	-3.94E-06	0.000192979	-8.11E-05
54	1.90E-05	-0.00016033	8.31E-05
55	-1.01E-05	0.000200014	-9.01E-05
56	1.77E-05	-0.00018868	9.32E-05
57	-1.34E-05	0.000209707	-9.73E-05

Tabel B.4 (Lanjutan) Data Training *yaw quadrotor*

58	1.70E-05	-0.00020486	9.90E-05
59	-1.49E-05	0.000214633	-0.0001
60	1.65E-05	-0.0002113	0.000101
61	-1.53E-05	0.000214697	-0.0001
62	1.60E-05	-0.00021185	0.000101
63	-1.54E-05	0.000212565	-0.0001
64	1.57E-05	-0.00021066	9.99E-05
65	-1.54E-05	0.000210802	-9.97E-05
66	1.55E-05	-0.0002099	9.95E-05
67	-1.54E-05	0.000210094	-9.95E-05
68	1.55E-05	-0.00020982	9.94E-05
69	-1.55E-05	0.000210053	-9.95E-05
70	1.55E-05	-0.00021002	9.95E-05
71	-1.55E-05	0.000210203	-9.96E-05
72	1.55E-05	-0.00021022	9.96E-05
73	-1.55E-05	0.000210305	-9.96E-05
74	1.55E-05	-0.0002103	9.96E-05
75	-1.55E-05	0.000210326	-9.96E-05
76	1.55E-05	-0.00021031	9.96E-05
77	-1.55E-05	0.000210308	-9.96E-05

Berikut adalah *coding* pada MATLAB :

Code Parameter NN

```
%% UJI QUAD
clc
clear all
close all
%% Konstanta Quadrotor
Ixx = 8.1e-3;     % momen inersia pada X
Iyy = 8.1e-3;     % momen inersia pada Y
Izz = 14.2e-3;    % momen inersia pada Z
Jtp = 104e-6;     % Total rotational moment of
inertia around the propeller axis
b   = 54.2e-6;    % Thrust factor
d   = 1.1e-6;     % Drag factor
l   = 0.165;      % Distance to the center of the
Quadrotor
m   = 1;          % Mass of the Quadrotor in Kg
g   = 9.81;       % akselarasi Gravitasi
%% NN Control Quadrotor
%% Param Control altitude/Z/z
    data_train =
xlsread('data_training.xls','z');
    [s1,s2] = size(data_train);
    x = data_train(:,1:2);
    y = data_train(:,3);
evalin('base','load(''nn_75_7k_0.5_0.05472_used_
z.mat'')');
    bobot1z = evalin('base','w1');
    bobot2z = evalin('base','w2');
    inminz = evalin('base','min');
    inmaxz = evalin('base','max');
    inminxz = evalin('base','xmin');
    inmaxxz = evalin('base','xmax');
%% Param Control roll/phi/p
    data_train =
xlsread('data_training.xls','p');
    [s1,s2] = size(data_train);
    x = data_train(:,1:2);
    y = data_train(:,3);
```

```
evalin('base','load(''nn_70_1k_0.25_0.0007759_us
ed_p.mat'')');
    bobot1p = evalin('base','w1');
    bobot2p = evalin('base','w2');
    inminp = evalin('base','min');
    inmaxp = evalin('base','max');
    inminxp = evalin('base','xmin');
    inmaxxp = evalin('base','xmax');
%% Param Control pitch/theta/q
    data_train =
xlsread('data_training.xls','q');
    [s1,s2] = size(data_train);
    x = data_train(:,1:2);
    y = data_train(:,3);
evalin('base','load(''nn_80_3k5_0.1_0.0007835_us
ed_q.mat'')');
    bobot1q = evalin('base','w1');
    bobot2q = evalin('base','w2');
    inminq = evalin('base','min');
    inmaxq = evalin('base','max');
    inminxq = evalin('base','xmin');
    inmaxxq = evalin('base','xmax');
```

Code Training NN

```
clear
clc
%%
%param
hidden_neurons = 75;
epochs = 500;
learn_rate = 0.5;

serror = [];
serror_q = [];
delta_W_Out = [];
%%
%input data
data_train = xlsread('data_training.xls','z');
[s1,s2] = size(data_train);
n_input = 2;          %number of input
n_output = 1;         %number of output
x = data_train(:,1:n_input);    %get data input
y = data_train(:,n_input+1:n_input+n_output);
%get data output

%%
inmax = max(max(x));          %max number input
inmin = min(min(x));          %min number input
outmax = max(max(y));         %max number output
outmin = min(min(y));         %min number output

norm_In = (2*(x(:,:)-inmin)/(inmax-inmin))-1;
%Nilai In yang sudah di normalisasi
norm_Out = (2*(y(:,:)-outmin)/(outmax-outmin))-
1;       %Nilai Out yang sudah di normalisasi
%%
%bobot and bias
jumlah_data_input = size(norm_In,1);
%jumlah data input
bias_In = ones(jumlah_data_input,1);
%jumlah bias pada input
norm_Input_NN = [norm_In bias_In];
%nilai normalisasi + bias
```

```
n_NN_input = size(norm_Input_NN,2);
%jumlah input NN
weight_hidden_input =
(randn(n_NN_input,hidden_neurons)/10);    %nilai
bobot Input
weight_hidden_output =
(randn(n_output,hidden_neurons+1)/10);   %nilai
bobot output
%%
% iterasi NN
for iterasi = 1:epochs
    % Train
       for Q = 1:jumlah_data_input
        step_1 = norm_Input_NN(Q,:);
        true = norm_Out(Q,:);
        hidden_value =
(tanh(step_1*weight_hidden_input))';
%perkalian pada hidden layer
        hidden_value2 = [hidden_value;1];
        step_2 =
hidden_value2'*weight_hidden_output';
%perkalian untuk output
        error = step_2 - true;
%error terhadap target

        delta_W_Out=[];

        for R=1:n_output
           delta_W_Out =
[delta_W_Out;(error(1,R).*weight_hidden_output(R
,:)).*((learn_rate).*hidden_value2')];
        end

        weight_hidden_output =
weight_hidden_output - delta_W_Out;
%update weight hidden input
```

```
        error_1 =
error*weight_hidden_output(:,1:hidden_neurons);
        delta_hidden_input=
learn_rate.*error_1'.*0.5.*(1-
(hidden_value.^2))*step_1;
        weight_hidden_input =
weight_hidden_input - delta_hidden_input';
%update weight hidden output
       end
    hidden_val =
(tanh(norm_Input_NN*weight_hidden_input))';
    h_hidden_val = [hidden_val;ones(1,s1)];
    out_not_norm =
weight_hidden_output*h_hidden_val;

    out_norm = ((outmax-
outmin)*(out_not_norm+ones(n_output,s1))/2)+outm
in; %de-normalisasi nilai output
    error_train = out_not_norm' - norm_Out;
    errorq = out_norm'-y;
    error_rmse =
((sum(sum(errorq.^2)))/jumlah_data_input).^0.5;
    error_rmse_2 =
((sum(sum(error_train.^2)))/jumlah_data_input).^
0.5;
    serror = [serror;error_rmse_2];
    serror_q = [serror_q;error_rmse];
end
%%
plot(serror_q);
assignin('base','w1',weight_hidden_input);
assignin('base','w2',weight_hidden_output);
assignin('base','min',outmin);
assignin('base','max',outmax);
assignin('base','xmin',inmin);
assignin('base','xmax',inmax);
uisave({'w1','w2','min','max','xmin','xmax'},'nn
');
```

NN Control Code

```
function A1   =
fcn(ez,dez,bobot1z,bobot2z,inminz,inmaxz,inminxz
,inmaxxz)
inz =(2*([ez dez]-inminxz)/(inmaxxz-inminxz))-1;
Bz = [tanh([inz 1]*bobot1z) 1]*bobot2z;
A1 = ((inmaxz-inminz)*(Bz+[1])/2)+inminz;

function A2 =
fcn(ep,dep,bobot1p,bobot2p,inminp,inmaxp,inminxp
,inmaxxp)
inp =(2*([ep dep]-inminxp)/(inmaxxp-inminxp))-1;
Bp = [tanh([inp 1]*bobot1p) 1]*bobot2p;
A2= ((inmaxp-inminp)*(Bp+[1])/2)+inminp;

function A3 =
fcn(eq,deq,bobot1q,bobot2q,inminq,inmaxq,inminxq
,inmaxxq)
inq =(2*([eq deq]-inminxq)/(inmaxxq-inminxq))-1;
Bq = [tanh([inq 1]*bobot1q) 1]*bobot2q;
A3= ((inmaxq-inminq)*(Bq+[1])/2)+inminq;
```

Penulis bernama Yuhara Rahmantya, lahir pada tanggal 29 Juli 1992 di Surabaya Jawa Timur. Riwayat pendidikan formal penulis dimulai pada tahun 1998 di SDN Barata Jaya 2 Surabaya selanjutnya pada tahun 2004 penulis melanjutkan pendidikan di SMP NEGERI 12 Surabaya dan pada tahun 2007 pendidikan di SMA NEGERI 16 Surabaya. Setelah lulus dari pendidikan menengah atas, penulis melanjutkan kuliah di Institut Teknologi Sepuluh Nopember Surabaya dengan prodi S1 Teknik Fisika. Selama menjalani pendidikan di ITS Surabaya, penulis melakukan kegiatan selayaknya mahasiswa pada umumnya, dengan keseharian ngopi dan rokok untuk penyegaran setelah beraktifitas sebagi admin dari salah satu lab di Teknik Fisika Laboratorium Simulasi dan Komputasi sebagai penanggung jawab untuk Jaringan jurusan. Dalam menerima tanggung jawab tersebut penulis tidak pernah menganggap bahwa itu sebagai beban atau masalah dalam keseharian dengan sibuknya perkuliahan, karena penulis menjalani itu semua dengan anggapan bahwa penulis masih bermanfaat untuk sekitarnya meskipun itu tidak berarti.