BAB II

LANDASAN TEORI

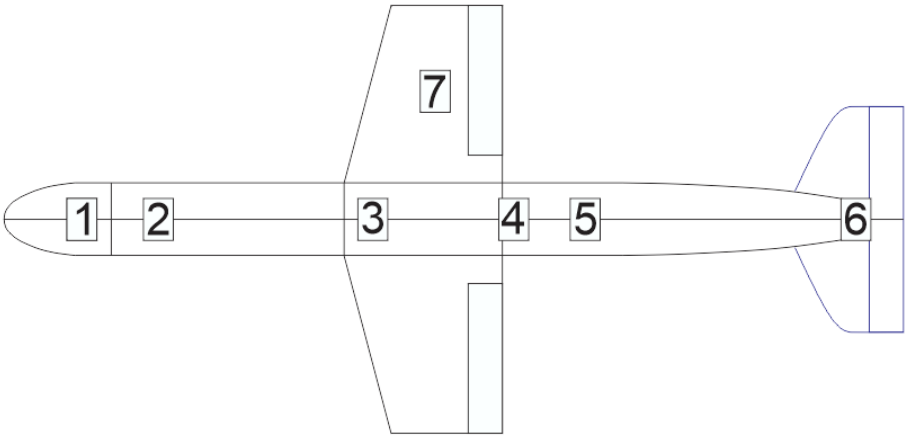
Pada bab ini akan menerangkan teori-teori penunjang untuk merancang sebuah *firmware* untuk kestabilan roket *Electric Ducted Fan (EDF)* pada saat *cruising* dengan sistem kendali, adapun teori yang akan dipaparkan sebagai berikut.

2.1 Roket dan Kendali Sikap Roket

Pada umumnya roket adalah sejenis sistem propulsi yang membawa bahan bakar dan oksigennya sendiri. Dorongan pada roket merupakan penerapan yang menarik dari hukum III Newton dan Kekekalan momentum yaitu dengan memancarkan aliran massa hasil pembakaran *propellant*. Roket memiliki tangki yang berisi bahan bakar hidrogen cair dan oksigen cair. Proses pembakaran dilakukan pada ruang tertentu, pembakaran tersebut menghasilkan gas buang yang disalurkan melalui mulut pipa pada ujung roket, maka terjadi perubahan momentum pada gas selama selang waktu tertentu.

Roket EDF adalah roket elektrik yang tidak menggunakan bahan bakar, roket EDF menggunakan baterai sebagai sumber tenaga untuk meluncur, roket ini meluncur dengan daya dorong dari motor EDF yang diatur dengan *driver* motor. Perbedaan roket bahan bakar dengan roket EDF hanya pada pendorong dan kontrol terbangnya, kebanyakan dari roket bahan bakar diatur pada ujung pengeluaran bahan bakar sedangkan roket EDF biasanya menggunakan pin kontrol.

Roket EDF terdiri dari beberapa bagian penting agar dapat meluncur dengan baik, bagian tersebut antara lain badan roket, kompoen elektronika, dan *firmware.* Berikut ini bagian-bagian badan roket yang di ilustrasikan pada Gambar 2.1.



Gambar 2.1 Bagian-bagian pada roket EDF

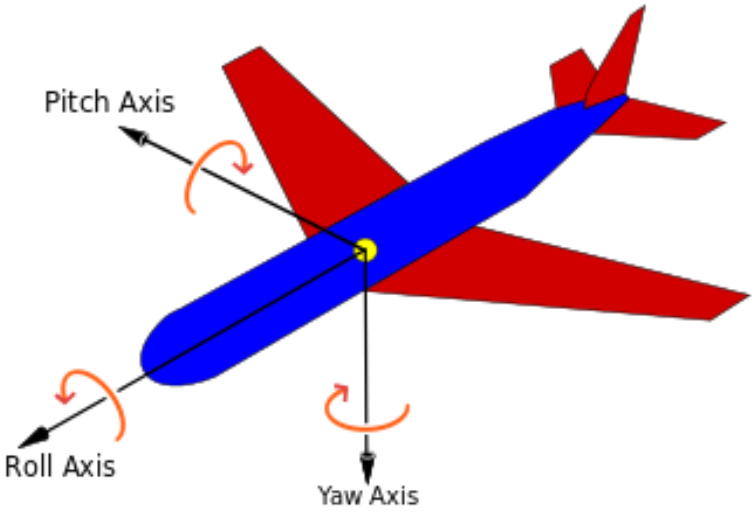
Berikut ini penjelasan dari setiap bagian pada Gambar 2.1

- Hidung roket (*Nose cone*) yaitu ujung roket atau bagian terdepan pada roket yang berbentuk kerucut yang bertujuan untuk memecah angin ketika meluncur.
- Badan roket yaitu tempat menyimpan muatan roket, pada roket EDF muatan yang dibawa berupa komponen-komponen elektronika dan parasut.
- Ekor roket yaitu ujung dari roket yang diberi sirip sebagai kontrol ketika roket terbang. Ekor roket biasanya berukuran lebih kecil dari badan agar roket meluncur lebih cepat.
- Penjelasan bagian nomor 1 pada gambar adalah bagian *nose* atau bagian depan roket yang berbentuk kerucut yang bertujuan untuk memecah angin ketika meluncur.
- Penjelasan nomor 2 pada gambar adalah tempat penyimpanan baterai agar roket seimbang jika diukur pada titik *Center of Gravity* (CG). CG adalah

pusat massa roket atau bisa dibilang titik keseimbangan roket ketika meluncur.

- Penjelasan nomor 3 pada gambar adalah tempat penyimpanan parasut, tempat penyimpanan parasut dilengkapi dengan pelontar untuk melontarkan parasut ketika akan melakukan separasi pada saat roket sudah mencapai target.
- Penjelasan nomor 4 pada gambar adalah ruang sistem roket, ruang sistem tersebut sebagai pusat kendali dari roket EDF ketika *autonomous.* didalam ruang sistem terdapat komponen berupa *flight controller* sebagai kontrol dari roket ketika terbang secara *autonomous.*
- Penjelasan nomor 5 pada gambar adalah bagian pendorong yang disimpan dibawah roket, pendorongnya menggunakan motor tipe EDF. Pendorong yang ideal yaitu pendorong yang memiliki perbandingan minimal 1:2 dengan berat roket keseluruhan.
- Penjelasan nomor 6 pada gambar adalah bagian sirip (*fin*) yang berfungsi sebagai pembentuk titik stabilisasi dari roket. Sirip ini digunakan juga sebagai kontrol *pitch* dan *yaw* pada saat roket terbang.
- Penjelasan nomor 7 pada gambar sama dengan bagian nomor 6, hanya berbeda penempatannya saja. Pada bagian nomor 7 disimpan di badan roket untuk kontrol *roll* roket.

Adapun gerakan yang dihasilkan oleh sirip roket yaitu gerakan *roll, pitch* dan *yaw* berikut ini adalah penjelasannya, seperti yang ditunjukkan pada Gambar 2.2.

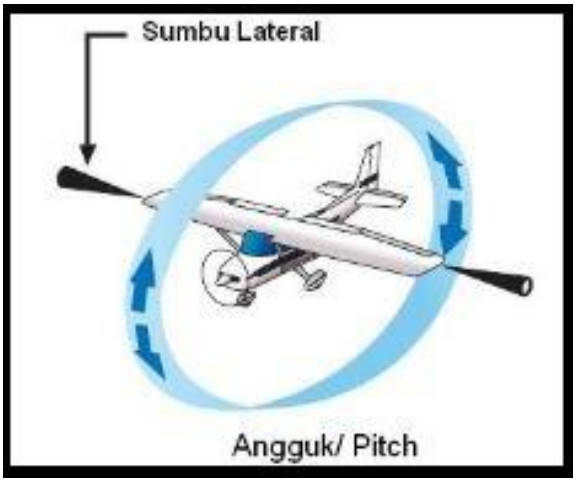


Gambar 2.2 Tiga sumbu kendali pada roket EDF

Gerakan kendali *roll* dihasilkan oleh *fin* pada sirip yang berada pada badan roket yang terletak pada bagian sayap kanan dan kiri. Sirip tersebut berfungsi sebagai *aileron* jika pada pesawat terbang, gerakan *roll* yaitu gerakan gerakan yang berorientasi berputar kearah kanan dan kiri[4]. Cara kerja pergerakan *roll* yaitu dengan menggerakan ujung sirip berlawanan arah, ujung sirip tersebut akan menjadi pengangkat utama roket. Jadi pada saat *fin* sirip yang satu dinaikan maka *fin* sirip yang satu lagi di turunkan, *fin* sirip yang naik akan terkena angin lalu membuat *fin* sirip tertekan kebawah dan *fin* sirip yang turun akan terkena angin dan membuat *fin* sirip tertekan keatas. Jadi gerakan memutarnya ditunjukan dengan *fin* sirip yang naik ke atas, jika *fin* sirip pada sayap kanan naik maka roket akan berorientasi memutar kearah kiri begitupun sebaliknya jika *fin* sirip pada sayap kiri naik maka roket akan berorientasi memutar kearah kanan seperti pada Gambar 2.3

Gambar 2.3 Gerakan kendali *roll*

Pada gerakan kendali *pitch* bertujuan untuk mengontrol roket naik dan turun, pada pesawat terbang dikenal dengan *elevator* dimana ketika pesawat akan lepas landas maka *elevator* akan naik dan ketika akan turun mendarat *elevator* akan turun[4]. Gambar 2.4 menunjukkan pergerakan *pitch*



Gambar 2.4 Gerakan kendali *pitch*

Gerakan kendali *yaw* dilakukan oleh sirip pada ekor roket namun yang vertikal dengan badan roket untuk mengarahkan roket ke kiri dan ke kanan, kendali *yaw* berbeda dengan kendali *roll* karena kendali *yaw* tidak memutarkan badan roket

ke kiri dan ke kanan, sirip tersebut jika pada pesawat terbang dikenal dengan *rudder*[4], berikut ini Gambar 2.5 menunjukkan pergerakan *yaw*

Gambar 2.5 Gerakan kendali *yaw*

2.2 Komponen Elektronika

Perancangan *firmware* pada roket EDF sangat bergantung pada komponen elektronik yang digunakan, karena *firmware* yang dirancang akan ditanam pada mikrokontroler. Mikrokontroler didukung oleh komponen elektronika lainnya yaitu sensor *gyroscope, accelerometer, barometer,* motor servo, *electric speed control* (ESC), motor EDF dan baterai. Berikut penjelasan tentang komponen-komponen elektronik yang mendukung perancangan *firmware* roket EDF.

2.2.1 Mikrokontroler

Mikrokontroler adalah *processing unit* seperti komputer versi mini dan untuk aplikasi khusus yang dikemas menjadi suatu *integrated circuit*, pada perancangan ini mikrokontroler befungsi untuk membaca data dari sensor, melakukan perhitungan matematis dan mengendalikan aktuator. Penggunaan utama

mikrokontroler adalah mengontrol operasi sebuah mesin mengunakan program yang disimpan dalam ROM dan tidak berubah sepanjang umur sistem tersebut.

Mikrokontroler merupakan komputer didalam chip yang digunakan untuk mengontrol peralatan elektronik, yang menekankan efisiensi dan efektifitas biaya. Secara harfiahnya bisa disebut "pengendali kecil" dimana sebuah sistem elektronik yang sebelumnya banyak memerlukan komponen-komponen pendukung seperti IC TTL dan CMOS dapat direduksi/diperkecil dan akhirnya terpusat serta dikendalikan oleh mikrokontroler ini.[5]

Mikrokonktroler digunakan dalam produk dan alat yang dikendalikan secara automatis, seperti sistem kontrol mesin, *remote controls*, mesin kantor, peralatan rumah tangga, alat berat, dll. Dengan mengurangi ukuran, biaya, dan konsumsi tenaga dibandingkan dengan mendesain menggunakan mikroprosesor memori, dan alat *input output* yang terpisah, kehadiran mikrokontroler membuat kontrol elektrik untuk berbagai proses menjadi lebih ekonomis. Mikrokontroler memerlukan komponen eksternal yang disebut dengan sistem minimum. Untuk membuat sistem minimal paling tidak dibutuhkan sistem clock dan reset, walaupun pada beberapa mikrokontroler sudah menyediakan sistem *clock internal*, sehingga tanpa rangkaian eksternal pun mikrokontroler sudah beroperasi. Yang dimaksud dengan sistem minimal adalah sebuah rangkaian mikrokontroler yang sudah dapat digunakan untuk menjalankan sebuah aplikasi. Sebuah IC mikrokontroler tidak akan berarti bila hanya berdiri sendiri[5].

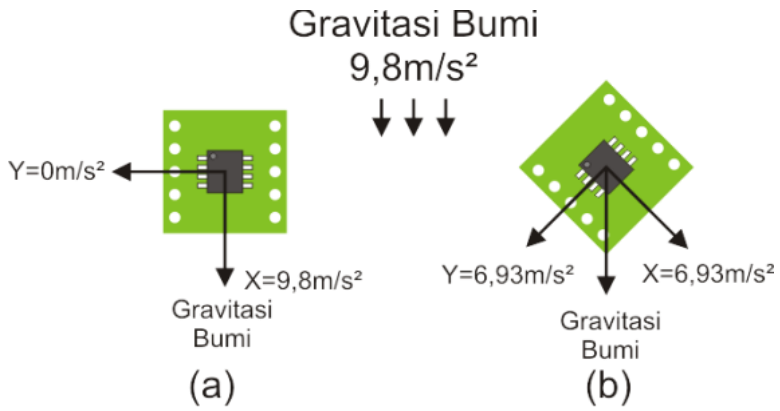
2.2.2 Sensor *Inertia Measurment Unit(IMU)*

Inertia Measurment Unit (IMU) merupakan gabungan beberapa sensor yang digunakan untuk mengukur perubahan inersia yang terjadi pada suatu benda. Kombinasi yang digunakan berupa sensor *accelerometer* dan *gyroscope* yang kadangkala ditambahkan beberapa sensor lain seperti *magnetometer* dan *barometer*. Sensor ini telah banyak digunakan pada beberapa perangkat navigasi pada kendaraan maupun *mobile device*.

2.2.2.1 Sensor *Accelerometer*

Sensor *accelerometer* adalah sensor yang mengukur akselerasi pada lingkungannya. Beberapa jenis *accelerometer* menggunakan konsep kerja pegas. Perubahan panjang yang terjadi pada pegas akan menjadi nilai akselerasi yang dihasilkan[6].

Selain mengukur akselerasi, *accelerometer* mampu mengukur kemiringan suatu benda dengan melakukan proses perbandingan terhadap gaya gravitasi bumi yang dapat dirasakan oleh sensor ini. Berikut ini ilustrasi pembacaan gravitasi bumi.



Gambar 2.6 Pembacaan gravitasi bumi pada sensor *accelerometr*

Pada gambar 2.6 tersebut terlihat adanya perubahan pembacaan percepatan gaya gravitasi bumi pada saat sensor *accelerometer* dimiringkan. Pada kondisi (a), sumbu x membaca keseluruhan gaya gravitasi bumi. Sedangkan pada kondisi (b), sumbu x dan sumbu y memiliki besar masing-masing pembacaan gaya gravitasi bumi. Dengan memanfaatkan persamaan dibawah ini, perubahan sudut yang terjadi pada sensor *accelerometer* dapat diketahui.

$$Degree = \tan^{-1}\left(\frac{Accelerometer\ X}{Accelerometer\ Y}\right)$$

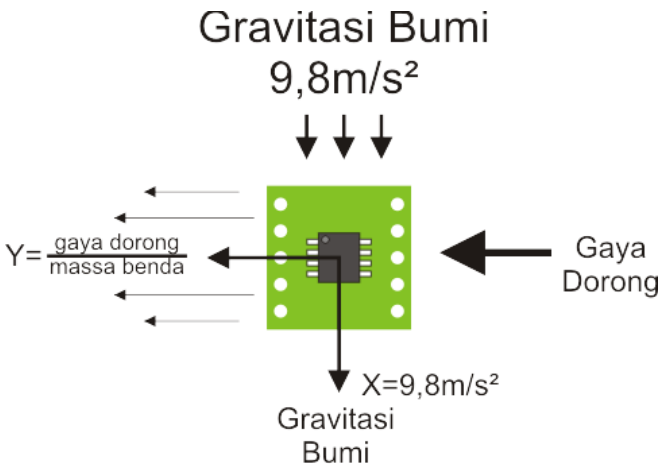
Pada kondisi (a):

$$Degree = \tan^{-1}\left(\frac{9{,}18\ m/s^2}{0\ m/s^2}\right) = \tan^{-1}(\infty) = 90°$$

Pada kondisi (b) :

$$Degree = \tan^{-1}\left(\frac{6{,}93\ m/s^2}{6{,}93\ m/s^2}\right) = \tan^{-1}(1) = 45°$$

Perhitungan tersebut membuktikan bahwa dengan memanfaatkan gaya gravitasi bumi, sensor *accelerometer* dapat mengukur sudut benda. Namun proses pembacaan akan terganggu disaat sensor mendapat *noise* berupa getaran atau akselerasi dari luar.



Gambar 2.7 Noise *Accelerometer*

Noise tersebut merupakan kelemahan dari sensor *accelerometer*. Penggunaan low-pass filter dan pembatasan besar percepatan merupakan salah satu solusi untuk mengatasi kelemahaan *accelerometer* untuk pembacaan sudut.

2.2.2.2 Sensor *Gyroscope*

Gyroscope adalah perangkat *Micro-machined Electro-Mechanical Systems* (MEMS) yang berfungsi untuk mengukur kecepatan sudut atau mempertahankan gerak rotasi dengan memanfaatkan gaya mekanis. Satuan kecepatan sudut yang diukur dalam derajat per detik (°/s) atau revolusi per detik (RPS).

Gyroscope juga dapat digunakan untuk menentukan orientasi, yang sering ditemukan di sebagian besar sistem navigasi otonom. Misalnya, jika ingin menyeimbangkan robot, *gyroscope* dapat digunakan untuk mengukur rotasi dari posisi seimbang dan mengirimkan koreksi ke motor.

Tiga sumbu *gyroscope*, dapat mengukur rotasi sekitar tiga sumbu: x, y, dan z. Beberapa *gyroscope* terdiri dari sumbu tunggal dan ganda, tetapi yang banyak digunakan adalah jenis *gyroscope* yang memiliki tiga sumbu dalam satu chip.

Dengan memanfaatkan data kecepatan sudut tersebut dapat diketahui kemiringan (*roll, pitch, yaw*) suatu benda. Untuk mengetahui nilai sudut, data kecepatan sudut harus terintegrasi. Persamaannya adalah berikut ini:

$$\theta_{(x,y,z)} = \int gyro(x,y,z)\, dt \quad \text{.....(2.1)}$$

$$dt = T = t_n - t_{n-1} \quad \text{.....(2.2)}$$

dimana

T = sample time atau waktu siklus

n = {1,2,3…} contoh nilai

$gyro_{(x,y,z)}$ = data giroskop (°/s)

2.2.2.3 Sensor *Barometer*

Barometer adalah sensor untuk mengukur tekanan udara dengan nilai keluaran berupa satuan Pa (pascal). Sensor ini berfungsi untuk mengukur tekanan udara disekitar, dengan adanya diafragma yang sensitif terhadap tekanan maka ketinggian dari diafragma diukur untuk menentukan tekanan di lingkungan sekitar sensor, dengan perhitungan matematis nilai tekanan tersebut dapat dikonversikan menjadi nilai ketinggian karena ketinggian berbanding terbalik dengan tekanan udara.

Persamaan untuk mengukur ketinggian menggunakan sensor barometer adalah sebagai berikut:

$$Altitude = 44330 \times 1 - \left(\left(\frac{p}{p_0}\right)^{\frac{1}{5.255}}\right) \quad \text{.....(2.3)}$$

Dimana :
p = tekanan (Pa)
p_0 = *sea level standard atmospheric pressure* (101325Pa)
altitude = nilai ketinggian (mdpl)

2.2.3 *Electronic Speed Control(ESC)*

Electronic Speed Control adalah sebuah modul untuk mengontrol kecepatan motor listrik, mengontrol arah putaran motor dan sebagai rem dinamis pada motor listrik. *ESC* biasa digunakan pada model radio yang dikendalikan secara elektrik dan paling sering digunakan untuk motor *brushless*, pada dasarnya *ESC* terhubung ke kontrol *throttle* penerima atau sering disebut sebagai *receiver* dan memberikan arus pada motor untuk memvariasikan kecepatan motor brushless.

Kinerja dari *ESC* sangatlah cepat untuk menghidupkan atau mematikan pulsa pada motor, cepatnya kinerja dari *ESC* dipengaruhi oleh komponen yang terdapat dalam *ESC* itu sendiri, *ESC* menggunakan *microprosessor* sebagai pusat kendali sehingga respon pada *ESC* sangat cepat dan bisa di program juga sesuai dengan kebutuhan.

2.2.4 *Motor Electric Ducted Fan (EDF)*

Electric Ducted Fan (EDF) adalah jenis motor *brushless* yang memiliki baling-baling yang dibungkus oleh sebuah tabung yang berfungsi untuk mengevisiensikan aliran udara yang mengalir, kecepatan dan tekanan udara, penggunaan tabung ini dapat mengurangi kerugian-kerugian yang dihasilkan pada penggunaan baling-baling biasa. *Propeller* atau baling-baling yang digunakan pada motor EDF adalah *blade* yang dilindungi dengan tabung dan *blade EDF* biasanya berjumlah 4 atau lebih sedangkan *propeller* hanya 2 atau 3.

Gambar 2.8 *Electric Ducted Fan(EDF)*

Motor EDF memiliki beberapa kelebihan dibandingkan dengan motor pada umumnya yaitu.

a. Kecepatannya lebih baik dibanding kekuatannya (torsi).

b. Responnya lebih cepat.

c. Lebih efisien.

d. Massa operasi lebih lama.

e. Tidak terlalu bising.

f. Rentang kecepatan pada motor EDF lebih luas.

Pada motor EDF tadi disebut memiliki *stator* dan *rotor* seperti motor pada umumnya. Berikut penjelasan mengenai *stator* dan *rotor*. *Stator* adalah laminasi besi yang diberi lilitan yang diletakkan pada pusat putaran motor atau inti motor yang diberi lilitan namun lilitan pada motor EDF berbeda dengan motor pada umumnya.

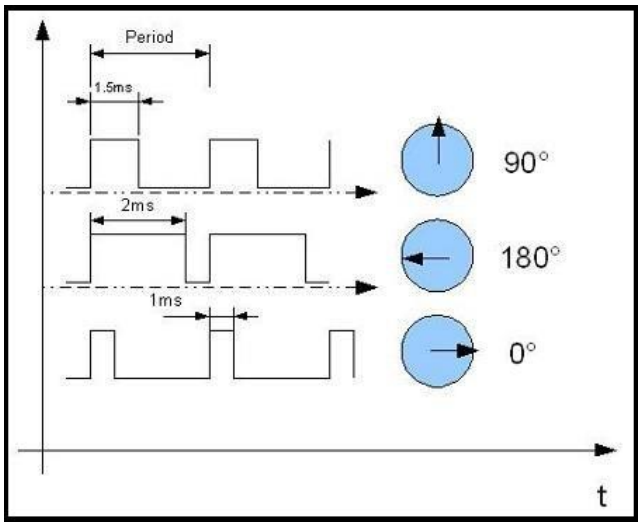
Motor pada umumnya memiliki dua keluaran lilitan pada *stator* namun pada motor EDF memiliki tiga keluaran lilitan yang terhubung dalam bentuk asterisk atau bintang. *Rotor* adalah bagian luar dari motor EDF yang terdiri dari lingkaran logam yang memiliki magnet permanen dua pasang atau lebih magnet kutub utara (U) dan selatan (S). bahan yang digunakan adalah *Ferrite magnet* atau bahan pembuat magnet permanen.

2.2.5 Motor Servo

Motor Servo adalah jenis motor DC yang tersusun dari motor DC, *gearbox*, dan potensiometer. Motor servo dalam wahana terbang digunakan untuk sistem aktuator pada *Aileron, Elevator* dan *Rudder*. Motor servo terintegrasi dengan *flight controller*, untuk dapat bergerak motor servo memerlukan tegangan dan sinyal *Pulse Width Modulation* (PWM) untuk kendali motor servo tersebut. Sinyal *Pulse Width Modulation* (PWM) yang didapat servo dari *flight controller* merupakan pengolahan data semua sensor yang diolah dan dikeluarkan dalam bentuk sinyal

Pulse Width Modulation (PWM) agar servo bergerak. Motor servo memiliki banyak jenis tergantung dari torsi atau daya angkat yang dihasilkan oleh motor servo.

Cara kerja dari motor servo bergantung pada nilai yang dihasilkan oleh komponen potensiometer yang akan diubah oleh komponen ADC menjadi data digital. Data tersebut akan dibandingkan dengan sinyal masukan oleh komparator. Hasil perbandingan komparator akan menjadi penentu arah pergerakan motor. Sinyal masukan pada motor servo hampir berkerja sama seperti pada *Electronic Speed Control*(ESC). Hanya saja, pada ESC terjadi perubahan kecepatan motor sedangkan pada motor servo terjadi perubahan sudut seperti pada Gambar 2.9 ini.



Gambar 2.9 Lebar pulsa motor servo

2.2.6 *Battery Lithiun-Polimer(Li-Po)*

Baterai adalah perangkat yang mengandung sel listrik yang dapat menyimpan energi yang dapat dikonversi menjadi daya. Baterai menghasilkan listrik melalui proses kimia. Baterai atau akumulator adalah sebuah sel listrik dimana didalamnya berlangsung proses elektrokimia yang *reversible* (dapat berkebalikan) dengan

efisiensinya yang tinggi. Yang dimaksud dengan reaksi elektrokimia *reversibel* adalah didalam baterai dapat berlangsung proses pengubahan kimia menjadi tenaga listrik (proses pengosongan) dan sebaliknya dari tenaga listrik menjadi tenaga kimia (proses pengisian) dengan cara proses regenerasi dari elektroda-elektroda yang dipakai yaitu, dengan melewatkan arus listrik dalam arah polaritas yang berlawanan didalam sel.

Baterai terdiri dari dua jenis yaitu, baterai primer dan baterai sekunder. Baterai primer merupakan baterai yang hanya dapat dipergunakan sekali pemakaian saja dan tidak dapat diisi ulang. Baterai yang digunakan adalah baterai jenis sekunder yaitu lipo (*lithium polymer*) yang memiliki kelebihan dapat mengeluarkan daya yang besar atau *discharge* yang cepat sesuai dengan keterangan yang tercantum pada baterai lipo.

2.3 Firmware

Firmware adalah sebuah perangkat lunak yang menyerupai sistem operasi pada perangkat lunak yang bersifat permanen yang terdapat dan ditanamkan pada perangkat keras elektronika. *Firmware* memiliki sifat yang hampir sama dengan sistem operasi yaitu berfungsi sebagai otak pada perangkat keras yang ditanamkan, dimana perangkat keras tidak akan bisa menjalankan berbagai perintah tanpa adanya *firmware* dan sistem operasi[3].

Firmware merupakan perangkat tegar yang mengacu pada perangkat lunak yang disimpan di *read-only memory* (ROM) yang ada pada perangkat keras (*hardware*), perangkat keras yang digunakan untuk *firmware* sistem kestabilan roket yaitu pixhawk. *Firmware* seperti *basic input/output system* (BIOS) pada

komputer. *Firmware* merupakan sekumpulan algoritma yang dapat memberikan intruksi intruksi pada perangkat keras, namun *firmware* harus sesuai dengan perangkat keras yang akan digunakan. Seperti BIOS yang menyesuaikan algoritmanya untuk memberikan intruksi pada perangkat komputer, tidak bisa jika BIOS memberikan intruksi pada pixhawk.

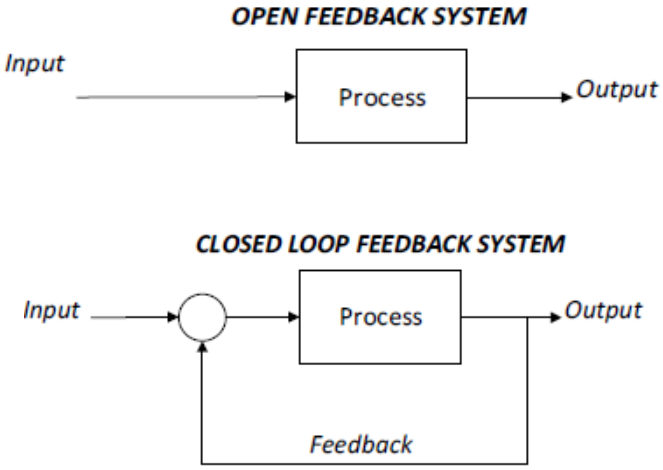
Firmware seringkali disamakan dengan *software* karena memiliki fungsi yang mirip, namun pada dasarnya *firmware* berbeda dengan *software*. Pada penggunaanya *firmware* digunakan untuk mengontrol aktuator seperti motor servo dan ESC, sedangkan *software* seperti *mission planner* yang menghubungkan pengguna dengan mikrokontroler untuk mengatur aktuator.

2.4 Sistem Kendali

Sistem kendali adalah suatu sistem yang mengatur, memerintah dan mengendalikan keadaan satu parameter atau lebih sehingga akan mendapatkan sebuah nilai atau harga pada suatu rangkuman tertentu. Dalam segi peralatan sistem kendali memiliki beberapa aspek fisis yang mengarahkan aliran energi ke suatu mesin sehingga menghasilkan suatu pengendalian yang diinginkan[2].

Pada dasarnya sistem kendali memiliki tujuan utama yaitu untuk mendapatkan nilai dari fungsi sistem kendali itu sendiri yaitu: pengukuran, pembandingan, perhitungan dan perbaikan.

Dalam sistem kendali terdapat 2 sistem kontrol yakni *open loop* dan *close loop* seperti pada gambar 2.10

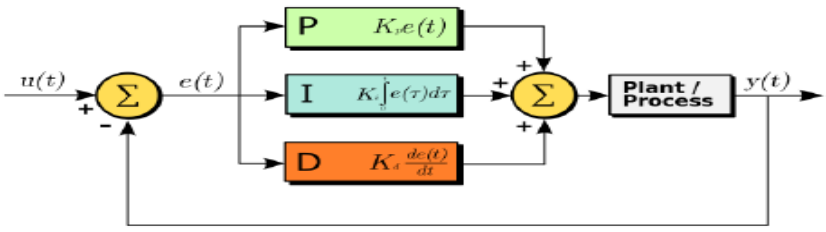


Gambar 2.10 Jenis kontrol

Sistem *open loop* memiliki keadaan dimana nilai *output* atau nilai keluaran tidak berpengaruh terhadap nilai *input* atau nilai masukkan pada kontrol[2]. Hal ini mengakibatkan nilai dari keluaran sistem hanya akan bergantung pada kalibrasi.

Sistem *close loop* memiliki keadaan dimana nilai *output* atau nilai keluaran dari sistem memiliki efek timbal balik pada nilai masukkan yang mengakibatkan pengaruh terhadap kontrol pada sistem tersebut[2]. Pada sistem *close loop* memiliki sinyal *error* yang dihasilkan dari nilai perbandingan perbedaan nilai keluaran dan nilai referensi pada sistem.

Ada jenis kontrol pada sistem *close loop* yaitu kontrol *Proportional Integral Derivative* (PID) yang telah umum digunakan pada sistem industri. Berikut adalah gambar dari blok diagram sistem kontrol PID.

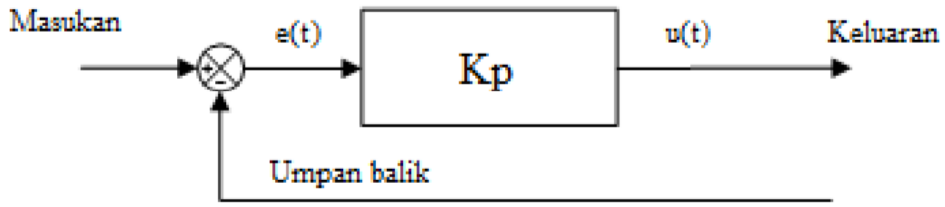


Gambar 2.11 Kontrol PID

Pada sistem kontrol PID memiliki beberapa komponen yang terdiri dari kontrol *proporsional*, kontrol *integral* dan kontrol *derivatif*. Ketiga komponen ini memiliki peran dan fungsi yang saling terkait satu dengan yang lainnya.

2.4.1 Kontrol Proportional

Pada kontrol *proportional* ini adalah pembesaran dari sinyal sinyal masukan pada sistem yang menggunakan kontrol *proportional*, dimana sinyal masukan akan dikalikan dengan nilai konstanta kontrol *proportional*.. Gambar 2.12 menunjukkan diagram blok kendali proporsional.



Gambar 2.12 Kendali Proporsional

Hubungan antara output kendali *proportional* u(t) dengan sinyal error e(t) dibuat oleh persamaan:

$$u(t) = Kp \cdot e(t) \quad \text{.....(2.4)}$$

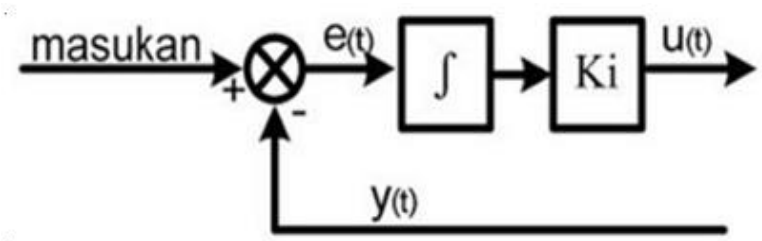
Ciri-ciri kendali *proportional* dapat dijelaskan sebagai berikut.

1. Jika nilai Kp kecil maka kendali *proportional* hanya mampu melakukan koreksi kesalahan yang kecil, sehingga akan menghasilkan respon sistem yang lambat.
2. Jika besar nilai Kp berlebihan akan menyebabkan ketidakstabilan pada sistem.

3. Seberapa besar nilai Kp tidak akan dapat menghilangkan *steady state error*.

2.4.2 Kontrol Integral

Suatu sistem yang tidak memiliki kontrol integral akan terdapat *steady-state error* atau nilai *offset. Steady-state error* adalah selisih dari nilai referensi dan nilai keluaran pada limit waktu tak hingga. Nilai offset menyebabkan sistem tidak dapat diam pada titik referensinya. Kontrol integral digunakan untuk mengeliminasi nilai offset pada sistem. Gambar 2.13 menunjukkan hubungan antara *output* kendali integral u(t) dengan sinyal *error* e(t).



Gambar 2.13 Kontrol Integral

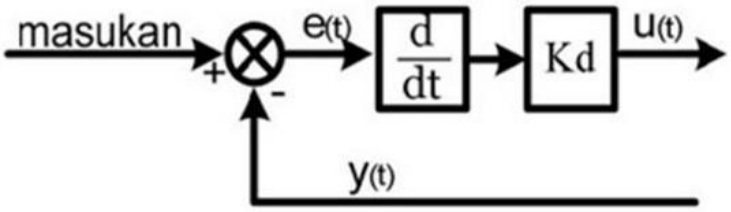
Diagram blok di atas dapat diwujudkan dalam bentuk persamaan yang menunjukkan hubungan antara output kendali integral u(t) dengan sinyal *error* e(t) sebagai berikut:

$$u(t) = Kt \int_0^t (t)dt \quad \text{.....(2.5)}$$

2.4.3 Kontrol Derivatif

Keluaran kendali derivatif memiliki sifat seperti halnya suatu operasi derivatif. Perubahan yang mendadak pada masukan pengendali,akan mengakibatkan perubahan yang sangat besar dan cepat juga pada keluaran. Pada

saat kontrol derivatif ditambahkan pada sistem yang menggunakan kontrol proporsional akan menghasilkan kontroler yang memiliki sensitifitas tinggi. Selain itu, kontrol derivatif menggunakan laju perubahan dari sinyal error sehingga kontroler yang dihasilkan dapat menghasilkan nilai koreksi yang signifikan sebelum sinyal kontroler menjadi terlalu besar Diagram blok dari kendali derivatif ditunjukkan oleh gambar 2.14.



Gambar 2.14 Kendali derivatif

Diagram blok di atas diwujudkan dalam bentuk persamaan yang menunjukkan hubungan antara *output* kendali integral u(t) dengan sinyal *error* e(t) sebagai berikut:

$$u(t) = Kd \frac{de(t)}{dt} \quad \text{.....(2.6)}$$

Dengan demikian pada metode ini terdapat tiga aksi kontrol yaitu aksi kontrol proportional dapat meningkatkan kinerja secara maksimum dalam waktu yang cepat. Aksi kontrol integral memiliki kemampuan untuk memperkecil terjadinya error dari aksi proportional. Aksi kontrol derivative memiliki keunggulan untuk memperkecil error dan mengurangi terjadinya overshoot. Aksi kontrol PID beserta fungsinya dapat dilihat pada Tabel 2.1 berikut

Tabel 2.1 Fungsi aksi kontrol pada PID

Pembesaran nilai kontrol	*Rise time*	*Overshoot*	*Settling time*	*Steady-state error*
Proportional	Menurunkan	Meningkatkan	Sedikit bertambah	Menurunkan
Integral	Menurunkan	Meningkatkan	Meningkatkan	Menurunkan
Derivatif	Sedikit berkurang	Menurunkan	Menurunkan	Menurunkan

Dalam bentuk idealnya, parameter tiap aksi kontrol yang nampak pada Tabel 2.1 diatas dapat dijelaskan sebagai berikut :

- *Rise time* : waktu yang diperlukan respondari 0 hingga mencapai 100% (set point)
- *Overshoot* : lonjakan maksimum yang dihasilkan oleh respon
- *Settling time* : waktu yang diperlukan respon untuk stabil antara 95% - 98% dari set point

2.5 Software

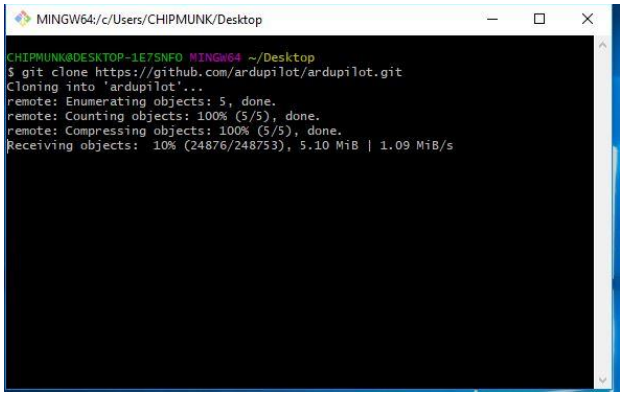
Software adalah perangkat lunak pada komputer yang berfungsi sebagai sarana media interaksi antara perangkat keras dan pengguna atau user. *Software* disebut juga sebagai penerjemah perintah-perintah yg dijalankan pengguna komputer untuk diteruskan atau diproses oleh perangkat keras. *Software* memiliki dua peran, disatu sisi berfungsi sebagai sebuah produk dan disisi lain sebagai pengontrol pembuatan sebuah produk. Sebagai produk, *Software* mengantarkan potensi perhitungan yang dibangun oleh *Software* komputer.

Software merupakan transformer informasi yang memproduksi, mengatur, memperoleh, memodifikasi, menampilkan atau memancarkan informasi, ini dapat sesederhana bit tunggal atau sekompleks sebuah simulasi multimedia. Sedangkan peran sebagai pengontrol yang dipakai untuk mengantarkan produk, *Software* berlaku sebagai dasar untuk kontrol komputer (sistem operasi), komunikasi informasi (jaringan), dan penciptaan serta kontrol dari program-program lain (peranti dan lingkungan *Software*).

Software untuk pembuatan *firmware* px4 menggunakan Git, visual studio dan px4 *toolchain*.

2.5.1 *Software* Git

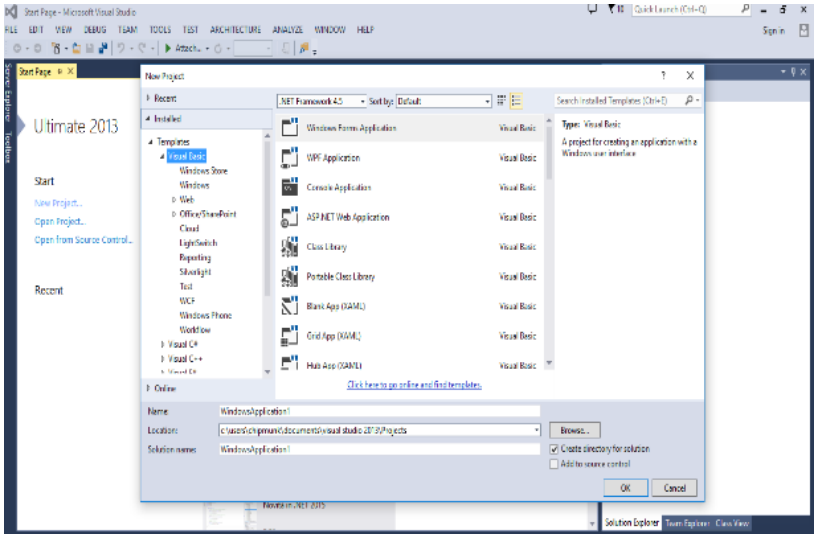
Git adalah sebuah sistem pengontrol versi pada proyek perangkat lunak yang diciptakan oleh *Linus Torvalds*. Pengontrol versi adalah mencatat setiap perubahan proyek yang dikerjakan secara individu ataupun banyak orang. Semua orang yang terlibat dalam pengerjaan proyek akan menyimpan *source code* atau pengkodean proyek di dalam database git, dimana setiap orang yang terlibat dapat mengakses setiap *source code* yang telah diubah oleh rekannya atau oleh orang lain yang ikut terlibat dalam proyek yang dikerjakan.



Gambar 2.15 *Software* Git

2.5.2 *Software* Visual Studio

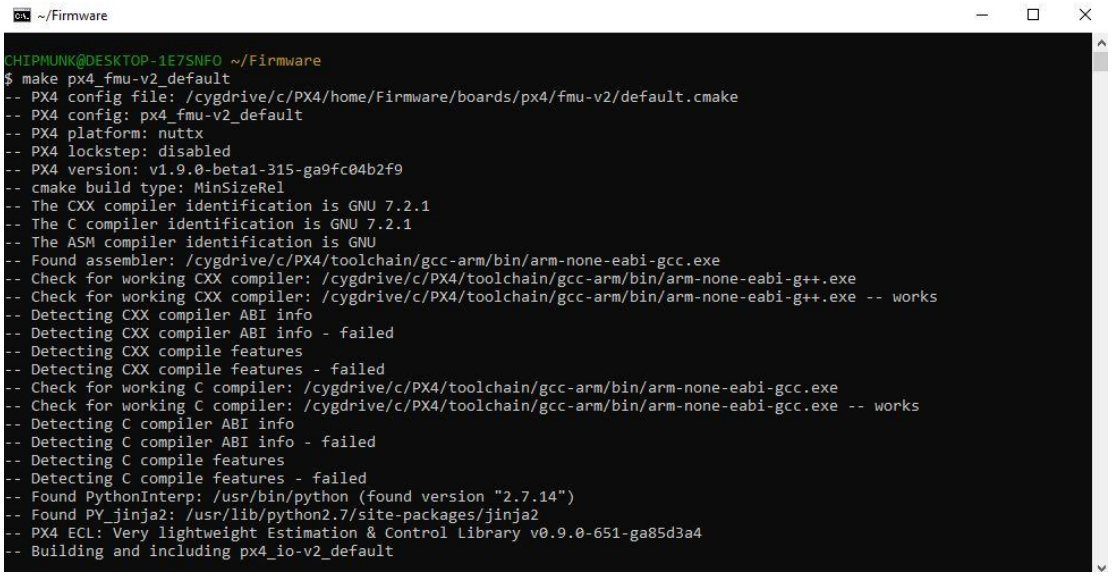
Microsoft Visual Studio merupakan *Software* yang digunakan untuk membuat file-file bertipe pemograman dan bisa digunakan untuk membuat sebuah *interface*. Microsoft Visual Studio dalam pembuatan *firmware* digunakan sebagai media untuk merancang pemograman yang akan dibuat. Microsoft Visual Studio ditunjukan seperti Gambar 2.16



Gambar 2.16 *Software* Visual Studio

2.5.3 Px4 *Toolchain*

Px4 *toolchain* adalah *Software* untuk membuat file dengan format px4 yang dikhususkan untuk *flight controller* pixhawk yang digunakan pada roket yang dibuat. Px4 *toolchain* adalah tools untuk mengenkripsikan semua data file format cpp dan file format header menjadi 1 file dengan format px4 yang akan di masukkan ke dalam *flight controller* pixhawk. Berikut tampilan px4 *toolchain*:



Gambar 2.17 *Software* Px4 *toolchain*