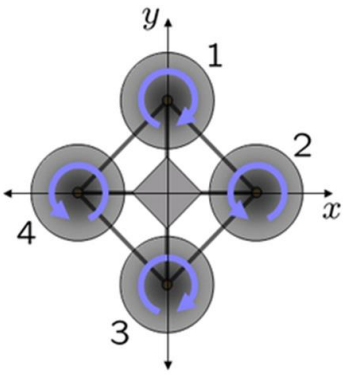
BAB II
DASAR TEORI

2.1 Quadcopter

Quadcopter adalah satu jenis wahana tanpa awak yang memiliki empat motor yang dilengkapi dengan empat *propeller* pada masing-masing motornya yang digunakan untuk terbang dan bermanuver. Masing-masing rotor (baling-baling dan motor penggeraknya) menghasilkan daya angkat dan memiliki jarak yang sama terhadap pusat massa wahana. Dengan daya angkat masing-masingrotor sebesar lebih dari sepeempat berat keseluruhan, memungkinkan *quadcopter* untuk terbang. Kecepatan *quadcopter* tergantung padaekuatan motor dan berat *quadcopter* itu sendiri.

Untuk menghindari terjadinya momen putar pada *body*, arah putaran baling-balng pada setiap rotornya berbeda seperti terlihat pada Gambar 2.1 terdapat 2 rotor yang bergerak searah jarum jam(CW) dan 2 rotor yagbergerak berlawanan arah jarum jam(CCW)



Gambar 2.1 Arah putaran baling-baling pada *quadcopter*

Sumber :dronezone.com

Konfigurasi yang paling sering digumakan adalah X-*Quadcopter*. Ketika quadcopter sedang terbang dan melayang di udara (*hovering*) kecepatan putar pada setiap rotornya adalah sama. Saat quadcopter melakua gerakan maju 2 buah baling-baling atau propeller yang berada dibelakang akan berputar lebih cepat sehingga body quadcopter akan miring ke depan,

Gaya dorong yang dihasilkan keempat propeller akan mempunyai komonen aya ke atas dan ke depan sehingga quadcopter akan tedorong ke arah depan sambil mempertaankan ketinggiannya. Gambar 2.2 adalah ilustrasi gerakan wahana yang dipengaruhi oleh kecepatan propeller. Berikut gerakan dasar pada quadcopter.

- *Throttle*

Gerak ini dapat dilakukan dengan menaikkan atau menurunkan kecepatan semua propeller dalam jumlah yang sama. Throttle digunakan untuk menaikan ketinggian quadcopter dan juga untuk menurunkan ketinggian quadcopter sehingga quadcopter dapat lepas landas dan mendarat.

- *Roll*

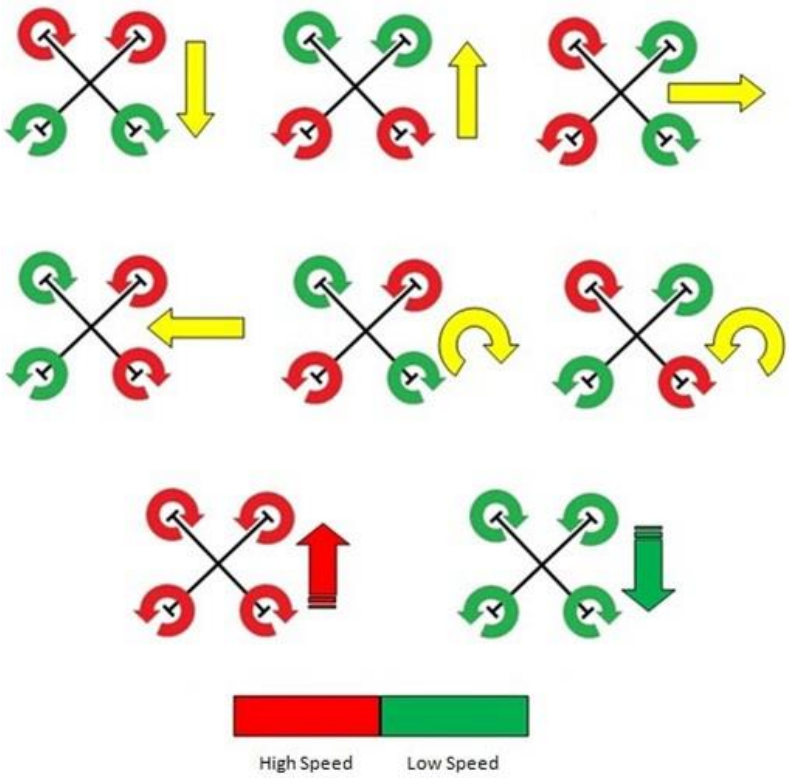
Gerak ini dapat dilakukan dengan menambah atau mengurangi kecepatan salah satu propeller yang kiri atau yang kanan. Roll membuat quadcopter menjadi lebih condong ke kanan atau ke kiri sehingga quadcopter dapat bergeser dari posisi awalnya.

- *Pitch*

Gerak ini dapat dilakukan dengan menambah atau mengurangi kecepatan salah satu propeller, yang depan atau yang belakang. Pitch adalah gerakan yang bertujuan untuk memajukan atau memundurkan posisi quadcopter dari titik asalnya atau melakukan putaran secara horizontal.

- *Yaw*

Gerak ini dapat dilakukan dengan menambah atau mengurangi propeller depan belakang dan kanan-kiri secara bersama. Yaw merupakan gerakan berputar quadcopter yang tegak lurus yang membuat quadcopter menghadap ke arah yang lain



Gambar 2.2 Gerakan dasar *quadcopter* berdasarkan kecepatan motor
Sumber : www.socialledge.com

2.1.1 Motor Brushless

Motor merupakan alat penggerak *quadcopter*. Pemilihan motor disesuaikan dengan kebutuhan, adapun yag digunakan merupakan *motor brushless/outrunner* type motor(yang berputar bagian luar). Gambar 2.3 memperlihatkan contoh motor *brushless*.

Biasanya motor yang digunakan ukuran KV=RPM/Volt dimana ukuran KV berbanding lurss dengan kecepatan putar motor (rpm). Nilai KV yang rendah menunjukkan RPM yang rendah dan torsi (daya angkat) yang besar. Namun kecepatan terbang yang rendah dikarenakan RPM yang rendah,. Namun kecepatan terbang yang tinggi menunjukkan RPM yang tinggi dan torsi (daya angkat) yang rendah.

Jumlah lilitan pada rotor sendiri berpengaruh pada besar torsi yang dihasilkan. Semakin banyak jumlah ikatannya semakin besar torsinya dan berlaku sebaliknya. Itulah penyebab mengapa motor dengan ukuran kecil cenderung mempunyai KV yang lebih besar dibandingkan dengan motor berukuran besar.

Seri pada motor brushless sendiri merupakan besar sebuah motor. Motor dengan ukuran 2212 berarti memiliki tinggi 22mm dan lebar 12mm.

Gambar 2.3 Contoh motor brushless

Sumber : www.hobbyking.com

2.1.2 Baterai *LiPo*

Gambar 2.4 Baterai LiPo

Sumber : Dokumen pribadi

LiPo adalah singkatan dari *Lithium Polimer* yaitu bahn uta yang digunakan sebagai sumber daya.baterai didalam dunia remote control. Keuntungan baterai ini adalah bobot yang ringan, kapasitas penyimpanan yang besar dan tingkat *discharge rate* energi yang tinggi.

Contoh pada Gambar 2.4 adala baterai LiPo Turnigy 4000mAh 3S 30C ”4000mAh” adalah arus yang diberikan baterai,”3S” adalah 3 *Cell* artinya baterai memberikan tegangan 11,1 Volt didapatkan dari tiap *cell*-nya 3,7 volt x 3 = 11,1 volt dan “30C” adalah *Discharge rate*.

Pada baterai 4000mAh dengan rating 30C maka baterai tersebut dapat menahan beban maksimum hingga 120 Ampere(20 x 4 Ampere = 120 Ampere). Artinya baterai dapat menerima ESC dibawah 120 Ampere. Untuk *Discharge* rate apabia sebuah baterai dengan *discharge rate* 10C baterai tersebut dapat di *discharge* 10 kali dari kapasitas baterai sebenarnya.

2.1.3 ESC

Electronic Speed Controller (ESC) adalah sebuah sensor elektronik yang berperan dalam mengontrol kecepatan motor brushless pada quadcopter. ESC bekerja dengan cara menghubungkan ESC diantara brushless motor dan mikrokontroler Arduino UNO.

Untuk menentukan ESC yang akan digunakan penting untuk mengetahui kekuatan dari ESC (*Peak Current)* dari motor. Kekuatan ESC yang digunakan seharusnyamelebihi kekuatan motor. Misalnya, dari data didapatkan kekuatan motor 15A (sesuai dengan datasheet motor) pada saat *throttle* terbuka penuh. Sebaiknya ESC yang digunakan adalah ESC yang berkekuatan 18A atau 20A. Jika dipaksakan menggunakan ESC 10A kemungkinan pada saat *throttle* dibuka penuh, ESC akan panas bahkan bisa terbakar komponennya.

Untk menghitung *max current*(amp) dari sebuah motor dapat dilakukan dengan persamaan

$$\boldsymbol{Amp = \frac{Watt}{Max\,Volt} + \left(\frac{Watt}{Max\,Volt}\ x\ 10\%\right)} \qquad \textbf{(2.1)}$$

Gambar 2.5 Salah satu jenis ESC

Sumber : www.ebay.com

2.1.4 Propeller

Propeller adalah pasangan untuk motor. Untuk Quadcoptr,Propeller yang digunakan ada dua jenis yaitu Clock Wise(CW)/searah jarum jam dan Counter Clock Wise (CCW)/Berlawanan Arah jarum. Gambar 2.7 Memperlihatkan contoh dari propeller

Ukurannya pun ada beraham biasanya dituliskan dengan format XXYY misalnya 1045,1150,1355,dll. Dimana nilai XX menunjukkan panjang propeller dan nilai YY menujukkan nilai Pitch dari propeller(dalam satuan inch) dan untuk memilih propeller juga harus disesuaikan dengan motor yang digunakan.

Gambar 2.6 Contoh dari propeller

Sumber : https://oscarliang.com/ctt/uploads/2013/06/images1.jpg

2.1.5 Frame

Frame sangat penting karena merupakan tempat untuk meletakkan komponen lain dari *Quadcopter*. Untuk dapat menentukan arah depan pada sebuah *Quadcopter,* maka biasanya pada frame diberi bola pingpong atau dengan cara memberikan warna yang berbeda terhadap propeller. Gambar 2.8 adalah contoh frame *quadcopter* bentuk X.

Gambar 2.7 Frame DJI f450

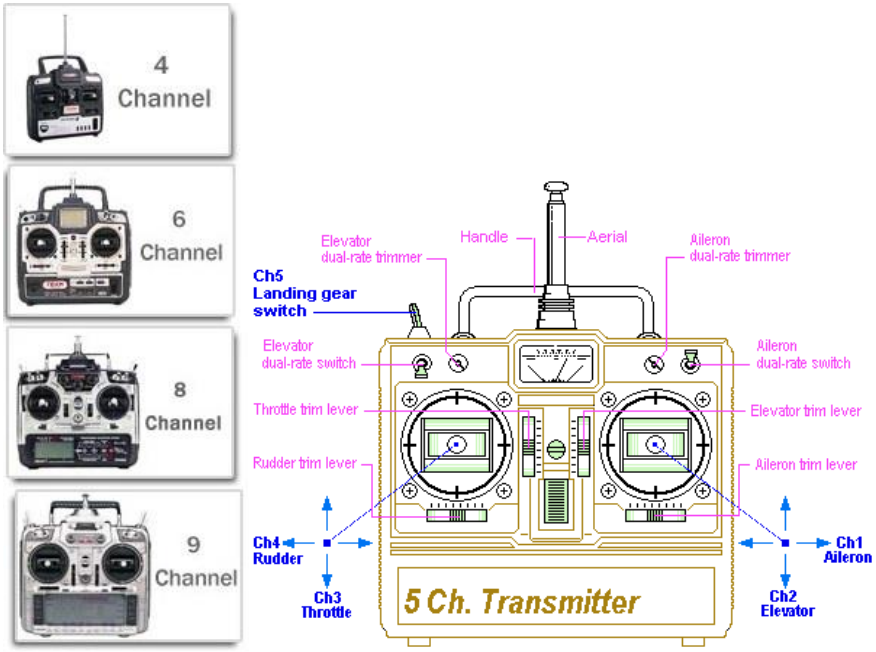
Sumber : www.quadrocopter.com

2.1.6 Remote Control

Sebelum merancang dan membangun quadcopter sebaiknya ditentukan terlebih dahulu transmitter *remote control* (RC) yang akan digunakan. RC digunakan untuk mengendalikan quadcopter secara langsung, dengan pancaran gelombang yang diterima oleh receiver yang kemudian diteruskan ke mikrokontroler. banyak hal yang perlu

dipertimbangkan dalam menentukan Rc yang akan digunakan seperti harga, jumlah *channel* yang tersedia, mode, frekuensi dan fitur lain yang dibutuhkan.

Seperti channel, setiap channel dapat melakukan satu gerakan yang dikendalikan pada quadcopter. Contohnya untuk mengontrol throttle kita membutuhkan satu channel, satu channel untuk belok kanan dan kiri, satu channel untuk maju ke depan dan mundur ke belakang, dan satu channel untuk berputar ke kiri dan ke kanan. Untuk kasus seperti itu maka RC yang dibutuhkan minimal memiliki 4 channel



Gambar 2.8 Channel pada remote control

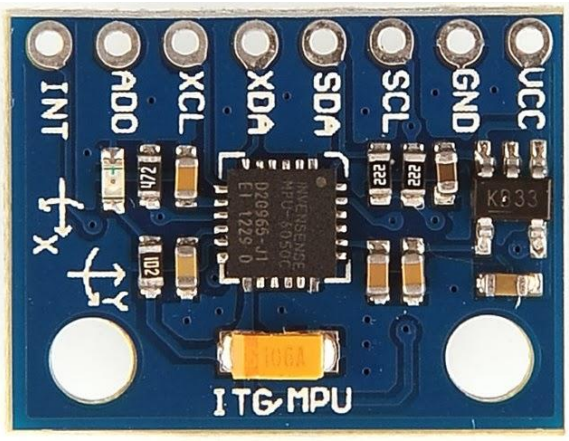
RC yang biasanya digunakan adalah yang memilki 4 atau 5 channel yang memiliki kendali untuk semua gerakan quadcopter. Pengontrolan quadcopter melalui RC juga dipengaruhi oleh frekuensi sinyal gelombang radio yang digunakan. Ada 2 tipe frekuensi yang umum terdapat di RC transmitter yaitu 72 Mhz dan 2,4 Ghz. 72 Mhz memungkinkan pengendalian untuk jarak yang jauh, tetepi RC dengan frekuensi tipe ini dapat terganggu dengan pengguna RC lain dengan tipe yang sama. 2,4 Ghz saat ini yang banyak digunakan, karena pada gelombang ini quadcopter dapat terbang dengan quadcopter yang lain tanpa terganggu pengendaliannya. Adapun RC yang digunakan adalah Turnigy 9x 2,4 Ghz satu paket dengan receivernya yaitu FS 9ch receiver.

2.1.7 6 dof IMU (Inertial Measurenment Unit) MPU6050

Modul sensor 6 dof IMU (*Inertial Measurenment Unit)* MPU6050 terdiri dari beberapa sensor yaitu 3 *axis Gyro* dan *3 axis Accelerometer* terlihat pada gambar 2.12 semua sensor berkomunikasi menggunakan protokol *I2C bus* sehingga dibutuhkan 4 jalur yaitu :

1. GND – Ground.

2. Supply voltage atau tegangan sumber +3.5 volt dan +5 volt.
3. SCL untuk I2C Clock.
4. SDA untuk I2C data

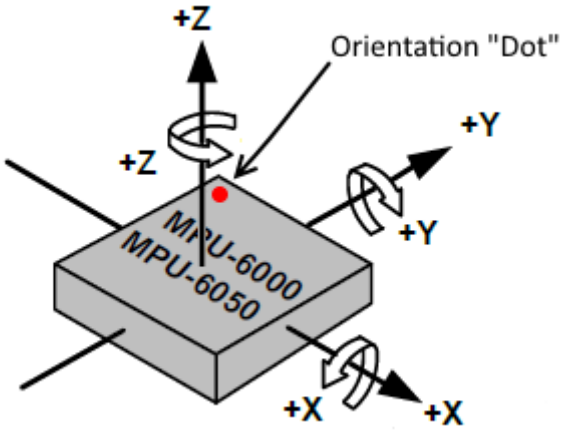


Gambar 2.9 Sensor Gyroscope MPU6050

Sumber : invensense.com

2.1.7.1 Gyro Sensor MPU6050

Gyro Sensor MPU6050 memiliki 3-axis angular rate atau rate 3 sudut sumbu XYZ, yang artinya sensor akan memberikan data jika ada perbandingan perubahan sensor terhadap sumbu X,Y,Z Gambar 2.13 Memperlihatkan 3 axis Angular rate pada sensor.

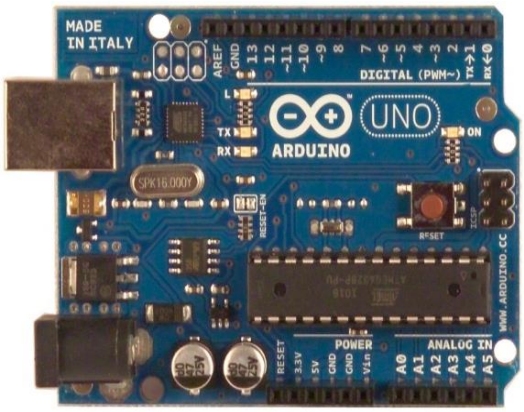


Gambar 2.10 Arah sudut rate sensor Gyro MPU 6050

Sumber: invensense.com

2.2 Arduino Uno

Arduino uno adalah *board* berbasis mikrokontroler pada ATmega328 (lihat Gambar 2.9). *Board* ini memiliki 14 digital *input/output* pin (dimana 6 pin dapat digunakan sebagai *output* PWM), 6 *input* analog, 16 MHz osilator kristal, koneksi USB, jack listrik tombol reset. Pin-pin ini berisi semua yang diperlukan untuk mendukung mikrokontroler, hanya terhubung ke komputer dengan kabel USB atau sumber tegangan bisa didapat dari adaptor AC-DC atau baterai untuk menggunakannya. Uno berbeda dengan semua board sebelumnya dalam hal koneksi USB-to-serial yaitu menggunakan fitur ATMega8 U2 yang diprogram sebagai konverter USB-to-serial berbeda dengan board sebelumnya yang menggunakan chip FTDI driver USB-to-serial.



Gambar 2.11 Arduino Uno Tampak Atas

Sumber: http://arduino.cc/en/Main/arduinoBoardUno

Nama "Uno" berarti *satu* dalam bahasa Italia, untuk menandai peluncuran Arduino 1.0 Uno dan versi 1.0 akan menjadi versi referensi dari Arduino. Uno adalah yang terbaru dalam serangkaian board USB Arduino, dan sebagai model referensi untuk *platform* Arduino, untuk perbandingan dengan versi sebelumnya.

2.2.1 Daya

Arduino Uno dapat diaktifkan melalui koneksi USB atau dengan catu daya eksternal (otomatis).Eksternal (non-USB) daya dapat berasal baik dari AC-ke adaptor-DC atau baterai. Adaptor ini dapat dihubungkan dengan menancapkan *plug jack* pusat-positif ukuran 2.1mm konektor POWER. Ujung kepala dari baterai dapat dimasukkan kedalam Gnd dan Vin pin header dari konektor POWER. Kisaran kebutuhan daya yang disarankan untuk *board* Uno adalah 7 sampai dengan 12 volt, jika diberi daya kurang dari 7 volt kemungkinan pin 5v Uno dapat beroperasi tetapi tidak stabil kemudian jika diberi daya lebih dari 12V, regulator tegangan bisa panas dan dapat merusak *board* Uno.

Pin listrik adalah sebagai berikut: N. Tegangan masukan kepada *board* Arduino ketika itu menggunakan sumber daya eksternal (sebagai pengganti dari 5 volt koneksi USB atau sumber daya lainnya).5V. Catu daya digunakan untuk daya mikrokontroler dan komponen lainnya. Sebuah pasokan 3,3 volt dihasilkan oleh regulator *on-board*.

GND. *Ground pin*.

2.2.2 Memori

ATmega328 memiliki 32 KB (dengan 0,5 KB digunakan untuk *bootloader*), 2 KB dari SRAM dan 1 KB EEPROM.

2.2.3 Input dan Output

Masing-masing dari 14 pin digital di Uno dapat digunakan sebagai *input* atau *output*, dengan menggunakan fungsi *pinMode (), digital Write (),* dan *digital Read (),* beroperasi dengan daya 5 volt. Setiap pin dapat memberikan atau menerima maksimum 40 mA dan memiliki internal *pull-up* resistor (secara default terputus) dari 20-50 kOhms. Selain itu, beberapa pin memiliki fungsi khusus:

a. *Serial*: 0 (RX) dan 1 (TX). Digunakan untuk menerima (RX) dan mengirimkan (TX) TTL data serial. Pin ini dihubungkan ke pin yang berkaitan dengan chip Serial ATmega8U2 USB-to-TTL.
b. *Eksternal menyela*: 2 dan 3. Pin ini dapat dikonfigurasi untuk memicu *interrupt* pada nilai yang rendah, dengan batasan tepi naik atau turun, atau perubahan nilai.*PWM*: 3, 5, 6, 9, 10, dan 11. Menyediakan output PWM 8-bit dengan fungsi *analogWrite ()*.
c. *SPI*: 10 (SS), 11 (Mosi), 12 (MISO), 13 (SCK). Pin ini mendukung komunikasi SPI menggunakan *SPI library*.
d. *LED*: 13. Ada *built-in* LED terhubung ke pin digital 13. Ketika pin bernilai nilai HIGH, LED on, ketika pin bernilai LOW, LED *off*.
e. Uno memiliki 6 masukan analog, berlabel A0 sampai dengan A5, yang masing-masing menyediakan 10 bit dengan resolusi (yaitu 1024 nilai yang berbeda). Selain itu, beberapa pin memiliki fungsi khusus:
f. *I2C: A4 (SDA) dan A5 (SCL)*. Dukungan I2C (TWI) komunikasi menggunakan perpustakaan *Wire*.
g. *Aref*. Tegangan referensi (0 sampai 5V saja) untuk *input* analog. Digunakan dengan fungsi *analogReference ()*.
h. *Reset*. Bawa baris ini LOW untuk me-reset mikrokontroler.

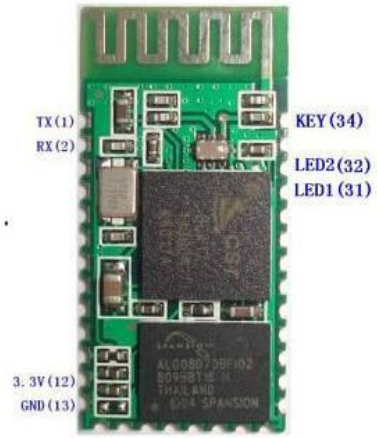
2.2.4 Komunikasi

Arduino Uno memiliki sejumlah fasilitas untuk berkomunikasi dengan komputer, Arduino lain, atau mikrokontroler lainnya. ATmega328 menyediakan UART TTL (5V) untuk komunikasi serial, yang tersedia di pin digital 0 (RX) dan 1 (TX). Sebuah ATmega8U2 sebagai saluran komunikasi serial melalui USB dan sebagai *port virtual* com untuk perangkat lunak pada komputer. *Firmware* '8 U2 menggunakan driver USB standar COM, dan tidak ada *driver* eksternal yang diperlukan. Namun, pada Windows diperlukan, sebuah file inf. Perangkat lunak Arduino terdapat monitor serial yang memungkinkan digunakan memonitor data tekstual sederhana yang akan dikirim ke atau dari board Arduino. LED RX dan TX di papan tulis akan berkedip ketika data sedang dikirim melalui chip USB-to-serial dengan koneksi USB ke komputer (tetapi tidak untuk komunikasi serial pada pin 0 dan 1).

Sebuah *Software Serial Library* memungkinkan untuk berkomunikasi secara serial pada salah satu pin digital pada board Uno's. ATmega328 juga mendukung I2C (TWI) dan komunikasi SPI. Perangkat lunak Arduino termasuk perpustakaan Kawat untuk menyederhanakan penggunaan bus I2C.

2.3 Modul Bluetooth HC-05

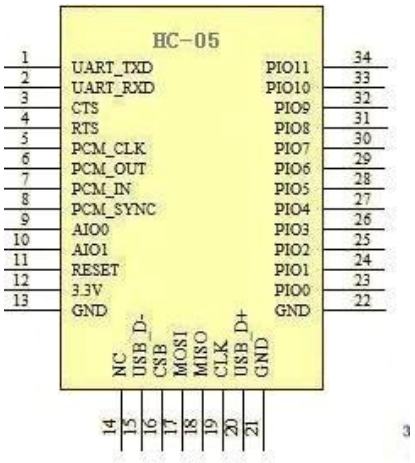
Bluetooth adalah protokol komunikasi wireless yang bekerja pada frekuensi 2.4 GHz untuk pertukaran data pada perangkat bergerak seperti PDA, laptop, HP dan lain-lain. Salah satu hasil contoh modul *bluetooth* HC-05 merupakan salah satu modul *Bluetooth* yang dapat ditemuan dipasaran harga relatif yang murah. Untuk gambar *modul bluetooth* dapat dilihat pada gamar 2.12 dibawah ini:



Gambar 2.12 *Modul Bluetooth* HC-05

(Sumber: https://mbed.org/users/edodm85/notebook/HC-05-bluetooth)

Modul *Bluetooth* HC-05 dengan *supply* tegangan sebesar 3,3 V ke pin 12 modul *bluetooth* sebagai VCC. Pin 1 pada modul *bluetooth* sebagai transmitter kemudian pin 2 pada *bluetooth* sebagai receiver.



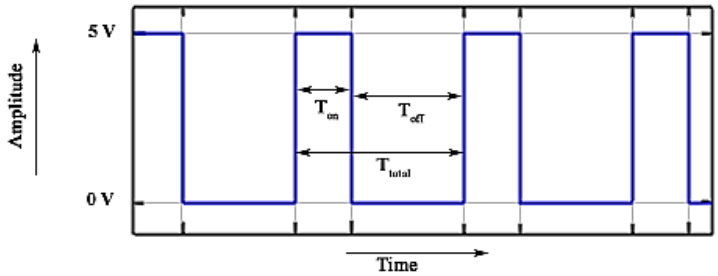
Gambar 2.13 Konfigurasi PIN HC-05

(Sumber: https://mbed.org/users/edodm85/notebook/HC-05-bluetooth)

2.4 Pulse Width Modulation

Pengaturan tegangan sumber biasanya menggunakan metode Pulse Width Modulation (PWM). Sinyal Pulse Width Modulation (PWM) adalah metode yang dapat digunakan untuk mengontrol kecepatan motor DC. Dimana kecepatan motor DC tergantung pada besarnya duty cycle yang diberikan pada motor DC tersebut.

Pada sinyal PWM, frekuensi sinyal konstan sedangkan duty cycle bervariasi dari 0%-100%. Duty cycle adalah besarnya sinyal kontrol yang diberikan pada motor. Persamaan untuk perhitungan duty cycle ditunjukkan pada Persamaan $Duty\ Cycle = \frac{T_{on}}{T} x100\%$ dengan T_on adalah periode logika tinggi, dan T adalah periode keseluruhan. Sinyal PWM secara umum dapat dilihat dalam Gambar 2.14 berikut.



Gambar 2.14 Sinyal PWM secara umum

Sumber: http://electroschematics.com

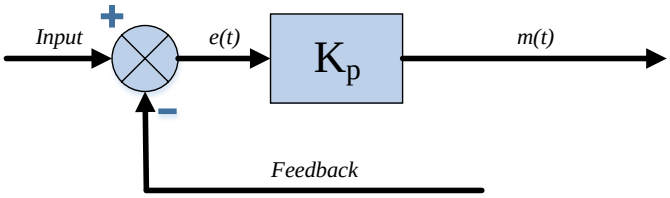
2.5 Kontroler

Kontroler dalam sebuah system mempunyai kontribusi yang sangat besar terhadap perilaku sistem. Hal ini disebabkan oleh tidak dapat di rubahnya komponen penyusu system tersebut. Artinya , karakteristik *plant* harus diterima sebagaimana adanya, perubahan perilaku system hanya dapat dilakukan melalui penambahan kontroler.

Prinsip kerja Kontroler adalah membandingkan nilai *output plant* dengan nilai *setpoint*, menentukan nilai kesalahan (nilai *error*) dan menghasilkan sinyal control untuk meminimalkan *error* (Ogata K, 1997: 197).

2.5.1 Kontrol Proporsional

Kontroler *proportional* memiliki *output* yang besarnya sebanding dengan besarnya sinyal *error*. *Output* kontroler merupakan hasil perkalian antara gain proporsional dengan sinyal *error*. Gambar 2.4 menunjukkan diagram blok kontroler *proportional* dan Persamaan 2-1 menunjukkan hubungan antara *output* kontroler dengan sinyal *error*.



Gambar 2.15 Diagram Blok Kontroler *Proportional*
(Ogata K.,1997: 201)

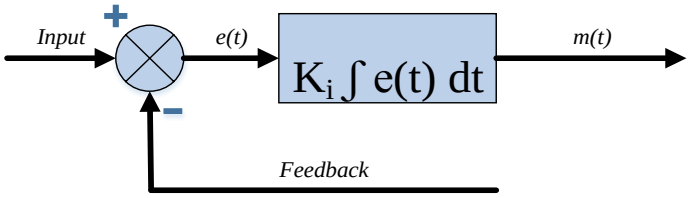
$$m(t) = Kp\ e(t) \qquad (2\text{-}2)$$

dimana:

Kp : gain proporsional
$e(t)$: sinyal *error*
$m(t)$: *output* kontroler

2.5.2 Kontrol Integral

Kontroler *integral* memiliki karakteristik seperti sebuah operasi *integral*. *Output* kontroler merupakan hasil perkalian gain integral dengan penjumlahan terus menerus dari perubahan sinyal *error*. Gambar 2.5 menunjukkan diagram blok kontroler *integral* dan Persamaan 2-2 menunjukkan hubungan antara *output* kontroler dengan sinyal *error*.



Gambar 2.16 Diagram Blok Kontroler *Integral*

(Ogata K., 1997: 202)

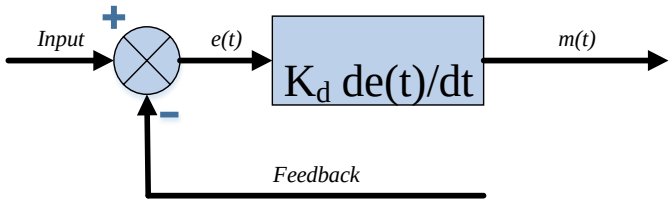
$$m(t) = Ki \int e(t)\, dt \qquad (2\text{-}3)$$

dimana:

Ki : gain integral
$e(t)$: sinyal *error*
$m(t)$: *output* kontroler

2.5.3 Kontrol Derivatif

Kontroler *derivative* memiliki sifat seperti suatu operasi *derivative* (turunan). Perubahan yang mendadak pada masukkan kontroler mengakibatkan perubahan yang sangat besar dan cepat. Kontroler ini tidak akan menghasilkan *output* pada saat sinyal *error* konstan sehingga tidak akan mempengaruhi keadaan mantap (*steady state*). Sehingga Kontroler ini digunakan untuk memperbaiki atau mempercepat respon *transient*. Gambar 2.6 menunjukkan diagram blok kontroler *derivative* dan Persamaan 2-3 menunjukkan hubungan antara *output* kontroler dengan sinyal *error*.



Gambar 2.17 Diagram Blok kontroler *Derivative*
(Ogata K., 1997: 203)

$$m(t) = Kd \frac{de(t)}{dt} \qquad (2\text{-}4)$$

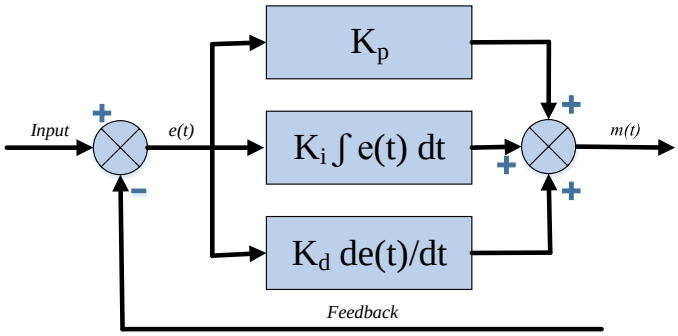
dimana:

Kd : gain *derivative*
$e(t)$: sinyal *error*
$m(t)$: *output* kontroler

Kontroler ini digunakan untuk memperbaiki atau mempercepat respon *transient*.

2.5.4 Kontrol Proporsional Integral Derivatif

Gabungan aksi kontrol *proportional*, *integral*, dan *derivative* yang terlihat dalam Gambar 2.18 mempunyai keunggulan dapat saling menutupi kekurangan dan kelebihan dari masing-masing kontroler. Persamaan kontroler PID ini dapat dinyatakan sebagai berikut (Persamaan 2-5):



Gambar 2.18 Diagram Blok Kontroler PID
(Ogata K., 1997: 204)

$$m(t) = Kp\,.e(t) + Ki.\int e(t)dt + Kd\frac{de(t)}{dt} \qquad (2\text{-}5)$$

dimana:

Kp	: gain *proportional*
Ki	: gain *integral*
Kd	: gain *derivative*
$e(t)$	: sinyal *error*
m(t)	: output kontroler

2.6 Respon Transien

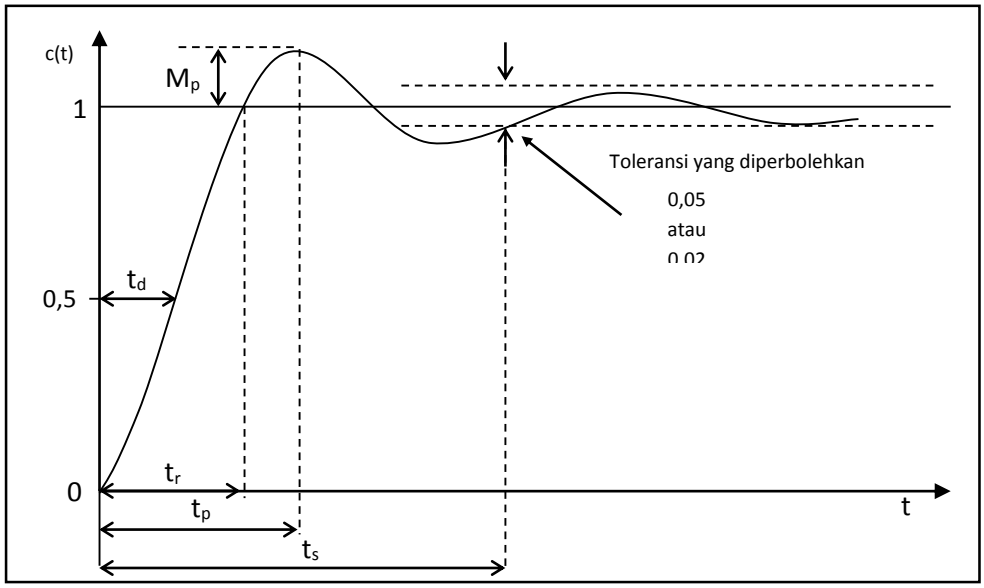
Karakteristik respons transien suatu sistem terhadap masukan *unit step* bergantung pada syarat awal. Untuk dapat membandingkan respons transien berbagai macam sistem, hal yang biasa dilakukan adalah menggunakan syarat awal standar bahwa sistem mula-mula dalam keadaan diam sehingga keluaran dan semua turunan terhadap waktunya pada awal respons akan sama dengan nol. Selanjutnya karakteristik respons secara mudah dapat dibandingkan.

Respons transien suatu sistem kontrol seringkali menunjukkan suatu osilasi teredam sebelum mencapai keadaan mantapnya. Dalam menentukan karakteristik respons transien suatu sistem kontrol terhadap masukan *unit step*, dicari parameter-parameter (performansi sistem) sebagai berikut:

1. Waktu penetapan (*Settling Time*), t_s : merupakan waktu yang diperlukan kurva respons untuk mencapai dan menetap dalam daerah di sekitar nilai akhir yang ukurannya ditentukan dengan prosentase mutlak dari nilai akhir (biasanya 5 % atau 2 %). Waktu penetapan ini dikaitkan dengan konstanta waktu terbesar dari sistem kontrol. Kriteria prosentase kesalahan yang akan digunakan ditentukan dari sasaran disain.
2. Waktu tunda (*Delay Time*), t_d : merupakan waktu yang diperlukan respons untuk mencapai setengah nilai akhir pada saat lonjakan yang pertama kali.

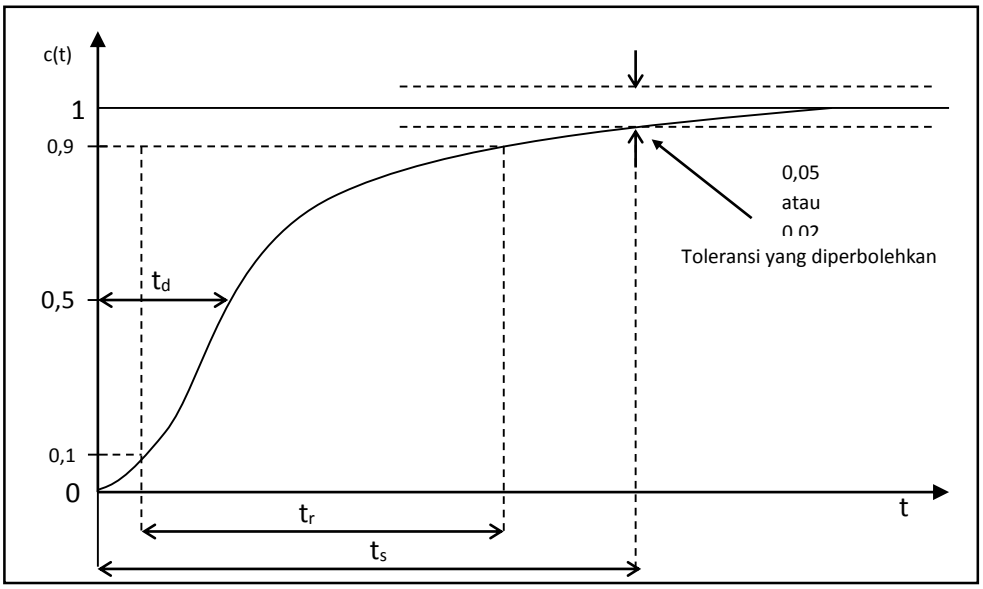
3. Waktu naik (*Rise Time*), t_r : merupakan waktu yang diperlukan respons untuk naik dari 10 sampai 90 %, 5 sampai 95 % atau 0 sampai 100 % dari nilai akhir. Untuk sistem orde dua redaman kurang (*Underdamped*) digunakan waktu naik 0-100 %, dan untuk sistem redaman lebih (*Overdamped*) digunakan waktu naik 10-90 %.
4. Waktu puncak (*Peak Time*), t_p : merupakan waktu yang diperlukan respons untuk mencapai puncak lewatan (lonjakan maksimal) yang pertama kali.
5. Lewatan maksimum (*Maximum Overshoot*), M_p : merupakan nilai puncak maksimum kurva respons yang diukur dari satu. Jika nilai keadaan mantap respons tidak sama dengan satu, maka dapat digunakan persen lewatan maksimum.

Gambar 2.19 menunjukkan kurva respons sistem orde dua kurang teredam (*underdamped)* dengan masukan *unit step.*



Gambar 2.19 Kurva Respons Sistem Orde Dua *Underdamped* dengan Masukan *Unit Step.*

Gambar 2.19 menunjukkan kurva respons sistem orde dua terlalu teredam (*overdamped)* dengan masukan *unit step*.

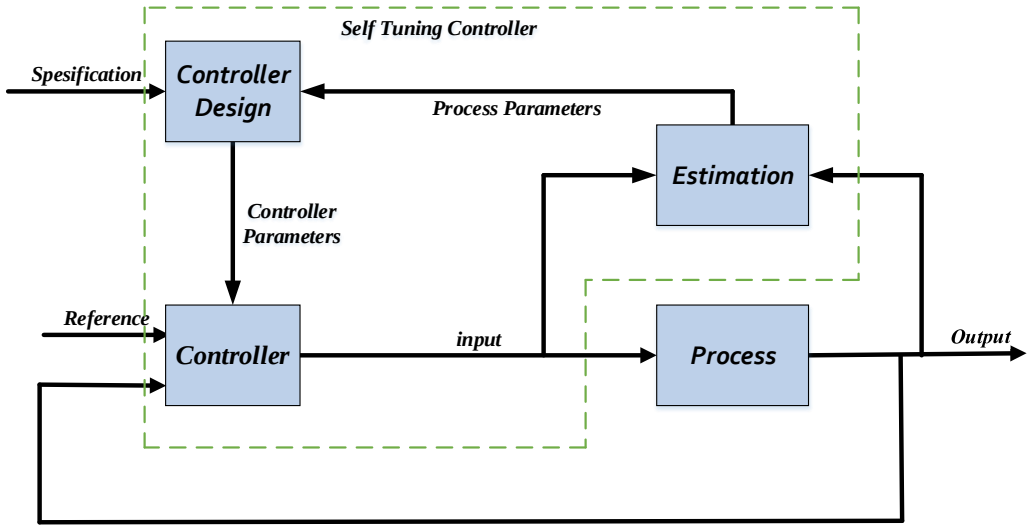


Gambar 2.20 Kurva Respons Sistem Orde Dua *Overdamped* dengan Masukan *Unit Step*.

2.7 Self-Tuning Controller (STC)

Kontroler adaptif adalah kontroler yang dapat melakukan *adjustable* parameter dan mekanisme untuk mengatur parameter atau dalam pengertian umumnya berarti mengubah tingkah laku atau karakteristik untuk menyesuaikan diri terhadap keadaan yang baru atau tidak diketahui (Oktafiani, 2013: 2). Sebuah pendekatan kontrol adaptif yang berdasarkan *recursive estimation* (estimasi rekursif) dari karakteristik sistem dan gangguan serta dapat mengubah-ubah estimasi, sehingga memungkinkan untuk di amati. Dengan menggunakan pendekatan tersebut, kita dapat menggunakannya untuk merancang kontroler yang optimal. Dimana kontroler mengidentifikasi proses yang tidak diketahui dan kemudian mensintesis sinyal kontrol (Adaptif kontrol dengan rekursif identifikasi) yang disebut dengan *Self-Tuning Controller* (STC) (Bobal, 2005: 13). Dalam skema ini, parameter diestimasi dan hasil dari estimasi parameter tersebut, dihitung setelah itu digunakan untuk mendapatkan parameter kontrol yang baru.

Pada STC terdapat dua loop, yaitu *inner loop* dan *outer loop*. *Inner loop* terdiri dari *plant* (proses) dan kontroler dengan umpan balik biasa, sedangkan pada *outer loop* berfungsi untuk mengubah parameter dari kontroler, dimana terdiri estimasi parameter rekursif dan perhitungan desain (Astrom, 2008: 21). Diagram Blok *Self* Tuning *Controller* dapat dilihat pada Gambar 2.21.



Gambar 2.21 Diagram Blok *Self-Tuning Controller*
(Astrom, 2008: 91)

Parameter dari model di identifikasi secara *on-line* pada blok "*Estimation*" yang memberikan sebuah estimasi parameter *plant*. Pada blok "*Controller Design*" berisi perhitungan yang dibutuhkan untuk desain dari kontroler dengan metode yang telah di tentukan dan beberapa desain parameter yang telah di pilih. Setelah perhitungan selesai, hasil parameter dari blok *controller design* akan di implementasikan pada blok "*Controller*" (Astrom, 2008: 90).

2.7 Dahlin PID *Controller* (Bobal, 2005:54-55,93-94)\

Fungsi alih dari kontroler Dahlin PID ini:

$$G(z) = \frac{b_1 z^{-1}}{1 + a_1 z^{-1} + a_2 z^{-2}} \qquad (2\text{-}6)$$

Algoritma dari kontroler ini, *output* kontroler dihitung menggunakan bentuk jumlahan dari persamaan diskretisasi dengan metode *forward rectangular* :

$$u(t) = K_P \left[e(t) + \frac{1}{T_I} \int_0^t e(t)dt + T_D \frac{de(t)}{dt} \right] \qquad (2\text{-}5)$$

Persamaan PID diatas dalam bentuk umum dalam fungsi waktu. Dimana $e(t) = w(t) - y(t)$ yang merupakan nilai selisih antara nilai yang diinginkan dengan nilai hasil proses atau nilai sebenarnya. $u(t)$ merupakan nilai keluaran dari kontroler. $y(t)$ merupakan nilai keluaran dari *plant* (proses) atau nilai sebenarnya. $w(t)$ merupakan sinyal referensi atau setpoint (nilai yang diinginkan). Parameter dari kontroler PID, yaitu Kp merupakan *gain* (penguatan) proporsional, T_I adalah konstanta waktu integral, dan T_D merupakan konstanta waktu diferensial.

Menggunakan transformasi *Laplace* persamaan 2-24 diubah ke dalam bentuk:

$$U(s) = K_P\left[1 + \frac{1}{T_I s} + T_D s\right]E(s) \qquad (2\text{-}6)$$

Dimana s merepresentasikan operator transformasi *laplace*. Dari persamaan 2-25 maka kita dapat menentukan fungsi alih dari kontroler PID :

$$G_R(s) = \frac{U(s)}{E(s)} = K_P\left[1 + \frac{1}{T_I s} + T_D s\right] \qquad (2\text{-}7)$$

Untuk mendapatkan persamaan dalam fungsi Z (digital) maka persamaan kontroler PID dalam fungsi waktu kita diskritisasi komponen integral dan diferensial pada persamaan 2-24 ketika periode sampling T_0 kecil dan *noise* (derau) dari sinyal *output* proses telah di filter, maka algoritma yang paling sederhana diperoleh dengan mengganti derivatif dengan diferensial dari orde pertama (*two-point, backward difference*)

$$\frac{de}{dt} \approx \frac{e(k) - e(k-1)}{T_0} = \frac{\Delta e(k)}{T_0} \qquad (2\text{-}8)$$

Dimana *e(k)* adalah nilai kesalahan pada saat *k* sampling (misal pada waktu t = kT_0).

Pendekatan termudah dari integral adalah dengan penjumlahan sederhana sehingga kita mendekati fungsi waktu kontinyu dengan sampling periode T_0 dari fungsi konstan (*step function* atau fungsi satuan). Dengan menggunakan *forward rectangular method* (FRM):

$$\int_0^t e(t)dt \approx T_0 \sum_{i=1}^{k} e(i-1) \qquad (2\text{-}9)$$

Sehingga persamaan diskrit (digital) untuk kontroler PID memiliki bentuk:

$$u(k) = K_P\left\{e(k) + \frac{T_0}{T_I}\sum_{i=1}^{k} e(i-1) + \frac{T_D}{T_0}\left[e(k) - e(k-1)\right]\right\} \qquad (2\text{-}10)$$

Dalam aplikasi pada industri persamaan (2-10) tidak praktis untuk digunakan dalam pengontrolan menggunakan komputer karena memerlukan memori untuk menyimpan semua nilai-nilai *error* sebelumnya. Oleh karena itu, nilai *output* kontroler $u(k)$ adalah hasil dari nilai *output* kontroler sebelumnya $u(k-1)$ ditambah dengan nilai selisih koreksi $\Delta u(k)$. Maka persamaan tersebut dapat dituliskan sebagai berikut.

$$u(k) = \Delta u(k) + u(k-1) \qquad (2\text{-}11)$$

Dimana,

$$\Delta u(k) = K_P\left\{e(k) - e(k-1) + \frac{T_0}{T_I}e(k) + \frac{T_D}{T_0}\left[e(k) - 2e(k-1) + e(k-2)\right]\right\} \qquad (2\text{-}12)$$

Sehingga persamaan (2-10) menjadi seperti yang dituliskan kedalam persamaan (2-13):

$$u(k) = K_P\left\{e(k) - e(k-1) + \frac{T_0}{T_I}e(k) + \frac{T_D}{T_0}\left[e(k) - 2e(k-1) + e(k-2)\right]\right\} + u(k-1)$$

Kontroler ini menggunakan vektor estimasi parameter dalam bentuk

$$\hat{\theta}^T(k) = \left[\hat{a}_1, \hat{a}_2, \hat{b}_1\right] \tag{2-14}$$

Dan karena $\hat{b}_2 = 0$, regresi vektornya adalah

$$\theta^T(k-1) = \left[-y(k-1), -y(k-2), u(k-1)\right] \tag{2-15}$$

Dari hubungan berikut di peroleh persamaan untuk parameter kontroler sebagai berikut:

$$K_P = -\frac{(\hat{a}_1 + 2\hat{a}_2)Q}{\hat{b}_1} \tag{2-16}$$

$$T_I = -\frac{T_0}{\frac{1}{\hat{a}_1 + 2\hat{a}_2} + 1 + \frac{T_D}{T_0}} \tag{2-17}$$

$$T_D = \frac{T_0 \hat{a}_2 Q}{K_P \hat{b}_1} \tag{2-18}$$

$$Q = 1 - e^{-\frac{T_0}{B}} \tag{2-19}$$

Dimana *B* di kenal sebagai *adjustment factor* (faktor pengatur) dimana menggambarkan konstanta dalam domain waktu dari fungsi alih sesuai dengan perubahan yang terjadi pada *output plant* (proses) pada *closed loop*. Semakin kecil nilai *B* maka respon akan semakin cepat.

Untuk menghindari osilasi pada *output plant* (proses) maka disarankan untuk memilih estimasi parameter kontrol awal menggunakan persamaan berikut. Pada dasarnya persamaan ini merupakan *invers* dari persamaan (2-16), (2-17) dan (2-18).

$$C = 1 + \frac{T_0}{T_I} + \frac{T_D}{T_0} \tag{2-20}$$

$$\hat{a}_1 = -\frac{1 + 2\frac{T_D}{T_0}}{C} \tag{2-21}$$

$$\hat{a}_2 = \frac{T_D}{T_0 C} \tag{2-22}$$

$$\hat{b}_1 = \frac{Q}{K_P C} \tag{2-23}$$

Oleh karena itu, ketika memulai merancang algoritma kita harus menghindari memilih nilai nol untuk parameter $\hat{b}_1$