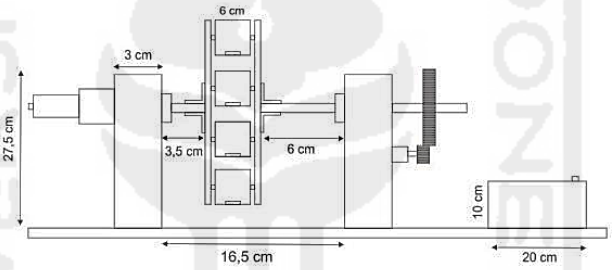
BAB 2
TINJAUAN PUSTAKA

2.1 Kajian Pustaka

Beberapa penelitian tentang pengendalian motor DC sebelumnya telah banyak dilakukan oleh beberapa mahasiswa dari berbagai perguruan tinggi. Pada penelitian (Putra, 2013) dijelaskan tentang bagaimana merancang sistem *rotary parking* dengan sistem elektriknya menggunakan modul Arduino Mega 2560 dengan bahasa pemrograman Arduino. Pada penelitian tersebut dilakukan penelitian dengan mengatur posisi motor DC dan posisi garasi berdasarkan kontrol PID agar sesuai dengan posisi tujuan. Bentuk dari model dari *rotary parking* dapat dilihat pada Gambar 2-1.

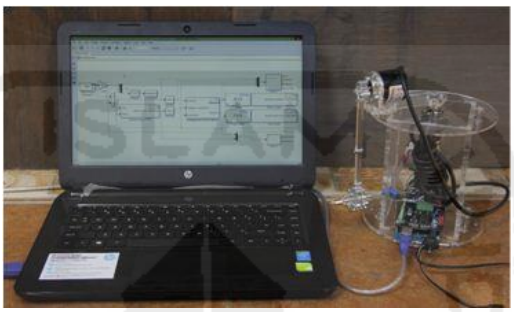


Gambar 2 - 1 Skema *rotary parking* tampak depan (Putra, 2013).

Berdasarkan penelitian yang dilakukan oleh Putra, diperoleh parameter kontroler PID dengan menggunakan metode kedua dari teori Ziegler-Nichols yaitu K_p = 4.8 , K_i = 8,9 , K_d = 0,64 dan toleransi *error* sebesar 2% - 10%. Hasil pengujian terhadap aplikasi kontroler PID ini menunjukkan bahwa respons sistem untuk tiap sudut mempunyai *error steady state* (e_{ss}) sebesar 0-0,146%. Sedangkan t_s sudut 60°=1,169 detik (s), sudut 120°= 2,2415 detik (s), sudut 180°= 3,507 detik (s), sudut 240°= 4,342 detik (s), dan sudut 300°= 5,331 detik (s). Hasil pengujian ini menunjukkan bahwa kontroler PID menghasilkan respons sesuai yang direncanakan dan dapat diaplikasikan pada *rotary parking* untuk mencapai posisi yang dituju.

Penelitian lain (Faekar, 2015) melakukan perancangan dan implementasi sistem kendali posisi untuk mereduksi ayunan pada simulator *rotary crane*

menggunakan Arduino berbasis PC. Pada penelitian tersebut dijelaskan bagaimana membuat simulator *crane* dan merancang sistem kendali untuk mereduksi ayunan yang diijinkan sebesar 5˚ saat simulator *tower crane* beroperasi dengan menggunakan kontroler PID dengan metode Ziegler-Nichols 2, dan *trial and error*. Pada penelitian juga digunakan *software* Matlab Simulink dan Arduino Mega 2560 R3 sebagai perangkat pemrosesnya. Pada Gambar 2-2 berikut adalah bentuk dari perangkat keras simulator yang dibuat.



Gambar 2 - 2 Simulator *tower crane* (Faekar, 2015).

Pada penelitian tersebut dihasilkan perhitungan kontroler PID pada setiap metode. Metode Ziegler-Nichols 2 didapat nilai K_p=56,4, K_i=829,41, dan K_d=0,095 dengan *percent overshoot* tanpa kendali ayun sekitar 15,11 %, *settling time* 3,99 detik, ayunan maksimum yang terjadi 15,75˚. Sedangkan untuk *percent overshoot* dengan kendali ayun sekitar 34,31˚, *settling time* 8,94 detik, ayunan maksimum yang terjadi 2,92˚ . Pada metode *trial and error* nilai K_p=30, K_i=100, K_d=5, *percent overshoot* tanpa kendali ayun 14,72 %, *settling time* 4,82 detik, ayunan maksimum 18,45˚, sedangkan untuk *percent overshoot* dengan kendali ayun 13,06 %, *settling time* 18,95 detik, ayunan maksimum 3,60˚. Kedua metode ini yang mampu mereduksi ayunan beban tidak lebih dari 5˚ dimana metode Ziegler-Nichols 2 mampu mengurangi ayunan beban hingga 2,92˚ sedangkan dengan metode *trial and error* yang hanya mampu mencapai 3,60˚.

Penelitian lain yang terkait dengan sistem kendali *crane* juga sudah dilakukan untuk menghilangkan ayunan yang berlebih saat *crane* beroperasi. Teknik *feed forward input shaping* digunakan untuk mereduksi sisa ayunan (Singhose, Porter, & Seering, 1997). Teknik *input shaping* juga digunakan pada *automatic overhead crane* untuk menghilangkan ayunan yang terjadi (Garrido, Abderrahim, Gimenez, & Balaguer, 2008). Sisa ayunan yang terjadi dihilangkan

dengan cara mentransformasi *step input* menjadi sinyal masukan yang memiliki kurva akselerasi yang terbentuk dari serangkain *impuls* yang muncul.

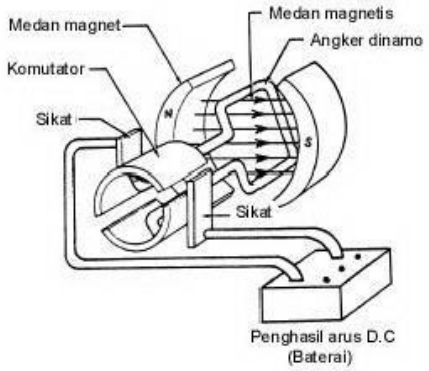
Berdasarkan penelitian sebelumnya, maka untuk penelitian selanjutnya yang akan dilakukan menggunakan motor DC mabuchi rs-385ph *with encoder* 448 AB *phase* sebagai aktuator, Autonics *incremental rotary encoder* sebagai sensor, Adafruit motor *shield* v1 sebagai driver. Kemudian untuk sistem kendali dan kontroler yang digunakan adalah kendali posisi linier simulator troli guna mereduksi posisi ayunan. Metode kontroler PID yang digunakan ada 2, yaitu metode Ziegler-Nichols 2 dan metode *trial and error*. Perangkat lunak yang digunakan adalah Arduino IDE, Matlab Simulink R2014a, dan Arduino Mega 2560 R3.

2.2 Dasar Teori

2.2.1 Motor DC

Motor listrik merupakan perangkat elektromagnetis yang mengubah energi listrik menjadi energi mekanik. Energi mekanik digunakan untuk, misalnya memutar baling-baling kipas angin, memutar *impeller* pompa, dan lain sebagainya. Motor listrik digunakan pada rumah tangga (contoh: kipas angin, bor listik, dan pompa air) dan pada dunia industri.

Motor DC memerlukan suplai tegangan yang searah pada kumparan medan untuk diubah menjadi energi mekanik. Kumparan pada motor DC disebut *stator* (bagian yang tidak berputar) dan kumparan jangkar disebut rotor (bagian yang berputar). Prinsip kerja dari arus searah adalah membalik phasa tegangan dari gelombang yang mempunyai nilai positif dengan menggunakan komutator, dengan demikian arus yang berbalik arah dengan kumparan jangkar yang berputar dalam medan magnet. Bentuk motor paling sederhana memiliki kumparan satu lilitan yang bisa berputar bebas di antara kutub-kutub magnet permanen. Gambar 2-3 menunjukan ilustrasi motor DC sederhana.

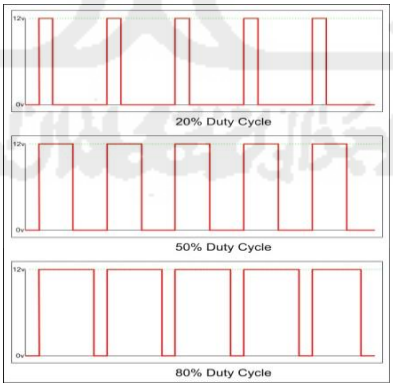


Gambar 2 - 3 Motor DC sederhana (biondiocta, 2012).

2.2.2 PWM (*Pulse Widht Modulation*)

PWM (*Pulse width Modulation*), adalah sebuah metode untuk pengaturan kecepatan perputaran, dalam hal ini adalah motor DC. PWM dapat dihasilkan oleh empat metode, yaitu analog, digital, IC diskrit, dan mikrokontroler.

Metode PWM dikerjakan oleh mikrokontroler. Metode ini akan mengatur lebar atau sempitnya periode pulsa aktif yang dikirimkan oleh mikrokontroler ke *driver* motor. Pada pengaturan kecepatan motor, nilai PWM mulai dari 0-255. Secara analog besaran PWM dihitung dalam persentase, nilai ini didapat dari perbandingan: T *high* / (T *high* + T *low*)*100%. Dimana T adalah periode atau waktu tempuh untuk sebuah pulsa, yang terbagi menjadi bagian puncak positif (T *high*) dan puncak negatif (T *low*).



Gambar 2 - 4 Ilustrasi PWM.

Semakin rapat periode antar pulsa, maka frekuensi yang dihasilkan akan semakin tinggi, ini berarti kecepatan akan bertambah. Semakin lebar jarak antar pulsa, maka frekuensi semakin rendah berarti kecepatan berkurang atau menurun.

2.2.3 Catu Daya

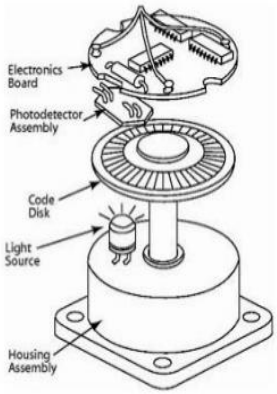
Catu daya adalah sebuah peralatan penyedia tegangan atau sumber daya untuk peralatan elektronika dengan prinsip mengubah tegangan listrik yang tersedia dari jaringan distribusi transmisi ke level yang diinginkan sehingga berimplikasi pada pengubahan daya listrik.

Dalam sistem pengubah daya, terdapat empat jenis proses yang telah dikenal yaitu sistem pengubah daya AC ke DC, DC ke DC, DC ke AC, dan AC ke AC. Masing-masing sistem pengubahan memiliki keunikan aplikasi tersendiri, tetapi ada dua yang implementasinya kemudian berkembang pesat dan luas yaitu sistem pengubahan AC ke DC (DC *power supply*) dan DC ke DC (DC-DC *converter*).

2.2.4 *Rotary Encoder*

Rotary inremental encoder adalah alat elektromekanik yang berfungsi membaca dan memonitor gerakan atau posisi benda yang berputar. Pada *rotary incremental encoder* biasanya terdapat sensor optik yaitu *optocoupler* yang berfungsi membaca jumlah pulsa yang dapat diartikan menjadi gerakan, posisi maupun arah yang kemudian diolah menjadi informasi kode digital dan dikirimkan atau diteruskan ke rangkaian kendali. *Rotary incremental encoder* tersusun dari suatu piringan tipis yang memiliki lubang-lubang pada bagian lingkaran piringan. LED ditempatkan di salah satu sisi piringan yang kemudian cahaya akan menuju ke piringan, disisi lainnya terdapat *photo-transistor* sehingga *photo-transitor* ini dapat mendeteksi cahaya yang ada dipancarkan oleh LED.

Piringan tipis kemudian dihubungkan dengan poros motor, atau perangkat berputar lainnya yang ingin kita ketahui posisinya, sehingga ketika motor berputar piringan juga akan ikut berputar. Apabila posisi piringan mengakibatkan cahaya dari LED dapat mencapai *photo-transistor* melalui lubang-lubang yang ada, maka *photo-transistor* akan mengalami saturasi dan akan menghasilkan suatu pulsa gelombang persegi. Gambar 2-5 menunjukkan bagan skematik sederhana dari *rotary incremental encoder*. Semakin banyak deretan pulsa yang dihasilkan pada satu putaran menentukan akurasi *rotary incremental encoder* tersebut.



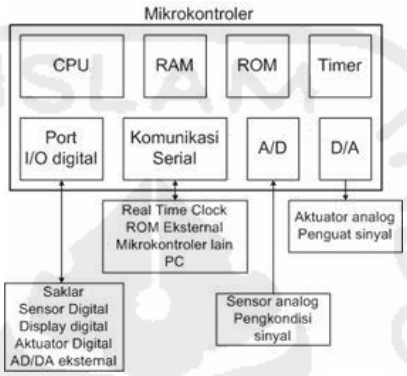
Gambar 2 - 5 *Incremental rotary encoder* (Rotary Encoders Information, 2016).

2.2.5 Mikrokontroler

Mikrokontroler adalah suatu chip berupa IC (*Integrated Circuit*) yang dapat menerima sinyal *input*, mengolahnya dan memberikan sinyal *output* sesuai dengan program yang diisikan ke dalamnya. Sinyal *input* mikrokontroler berasal dari sensor yang merupakan informasi dari lingkungan sedangkan sinyal *output* ditujukan kepada aktuator yang dapat memberikan efek ke lingkungan. Jadi secara sederhana mikrokontroler dapat diibaratkan sebagai otak dari suatu perangkat/produk yang mampu berinteraksi dengan lingkungan sekitarnya.

Mikrokontroler pada dasarnya adalah komputer dalam satu chip, yang didalamnya terdapat perangkat mikroprosesor, memori, jalur *input/output, (I/O)* dan perangkat pelengkap lainnya. Kecepatan mikrokontroler pada umumnya berkisar antara 1 - 16 Mhz, begitu juga kapasitas RAM dan ROM hanya berkisar pada orde byte/kbyte. Mikrokonktroler juga digunakan dalam produk dan alat yang dikendalikan secara automatis, seperti sistem kontrol mesin, *remote controls*, mesin kantor, peralatan rumah tangga, alat berat, dan mainan. Bentuk dari mikrokontroler dapat dilihat pada gambar 2-6 dibawah ini.

Gambar 2 - 6 Bentuk dari mikrokontroler (Immersa Lab, 2016)



Gambar 2 - 7 Komponen dari suatu mikrokontroler (Budiman, 2013).

Pada Gambar 2-7 diatas menunjukan komponen-komponen dari suatu mikrokontroler yang mempunyai fasilitas lengkap beserta peranti eksternal yang biasanya dihubungkan ke/dari mikrokontroler. Tidak semua mikrokontroler mempunyai semua komponen tersebut, misalnya converter A/D dan D/A hanya terdapat pada beberapa jenis mikrokontroler tertentu.

2.2.6 Arduino

Untuk memahami Arduino, terlebih dahulu kita harus memahami terlebih dahulu apa yang dimaksud dengan *physical computing*. *Physical computing* adalah membuat sebuah sistem atau perangkat fisik dengan menggunakan *software* dan *hardware* yang sifatnya interaktif yaitu dapat menerima rangsangan dari lingkungan dan merespons balik. *Physical computing* adalah sebuah konsep untuk memahami hubungan yang manusiawi antara lingkungan yang sifat alaminya adalah analog dengan dunia digital. Pada prakteknya konsep ini diaplikasikan dalam desain-desain alat yang menggunakan sensor dan mikrokontroler untuk menerjemahkan *input* analog ke dalam sistem *software*

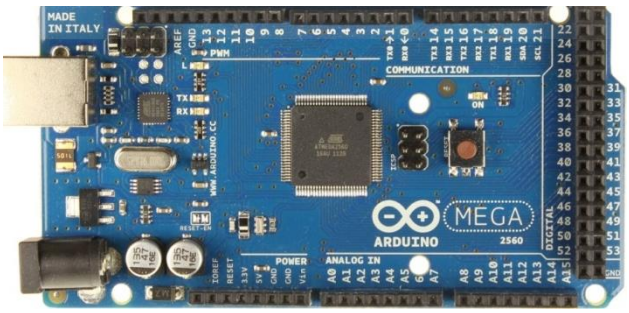
untuk mengontrol gerakan alat elektromekanik seperti lampu, motor, dan sebagainya.

Arduino dikatakan sebagai sebuah *platform* dari *physical computing* yang bersifat *open source*. Arduino tidak hanya sekedar sebuah alat pengembang, tetapi ia adalah kombinasi dari *hardware*, bahasa pemograman dan *Integrated Development Environment* (IDE) yang canggih. IDE adalah sebuah *software* yang sangat berperan untuk menulis program, meng-*compile* menjadi kode biner dan meng-*upload* ke dalam *memory* mikrokontroler. Ada banyak projek dan alat-alat dikembangkan oleh akademis dan professional dengan menggunakan Arduino, selain itu juga banyak modul-modul pendukung (sensor, tampilan, penggerak dan sebagainya) yang dibuat oleh pihak lain untuk bisa disambungkan dengan Arduino. Arduino dikembangkan oleh sebuah tim yang beranggotakan orang-orang dari belahanan dunia. Anggota inti dari tim ini adalah Massimo Banzi (Italy), David Cuartielles Malmoe (Sweden), Tom Igoe (NY, US), Gianluca Martino Torino (Italy), dan David A. Mellis Boston (MA, USA) (Tim Arduino, 2016).

Secara umum Arduino terdiri dari 2 bagian, yaitu :

1. *Hardware* : papan *input/output* (I/O)
2. *Software* : *software* Arduino meliputi IDE untuk menulis program, *driver* untuk koneksi dengan komputer, contoh program dan *library* untuk pengembangan program.

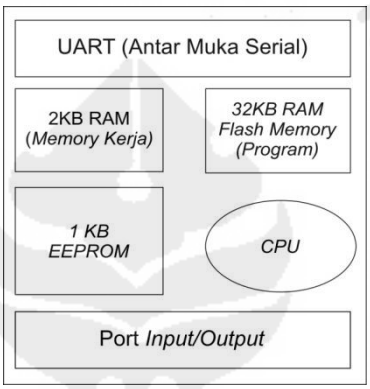
Bentuk dari salah satu jenis perangkat Arduino tersebut dapat kita lihat pada Gambar 2-8.



Gambar 2 - 8 Salah satu jenis papan Arduino (geeetech.com, 2014).

Komponen utama didalam papan Arduino adalah sebuah mikrokontroler 8 bit dengan merk Atmega yang dibuat oleh perusahaan Atmel Corporation. Berbagai papan Arduino menggunakan tipe Atmega yang berbeda-beda tergantung dari spesifikasinya, sebagai contoh Arduino Uno menggunakan Atmega 328 sedangkan Arduino Mega 2560 yang lebih canggih menggunakan Atmega 2560.

Untuk memberikan gambaran mengenai apa saja yang terdapat didalam sebuah mikrokontroler, pada Gambar 2-9 berikut ini diperlihatkan contoh diagram blok sederhana dari mikrokontroler Atmega328 (dipakai pada Arduino Uno).



Gambar 2 - 9 Diagram blok sederhana Arduino Uno

Blok-blok diatas dijelaskan sebagai berikut :

1. UART (*Universal Asynchronous Receive-Transmitter)* adalah antar muka yang digunakan untuk komunikasi serial seperti pada RS-232, RS-422 dan RS-485.
2. 2KB RAM pada memori kerja bersifat *volatile* (hilang saat daya dimatikan), digunakan oleh variabel-variabel didalam program.
3. 32KB RAM *flash memory* bersifat *non-volatile*, digunakan untuk menyimpan program yang dimuat dari computer. Selain program, *flash memory* juga menyimpan *bootloader*. *Bootloader* adalah program inisiasi yang ukurannya kecil, dijalankan oleh CPU saat daya dihidupkan. Setelah *bootloader* selesai dijalankan, berikutnya program dalam RAM akan dieksekusi.

4. 1KB EEPROM bersifat *non-volatile*, digunakan untuk menyimpan data yang tidak boleh hilang saat daya dimatikan.
5. CPU (*Central Processing Unit*), bagian dari mikrokontroler untuk menjalankan setiap instruksi dari program.

Port (I/O) *Input/Output*, pin-pin untuk menerima data (*input*) digital atau analog, dan mengeluarkan data (*output*) digital atau analog.

2.2.7 Komponen Dasar Sistem Kendali

Komponen dasar sistem kendali adalah sebagai berikut :

1. Masukan (*Input*)
2. Sistem kontrol
3. Hasil atau keluaran (*Output*)

Hubungan dasar antara ketiga komponen ini diterangkan pada gambar 2-10. Masukan (*input*) adalah rangsangan dari luar yang diterapkan ke sebuah sistem kendali untuk memperoleh tanggapan tertentu dari sistem, dan hasilnya disebut keluaran (*output*) atau variabel yang dikendalikan. Secara umum, tujuan sistem kendali adalah untuk mengendalikan keluaran dengan berbagai masukan tertentu melalui unsur-unsur sistem kendali.

Gambar 2 - 10 Komponen dasar sistem kendali

2.2.8 Sistem Kendali *Open Loop*

Sistem kendali *open loop* adalah sistem kendali yang sinyal keluarannya tidak berpengaruh terhadap aksi pengendaliannya (Ogata, 2002). Dalam hal ini sinyal keluaran tidak diukur atau di umpan balikkan untuk dibandingkan dengan sinyal masukannya. Gambar 2-11 menunjukan hubungan dari masukan dan keluaran suatu sistem kendali *open loop*.

Sebuah contoh praktis adalah memanggang/pemanas roti atau *toaster*. Prinsip kerja dari *toaster* hanya memanaskan roti yang dimasukan yang berperan

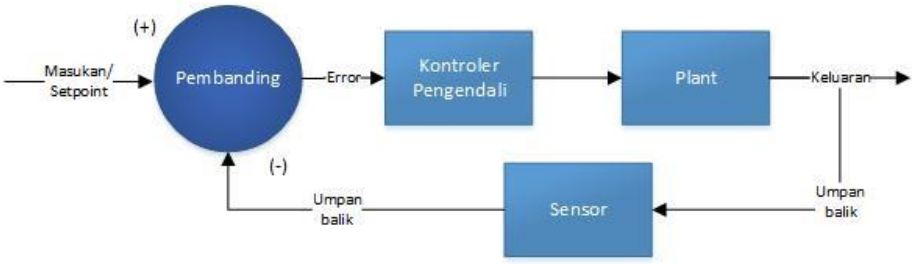
sebagai input. Input pada *toaster* merupakan waktu berapa lama roti dipanaskan. Setelah waktu diatur, roti tersebut akan dipanaskan selama waktu yang ditentukan dengan panas yang tetap. *Toaster* tidak memberikan pilihan beberapa tingkat kematangan roti akan keluar setelah dipanaskan melainkan hanya bekerja memanaskan roti dengan panas yang tetap pada waktu yang telah ditentukan.

Gambar 2 - 11 Diagram blok sistem kendali *open loop*

Jadi pada sistem kendali *open loop*, keluaran tidak dibandingkan dengan masukan acuannya. Oleh sebab itu, untuk setiap masukan acuan terdapat suatu kondisi operasi yang tetap. Perlu diketahui bahwa sistem kendali *open loop* harus dikalibrasi dengan hati-hati, agar ketelitian sistem tetap terjaga dan berfungsi dengan baik. Dengan adanya gangguan, sistem kendali *open loop* tidak dapat bekerja seperti yang diharapkan. Sistem kendali *open loop* dapat digunakan dalam praktek hanya jika hubungan masukan dan keluaran diketahui dan jika tidak terdapat gangguan internal maupun gangguan eksternal. Dengan demikian jelas bahwa sistem semacam ini bukan sistem kendali berumpan-balik. Demikian pula bahwa setiap sistem kendali yang bekerja berdasar basis waktu adalah sistem kendali *open loop*.

2.2.9 Sistem Kendali *Closed Loop*

Sistem kendali *closed loop* adalah sistem kendali yang sinyal keluarannya mempunyai pengaruh langsung terhadap aksi pengendaliannya. Dengan kata lain sistem kendali *closed loop* adalah sistem kendali berumpan-balik. Sinyal kesalahan *(error)*, yang merupakan selisih antara sinyal masukan (+) dan sinyal umpan-balik (-), diumpankan ke elemen kendali untuk memperkecil kesalahan dan membuat agar keluaran sistem mendekati harga yang diinginkan. Hal ini berarti bahwa pemakaian aksi umpan-balik pada sistem kendali *closed loop* bertujuan untuk memperkecil kesalahan sistem. Diagram blok sistem kendali *closed loop* dapat dilihat dari Gambar 2-12.



Gambar 2 - 12 Diagram blok sistem kendali *closed loop*

Berikut ini adalah uraian mengenai fungsi-fungsi elemen di atas :

1. Elemen Pembanding

   berfungsi untuk membandingkan nilai yang dikehendaki (*setpoint*/posisi target) dari variabel yang sedang dikontrol dengan nilai terukur yang diperoleh dan menghasilkan sebuah sinyal *error*. Sinyal *error* merupakan hasil dari sinyal dengan nilai yang dikehendaki (*setpoint*/posisi target) dikurangi dengan nilai aktual yang terukur.

2. Kontroler Pengendali

   Elemen kontroler pengendali menentukan aksi atau tindakan apa yang akan diambil bila diterima sebuah sinyal *error*. Kontrol akan mengarahkan/mengatur sistem untuk menghilangkan sinyal *error*.

3. *Plant*/Proses

   Plant/Proses adalah sebuah variabel atau objek yang dikontrol.

4. Sensor

   Elemen sensor dimaksudkan untuk dapat menghasilkan sebuah sinyal yang berhubungan dengan kondisi variabel dari proses yang sedang dikontrol.

2.2.10 Kontrol PID (*Proportional, Integrative, Derivative*)

Komponen dasar pembentuk kontroler PID adalah *Proporsional, Integral,* dan *derivative*. Setiap komponen memiliki fungsi yang berbeda dan memiliki efek yang berbeda pula pada suatu sistem (Wescott, 2000). Gambar 2-13 menunjukan diagram blok PID dalam suatu sistem kendali. Persamaan sinyal keluaran kontroler PID dapat dituliskan sebagai berikut :

$$U(t) = K_p e(t) + K_i \int_0^t e\,(t)dt + K_d \frac{de(t)}{dt} \qquad (2\text{-}1)$$

Transformasi Laplace digunakan untuk menghasilkan fungsi alih :

$$U(s) = \left(K_p + \frac{K_i}{s} + K_d s\right) E(s) \qquad (2\text{-}2)$$

Dimana :

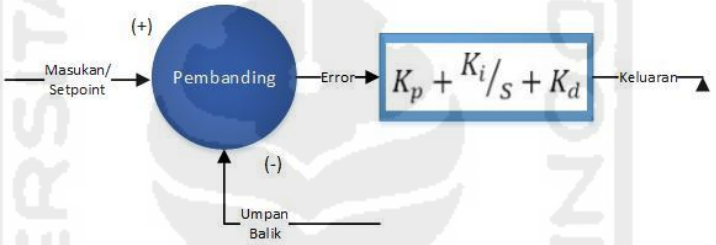
$U(s)$ = Sinyal keluaran

K_p = Konstanta proporsional

K_i = Konstanta integral

K_d = Konstanta derivatif

$e(t)$ = Sinyal *error*



Gambar 2 - 13 Diagram blok kendali PID

Seperti dalam penjelasan kontrol PID terbagi atas 3 komponen utama yaitu :

1. Kontrol *Proporsional*

   Kontrol proporsional berfungsi sebagai *gain* (penguat) *output* agar responsnya mendekati nilai *input*, P juga memelihara sistem untuk selalu dalam keadaan stabil.

2. Kontrol *Integrative*

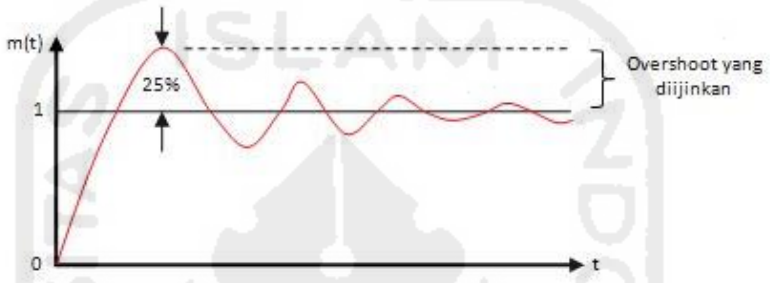
   Kontrol integral berfungsi memaksa sistem untuk tetap berada pada *steady state* dan juga mengurangi *error* atau kesalahan. Integral memperkokoh sistem terhadap gangguan.

3. Kontrol *Derivative*

   Kontrol derivatif berfungsi menambah nilai *zerro* sehingga parameter ini mempercepat respons. Selain itu juga mengurangi osilasi akibat dari *gain.*

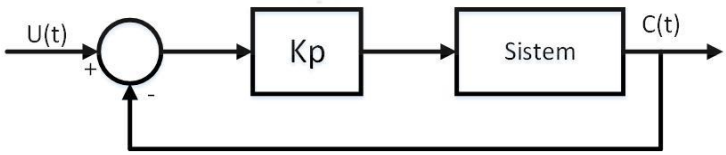
2.2.11 Metode Ziegler-Nichols 2

Pada metode Ziegler-Nichols penalaan parameter kontrol PID didasarkan pada respons fungsi tangga satuan eksperimental atau pada nilai K_p yang menghasilkan kestabilan marginal dengan hanya menggunakan tindakan kontrol proporsional. Aturan-aturan pada metode Ziegler-Nichols berdasarkan pada karakteristik respons transien suatu sistem yang diketahui. Metode ini ditujukan dalam pencapaian 25% lonjakan maksimum dalam respons step. Seperti pada gambar 2-14 dibawah ini.



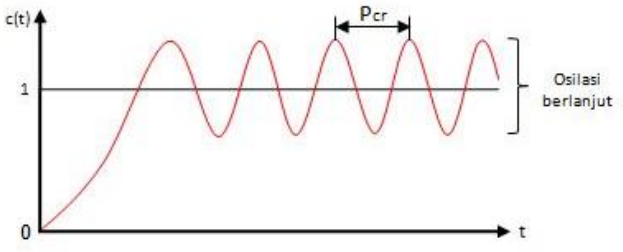
Gambar 2 - 14 Kurva respons tangga satuan memperlihatkan 25% lonjakan maksimum.

Metode osilasi ini didasarkan pada reaksi sistem *closed loop*. *Plant* disusun serial dengan kontroler PID. Semula parameter-parameter integrator disetel tak berhingga dan parameter diferensial disetel nol (T_i = ~ ; T_d = 0). Parameter proporsional kemudian dinaikkan bertahap. Mulai dari nol sampai mencapai nilai yang mengakibatkan reaksi sistem berosilasi secara berkesinambungan. Reaksi sistem harus berosilasi dengan *magnitude* tetap (*sustain oscillation*). Gambar 2- 15 menunjukkan rangkaian sistem *closed loop* pada metode osilasi.



Gambar 2 - 15 Sistem kendali *closed loop* dengan kontrol proporsional

Nilai penguatan proporsional pada saat sistem mencapai kondisi *sustain oscillation* disebut *ultimate gain* atau penguatan kritis *(K_{cr}).* Periode dari *sustained oscillation* disebut *ultimate period* atau periode *(P_{cr}).* Gambar 2-16 menggambarkan kurva reaksi sistem *closed loop* ketika berosilasi.



Gambar 2 - 16 Osilasi berkesinambungan dari periode P_{cr}.

Ziegler dan Nichols menyarankan penyetelan nilai parameter K_p, T_i, dan T_d berdasarkan rumus yang diperlihatkan pada Tabel 2-2.

Tabel 2-2 Aturan penalaan Ziegler-nichols didasarkan pada penguatan K_{cr} dan Periode P_{cr} kritis (Ogata, 2002).

Tipe kontroler	K_p	T_i	T_d
P	$0{,}5\ K_{Cr}$	∞	0
PI	$0{,}45\ K_{Cr}$	$\frac{1}{1{,}2} P_{Cr}$	0
PID	$0{,}6\ K_{cr}$	$0{,}5\ P_{Cr}$	$0{,}125\ P_{Cr}$

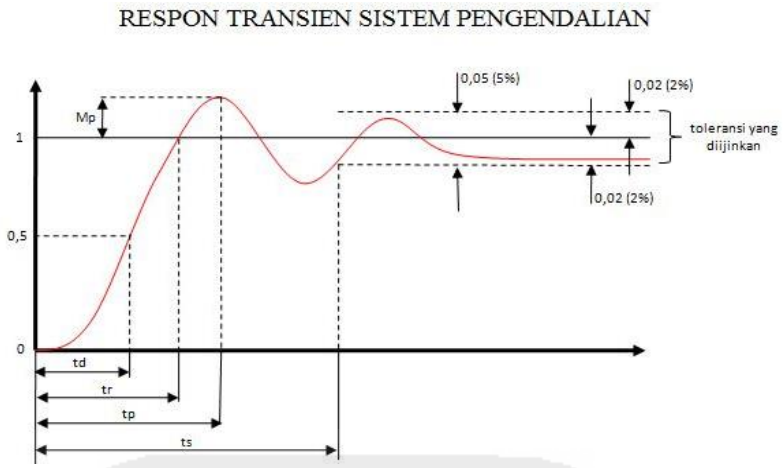
Untuk mencari nilai K_i dan K_d menggunakan persamaan 2-3 dan 2-4.

$$K_i = \frac{K_p}{T_i} \quad (2\text{-}3)$$

$$K_d = K_p \times T_d \quad (2\text{-}4)$$

2.2.12 Karakteristik Respons Sistem

Salah satu teknik untuk mengetahui karakteristik sistem adalah dengan memberinya sinyal uji sebagai *input* dan mengamati sinyal *output*/keluaran atau respons alihnya (transien). Respons tersebut dapat ditunjukan pada gambar 2-17.



Gambar 2 - 17 Respons transien sistem pengendalian.

Berikut keterangan karakteristik respons transien yang terdapat pada Gambar 2-18.

1. *Delay time*, t_d : waktu tunda, ukuran waktu yang menyatakan faktor keterlambatan respons *output* terhadap *input*. Diukur mulai, t = 0 sampai dengan respons mencapai 50% dari respons *steady state.*
2. *Risetime*, t_r : Waktu naik, ukuran waktu yang diukur mulai dari respons t = 0 sampai dengan respons memotong sumbu *steady state* yang pertama.
3. *Peak time*, t_p : Waktu puncak, ukuran waktu yang diperlukan respons mulai dari t = 0 hingga mencapai puncak pertama *overshoot*.
4. *Maximum overshoot*, M_p : Respons lonjakan maksimum/nilai relatif yang menyatakan perbandingan harga maksimum respons yang melampaui harga *steady state* dibanding dengan nilai *steady state.*
5. *Settling time*, t_s: Waktu tunak, ukuran waktu yang menyatakan respons telah masuk ±5%, ±2%, atau ±0,5% dari keadaan *steady state*.

2.2.13 *Crane*

Crane dikategorikan sebagai mesin yang dipergunakan untuk mengangkat beban, memindahkan secara horisontal dan menurunkan ketempat yang dituju dengan jangkauan terbatas. *Crane* digunakan dalam pekerjaan transportasi, industri, dan konstruksi. Dalam bidang transportasi *crane* digunakan untuk bongkar muat barang (*loading and unloading*) di pelabuhan, ataupun terminal

kontainer. Agar lebih jelasnya bagian-bagian *crane* dapat dilihat pada gambar 2-18.

Bagian-bagian *crane* dan fungsinya, antara lain :

1. Pemberat penyeimbang

   Untuk menyeimbangkan lengan *crane* (jib) ketika mengangkat beban.

2. Jib

   Lengan panjang yang mampu berputar 360 derajat secara horisontal.

3. Ruang Operator

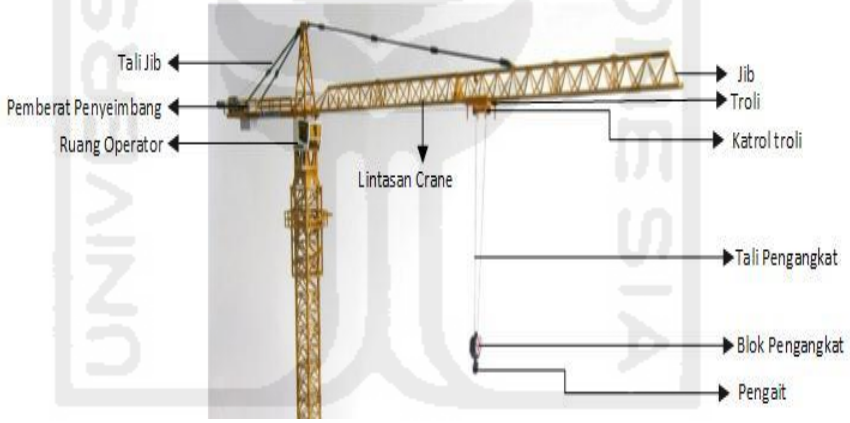
   Tempat pengendali/kontrol *crane* yang dikendalikan oleh operator.

4. Katrol troli

   Tempat beradanya kabel yang terhubung dengan blok pengangkat dan pengait untuk mengangkat beban yang mampu bergerak secara linier.

5. Lintasan *crane*

   Tempat berjalannya katrol troli secara linier.



Gambar 2 - 18 Bagian-bagian *crane* (Morrow Company Store, 2016).