PATH TRACKING PADA *MOBILE ROBOT* DENGAN UMPAN BALIK *ODOMETRY*

Bayu Sandi Marta[(1)], Fernando Ardilla[(2)], A.R. Anom Besari[(2)]

[(1)]Mahasiswa Program Studi Teknik Komputer, [(2)] Dosen Program Studi Teknik Komputer
Prodi Teknik Komputer, Politeknik Elektronika Negeri Surabaya
Kampus PENS-ITS Sukolilo, Surabaya
bayoesandi@yahoo.com

Abstrak

Makalah ini meyajikan suatu model pergerakan dari suatu robot yang menggunakan sistem penggerak diferensial dengan sensor rotary encoder yang akan diolah dengan metode odometry sehingga menghasilkan titik koordinat (path) dari posisi relatif robot secara real time. Nantinya robot tersebut akan bergerak dari posisi awal ke path tujuan. Untuk bergerak menuju path tujuan, robot ini akan melewati banyak path sehingga penulis memadukan dengan metode PID sebagai kontroler dalam menjajaki setiap path yang dilewati Hasil yang didapat dengan mencoba pada model lintasan yang berbeda dengan kecepatan 0,25 m/s menunjukkan bahwa robot dapat mengikuti jalur yang dibuat dengan simpangan terjauh sebesar 10cm pada sumbu x negatif sampai 10cm pada sumbu x positif serta 14cm pada sumbu y positif dan 7cm pada sumbu y negatif serta dengan menggunakan kecepatan 0,38 m/s simpangan terjauhnya sebesar 10cm pada sumbu x positif sampai 10cm pada sumbu x negatif serta 10cm pada sumbu y positif serta 7cm pada sumbu y negatif.

Kata kunci : Robot, Rotary encoder, Path, Real Time, Odometry, PID.

1. PENDAHULUAN

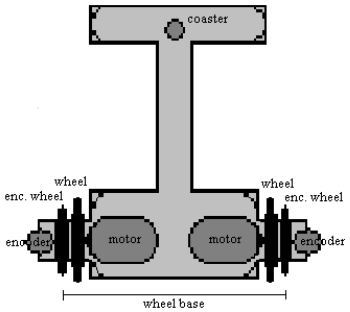
Pada aplikasi robot dikenal istilah posisi relatif. Posisi relatif ini biasanya didasarkan pada perhitungan rotasi roda. Posisi relatif ini bukan untuk menentukan posisi absolut dari robot tetapi hanya memperkirakan saja. Sensor yang umum digunakan untuk mendapatkan data posisi dari suatu robot adalah *rotary encoder* yang mana data dari sensor ini akan dimasukkan dalam perhitungan *odometry* sehingga menghasilkan posisi relatif dari robot tersebut. *Odometry* adalah penggunaan data dari sensor pergerakan untuk memperkirakan perubahan posisi dari waktu ke waktu. *Odometry* ini akan memetakan posisi robot dalam sumbu Cartesian. Sehingga akan didapatkan data posisi berupa titik koordinat (*path*) dan arah hadap (*heading*) dari robot tersebut. David P. Anderson membuat robot yang dalam navigasinya tidak menggunakan referensi dari luar seperti GPS dan lainnya [1].

Dalam bergerak robot akan mengukur perubahan *path* atau posisi dari waktu ke waktu, sehingga akan diketahui *error* atau kesalahan dari arah hadap robot terhadap *path* tujuan. Untuk dapat menuju *path* tujuan, seharusnya *error* dari arah hadap robot ini adalah nol. Untuk itu maka penulis menggunakan kontrol *Proportional-Integral-Detivative (PID)* untuk menjaga agar *error* yang dihasilkan selama perjalanan dari robot tersebut mendekati dan atau sama dengan nol. Kontrol *PID* ini akan mengatur kecepatan dari roda kanan dan roda kiri dengan umpan balik *error* dari arah hadap (*heading error*) robot tersebut terhadap *heading* yang seharus berdasarkan perhitungan *odometry*.

2. MENCARI POSISI DAN *TRACKING PATH*

2.1. Desain Robot

Robot yang digunakan menggunakan sistem penggerak diferensial dan memiliki dua pasang roda yaitu roda penggerak utama dan roda *encoder*. roda *encoder* dibuat lebih tipis sehingga mengurangi resiko ketidakpastian dari jarak antara roda *encoder* kanan dan roda *encoder* kiri (*wheel base*). roda *encoder* ini juga dibuat untuk dapat bergerak secara vertikal untuk meminimalkan faktor selip roda. parameter dalam *odometry* nantinya akan berasal dari roda *encoder* ini.



Gambar 2.1. Desain robot

2.2. *Odometry*

Odometry adalah penggunaan data dari pergerakan aktuator untuk memperkirakan perubahan posisi dari waktu ke waktu. *odometry* digunakan untuk memperkirakan posisi relatif terhadap posisi awal.

Untuk memperkirakan posisi relatif robot, digunakan perhitungan jumlah pulsa yang dihasilkan oleh sensor *rotary encoder* setiap satuan ukuran yang kemudian dikonversi menjadi satuan *millimeter*.

Untuk mendapatlan jumlah pulsa setiap satu kali putaran roda digunakan rumus sebagai berikut:

$$K\ roda = 2 \times \pi \times r \quad (1)$$
$$pulsa\_per\_mm = resolusi\ enc / K\ roda \quad (2)$$

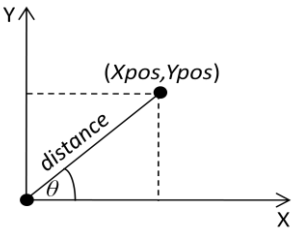
Pada sistem penggerak diferensial terdapat dua roda,yaitu roda kanan dan roda kiri dan dimisalkan jumlah pulsa_per_mm untuk roda kanan adalah *right_encoder* dan roda kiri adalah *left_encoder* dan jarak antara dua roda adalah *wheel_base* maka didapatkan jarak tempuh (*distance*) dan sudut orientasi (θ). Rumusnya adalah sebagai berikut.

$$distance = (left\_enc + right\_enc) / 2 \quad (3)$$
$$\theta = (left\_enc - right\_enc) / wheel\_base \quad (4)$$

Karena θ adalah sudut dalam radian maka untuk mengetahui sudut dalam derajat (*heading*) digunakan rumus sebagai berikut :

$$heading = \theta \times \frac{180}{\pi} \quad (5)$$

Dari ketentuan diatas didapatkan bahwa nilai *heading* akan bernilai negatif (-) ketika robot berputar melawan arah jarum jam dan akan bernilai positif (+) ketika robot berputar searah dengan jarum jam. Dengan mengetahui jarak dan sudut (*distance* dan θ) maka kita dapat mengetahui koordinat X dan koordinat Y dengan persamaan trigonometri.



Gambar 2.2. Ilustrasi pada sumbu cartesian

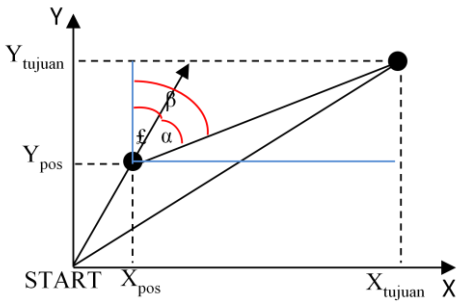
Dari ilustrasi diatas maka koordinat dari robot dapat kita ketahui dengan rumus :

$$Xpos = distance \times sin\ (\theta) \quad (6)$$
$$Ypos = distance \times cos\ (\theta) \quad (7)$$

Untuk menentukan *error* arah hadap dari robot terhadap titik tujuan maka digunakan teorema *phytagoras* yang akan menghasilkan posisi (*path*) saat ini dan jarak terhadap titik tujuan, berikut perhitungannya:

$$x = X\_tujuan - Xpos \quad (8)$$
$$y = Y\_tujuan - Ypos \quad (9)$$
$$target\_distance = \sqrt{x^2 + y^2} \quad (10)$$

Heading dari robot yang telah diketahui sehingga kita dapat menghitung *error* arah hadap (*heading error*) robot terhadap titik tujuan.



Gambar 2.3. Sudut *α, β,* dan £

Pada gambar 4.2. menunjukkan ilustrasi untuk mencari *heading error* (α) dimana β adalah *target bearing* yaitu sudut antara posisi robot saat ini terhadap titik tujuan. Sedangkan garis berwarna biru adalah garis bantu yang masing-masing sejajar dengan sumbu X dan sumbu Y.

Untuk mendapat nilai dari β, maka digunakan rumus sebagai berikut :

$$\beta = arctan\ \frac{(Ytujuan - Ypos)}{(Xtujuan - Xpos)} \quad (11)$$

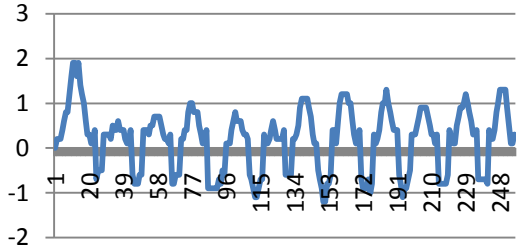
sehingga

$$\alpha = \beta - £ \quad (12)$$

2.3. Penalaan kostanta *PID*

Pada penelitian ini, dalam bergerak robot akan mencari *heading*-nya dan sudut terhadap posisi tujuan secara terus-menerus. sehingga akan didapatkan *heading error*.untuk mempertahankan agar *heading error* ini tetap bernilai nol maka digunakan kontrol *PID*.

Sesuai dengan aturan Ziegler-Nichols yang kedua, untuk mendapatkan nilai konstanta *Kp*, *Ki* dan *Kd* maka sistem diberi kontroler P. nilai *Kp* ditambahkan sampai sistem tersebut berosilasi secara berkesinambungan.



Gambar 4. Kurva Respon Dengan Nilai Kp 0,7.

Sesuai dengan aturan Ziegler-Nichols 2 maka :

Kcr = 0,7
Pcr = 21 x 10ms = 210ms

T_i = 0,5 x 210 = 105
T_d = 0,125 x 210 = 26,25

Fungsi alih kontroler *PID* plant sebagai berikut:

$$U = Kp(1 + \frac{1}{T_i s} + T_d s) \quad (13)$$

Kp = 0,6 x 0,7 = 0,42
Ki = 0,42 / 105 = 0,004
Kd = 0,42 x 26,25

Dalam aplikasinya, maka peran dari kontroler ini dapat diterapkan dalam program dengan formulasi seperti berikut:

$$U = Kp \times err + Ki \times i_{err} + Kd \times d_{err} \quad (14)$$

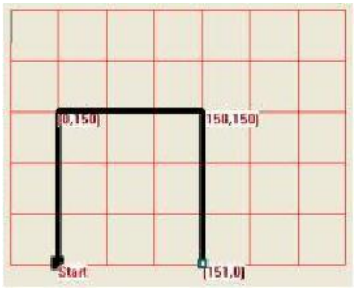
3. PENGUJIAN DAN ANALISA

Pada pengujian ini robot akan diberi jalur berupa beberapa titik koordinat (X,Y) yang harus ditempuh. titik pertama yang diberikan akan menjadi titik acuan yang mana titik acuan ini selalu bernilai (0,0).dari titik awal ini secara umum seluruh lintasan robot memiliki empat kuadran yaitu kuadran satu, dua, tiga dan empat. berikut pemetaan koordinat berdasarkan kuadrannya :

Kuadran	X	Y	Letak terhadap titik awal
1	+	+	Kanan depan
2	-	+	Kiri depan
3	-	-	Kiri belakang
4	+	-	Kanan belakang

Tabel 3.1. Pembagian kuadran

Pengujian pertama dilakukan dengan memberikan empat titik yaitu (0,0), (0,150), (150,150) dan (151,0). satuan jaraknya dalam centimeter.



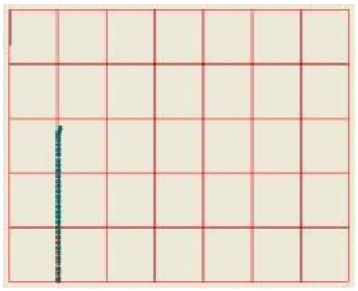
Gambar 3.1. Koordinat input

Grid berwarna merah pada gambar diatas memiliki ukuran 50 x 50 yang mana grid tersebut adalah gambaran dari lapangan robot pada kondisi nyata.

Gambar 3.2. Posisi start pada lapangan

Dari kedua gambar diatas maka robot jalur yang harus dilalaui robot adalah garis berwarna hitam sehingga setelah tiga kotak robot akan belok kanan dan tiga kotak kemudian robot akan belok kanan dan tiga kotak kemudian robot akan berhenti.ukuran kotak pada gambar 3.2. adalah 50 x 50 dalam satuan centimeter.

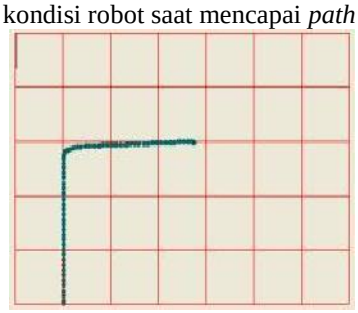
Berikut kondisi robot saat mencapai titik koordinat (path) kedua:



Gambar 3.3. Jalur berdasarkan perhitungan *odometry* dari robot pada *path* kedua

Gambar 3.4. Kondisi robot pada *path* kedua

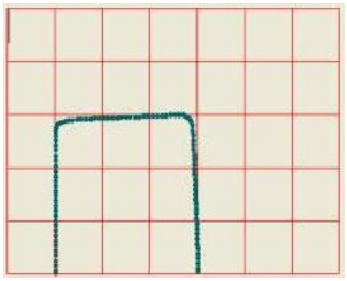
Berikut kondisi robot saat mencapai *path* ketiga:



Gambar 3.5. Jalur berdasarkan perhitungan *odometry* dari robot pada *path* ketiga

Gambar 3.4. Kondisi robot pada *path* ketiga

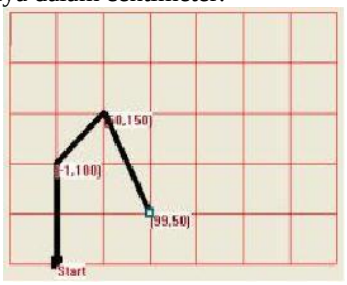
Berikut kondisi robot saat mencapai *path* keempat:



Gambar 3.5. Jalur berdasarkan perhitungan *odometry* dari robot pada *path* keempat

Gambar 3.6. Kondisi robot pada *path* keempat

Pengujian kedua dilakukan dengan memberikan empat titik yaitu (0,0), (0,100), (50,150) dan (99,50). satuan jaraknya dalam centimeter.



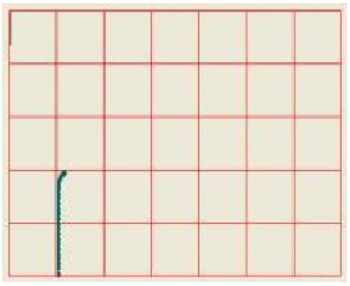
Gambar 3.1. Koordinat input

Grid berwarna merah pada gambar diatas memiliki ukuran 50 x 50 yang mana grid tersebut adalah gambaran dari lapangan robot pada kondisi nyata.

Gambar 3.2. Posisi start pada lapangan

Dari kedua gambar diatas maka robot jalur yang harus dilalaui robot adalah garis berwarna hitam sehingga setelah tiga kotak robot akan belok kanan dan tiga kotak kemudian robot akan belok kanan dan tiga kotak kemudian robot akan berhenti.ukuran kotak pada gambar 3.2. adalah 50 x 50 dalam satuan centimeter.

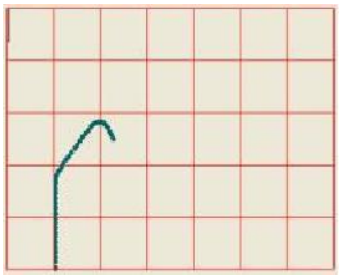
Berikut kondisi robot saat mencapai titik koordinat (path) kedua:



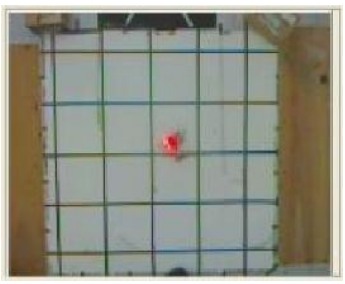
Gambar 3.3. Jalur berdasarkan perhitungan *odometry* dari robot pada *path* kedua

Gambar 3.4. Kondisi robot pada *path* kedua

Berikut kondisi robot saat mencapai *path* ketiga:



Gambar 3.5. Jalur berdasarkan perhitungan *odometry* dari robot pada *path* ketiga



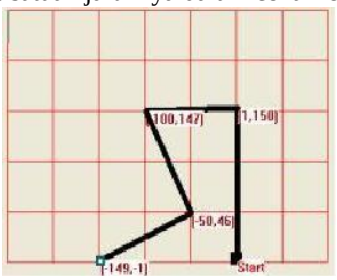
Gambar 3.4. Kondisi robot pada *path* ketiga

Berikut kondisi robot saat mencapai *path* keempat:

Gambar 3.5. Jalur berdasarkan perhitungan *odometry* dari robot pada *path* keempat

Gambar 3.6. Kondisi robot pada *path* keempat

Pengujian ketiga dilakukan dengan memberikan empat titik yaitu (0,0), (0,150), (-100,147), (-50,46) dan (-149,-1). satuan jaraknya dalam centimeter.



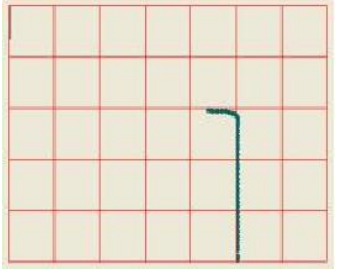
Gambar 3.1. Koordinat input

Grid berwarna merah pada gambar diatas memiliki ukuran 50 x 50 yang mana grid tersebut adalah gambaran dari lapangan robot pada kondisi nyata.

Gambar 3.2. Posisi start pada lapangan

Dari kedua gambar diatas maka robot jalur yang harus dilalaui robot adalah garis berwarna hitam sehingga setelah tiga kotak robot akan belok kanan dan tiga kotak kemudian robot akan belok kanan dan tiga kotak kemudian robot akan berhenti.ukuran kotak pada gambar 3.2. adalah 50 x 50 dalam satuan centimeter.

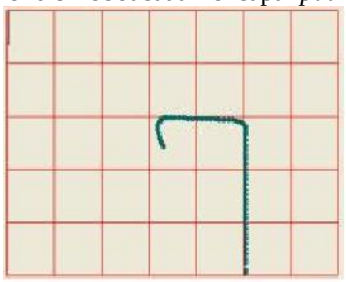
Berikut kondisi robot saat mencapai titik koordinat (path) kedua:



Gambar 3.3. Jalur berdasarkan perhitungan *odometry* dari robot pada *path* kedua

Gambar 3.4. Kondisi robot pada *path* kedua

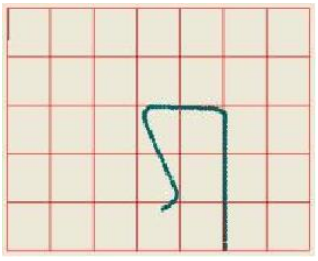
Berikut kondisi robot saat mencapai *path* ketiga:



Gambar 3.5. Jalur berdasarkan perhitungan *odometry* dari robot pada *path* ketiga

Gambar 3.4. Kondisi robot pada *path* ketiga

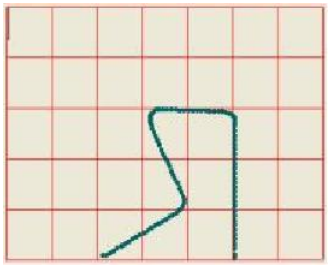
Berikut kondisi robot saat mencapai *path* keempat:



Gambar 3.5. Jalur berdasarkan perhitungan *odometry* dari robot pada *path* keempat

Gambar 3.6. Kondisi robot pada *path* keempat

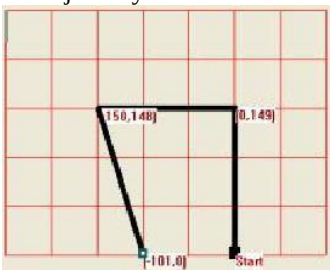
Berikut kondisi robot saat mencapai *path* kelima:



Gambar 3.5. Jalur berdasarkan perhitungan *odometry* dari robot pada *path* kelima

Gambar 3.6. Kondisi robot pada *path* kelima

Pengujian keempat dilakukan dengan memberikan empat titik yaitu (0,0), (0,100), (50,150) dan (99,50). satuan jaraknya dalam centimeter.



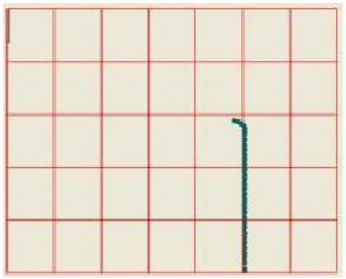
Gambar 3.1. Koordinat input

Grid berwarna merah pada gambar diatas memiliki ukuran 50 x 50 yang mana grid tersebut adalah gambaran dari lapangan robot pada kondisi nyata.

Gambar 3.2. Posisi start pada lapangan

Dari kedua gambar diatas maka robot jalur yang harus dilalaui robot adalah garis berwarna hitam sehingga setelah tiga kotak robot akan belok kanan dan tiga kotak kemudian robot akan belok kanan dan tiga kotak kemudian robot akan berhenti.ukuran kotak pada gambar 3.2. adalah 50 x 50 dalam satuan centimeter.

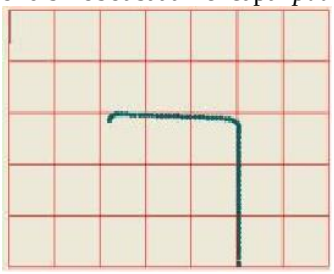
Berikut kondisi robot saat mencapai titik koordinat (path) kedua:



Gambar 3.3. Jalur berdasarkan perhitungan *odometry* dari robot pada *path* kedua

Gambar 3.4. Kondisi robot pada *path* kedua

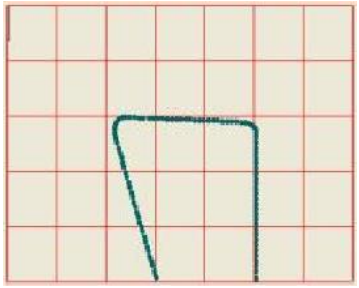
Berikut kondisi robot saat mencapai *path* ketiga:



Gambar 3.5. Jalur berdasarkan perhitungan *odometry* dari robot pada *path* ketiga

Gambar 3.4. Kondisi robot pada *path* ketiga

Berikut kondisi robot saat mencapai *path* keempat:



Gambar 3.5. Jalur berdasarkan perhitungan *odometry* dari robot pada *path* keempat

Gambar 3.6. Kondisi robot pada *path* keempat

Dari percobaan diatas secara perhitungan *odometry* pada robot jalur dibuat dari empat *path* diatas sama dengan jalur masukan pada gambar 3.1. namun secara riil, posisi robot masih memiliki *error* kurang lebih sekitar 3cm. hal ini lebih dikarenakan dalam satu siklus perhitungan *odometry* pada robot yang memerlukan waktu sehingga terkadang menyebabkan keterlambatan proses *sampling rotary encoder*.

Percobaan diatas dilakukan di tempat yang rata agar perhitungan *odometry* pada robot tidak kacau. dari empat macam path yang telah dicoba tersebut, error titik akhir dari robot berkisar antara -3 sampai 3 cm.

4. KESIMPULAN

Setelah dilakukan pengujian dan analisa, maka dapat diambil kesimpulan bahwa:

1. *Odometry* memiliki batasan yaitu dapat bekerja dengan baik jika lintasan nya berupa bidang datar dan tidak menyebabkan *rotary encoder* selip.
2. *Path tracking* dengan menggunakan kontrol *PID* memiliki tingkat akurasi yang baik.
3. Secara perhitungan di mikrokontroller (robot) hasil yang didapat mendekati 100% namun pada kondisi riil robot sering melenceng dari posisinya. hal ini dikarenakan keterlambatan dalam menyelesaikan satu siklus *odometry* dan *PID* sehingga mempengaruhi pembacaan *rotary encoder*.

5. DAFTAR PUSTAKA

[1] Anderson, David, IMU Odometry, http://www.geology.smu.edu/%7Edpawww/dpa.html , Texas, 2006

[2] P. Jim_enez, F. Thomas dan C. Torras," *Research on 6-DOF motion platform on PMAC*"

[3] Barış GÖKÇE,"*Odometry Motion Model and Motion Model with Map*", File PPT

[4] Kok Seng *CHONG* Lindsay KLEEMAN," *Accurate Odometry and Error Modelling for a Mobile Robot*",1997

[5] Ogata, Katsuhiko,"*Modern Control Engineering Third Edition*",1997

[6] Rachmadyanti, Nita,"*Kontrol PID Untuk Pengaturan Kecepatan Motor Pada Prototype Ayunan Bayi Otomatis*", Tugas Akhir PENS ITS 2010.

[7] Tsukasa.Watanabe1, Hiroyuki.Fujimoto1, Tadashi.Masuda2,"*Self-Calibratable Rotary Encoder*",2005

[8] Borenstein, J; Everett, H. R. and Feng, L, Contributing authors: S. W. Lee and R. H. Byrne,"*Where am I ? Sensors and Methods for Robot Positioning*", Michigan, 1996.