Pembuatan Robot Penjejak Garis Berbasis Visual Menggunakan *Fuzzy logic controller*

Riza Agung Firmansyah

rizaagungf@itats.ac.id

Jurusan Teknik Elektro Institut Teknologi Adhi Tama Surabaya

Abstrak

Robot penjejak garis merupakan salah satu aplikasi robot yang sederhana. Robot ini biasa digunakan untuk media pembelajaran dan aplikasi industri. Untuk mengikuti garis maka robot harus dilengkapi sensor garis. Sensor yang biasa digunakan adalah photodioda. Penggunaan sensor photodioda memiliki masalah ketelitian karena keterbatasan mengenali posisi aktual robot. Hal tersebut dipengaruhi keterbatasan jumlah sensor yang digunakan. Salah satu sensor yang bisa mengatasi masalah ini adalah kamera. Dengan ketelitan yang lebih baik maka kombinasi masukan sistem kontrol yang digunakan lebih bervariasi. Sistem kendali yang dapat digunakan adalah *fuzzy logic controller*. Dengan penerapan *fuzzy logic controller* diharapkan mampu memperbaiki sistem kendali konvensional yang sudah biasa digunakan. Selain itu *fuzzy logic controller* mudah diatur dan tidak perlunya mengetahui model matematis dari robot. Dari pengujian yang telah dilakukan, robot mampu menuju *steady state* setelah melewati tikungan dalam rata-rata waktu 0,74 detik dengan nilai rata-rata *error* 74,7.

Kata Kunci — robot penjejak garis, kamera, *fuzzy logic controller*.

Abstract

Line tracer robot is one of robot simple application. This robot ussualy used for learning media and industrial aplication. To trace a line, robot must be equipped with line sensor. Sensor that commonly used is photodiode. Photodiode sensor has accuracy problems due to limitations to identify the actual position of the robot. Sensor that can solve this problem is camera. With better accuration, combination of control systems input that used is more varies. Control system that can be used is a fuzzy logic controller. Fuzzy logic controller is expected to improve the conventional control system that has been used. Furthermore, fuzzy logic controller easily tuned and do not need to find out mathematical model of the robot. From the experiment, robot is able to steady state after passing a curve in average time 0,74 second with average error value 74,7.

Keywords — line tracer robot, camera, fuzzy logic controller.

II. PENDAHULUAN

Line tracer robot atau robot penjejak garis sudah banyak digunakan dalam dunia pendidikan hingga industri. Robot ini memiliki kinerja yang cukup handal dan efisien dalam melakukan tugasnya. Hal ini dikarenakan robot hanya memerlukan algoritma pembacaan garis dan sistem kendali pergerakan. Di dalam dunia pendidikan, *line tracer* merupakan salah satu robot yang mudah untuk dibuat sehingga tepat untuk dijadikan bahan belajar pembuatan robot dan pemrograman. Sedangkan aplikasinya dalam industri umumnya digunakan untuk mengangkut barang dari satu tempat ke tempat lain [1].

Sistem deteksi garis merupakan salah satu bagian terpenting dalam robot pengikut garis. Keberhasilan robot untuk berjalan dengan baik atau tidak sangat tergantung pada sistem deteksi garis yang dibuat. Pada umumnya sistem deteksi garis pada *line tracer robot* menggunakan sensor warna [2], [3], *magnetic sensor* [4], maupun menggunakan kamera [5][6][7][8].

Permasalahan yang muncul dalam sistem deteksi garis adalah gangguan pencahayaan. Hal ini berakibat robot tidak mampu mendeteksi garis melengkung saat cahaya redup [2]. Tingkat ketelitian sensor juga mempengaruhi performa robot. Kemampuan robot dalam mendeteksi variasi posisi aktual berbanding lurus dengan jumlah sensor yang digunakan. Penelitian [3] dan [4], memiliki kemampuan mendeteksi garis dalam lingkungan redup namun hanya menggunakan 3 sensor. Hal ini menyebabkan robot hanya mampu mengenali 3 posisi yaitu kanan, tengah, dan kiri. Keterbatasan informasi mengenai posisi aktual robot menyebabkan sistem kendali yang dibuat kurang fleksibel.

Penelitian [5] menggunakan kamera sebagai sensor pendeteksi garis. Sensor garis menggunakan kamera akan memberikan variasi posisi aktual yang lebih banyak. Robot yang telah dibuat juga mampu mengenali garis dengan baik. Namun penerapan sistem kendali yang digunakan masih menggunakan kontrol proporsional. Menyebabkan kemampuan robot saat melewati tikungan kurang baik. Sehingga untuk meningkatkan kemampuan robot dalam melewati tikungan, diperlukan sistem kendali yang lebih baik. Salah satu sistem kendali yang bisa digunakan adalah *fuzzy logic controller (FLC)*.

Penelitian [6][7] telah menerapkan *fuzzy logic controller* pada robot penjejak garis dengan sensor kamera. *FLC* yang digunakan diterapkan pada sebuah laptop sehingga dimensinya cukup besar. Kamera yang digunakan juga harus diproses menggunakan laptop. Robot dalam penelitian [8] menggunakan kamera OV7670 dengan unit pemroses STM32F4 sehingga dimensinya lebih kecil. Namun kamera OV7670 yang digunakan hanya mampu menampilkan gambar dengan resolusi 176 x 144.

Berdasarkan permasalahan tersebut maka dalam penelitian ini dibuat sebuah sistem kendali *FLC* yang diterapkan pada sebuah robot penjejak garis. Sensor yang digunakan adalah kamera *Raspberry Camera* yang mampu menampilkan gambar dengan resolusi 640 x 480. Unti pemroses yang digunakan adalah mini komputer Raspberry Pi sehingga dimensi robot lebih kecil.

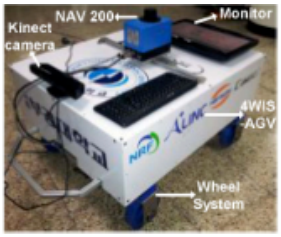
III. TINJAUAN PUSTAKA

Robot penjejak garis atau *line tracer robot* merupakan robot yang mampu bergerak mengikuti garis yang telah dibuat. Robot ini pada umumnya digunakan untuk media pembelajaran pada tingkat sekolah maupun universitas. Selain itu, *line tracer robot* juga digunakan di dalam industri seperti yang ditunjukan pada Gambar 1. Dalam industri, biasanya robot ini digunakan untuk mengangkut barang sesuai dengan line produksi [1], [9].

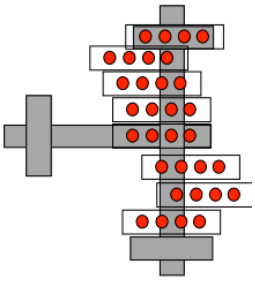
Pada umumnya *robot line tracer* menggunakan sensor garis yang terbuat dari pasangan LED dan sensor cahaya [3][10]. Sensor cahaya yang digunakan pada umumnya adalah photodiode, LDR, maupun sensor warna. Sensor tersebut membaca intensitas pantulan cahaya yang dipancarkan oleh LED. Saat cahaya LED mengenai warna hitam maka pantulan yang diterima oleh sensor lebih sedikit dari pada pantulan saat cahaya LED mengenai warna putih. Sensor cahaya lain yang biasa digunakan adalah sensor warna. Pada penelitianya, [2] memiliki kendala pada cahaya redup. Saat cahaya redup, sensor tidak mampu mendeteksi garis lengkung.

Penelitian [4] yang menggunakan *magnetic sensor* tidak terpengaruh oleh pencahayaan namun jalur yang digunakan harus menggunakan garis yang terbuat dari logam. Selain itu jumlah posisi yang mampu dideteksi robot hanya sejumlah sensor yang digunakan. Penggunaan sensor yang hanya mampu mengenali posisi yang terbatas menyebabkan sistem kendali yang dibuat kurang fleksibel. Permasalahan ini juga dialami [3][2] yang hanya menggunakan 4 pasang sensor. Sehingga untuk mendapatkan sistem kendali yang lebih fleksibel maka sensor yang digunakan harus mampu mengenali lebih banyak posisi robot. Salah satu sensor yang bisa digunakan adalah kamera.

Kamera mampu mengambil gambar dengan resolusi tertentu. Untuk mendeteksi lebar garis, bisa ditentukan berdasarkan jumlah piksel pada sumbu horizontal. Sebagai contoh, jika kamera memiliki resolusi 640x480 piksel maka kamera tersebut memiliki jumlah piksel pada sumbu horizontal sebesar 640 piksel. Sehingga dengan menggunakan kamera tersebut maka robot mampu mengenali 640 posisi. Pada penelitianya [5] menggunakan kamera webcam dengan unit pemroses data menggunakan *mini* komputer raspberry pi. Robot yang dibuat mampu mengenali garis dengan baik. Namun penerapan sistem kendali yang digunakan masih menggunakan kontrol proporsional. Hal tersebut berakibat robot memiliki keberhasilan yang rendah saat melewati tikungan.



Gambar 1. AGV dengan sensor kinect pada penelitian [9]



Gambar 2. Kombinasi pembacaan sensor penelitian [3]

Dengan memanfaatkan kamera sebagai sensor maka lokasi garis dapat diketahui dengan mengolah citra atau gambar yang telah diambil kamera. Proses pengolahan citra ini dilakukan secara *software* yang dijalankan komputer. pada umumnya proses pengolahan citra diawali dengan *preprosesing* yang meliputi konversi RGB ke *grayscale*, filter, segmentasi dan lain-lain. Untuk mempermudah proses tersebut, tahap *preprosesing* dan proses pengolahan citra lainya dapat mengguanakan *tools* opencv.

Penelitian [5] menggunakan *mini* komputer raspberry pi untuk unit pemroses data. Raspberry PI merupakan salah satu jenis *mini* komputer yang sering digunakan untuk keperluan *embedded system*. Hal ini didukung oleh beberapa fitur raspberry yang hampir setara dengan personal komputer pada umumnya [11]. Selain kinerja prosesor yang cukup baik, raspberry PI mampu untuk menjalankan program pengolahan citra digital. Selain itu, raspberry PI juga mendukung atau kompatible dengan *tools* pengolahan citra digital opencv.

Penelitian [6] menggunakan laptop sebagai unit pemroses. Sensor yang digunakan adalah kamera yang memiliki resolusi 240 x 320. Kamera tersebut terhubung secara langsung melalui *port* USB. Gambar yang diambil oleh kamera diproses oleh laptop dan selanjutnya dijadikan acuan sistem kendali. Dimensi robot yang dibuat relatif besar sebab laptop harus berada dalam robot.

Robot dalam penelitian [7] memiliki konsep yang hampir sama dengan [6]. Namun hubungan antara kamera dengan laptop menggunakan komunikasi *wireless*. Dengan cara tersebut, dimensi robot relatif lebih kecil dibandingkan robot [6] karena laptop dan robot terpisah. Penelitian [8] menggunakan kamera OV7670 dengan unit pemroses mikrokontroler STM32F4. Dimensi robot lebih kecil dibandingkan robot[6] dan [7]. Namun resolusi kamera yang digunakan cukup kecil yaitu 176 x 144.

Penelitian [6], [7], dan [8] menggunakan *fuzzy logic controller* sebagai sistem kendalinya. Penelitian [6], dan [7] menerapkan *fuzzy logic controller* menggunakan *software*

matlab. Dengan matlab pembuatan *fuzzy logic controller* jauh lebih mudah. Namun untuk aplikasi *embedded* hal ini tidak bisa dilakukan. Sebab matlab hanya dapat dioperasikan di sistem operasi windows. [8] menerapkan sistem kendalinya pada mikrokontroler sehingga bisa untuk aplikasi *embedded*. Namun hanya menggunakan tiga jenis variabel linguistik pada *input membership function* seperti yang ditunjukan Gambar 3.

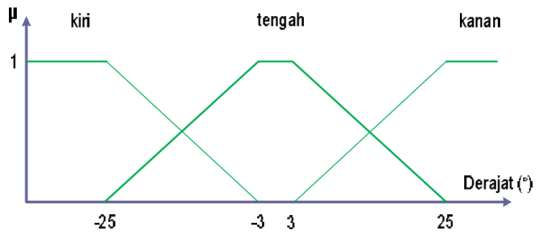
Fuzzy logic controller dibuat melalui tiga langkah utama yaitu fuzzifikasi, evaluasi *rule* atau *inference*, dan defuzzifikasi. Fuzzifikasi merupakan proses untuk mengubah variabel crisp (variabel numerik) menjadi variabel fuzzy (variabel linguistik). Nilai masukan yang dikuantisasi sebelumnya diolah oleh *FLC* dan kemudian diubah ke dalam variabel *fuzzy*. Melalui *membership function* (fungsi keanggotaan) yang telah disusun, maka dari masukan tersebut didapatkan derajat keanggotaan bagi masing-masing masukan.

Membership function adalah suatu fungsi (kurva) yang menunjukkan pemetaan titik-titik *input* data ke dalam nilai keanggotaannya (derajat keanggotaan) yang memiliki interval antara 0 sampai 1. *Membership function* yang digunakan pada umumnya adalah fungsi segitiga, trapesium, sigmoid, dll. Fungsi segitiga menghasilkan nilai keanggotaan yang linier. Persamaan (1) menunjukan cara untuk mendapatkan nilai keanggotaan dengan fungsi segitiga.

$$\mu_{input} = 1 - \frac{2 * |x_{crisp} - b|}{c - a} \quad \text{..... (1)}$$

Langkah berikutnya adalah proses evaluasi *rule* atau *inference*. Masing-masing *rule* merepresentasikan hubungan antara *input* dan *output* menggunakan aturan *if-then*. Proses evaluasi *rule* mengevaluasi derajat keanggotaan tiap-tiap fungsi keanggotaan himpunan fuzzy masukan ke dalam basis aturan yang telah ditetapkan. Dengan aturan tersebut kemudian disusun sebuah tabel evaluasi *rule* untuk mempermudah proses pemrograman kontrol. Tujuan dari evaluasi aturan ini adalah menentukan derajat keanggotaan dari keluaran *fuzzy*.

Setelah mendapatkan derajat keanggotaan dan melakukan evaluasi *rule* maka langkah berikutnya adalah proses defuzzifikasi. Proses defuzzifikasi adalah proses mengubah variabel fuzzy menjadi variabel crisp. Hingga proses evaluasi *rule* nilai yang didapatkan masih berupa variabel fuzzy (linguistik) sehingga agar perlu dikonversi lagi menjadi variabel crisp (numerik). Defuzzifikasi dapat dilakukan dengan beberapa cara salah satunya adalah *centre of gravity*. Defuzzifikasi dengan *centre of gravity* dilakukan dengan (2).

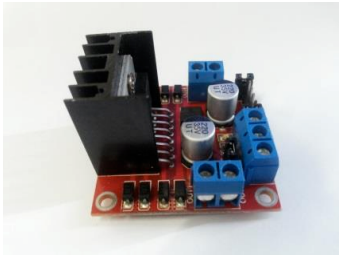


Gambar 3. Variabel lingusitik pada penelitian [8]

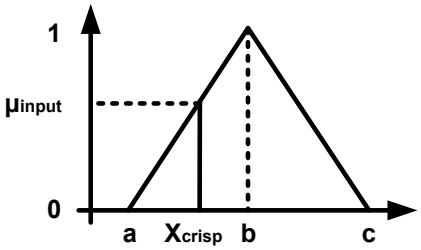
$$z_{output} = \frac{\sum \mu c(z) z dz}{\sum \mu c(z) dz} \quad \text{..... (2)}$$

Pada umumnya keluaran sebuah sistem kendali adalah berupa sinyal untuk mengendalikan motor. Namun sinyal kendali yang dihasilkan tidak selalu dapat dihubungkan ke motor. Maka untuk menghubungkan sinyal kendali dengan motor dibutuhkan sebuah rangkaian driver motor. Penelitian [5] menggunakan driver motor L298 yang merupakan *dual H-bridge motor driver*. Rangkaian driver ini mampu mengendalikan dua buah motor dengan arus hingga 3A. Modul driver L298 ditunjukan pada Gambar 5.

Motor DC akan berputar saat muncul medan magnet pada kumparan jangkar (rotor) dan kumparan medan (stator). Medan magnet yang dihasilkan berasal dari arus listrik yang disuplai ke dua kumparan tersebut. Untuk meringkas konstruksi motor DC, maka kumparan medan dapat diganti dengan magnet permanen. Beberapa jenis motor dc telah dilengkapi dengan *gearbox* sehingga beban langsung bisa dihubungkan ke poros motor. Motor jenis ini salah satunya adalah Mabuchi RF-370CA-22170 seperti yang ditunjukan pada Gambar 6. Motor dc ini memiliki kecepatan tanpa beban hingga 6000 RPM dengan torsi 137 gr-cm. Tegangan kerja motor ini sebesar 12V dengan konsumsi arus 620mA [12].



Gambar 5. Rangkaian driver motor L298[5]



Gambar 4. Fungsi keanggotaan segitiga

Gambar 6. Bentuk fisik Mabuchi RF-370CA-22170 [9].

IV. METODOLOGI

A. Perancangan Sistem Mekanik Robot

Tahap pertama penelitian ini adalah membuat sistem mekanik robot. Sistem mekanik yang dibuat meliputi *chassis*, sistem kemudi, serta penentuan posisi roda, dan sensor. Robot dibuat menggunakan *chassis* berbahan acrylic dengan dimensi panjang 22 cm dengan lebar 18 cm. Pemilihan bahan acrylic disebabkan acrylic berbobot ringan dan mudah untuk dipotong atau dibentuk sesuai keinginan. Hasil robot yang dibuat dapat dilihat pada Gambar 7.

Sistem kemudi menggunakan *differential drive* sehingga robot memiliki dua buah roda penggerak dan roda bebas. Roda yang digunakan memiliki diameter 65mm langsung dihubungkan ke poros motor. Hal ini dapat dilakukan karena motor dc yang digunakan telah dilengkapi dengan *gearbox*. Jarak roda dengan ujung depan robot cm 15 dan antar roda sejauh 20 cm. Kamera diletakan di bagian depan robot menghadap lintasan dengan sudut 45^{O}.

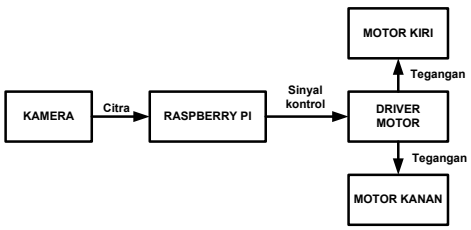
B. Perancangan Hardware

Robot yang dibuat memiliki beberapa bagian utama antara lain rangkaian regulator, raspberry pi, kamera, motor dc dan *driver* motor. Rangkaian regulator digunakan untuk menurunkan tegangan catu daya menjadi 5V. Regulator diperlukan karena raspberry pi dan kamera memerlukan tegangan 5V. Selain raspberry pi dan kamera, rangkaian *driver* motor juga memerlukan tegangan 5V untuk tegangan logika. Rangkaian regulator yang digunakan adalah regulator LM2576 yang mampu mensuplai arus hingga 3A. Blok diagram *hardware* penelitian ini ditunjukan pada Gambar 8.

C. Perancangan Fuzzy logic controller

Untuk menjalankan robot, digunakan sebuah sistem kendali. Sistem kendali yang digunakan adalah *fuzzy logic controller*. Pemilihan *fuzzy logic controller* ini dikarenakan tidak memerlukan model matematika dari robot. Selain itu *fuzzy logic controller* juga lebih sederhana dibandingkan sistem kendali cerdas lainya. Blok diagaram sistem kendali dapat dilihat pada Gambar 9.

Gambar 7. Robot yang telah dibuat dari bahan acrylic

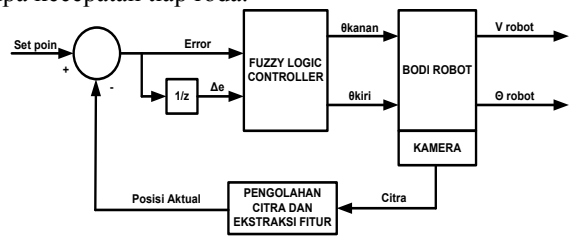


Gambar 8. Blok diagram *hardware*

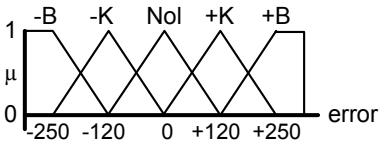
Hasil ekstraksi fitur merupakan pembacaan aktual sensor dan dibandingkan dengan set poin. Selisih pembacaan aktual sensor dan set poin adalah nilai *error*. Nilai *error* ini dijadikan masukan *fuzzy logic controller*. Keluaran *fuzzy logic controller* ini adalah nilai *duty cycle pulse width modulation* (PWM) yang digunakan untuk mengatur kecepatan motor. Putaran motor mengakibatkan posisi aktual robot akan berubah sehingga citra yang diambil kamera juga berbeda dengan *frame* sebelumnya.

Proses fuzzifikasi dalam *fuzzy logic controller* yang diterapkan menggunakan *input membership function* seperti yang ditunjukan pada Gambar 10 dan 11. Sedangkan *output membership function* yang digunakan ditunjukan pada Gambar 12. Dari *membership function* tersebut akan didapatkan nilai atau derajat keanggotaan. Masing-masing nilai keanggotaan tersebut kemudian dimasukan sebuah proses evaluasi *rule* yang nilainya ditunjukan Tabel I dan Tabel II. Setelah proses evaluasi *rule* dilakukan selanjutnya dilakukan proses defuzzifikasi menggunakan (2). Hasil proses ini adalah didapatkanya nilai *duty cycle* sinyal PWM yang diberikan ke rangkaian *driver* motor.

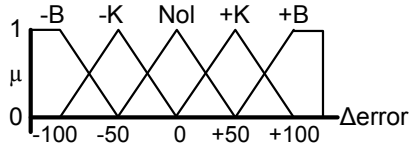
Masukan *fuzzy logic controller* berupa *error* yang terjadi antara set poin dengan pembacaan aktual. Selain nilai *error*, nilai selisih *error* aktual dengan *error* kondisi sebelumnya juga dijadikan sebagai masukan. Nilai ini disebut selisih *error* atau *delta error* (Δe). Kombinasi dari kedua masukan ini dinyatakan ke dalam bentuk evaluasi *rule* dengan keluaran berupa kecepatan tiap roda.



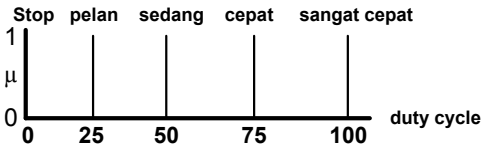
Gambar 9. Blok diagram *fuzzy logic controller*



Gambar 10. Fungsi keanggotaan *input error*



Gambar 11. Fungsi keanggotaan *output* Δ*error*



Gambar 12. Fungsi keanggotaan keluaran

D. Ekstraksi Fitur

Untuk mendapatkan pembacaan aktual posisi robot berdasarkan pembacaan kamera diperlukan proses ekstraksi fitur. Proses ini diawali dengan mengolah citra yang telah diambil oleh kamera. Citra pembacaan kamera memiliki format warna RGB. Citra RGB selanjutnya dikonversi menjadi *gryascale* untuk mempermudah proses ekstraksi fitur. Citra *grayscale* hasil konversi ditunjukan pada Gambar 13.

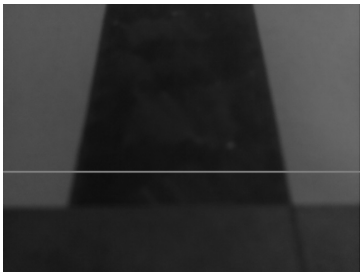
Citra yang didapatkan dikonversi menjadi citra *grayscale* kemudian melakukan proses *thresholding*. Proses *thresholding* dilakukan untuk membedakan garis lintasan dengan *background*. Tiap piksel citra *grayscale* yang memiliki intensitas di atas nilai *threshold* diubah menjadi warna putih dan jika dibawah nilai *threshold* maka diubah menjadi warna hitam. Hasil proses *thresholding* dapat dilihat pada Gambar 14.

TABEL I
EVALUASI *RULE* RODA KIRI

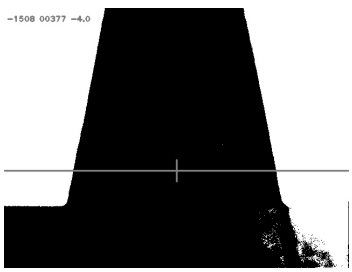
		Δerror				
		- Besar	**- Kecil**	**Nol**	**+ Kecil**	**+ Besar**
error	**- B**	sedang	sedang	cepat	cepat	cepat
	- K	sedang	sedang	cepat	cepat	cepat
	Nol	pelan	sedang	cepat	cepat	cepat
	+ K	pelan	pelan	sedang	sedang	cepat
	+ B	stop	pelan	pelan	pelan	sedang

TABEL II
EVALUASI *RULE* RODA KANAN

		Δerror				
		- Besar	**- Kecil**	**Nol**	**+ Kecil**	**+ Besar**
error	**- B**	sedang	pelan	pelan	pelan	stop
	- K	cepat	sedang	sedang	pelan	pelan
	Nol	cepat	cepat	cepat	sedang	pelan
	+ K	cepat	cepat	cepat	sedang	sedang
	+ B	cepat	cepat	cepat	sedang	sedang



Gambar 13. Citra *grayscale* hasil konversi.



Gambar 14. Citra hasil proses *thresholding*.

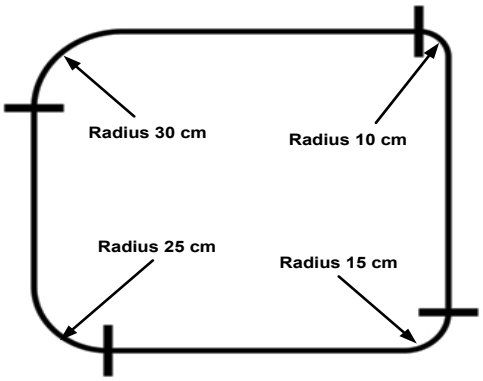
Piksel yang dibaca dan dijadikan sebagai data posisi aktual robot hanya yang berada sepanjang garis abu-abu horizontal pada Gambar 14. Sehingga jumlah piksel yang digunakan sebagai acuan hanya 640. Hal ini disebabkan resolusi citra yang diambil sebesar 640x480 piksel. Titik tengah acuan berada pada piksel di kolom 320 yang ditunjukan garis abu-abu vertikal pada Gambar 14. Piksel yang berada di sisi kiri garis vertikal memberikan posisi aktual negatif sedangkan sisi kanan memberikan posisi aktual positif. Dengan cara tersebut maka rentang pembacaan posisi aktual memiliki nilai antara -320 hingga +320.

V. HASIL DAN PEMBAHASAN

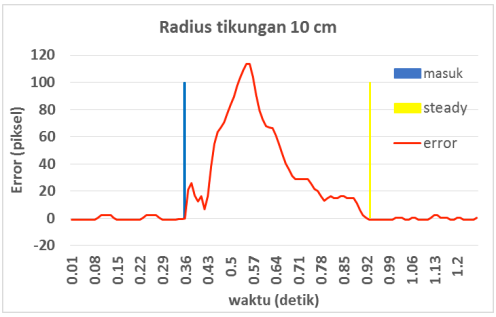
Pengujian dilakukan dengan menggunakan sebuah lintasan yang memiliki empat tikungan dengan radius yang berbeda. Radius ke empat tikungan masing-masing sebesar 10 cm, 15 cm, 25 cm, dan 30 cm. Lintasan yang digunakan ditunjukan pada Gambar 15. Garis melintang diletakan dalam lintasan tepat sebelum masuk tikungan. Garis ini berfungsi untuk memberikan informasi kepada robot bahwa telah memasuki tikungan. Sehingga saat robot mendeteksi garis ini, robot mengaktifkan *timer* dan menghitung waktu tempuh hingga kondisi *steady state.*

Pengambilan data dilakukan pada tiap tikungan. Data yang diambil dalam pengujian ini adalah *settling time* robot dalam melewati tikungan. *Settling time* dihitung setelah robot memasuki tikungan hingga menuju ke keadaan *steady state*. Selain *settling time*, data yang diamati adalah nilai *error* maksimal yang terjadi. Gambar 16 menunjukan performa robot saat melewati tikungan dengan radius 10 cm. Garis berwarna merah menunjukan nilai *error* yang terjadi. Garis vertikal warna biru menunjukan waktu robot saat masuk tikungan dan warna kuning menunjukan waktu robot saat mencapai *steady state*. Dari Gambar 16 *settling time* yang didapatkan adalah 0,56 detik.

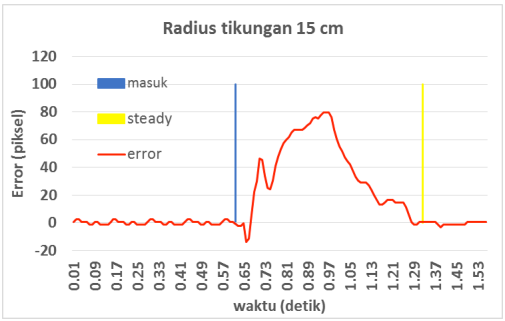
Pada tikungan dengan radius 15 cm, robot membutuhkan waktu 0,69 detik untuk menuju steady *state*. Nilai *error* terbesar yang terbaca oleh robot adalah 80. *Error* ini lebih rendah jika dibandingkan dengan radius 10 cm namun *settling time* yang dibutuhkan lebih lama. Performa robot saat melewati radius 15 cm, 25 cm, dan 30 cm masing-masing ditunjukan pada gambar 17, 18, dan 19.



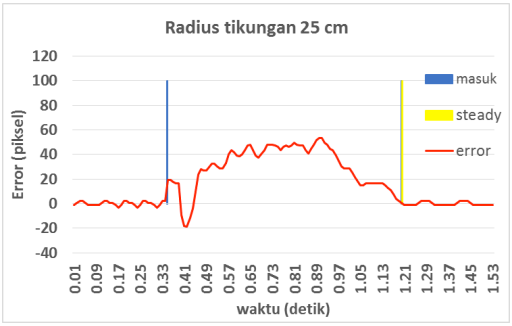
Gambar 15. Lintasan yang digunakan



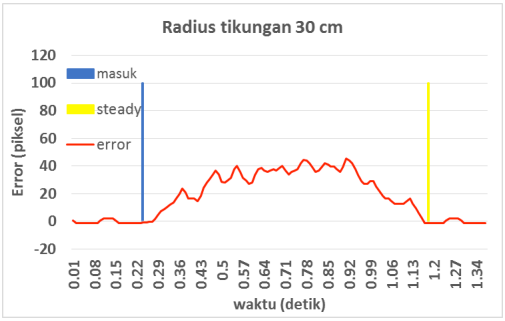
Gambar 16. Hasil pengujian dengan radius tikungan 10 cm.



Gambar 17. Hasil pengujian dengan radius tikungan 15 cm.



Gambar 18. Hasil pengujian dengan radius tikungan 25 cm.



Gambar 19. Hasil pengujian dengan radius tikungan 30 cm.

Pengujian yang dilakukan sebanyak 50 kali putaran. Pada setiap putaran robot membaca 4 jenis tikungan. Setelah melewati garis penanda masuk tikungan, robot mengitung waktu yang dibutuhkan hingga menuju *steady state*. Waktu tersebut kemudian disimpan dalam format *textfile* sehingga semua data dari putaran pertama hingga 50 bisa diamati. Dari semua data yang telah didapatkan, dihitung nilai rata-rata tiap tikungan dan rata-rata akhir. Nilai rata-rata yang didapatkan ditunjukan pada tabel III.

TABEL IIII
HASIL PENGUJIAN

No	Radius Tikungan (cm)	*Settling time* (detik)		Maksimal error (piksel)	
		Rata-rata	Std deviasi	Rata-rata	Std deviasi
1	10	0,60	0,026	116,91	3,65
2	15	0,67	0,014	84,40	6,05
3	25	0,82	0,033	54,36	5,89
4	30	0,86	0,036	43,14	43,14
Rata-rata pada seluruh tikungan		**0,74**	**0,126**	**74,70**	**29,34**

VI. KESIMPULAN (PENUTUP)

Dalam pengujian yang telah dilakukan dalam penelitian ini, robot mampu melewati tikungan dengan baik. Semakin besar radius tikungan *error* yang terjadi cenderung lebih kecil namun waktu menuju *steady state* lebih lama. Sebaliknya, semakin kecil radius tikungan maka *error* yang terjadi semakin besar namun *settling time* lebih cepat. Rata-rata waktu yang dibutuhkan untuk *steady state* adalah 0,74 detik dengan *error* rata-rata 74,7.

REFERENSI

[1] Kumar, K.Kishore et al, "*Design of Automatic Guided Vehicles*", international Journal of Mechanical Engineering and Technology (IJMET) Volume 3, Issue 1, January- April , pp. 24-32, 2012.

[2] Romadhon, A.S, dan Fuad, Muhammad, "*Perancangan Sistem Kontrol Gerakan Pada Robot Line Tracer*", Jurnal Ilmiah Mikrotek Vol.1, No.1 pp. 53-58, 2013.

[3] Janis, D.A.N et al, "*Rancang Bangun Robot Pengantar Makanan Line Follower*", e-journal Teknik Elektro dan Komputer UNSRAT pp.1-10 2014.

[4] Rashid, M.Z.A. et al, "*Metal Line Detection: A New Sensory System For Line Following Mobile Robot*",Journal of Theoretical and Applied Information Technology Vol.64 No.3 pp.756-764, 2014.

[5] Karim, Muhammad. et al, "*Robot Line Follower Berbasis Raspberry Pi Dengan Sensor Kamera*", jurnal skripsi dan tugas akhir STMIK GIMDP, 2015.

[6] INK Wardana, et al, "*Laptop-Based robot Sebagai Pramusaji Restoran Dengan Menerapkan Metode Pengolahan Citra Dan Kontrol Fuzzy*", Proceedings Seminar Nasional Teknik Elektro (FORTEI 2016) Departemen Teknik Elektro Undip, 2016.

[7] Uzer, MS and Yilmaz, N.,"*A real-time object tracking by using fuzzy controller for vision-based mobile robot*", Scientific Research and Essays Vol. 6(22), pp. 4808-4820, 7 October, 2011.

[8] Rizal, M.,Djuriatno,W.,Rif'an,M."*Implementasi Kamera OV7670 Sebagai Pendeteksi Garis Pada Robot Line Follower*", Jurnal Mahasiswa TEUB Vol 1, No 5 (2013).

[9] Gulalkari, A.V. et al, "*Kinect Camera Sensor-based Object Tracking and Following of Four wheel Independent Steering Automatic Guided Vehicle Using Kalman Filter*",International Conference on Control, Automation and Systems pp.1650-1655, 2015.

[10] Sen, P.K. et al, "*Solar Using Line Follower Robot with Robotic Arm*", International Journal of Novel Research in Electrical and Mechanical Engineering Vol. 3, Issue 1, pp: (35-44) 2016.

[11] RS Component, "*Raspberry pi 3 model B datasheet*", 2016.

[12] Kysan electonics, "*Datasheet Mabuchi RF-370CA-22170*", 2015.