Available at https://jtsiskom.undip.ac.id (5 May 2020)
DOI:10.14710/jtsiskom.8.3.2020.13643

Jurnal Teknologi dan Sistem Komputer, 8(3), 2020, 185-191

Navigasi robot bergerak berdasarkan landmark garis menggunakan kontroler Braitenberg dan pengolahan citra

Mobile robot navigation based on line landmarks using the Braitenberg controller and image processing

Ali Rizal Chaidir*), Gamma Aditya Rahardi, Khairul Anam

Program Studi Teknik Elektro, Fakultas Teknik, Universitas Jember
Jl. Kalimantan No. 37, Kampus Tegal Boto, Jember, Indonesia 68121

Cara sitasi: A. R. Chaidir, G. A. Rahardi, and K. Anam, "Navigasi robot bergerak berdasarkan landmark garis menggunakan kontroler Braitenberg dan pengolahan citra," *Jurnal Teknologi dan Sistem Komputer*, vol. 8, no. 3, pp. 185-191, 2020. doi: 10.14710/jtsiskom.2020.13643, [Online].

Abstract – Line following and lane tracking are robotic navigation techniques that use lines as a guide. The techniques can be applied to mobile robots in the industry. This research applied the Braitenberg controller and image processing to control and obtain line information around the mobile robot. The robot was implemented using Arduino Uno as a controller. A webcam was connected to a computer that performs image processing using canny edge detection and sends the data to the robot controller via serial communication. The robot can navigate on the side of the line, and the success rate of the system is 100 % at a turn of 135 ° and 80 % at a turn of 90 °.

Keywords *– mobile robot navigation; image processing; Braitenberg controller*

Abstrak – Pengikut garis dan pelacakan jalur adalah teknik navigasi robot yang menggunakan garis sebagai pemandunya. Teknik tersebut dapat diterapkan pada robot bergerak di industri. Penelitian ini bertujuan menerapkan kontroler Braitenberg dan pengolahan citra untuk mengendalikan dan mendapatkan informasi garis di sekitar robot. Implementasi robot menggunakan Arduino Uno sebagai kontroler. Sebuah webcam disambungkan ke komputer yang melakukan proses pengolahan citra menggunakan deteksi tepi canny dan mengirimkan datanya ke kontroler melalui komunikasi serial. Hasil yang diperoleh adalah robot mampu bernavigasi di samping garis, dan tingkat keberhasilan pengujian 100 % pada belokan 135 ° dan 80 % pada belokan 90 °.

Kata Kunci *– navigasi robot bergerak; pengolahan citra; kontroler Braitenberg*

I. PENDAHULUAN

Salah satu teknologi yang dapat membantu dan menggantikan beberapa tugas manusia yang sederhana adalah robot [1]. Robot dapat diterapkan untuk industri manufaktur, medis [2], dan bidang pertanian [3]. Di industri, robot digunakan untuk mempercepat proses produksi dan pemindahan barang. *Automated Guided Vehicle* (AGV) adalah robot atau kendaraan pemandu di industri. AGV memiliki peran penting karena lebih mudah digunakan dalam perubahan rencana jalur distribusi pengangkutan barang [4].

Robot dapat bergerak secara mandiri berdasarkan lingkungannya seperti [5] dan bergerak manual berdasarkan perintah dari operator seperti [6]. AGV adalah salah satu robot yang bergerak secara mandiri berdasarkan lingkungan di sekitarnya [4], [7]. Salah satu cara untuk mendapatkan informasi di sekitarnya adalah dengan menggunakan teknik pengindraan visual yang tidak membutuhkan banyak sensor [4], [8].

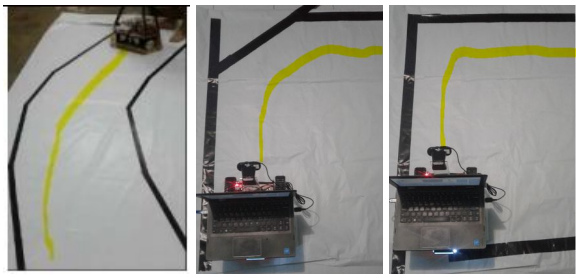
Beberapa teknik navigasi robot AGV telah diaplikasikan, yaitu navigasi berbasis perilaku (*behaviors*), petunjuk daerah (*landmark*), dan berbasis penglihatan (*vision*) [4]. Navigasi berbasis perilaku lebih cocok digunakan di lingkungan yang tidak terstruktur. Navigasi berbasis *landmark* membutuhkan sebuah tanda, misalnya garis. Navigasi berbasis penglihatan menggunakan kamera untuk mengetahui lingkungan di sekitarnya, misalnya warna dan jarak.

Beberapa kajian telah menggunakan garis sebagai pemandu robot, di antaranya [9], [10], yang mengembangkan robot yang dapat bernavigasi di antara dua garis (*lane tracking*) dan robot pengikut garis dalam [11]. Namun, algoritme yang digunakan oleh robot tersebut masih memiliki kelemahan, yaitu robot *lane tracking* hanya dapat berbelok jika garis pandunya memiliki belokan lebih besar dari 145 °, seperti yang ditunjukkan pada Gambar 1 [12]. Selain itu, untuk robot pengikut garis, garis harus ada di tengah robot seperti yang ditunjukkan pada Gambar 2 [7], [9].

Kajian bertujuan untuk menerapkan algoritme navigasi dengan memanfaatkan pengolahan citra dan kontroler Braitenberg untuk mengendalikan robot sehingga mampu bernavigasi di samping sebuah *landmark* garis dengan sudut belokan lebih kecil dari 145°. Selain itu, robot harus dapat bergerak di samping *landmark* sehingga dua robot yang melintasi *landmark*

*) Penulis korespondensi (Ali Rizal Chaidir)
Email: ali.rizal@unej.ac.id

Submitted: 20 January 2020; Revised: 22 April 2020; Accepted: 23 April 2020; Published: 31 July 2020



Gambar 1. Jalur-jalur robot yang digunakan untuk berbelok

tersebut tidak harus bergerak dengan arah dan kecepatan yang sama.

II. METODE PENELITIAN

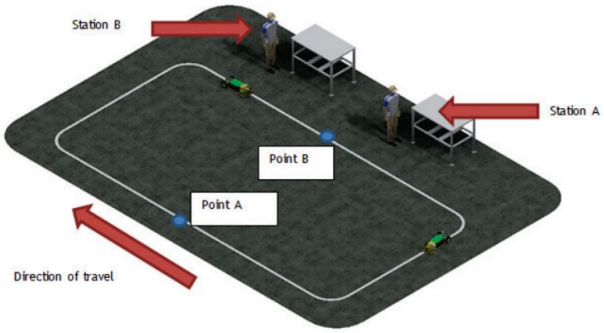
Blok sistem perangkat keras yang digunakan dalam kajian ditunjukkan pada Gambar 3. Algoritme navigasi atau alur kinerja perangkat lunak untuk mengatur gerak robot terdiri atas algoritme untuk pengolahan citra dan pengiriman data dari PC ke mikrokontroler serta dari mikrokontroler ke driver motor BTS7960B. Sebuah *webcam* digunakan untuk mengambil sebuah citra dan diproses di dalam sebuah komputer. Hasil dari pengolahan citra di komputer dikirimkan ke mikrokontroler menggunakan komunikasi serial. Pada penelitian ini, kontroler robot menggunakan Arduino Uno. Proses di Arduino Uno menghasilkan nilai digital dan PWM yang digunakan untuk mengendalikan motor DC melalui modul driver .

Tahapan kendali robot terdiri atas beberapa langkah untuk menghasilkan parameter citra yang digunakan untuk mengendalikan robot. Pertama adalah akuisisi citra dan melakukan proses perbaikan citra untuk menghilangkan derau pada citra. Citra tersebut diubah menjadi sebuah citra keabuan dan diperoleh parameter-parameter yang diperlukan. Paremeter-parameter tersebut dikirimkan ke Arduino melalui komunikasi serial, diolah, dan dikirimkan ke modul driver.

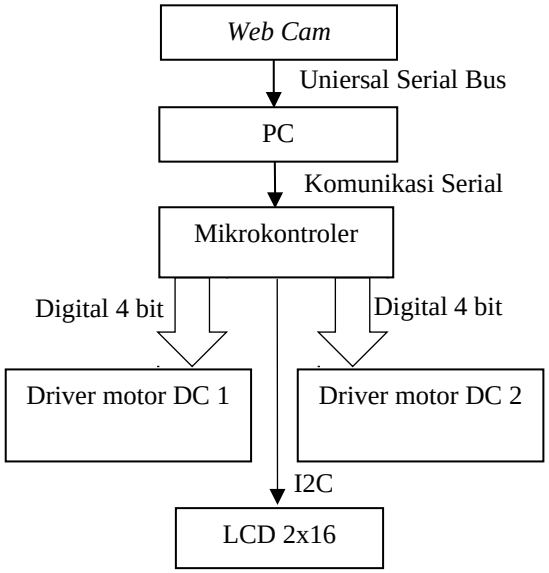
A. Robot beroda dan rangkaian elektronika

Robot yang digunakan untuk menguji algoritme dalam kajian ini adalah robot bergerak dengan penggerak roda tank (Gambar 4). Robot memiliki dimensi tinggi 19 cm, lebar 23 cm, dan panjang 30 cm. Ketinggian kamera adalah 33 cm dengan sudut 45°. Selain itu, robot memiliki rangkaian elektronika sebagai pemroses data dari hasil pengolahan citra dan pengendali dua motor DC.

Rangkaian elektronik untuk mendukung kinerja robot ditunjukkan pada Gambar 5. Komponen utamanya adalah Arduino Uno yang digunakan untuk menerima, memproses, dan mengirimkan data dari komputer ke kedua modul driver motor DC BTS7960B. Antarmuka komponen robot dan Arduino Uno tersebut ditunjukkan pada Tabel 1.



Gambar 2. Model jalur AGV

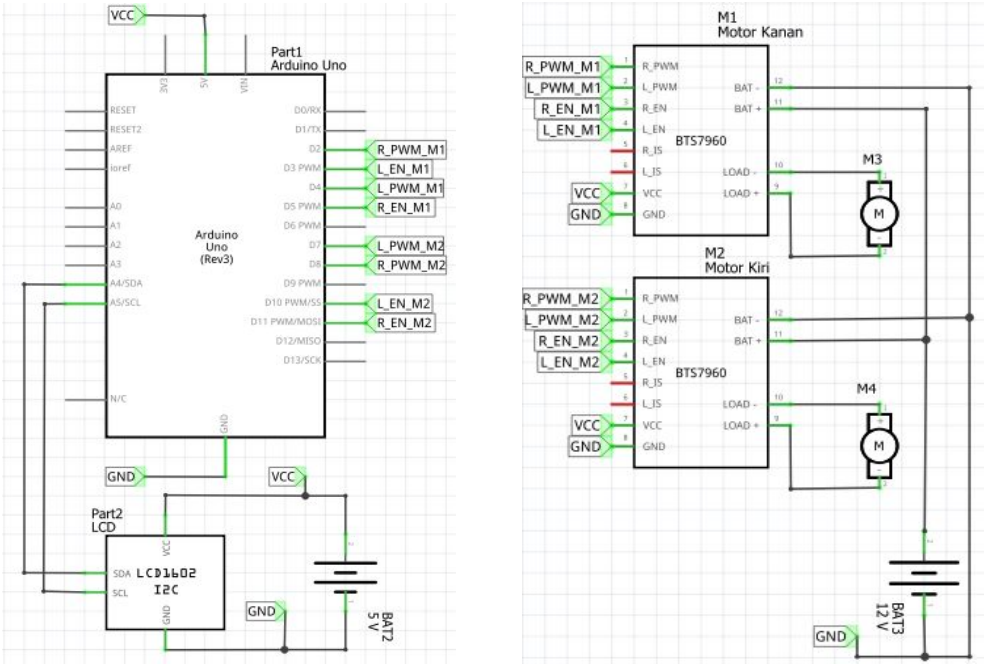


Gambar 3. Alur kinerja perangkat keras robot

Gambar 4. Robot beroda untuk menguji algoritme

B. Pengolahan citra

Robot bergerak berdasarkan informasi yang diterimanya melalui sebuah webcam. Informasi tersebut adalah berupa sebuah citra RGB dengan ukuran 320x240 piksel. Citra tersebut diolah untuk mendapatkan parameter-parameter yang diperlukan. Proses awal adalah melakukan perbaikan citra. Setiap citra dimungkinkan terdapat derau yang berbentuk



Gambar 5. Skematik rangkaian kendali robot

bintik-bintik. Derau tersebut dikurangi dengan menggunakan filter median seperti dinyatakan dalam Persamaan 1. Parameter *nbor[x,y]* menyatakan subcitra dari citra RGB dan I_{orig} *[i,j]* menyatakan sebuah citra RGB. Filter median ini berguna untuk menghilangkan bintik-bintik pada sebuah citra [13].

$$Y[x,y] = median(I_{orig}[i,j], i,j \in nbor[x,y]) \quad (1)$$

Proses selanjutnya adalah mendeteksi tepi citra menggunakan algoritme deteksi tepi. Deteksi tepi digunakan untuk mengenali dan mengelompokkan tepi citra, yaitu berdasarkan titik-titik piksel tepi yang terputus pada citra keabuan. Algoritme deteksi tepi memiliki titik tepi dengan pendekatan ambang batas sehingga nilai ambang memiliki pengaruh penting pada hasil deteksi tepi. Algoritme yang digunakan adalah deteksi tepi canny dari pustaka EmguCv OpenCV [14].

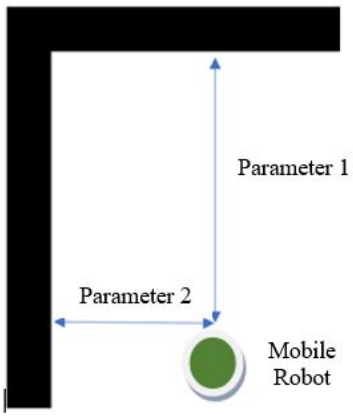
Robot menggunakan dua parameter berupa jumlah piksel untuk bernavigasi. Parameter pertama adalah jumlah piksel antara garis horizontal dan garis tepi depan robot, sedangkan parameter kedua adalah jumlah piksel antara garis vertikal citra dan garis tepi kiri yang tertangkap kamera. Ukuran citra yang digunakan adalah 320x240. Posisi garis tengah vertikal berada pada piksel ke-160 pada lebar piksel citra, sedangkan posisi garis horizontal berada pada 150 pada tinggi piksel citra. Ilustrasi citra dengan parameternya ditunjukkan pada Gambar 6. Proses untuk mendapatkan kedua parameter tersebut dinyatakan dalam Algoritme 1 dan Algoritme 2, yaitu berupa parameter *t_depan* dan *t_kiri*.

C. Pengendalian robot dengan Braitenberg

Setelah mendapatkan dua parameter tersebut, proses selanjutnya adalah pengendalian gerakan robot. Teknik pengendalian robot yang digunakan adalah kontroler Braitenberg. Kontroler ini memungkinkan untuk mengendalikan kecepatan motor DC kanan dan kiri atau

Tabel 1. Antarmuka komponen robot

Arduino Uno	Sinyal	Tipe sinyal	Antarmuka
SDA	SDA	Serial	I/O SDA LCD I2C
SCL	SCL	Serial	Keluaran SCL LCD I2C
D2	Digital	Digital (O)	R_PWM BTW7960_1
D3	PWM	Digital (O)	L_EN BTW7960_1
D4	Digital	Digital (O)	L_PWM BTW7960_1
D5	PWM	Digital (O)	R_EN BTW7960_1
D7	Digital	Digital (O)	L_PWM BTW7960_2
D8	Digital	Digital (O)	R_PWM BTW7960_2
D10	PWM	Digital (O)	L_EN BTW7960_2
D11	PWM	Digital (O)	R_EN BTW7960_2



Gambar 6. Parameter yang digunakan untuk menentukan gerakan robot

gerakan robot berdasarkan sensor kanan dan kiri [15], [16]. Parameter 2 (*t_kiri*) dalam kajian ini digunakan untuk mengendalikan kecepatan motor dc.

Kedua paramater yang telah diperoleh komputer dikirimkan ke Arduino. Data yang dikirim berjumlah tiga buah, yaitu nilai variabel *t_kiri*, *t_depan*, dan *t_kanan*, yang dipisahkan dengan penanda '#' seperti dinyatakan dalam Algoritme 3 dan Algoritme 4. Nilai *t_kanan* diperoleh dengan mengurangkan nilai 280 dengan nilai pada variabel *t_kiri*. Variabel *t_kiri* digunakan untuk menentukan kecepatan roda kanan pada robot, sedangkan *t_kanan* untuk menentukan kecepatan roda kiri robot. Variabel *t_depan* merupakan parameter 1 yang digunakan untuk menentukan kapan kontroler Braitenberg bekerja, yaitu ketika nilai pada variabel *t_depan* lebih besar dari 120 piksel.

Setiap variabel tersebut dikirim secara serial ke Arduino dengan dipisahkan menggunakan tanda "#". Algoritme 4 menunjukkan pembacaan data oleh Arduino yang dikirim secara serial dari komputer. Setiap tipe data yang diterima Arduino adalah karakter sehingga perlu dikonversi ke dalam bentuk tipe data *float* agar data yang diterima dapat digunakan untuk mengatur nilai PWM. Ketiga variabel tersebut disimpan di variabel larik *pwm_motor*. Nilai *t_depan* disimpan di variabel *pwm_motor[0]*, *t_kiri* disimpan di variabel *pwm_motor[1]*, dan *t_kanan* disimpan di variabel *pwm_motor[2]*.

Algoritme 5 digunakan untuk menentukan keputusan pergerakan robot. Jika jumlah piksel parameter 1 lebih kecil dari 120 atau menujukkan bahwa di depan robot terdapat sebuah garis, maka robot harus berbelok. Jika jumlah piksel parameter 1 lebih besar dari 120 atau garis masih jauh dari robot, maka robot bergerak sesuai dengan kecepatan setiap rodanya yang berdasarkan nilai variabel *t_kiri* dan *t_kanan*.

III. HASIL DAN PEMBAHASAN

Tahapan pengujian dalam kajian ini meliputi pengujian pergerakan, pengujian algoritme perangkat lunak, dan pengujian gerakan robot pada sebuah jalur.

Pengujian pergerakan robot dilakukan untuk mengamati pergerakan robot terhadap nilai masukan berupa nilai analog dan digital dari Arduino. Pergerakan robot yang dimaksud adalah maju, mundur, belok kanan dan kiri, serong kanan dan kiri, serta diam. Tabel 2 menunjukkan hasil dari pengujian pergerakan robot. Pergerakan robot diatur dengan cara memberikan nilai digital dan PWM. Pengaturan nilai digital mempengaruhi arah putaran roda robot sehingga nilai tersebut digunakan untuk memberikan pengaruh terhadap arah gerakan robot. Pengaturan gerakan robot berbelok serong menggunakan pin PWM.

Pengujian algoritme robot bertujuan untuk mengukur keberhasilan algoritme program yang diimplementasikan dalam robot. Algoritme dapat dikatakan siap digunakan jika dapat memberikan informasi parameter 1 dan 2 yang menunjukkan selisih jumlah piksel antara titik pusat citra yang digunakan

Algoritme 1. Proses untuk mendapatkan parameter 1 (*t_depan*)

```
1: for piksel dari koordinat (160,150) to (160,0)
2:     pindai dan hitung piksel
3:     if (piksel putih) then
4:         t_depan = hasil hitung piksel
5:     else
6:         t_depan = 150
7:     endif
8: endfor
9: return t_depan
```

Algoritme 2. Proses untuk mendapatkan parameter 2 (*t_kiri*)

```
1: for piksel dari koordinat (160,150) to (0,150)
2:     pindai dan hitung piksel
3:     if (piksel putih) then
4:         t_kiri = hasil hitung piksel
5:     else
6:         t_kiri = 160
7:     endif
8: endfor
9: return t_kiri
```

Algoritme 3. Pengiriman data ke Arduino

```
1: t_kiri = t_kiri
2: t_depan = t_depan
3: t_kanan = 280 – t_kiri
4: str1 = hasil konversi tipe data integer menjadi
   karakter pada variabel t_depan
5: str2 = hasil konversi tipe data integer menjadi
   karakter pada variabel t_kiri
6: str3 = hasil konversi tipe data integer menjadi
   karakter pada variabel t_kanan
7: TextBox4.Text = str1 + "#" + str2 + "#" + str3 + "#"
8: kirim data di TextBox4.Text secara serial
```

Algoritme 4. Penerimaan data dari komputer

```
1: byteread = data dari port serial
2: if (byteread == '#') then
3:     buff = hasil konversi tipe data karakter menjadi float
4:     pwm_motor [0] = data pertama
5:     pwm_motor [1] = data kedua
6:     pwm_motor [2] = data ketiga
7: else
8:     buff = byteRead
9: endif
```

Algoritme 5. Penentuan gerakan robot

```
1: if (pwm_motor [0] < 120) then // belok kanan
2:     D7 = High; D8 = Low; // EN L dan R motor 2
3:     D10 = 180; D11 = 180; // PWM L dan R motor 2
4:     D4 = High; D2 = Low;  // EN L dan R motor 1
5:     D3 = 180;  D5 = 180; // PWM L dan R motor 1
6:  else // kecepatan roda kanan = parameter 2, roda kiri = 280-
         parameter 2
7:     D7 = Low; D8 = High; // EN L dan R motor 2
8:     D10 = pwm_motor[1];
9:     D11 = pwm_motor[1]; // PWM L dan R motor 2
10:    D4 = High; D2 = Low;  // EN L dan R motor 1
11:    D3 = pwm_motor[2];
12:    D5 = pwm_motor[2]; // PWM L dan R motor 1
13: endif
```

dengan garis sebelah kiri dan depan robot. Pengujian dilakukan menggunakan sebuah webcam. Jalur yang digunakan untuk navigasi robot dalam kajian ini menggunakan sebuah garis warna hitam.

Hasil dari pengujian algoritme ini ditunjukkan dalam Tabel 3. Jumlah piksel antara garis horisontal dan garis tepi depan (parameter 1) atau nphd serta jumlah piksel antara garis vertikal citra dan garis tepi kiri (parameter 2) menunjukkan selisih jumlah piksel antara titik pusat yang digunakan terhadap garis hitam di sebelah kiri dan depan. Selain itu, ketika webcam pada robot tidak dapat menangkap citra garis, maka parameter 1 bernilai 149, dan parameter 2 bernilai 159. Kondisi tersebut sesuai dengan Algoritme 1 dan Algoritme 2, yaitu jika garis belum tertangkap kamera, maka nilai parameter 1 adalah 150 dan parameter 2 adalah 160.

Hasil tersebut juga menunjukkan kinerja penggunaan filter median dan deteksi tepi canny. Penggunaan teknik tersebut mempermudah pembuatan algoritme deteksi tepi [14] dan mengurangi derau yang dapat mengurangi kesalahan proses pembacaan parameter dengan waktu pemrosesan yang singkat [13]. Nilai parameter 1 menjadi sebuah acuan untuk mendapatkan informasi tentang jarak garis pandu di depan robot. Robot berbelok dengan putaran roda kanan dan kiri berlawanan arah jika nilai parameter 1 lebih kecil dari 120 (Algoritme 5). Jika parameter 1 bernilai lebih besar dari 120, maka kontroler Braitenberg bekerja. Gerakan robot berdasarkan kecepatan kedua motor DC sesuai nilai PWM setiap motor DC yang diterima. Nilai PWM tersebut berdasarkan nilai parameter 2.

Pengujian gerakan robot pada sebuah jalur dilakukan untuk mengamati respons robot terhadap lintasan yang telah disiapkan berdasarkan algoritme yang telah diterapkan pada robot. Tiga jenis lintasan digunakan untuk melakukan pengujian ini, yaitu lintasan dengan sudut belokan sebesar 135 °, lintasan dengan belokan 90 °, dan lintasan tertutup. Pengujian dilakukan sebanyak lima kali untuk setiap jenis sudut belokan. Pergerakan robot pada setiap pengujian direkam, diamati, dan dianalisis.

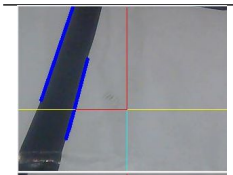
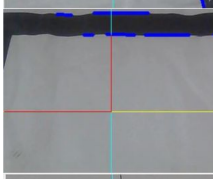
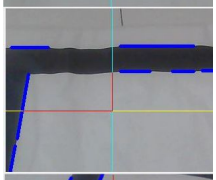
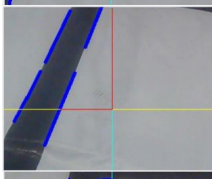
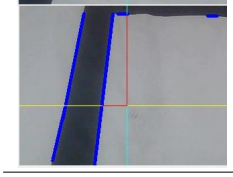
Gambar 7 menunjukkan hasil pengujian gerakan robot pada lintasan dengan belokan 135 °, sedangkan Gambar 8 hasil pergerakan pada lintasan dengan belokan 90 °. Garis kuning menunjukkan lintasan yang dilalui oleh robot. Robot mampu melakukan penjejakan garis di setiap pengujian pada belokan 135 °, sedangkan pada belokan 90 ° robot juga berhasil, kecuali di pengujian kedua yang gagal melakukan penjejakan garis dan mengenali garis di depannya sehingga robot melewati garis di depannya. Robot juga berhasil melakukan navigasi pada lintasan tertutup seperti ditunjukkan dalam Gambar 9. Hal tersebut disebabkan salah satunya oleh penggunaan parameter 1.

Secara umum, kontroler Braitenberg dapat bekerja dengan baik untuk mengikuti garis pandu yang ada di sampingnya dengan belokan lebih kecil dari 145 °, yaitu 90° dan 135°. Hasil tersebut menunjukkan kinerja yang sama dengan [15], [16] yang menggunakan kontroler

Tabel 2. Pengujian pergerakan robot

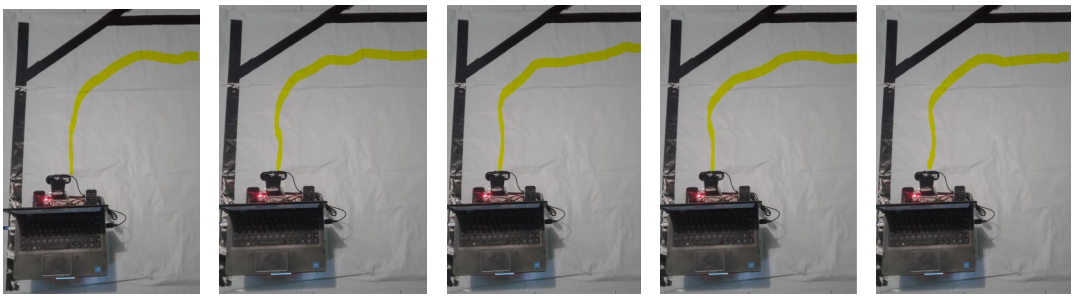
Pin Digital Arduino				Pin PWM Arduino				Gerakan Robot
2	4	7	8	3	5	10	11	
0	1	0	1	160	160	160	160	Maju
0	1	1	0	160	160	160	160	Belok Kanan
1	0	0	1	160	160	160	160	Belok Kiri
0	1	0	1	160	160	80	80	Serong kanan
0	1	0	1	80	80	160	160	Serong kiri
0	0	0	0	x	x	x	x	Diam

Tabel 3. Pengujian algoritme pendeteksi garis untuk navigasi robot

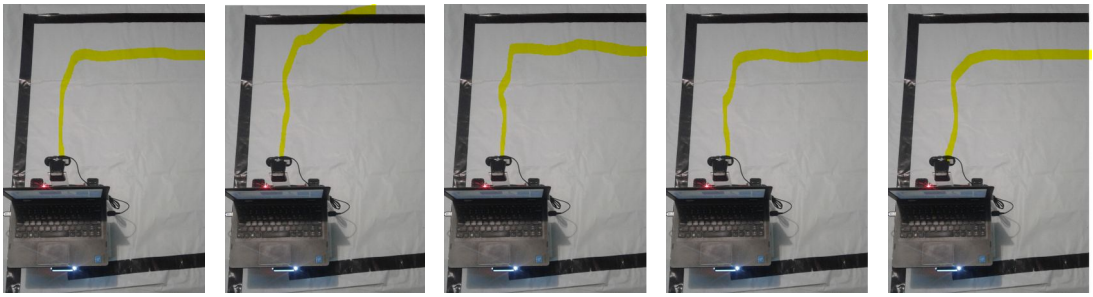
Citra	Jumlah piksel	
	Garis horisontal dan tepi depan	garis vertikal dan tepi kiri
	149	77
	149	159
	113	159
	57	135
	149	77
	138	37

Braitenberg. Hasil tersebut mengatasi permasalahan dalam [12] yang belum berhasil menjejaki jalur pemandu dengan belokan lebih kecil dari 145 ° dan dalam robot bernavigasi dengan garis pemandu berada di tengah robot [7].

Teknik navigasi robot pada kajian ini dapat digunakan tidak hanya pada robot beroda. Hal ini disebabkan penggunaan kontroler Braitenberg yang sederhana sehingga masukan berupa jumlah piksel dapat digunakan dengan mudah pada robot dengan garis



Gambar 7. Pengujian gerakan robot pada jalur dengan belokan sebesar 135 °



Gambar 8. Pengujian gerakan robot pada jalur dengan belokan sebesar 90 °

sebagai *landmark* pandunya. Penggunaan garis sebagai *landmark* pandu navigasi robot memudahkan operator atau pengguna untuk menentukan jalur robot yang harus dilalui sehingga dapat lebih mudah untuk menambah fungsi atau tugas tambahan dari robot untuk bidang lainnya, misalnya pertanian atau industri manufaktur.

IV. KESIMPULAN

Navigasi robot beroda berdasarkan *landmark* garis menggunakan kontroler Braitenberg dan pengolahan citra dengan deteksi tepi canny mampu mengikuti garis pandu dengan belokan lebih kecil dari 145 °, yaitu sudut belokan sebesar 135 ° dan 90 °, dan lintasan tertutup. Tingkat keberhasilan yang diperoleh adalah 100 % pada lintasan yang memiliki sudut belokan 135 ° dan 80 % pada lintasan yang memiliki sudut belokan 90 °.

Gambar 9. Pengujian gerakan robot pada lintasan tertutup

DAFTAR PUSTAKA

[1] B. Siciliano, L. Sciavicco, G. Oriolo, and L. Villani, *Robotics: modelling planning and control*. Springer, 2009. doi: 10.1007/978-1-84628-642-1

[2] O. Khatib, *Springer handbook of robotics.* Springer, 2008.

[3] M. U. Hassan, M. Ullah, and J. Iqbal, "Towards autonomy in agriculture: design and prototyping of a robotic vehicle with seed selector," in *2nd International Conference on Robotics and Artificial Intelligence*, Rawalpindi, Pakistan, Nov. 2016, pp. 37-44. doi: 10.1109/ICRAI.2016.7791225

[4] S. K. Das and M. K. Pasan, "Design and methodology of automated guided vehicle - a review," *IOSR Journal of Mechanical and Civil Engineering (IOSR-JMCE)*, Special Issue, pp. 29-33, 2016. doi: 10.9790/1684-15010030329-35

[5] W. Tushar and S. Pranav, "Design of lane detecting and following autonomous robot," *IOSR Journal of Computer Engineering (IOSRJCE)*, vol. 2, no. 2, pp. 45-48, 2012. doi: 10.9790/0661-0224548

[6] A. R. Chaidir, A. B. Satriya, and G. D. Kalandro, "Design of a gripping imitator robotic arm for taking an object," dalam *4th International Conference on Information and Communication Technology (ICoICT)*, Bandung, Indonesia, May 2016, pp. 1-5. doi: 10.1109/ICoICT.2016.7571940

[7] T. Ferreira and I. Gorlach, "Development of an automated guided vehicle controller using a model-based systems engineering approach," *South African Journal of Industrial Engineering*, vol. 27, no. 2, pp. 206-217, 2016. doi: 10.7166/27-2-1327

[8] H. Afrisal, "Metode pengenalan tempat secara visual berbasis fitur CNN untuk navigasi robot di dalam gedung," *Jurnal Teknologi dan Sistem Komputer*, vol. 7, no. 2, pp. 47-55, 2019. doi: 10.14710/jtsiskom.7.2.2019.47-55

[9] A. Kondakor, Z. Torcsvari, and A. Nagy, "A lane tracking algorithm based on image processing," in *12th International Symposium on Applied Computational Intelligence and Informatics (SACI)*, Timisoara, Romania, May 2018, pp. 39-44. doi: 10.1109/SACI.2018.8440975

[10] G. Ko, K.-H. Oh, and H.-S. Ahn, "Image-based lane tracking in quadcopter," in *2016 IEEE 25th International Symposium on Industrial Electronics (ISIE)*, Santa Clara, USA, Jun. 2016, pp. 387-392. doi: 10.1109/ISIE.2016.7744921

[11] L. A. P. Sánchez, C. A. A. Moreno, and H. A. B. Daza, "Design and construction of a line follower robot guided by pixels values of a camera connected to an FPGA," in *20th Symposium on Signal Processing, Images and Computer Vision (STSIVA)*, Bogota, Colombia, Sept. 2015, pp. 1-5. doi: 10.1109/STSIVA.2015.7330453

[12] A. R. Chaidir, K. Anam, and G. A. Rahardi, "Lane tracking pada robot beroda holonomic menggunakan pengolahan citra," *ELKOMIKA*, vol. 8, no. 1, pp. 69-79, 2020. doi: 10.26760/elkomika.v8i1.69

[13] E. Ramaraj and A. S. Rajan, "Median filter using open multiprocessing in agriculture," in *IEEE 10th International Conference on Signal Processing*, Beijing, Beijing, China, Oct. 2010, pp. 42-45. doi: 10.1109/ICOSP.2010.5656718

[14] Z. Xu, X. Baojie, and W. Guoxin, "Canny edge detection based on OpenCV," in *13th IEEE International Conference on Electronic Measurement & Instruments*, Yangzhou, China, Oct. 2018, pp. 53-56. doi: 10.1109/icemi.2017.8265710

[15] Y. Takei, Y. Shimizu, K. Hirasawa, and H. Nanto, "Braitenberg's vehicle-like odor plume tracking robot," in *IEEE SENSORS*, Valencia, Spain, Nov. 2014, pp. 1276-1279. doi: 10.1109/ICSENS.2014.6985243

[16] T. Salumäe, I. Rañó, O. Akanyeti, and M. Kruusmaa, "Against the flow: A Braitenberg controller for a fish robot," in *2012 IEEE International Conference on Robotics and Automation*, Saint Paul, USA, May 2012, pp. 4210-4215. doi: 10.1109/ICRA.2012.6225023