ELKOMIKA: Jurnal Teknik Energi Elektrik, Teknik Telekomunikasi, & Teknik Elektronika

ISSN(p): 2338-8323 | ISSN(e): 2459-9638 | Vol. 12 | No. 4 | Halaman 923 - 937
DOI : http://dx.doi.org/10.26760/elkomika.v12i4.923 Oktober 2024

Penerapan *Contour Detection* dengan Pengontrol PID pada Robot *Line Follower*

MIGUEL TANUJAYA, RAFAEL PATRICK, HANS SEBASTIAN, FAISAL WAHAB*

Universitas Katolik Parahyangan, Indonesia
Email : 6152001004@student.unpar.ac.id, faisal.wahab@unpar.ac.id*
*Penulis Korespondensi

Received 9 Agustus 2024 **|** *Revised* 27 September 2024 **|** *Accepted* 3 Oktober 2024

ABSTRAK

Sebuah robot line follower harus mampu mendeteksi garis dan bergerak mengikuti garis tersebut. Warna garis dan lantai yang tidak begitu kontras satu sama lain cukup sulit untuk dibedakan. Salah satu sensor yang dapat digunakan pada kasus tersebut adalah kamera. Dengan memanfaatkan pengolahan citra, robot dapat mengenali garis yang perlu diikuti. Untuk menghasilkan sistem line following yang konsisten dan rendah error, digunakan pengontrol PID yang bertujuan untuk menjaga posisi robot agar tetap pada lintasan yang diinginkan. Robot yang digunakan dalam penelitian ini adalah robot dengan empat roda mecanum. Perangkat keras yang digunakan meliputi kamera sebagai sensor utama, mikrokontroler Raspberry Pi 4 sebagai pusat pengolahan data dan pengendalian gerak motor. Hasil penelitian menunjukkan bahwa sistem line following yang dirancang memiliki rata-rata error posisi adalah 13,507%.

***Kata kunci**: line follower, pengolahan citra, contour detection, PID, mecanum*

ABSTRACT

A line-following robot must be able to detect a line and move along it. Lines and floors with low contrast are challenging to distinguish. One sensor that can be used in this case is a camera. By utilizing image processing, the robot can recognize the line it needs to follow. To produce a consistent and low-error line-following system, a PID controller is employed to keep the robot on the desired path. The robot used in this study is a four-wheeled mecanum robot. The hardware includes a camera as the primary sensor and a Raspberry Pi 4 microcontroller for data processing and motor control. The research results show that the designed line-following system has an average position error of 13.507%.

***Keywords**: line follower, image processing, contour detection, PID, mecanum*

1. PENDAHULUAN

Robot *line follower* adalah robot yang mampu bergerak dan bernavigasi dengan mengikuti garis bantu pada lantai. Sensor yang digunakan pada *line follower* pada umumnya adalah sensor *reflectance* yang menggunakan cahaya inframerah. Sensor inframerah bekerja dengan memberikan sinyal digital berdasarkan pantulan cahaya inframerah yang diterima sensor **(Shirmohammadi & Baghbani, 2024)**. Pada permukaan berwarna gelap seperti warna hitam, pantulan cahaya inframerah tidak diterima oleh sensor, sedangkan pada permukaan berwarna cerah pantulan dapat diterima. Bila warna lantai dan garis sama-sama cerah atau gelap maka sensor inframerah tidak akan bisa membedakan keduanya **(Khade, dkk, 2017)(Nugraha, dkk, 2015)**.

Untuk mengatasi hal tersebut, sensor kamera digunakan sebagai *input* bagi sistem *line following* **(Gomes, dkk, 2016)**. Pada penggunaan kamera, diperlukan pengolahan citra atau *image processing* supaya data dari kamera dapat diolah sehingga mendapatkan informasi berupa warna dan kontur dari lintasan robot *line follower*. Proses pengolahan citra akan memisahkan antara garis dan komponen selain garis. Metode yang umum digunakan adalah dengan melakukan *color thresholding,* namun tidak menggunakan model warna *Red Green Blue* (RGB) melainkan pada model warna *Hue Saturation Value* (HSV). Dengan mengubah model dari RGB menjadi HSV, gambar menjadi tidak rentan terhadap perubahan pencahayaan pada ruangan **(Deswal & Sharma, 2014)**. Garis yang telah dipisahkan dari objek lain pada gambar kemudian dicari posisi titik tengahnya. Kemudian selisih antara posisi tersebut dan titik acuan menjadi *error* posisi robot.

Supaya dapat meminimalisir nilai *error,* diperlukan sebuah kendali yang dapat menjaga nilai *error* tetap pada nol. Terdapat beberapa metode yang digunakan pada kendali robot *line follower*, diantara penggunaan logika Fuzzy pada robot line follower **(David, 2016)(Supriadi & Rizal, 2017)** dan penggunaan metode *Proporsional, Integral* dan *Derivatif* (PID) **(Miftahul, 2016)** dimana penggunaan PID dapat mengkoreksi nilai *error* dari hasil pengukuran sensor supaya keluaran yang dihasilkan sesuai dengan nilai yang diinginkan sehingga akan menghasilkan nilai *error* seminimal mungkin. Pada penelitian ini, akan digunakan metode PID sebagai penjejak garis pada robot *line follower*. dimana selisih *error*, akumulasi *error,* dan *error* akan menjadi penentu bagi kecepatan setiap motor **(Erbay, dkk, 2024)**. Secara prinsip dasar, cara kerja PID adalah semakin besar penyimpangan yang terjadi pada robot maka semakin besar juga usaha yang dilakukan robot untuk mengembalikan posisinya ke posisi tanpa *error.*

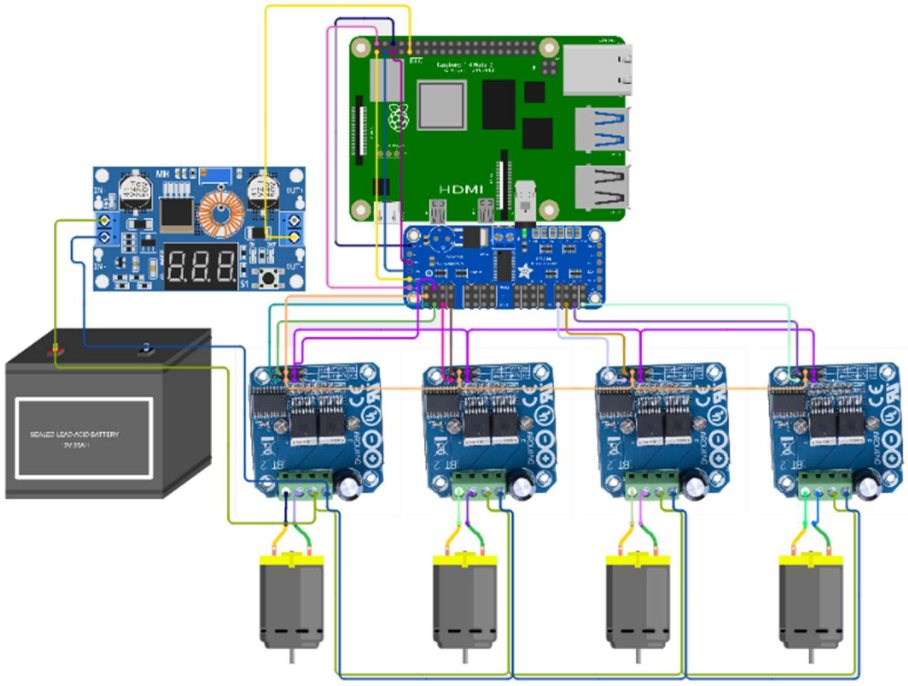
Jenis Roda yang digunakan pada robot *line follower* pada umumnya menggunakan roda differensial **(Latif, dkk, 2020)**, yang termasuk dalam kategori robot *non-holonomic* yaitu robot dimana robot diperlukan perubahan orientasi saat robot bergerak. Berbeda dengan robot kategori *holonomic*, dimana robot dapat bergerak ke segala arah tanpa perlu mengubah orientasi dari robotnya itu sendiri **(Manavaalan, dkk, 2023**). Salah satu jenis roda yang digunakan pada robot holonomic ialah Roda *mecanum* **(Adnan, dkk, 2020)**. Roda *mecanum* dirancang dengan dikelilingi *roller* yang membentuk sudut 45^{o}, sehingga arah dan kecepatan setiap roda akan menghasilkan resultas gaya. Untuk dapat mengontrol robot dengan roda *mecanum* dan supaya dapat bergerak sesuai dengan keinginan, robot dikendalikan berdasarkan kinematika robot **(Fahmizal, dkk, 2022)**. Dengan pengimplementasian roda *mecanum* para robot *line follower*, maka pegerakan robot *line follower* dapat lebih cepat dan efisien. Pada penelitian ini akan diterapkan sebuah kamera sebagai pendeteksi garis pada robot *line follower* menggunakan pengolahan citra metode contour detection. Selain itu,pada robot ini akan diterapkan roda *mecanum*.

Penelitian ini bertujuan untuk mengembangkan robot line follower berbasis kamera yang dapat mendeteksi dan mengikuti garis secara efektif melalui pengolahan citra menggunakan metode *contour detection*, serta mengaplikasikan roda mecanum agar pergerakan robot menjadi lebih fleksibel dan efisien. Selain itu, implementasi kendali PID diharapkan dapat meminimalkan *error* pada posisi robot sehingga robot dapat mengikuti lintasan dengan akurat. Ruang lingkup penelitian ini mencakup penggunaan kamera sebagai sensor utama untuk pendeteksi garis dengan metode *contour detection*, penerapan kendali PID, dan penggunaan roda mecanum untuk dapat mengikuti garis sambil mempertahankan orientasi.

2. METODE

2.1 Desain Sistem

Robot memiliki sebuah komputer yaitu Raspberry Pi yang berguna untuk menerima gambar, mengolah gambar, dan mengontrol motor. *Input* sistem berupa sebuah kamera yang berguna untuk menerima gambar lapangan. *Output* sistem adalah berupa gerakan motor yang dikontrol melalui motor driver dengan sinyal PWM yang berasal dari Raspberry Pi. Rangkaian elektrik robot dapat dilihat pada Gambar 1. Selain itu juga terdapat kamera yang terhubung dengan Raspberry Pi.



Gambar 1. Rangkaian sistem elektrik robot *line follower*.

Untuk menggabungkan seluruh komponen menjadi sebuah robot yang fungsional, diperlukan sebuah program yang mampu mendukung kebutuhan tersebut. Terdapat 3 buah blok utama pemrograman pada sistem ini: pengolahan gambar, pengendalian motor, dan penerapan PID. Pengolahan gambar dilakukan dengan gambar yang diperoleh melalui kamera Raspberry Pi.

2.2. Langkah Penelitian

Penelitian ini dilakukan melalui beberapa tahapan, yang dapat dijabarkan sebagai berikut:

1. **Pengumpulan Data**: Pengambilan data awal dilakukan dengan pengujian robot pada lintasan line follower sederhana berupa lintasan lurus, berbelok, dan persimpangan.
2. **Pengembangan Algoritma Pengolahan Citra**: Algoritma pengolahan citra dikembangkan untuk mendeteksi garis bantu dan persimpangan dengan metode color thresholding dan contour detection pada gambar dari kamera.
3. **Implementasi Kendali PID**: Parameter kendali PID dioptimalkan untuk menjaga posisi robot agar mengikuti garis dengan error yang minimal. Pengujian dilakukan dengan beberapa variasi parameter (*trial* and *error*), dan hasil *error* dibandingkan untuk mendapatkan konfigurasi terbaik.
4. **Pengujian dan Evaluasi Sistem**: Robot diuji pada lintasan nyata dengan berbagai konfigurasi untuk mengukur efektivitas kendali dan pengolahan citra, serta mendokumentasikan nilai *error* rata-rata yang dihasilkan.

2.3 Teknik *Image Processing*

Dalam perancangan sistem, pertama-tama perlu ditentukan daerah *masking* untuk mengurangi waktu komputasi dan menghindari gangguan objek yang menyebabkan kesalahan pendeteksian garis. Kemudian, gambar diubah dari gambar RGB ke gambar HSV, dimana dalam prosesnya gambar yang tadinya menunjukkan komposisi warna merah, hijau, dan biru, berubah menjadi representasi warna berdasarkan rona, saturasi, dan nilai kecerahan. Konversi tersebut memudahkan pendeteksian warna tertentu, seperti garis putih. Hal tersebut karena HSV memudahkan isolasi warna tertentu karena komponen warna dan kecerahan telah dipisahkan **(Deswal & Sharma, 2014)**.

Teknik *masking* merupakan teknik pengolahan citra untuk mengisolasi daerah tertentu yang ingin diproses. Setelah diisolasi, informasi yang ada dalam daerah isolasi akan diekstrak sementara semua informasi yang diluar daerah isolasi akan dihilangkan. Pada umumnya teknik *masking* digunakan untuk memanipulasi citra. Teknik *masking* digunakan dalam pengolahan citra agar mengurangi kemungkinan kesalahan dalam pendeteksian garis berwarna putih **(Risah Prayogi, dkk, 2019)**.

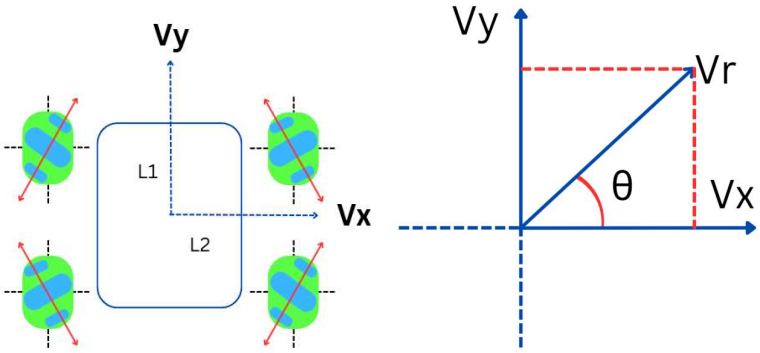
Color thresholding digunakan untuk mengekstrak jangkauan warna yang diinginkan. Dengan *color thresholding*, citra diolah menjadi gambar biner yang pikselnya berwarna putih atau hitam. Setelah melakukan *colour thresholding*, dilakukan pendeteksian kontur untuk menggambarkan garis kontur pada hasil *colour thresholding* **(Soans, dkk, 2017)**.

2.4 Kinematika Robot *Mecanum*

Kinematika robot *mecanum* berguna untuk mengetahui relasi antara arah gerak robot dan gerak masing-masing roda. Robot *mecanum* dapat bergerak ke segala arah dalam berbagai orientasi. Hal tersebut dikarenakan rodanya yang berbentuk *roller* dengan sudut orientasi 45 derajat. Untuk gambaran roda *mecanum* dapat dilihat pada Gambar 2.

Gambar 2. Roda *mecanum*.

Pergerakan robot *mecanum* ditentukan oleh kombinasi kecepatan dan arah gerak masing-masing motor. Gaya pada masing-masing roda *mecanum* akan digabung menjadi hasil resultan gaya robot. Kecepatan putar pada masing-masing motor juga mempengaruhi hasil resultan gaya.



(a) **(b)**

Gambar 3. Konfigurasi masing-masing motor dan resultan gaya robot *mecanum*

Karena robot *mecanum* dikategorikan sebagai robot *holonomic*. Kerangka Koordinat Lokal (KKL) perlu ditentukan agar arah gerak dan persamaan kinematika dapat didefinisikan. KKL berada pada robot *mecanum*, dimana titik tengah KKL berada pada titik tengah robot seperti pada Gambar 3. Persamaan kinematika dapat dilihat pada Persamaan (1) untuk kinematika maju (*forward Kinematics*) dan Persamaan (2) untuk kinematika terbalik (*inverse kinematics)* **(Fahmizal, dkk, 2022)**.

$$\begin{bmatrix} v_x \\ v_y \\ \omega \end{bmatrix} = \frac{R}{4}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & -1 & 1 \\ -\frac{1}{(L_1+L_2)} & \frac{1}{(L_1+L_2)} & -\frac{1}{(L_1+L_2)} & \frac{1}{(L_1+L_2)} \end{bmatrix}\begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \end{bmatrix} \quad (1)$$

$$\begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{bmatrix} = \begin{bmatrix} R\omega_1 \\ R\omega_2 \\ R\omega_3 \\ R\omega_4 \end{bmatrix} = \begin{bmatrix} -1 & 1 & (L_1+L_2) \\ 1 & 1 & -(L_1+L_2) \\ -1 & 1 & -(L_1+L_2) \\ 1 & 1 & (L_1+L_2) \end{bmatrix}\begin{bmatrix} v_x \\ v_y \\ \omega \end{bmatrix} \quad (2)$$

Dari persamaan kinematika terdapat parameter R yang merupakan jari-jari roda *mecanum*, L_1 merupakan jarak vertikal antara pusat badan robot dan sumbu roda *mecanum*, L_2 merupakan jarak horizontal antara pusat badan robot dan sumbu roda *mecanum*, serta ω_i merupakan kecepatan sudut pada motor ke$-i$ $(i = 1,2,3,4)$.

2.5 Kamera

Sensor inframerah pada umumnya membedakan antara gelap dan terang, maka pada umumnya digunakan garis dan alas yang berwarna berkebalikan yaitu hitam dan putih. Pada arena yang perlu dilintasi oleh robot memiliki garis bantu berwarna putih dengan latar belakang berbagai warna sehingga diperlukan sensor lain untuk pendeteksian garis. Salah satu sensor yang dapat menjadi solusi adalah sensor kamera. Kamera mampu menangkap informasi mengenai warna dari suatu area dan menyimpan nilai warna tersebut untuk setiap piksel. Piksel dalam suatu kamera itu sendiri merupakan unit terkecil dari gambar yang menyimpan informasi tentang warna dan intensitas cahaya. Besarnya area yang tercakup dalam suatu piksel berkaitan dengan posisi dan jarak kamera pada suatu area. Pada sistem ini kamera akan digunakan untuk menentukan besarnya penyimpangan robot terhadap garis. Jenis kamera yang digunakan adalah Raspberry pi kamera 5 MP yang dapat langsung dihubungkan pada Raspberry Pi.

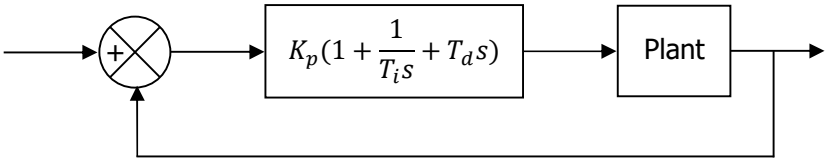
2.6 Raspberry Pi

Raspberry Pi merupakan sejenis komputer tunggal kecil yang digunakan untuk melakukan komputasi. Komputasi yang dimaksud khususnya pemrosesan masukan sinyal dari sensor kamera menjadi keluaran gerakan robot roda *mecanum* untuk *line following*. Raspberry pi tergantung tipe dan spesifikasi memiliki *port* Micro USB untuk catu daya, beberapa *port* USB, puluhan pin GPIO, CSI *camera port* untuk Raspberry Pi *camera*, DSI *display port* untuk Raspberry Pi *touchscreen display*, komunikasi *Bluetooth Low Energy* (BLE) dan *wireless* LAN serta *port* Micro SD untuk penyimpanan data dan sistem operasi yang terlihat pada gambar 4.

Gambar 4. Gambar RasPberry Pi 4 Model B

2.7 PID

PID (*Proportional-Integrative-Derivative*) merupakan jenis pengontrol yang banyak digunakan dalam sistem kendali otomatis. Pengontrol PID memanfaatkan umpan dari sistem yang diatur untuk menghitung sinyal kendali yang dibutuhkan agar mempertahankan sistem pada titik *setpoint* yang diinginkan. PID berfungsi untuk menentukan seberapa besar kompensasi yang dilakukan robot untuk mengembalikan posisi ke garis acuan. Untuk ilustrasi kendali diagram blok PID dapat dilihat pada Gambar 5.



Gambar 5. Diagram Blok PID

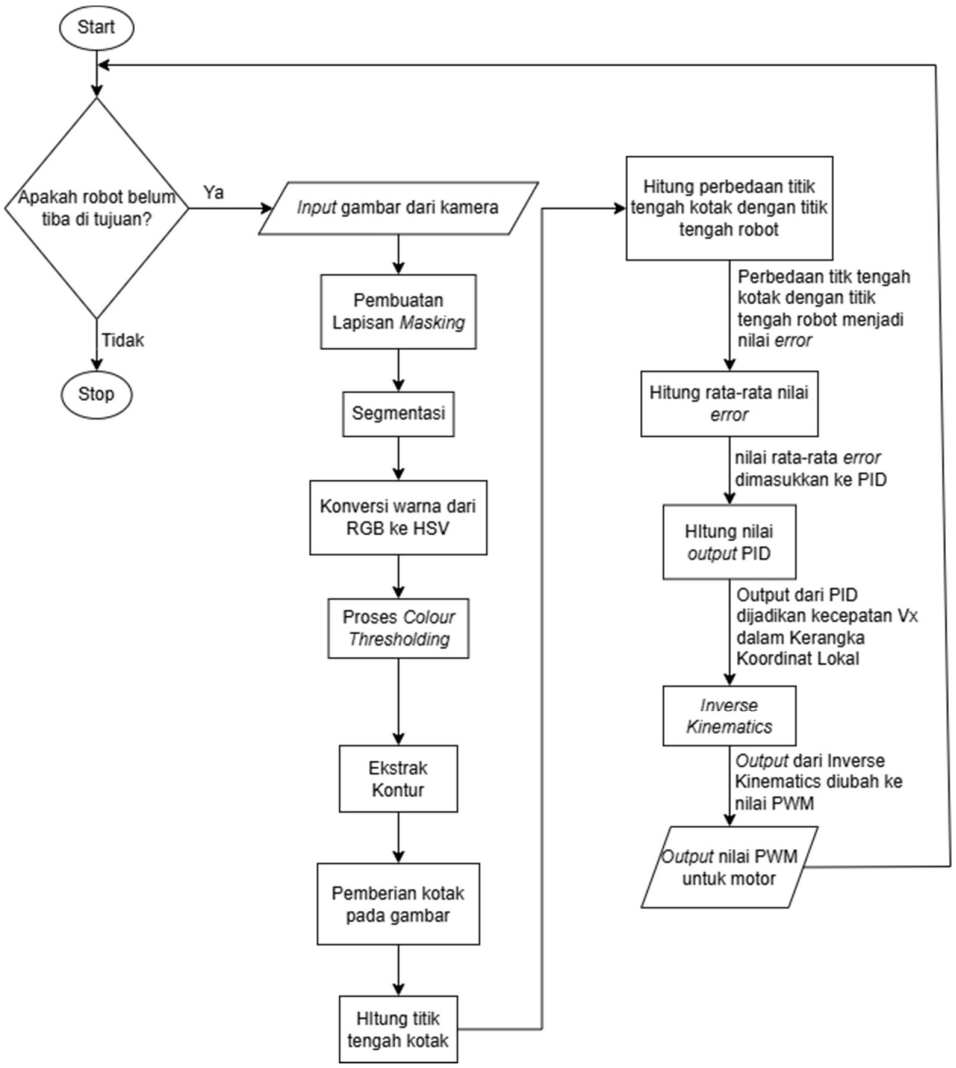
Untuk pengontrol PID dalam waktu diskrit dapat dituliskan pada Persamaan (3) :

$$K_p \cdot e[n] + K_i \cdot T_s \cdot \sum_{i=0}^{n} e[i] + K_d \cdot \frac{e[n] - e[n-1]}{T_s} \quad (3)$$

Dimana K_p merupakan konstanta proporsional, K_i merupakan konstanta integral, K_d merupakan konstanta derivatif, dan T_s merupakan *Time Sampling*. Komponen $K_p \cdot e[n]$ merupakan komponen proporsional yang nilai keluarannya sebanding dengan nilai *error*. Komponen $K_i \cdot T_s \cdot \sum_{i=0}^{n} e[i]$ merupakan komponen integral yang nilai keluarannya berupa akumulasi nilai *error*. Komponen $K_d \cdot \frac{e[n]-e[n-1]}{T_s}$ merupakan komponen derivatif yang nilai keluarannya berupa perbedaan nilai *error* dalam setiap waktu **(Ogata, 2010)**.

2.8 Diagram Alir sistem

Gambar yang diperoleh melalui kamera diolah menjadi nilai *error* melalui proses yang dapat dilihat pada diagram alir pada Gambar 6.

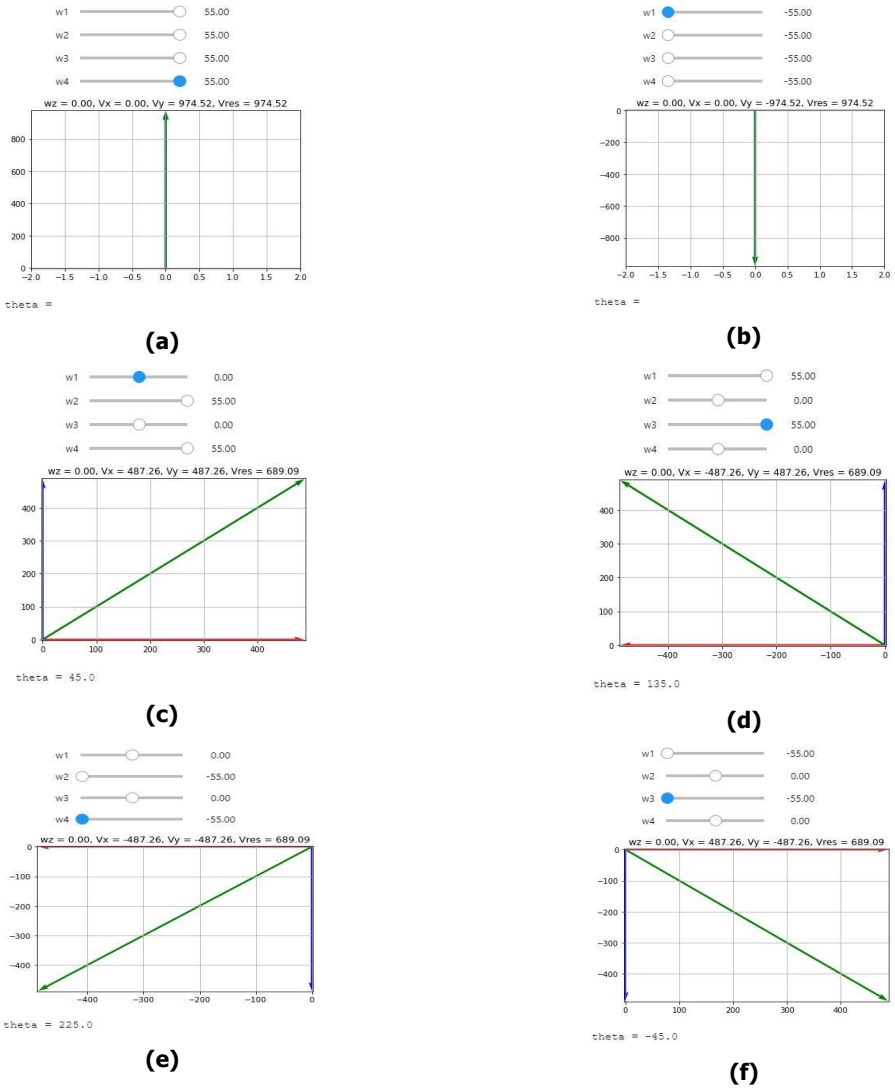


Gambar 6. Diagram Alir Sistem

3. HASIL DAN PEMBAHASAN

3.1 Pergerakan Dasar Robot *Mecanum*

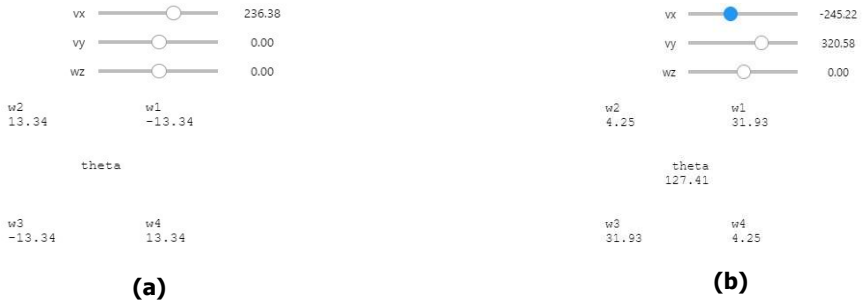
Pergerakan dasar robot *mecanum* akan disimulasikan menggunakan bahasa pemrograman Python pada Jupyter Notebook untuk menggambarkan pergerakan dan kinematika robot *mecanum*. Pada program Python *Forward Kinematic* untuk menggambarkan arah serta resultan vektor berdasarkan kecepatan motor masing-masing pada robot *mecanum*. Motor masing-masing dikonfigurasi agar mendapatkan hasil yang sesuai. Konfigurasi motor sesuai dengan dasar teori pada Gambar 7.



Gambar 7. Hasil output program python untuk *forward kinematic*

Hasil *output* pada Gambar 7 terdapat arah gerak secara vertikal, horizontal, arah resultan gerak, serta sudut theta dari sumbu x positif pada Kerangka Koordinat Lokal. *Input* dimasukan pada setiap motor dengan satuan rotasi per detik. Untuk hasil gerak wz merupakan arah putar pada robot *mecanum* dengan nilai wz positif ketika robot berputar berlawanan arah jarum jam, dan nilai wz bernilai negatif ketika robot berputar searah jarum jam.

Program Python pada *Inverse Kinematic* menggambarkan hasil dari kinematika mundur. Dengan menggunakan persamaan kinematika mundur pada Persamaan (2) yang telah dimasukan ke dalam program Python, didapatkan hasil *output* pada Gambar 8.

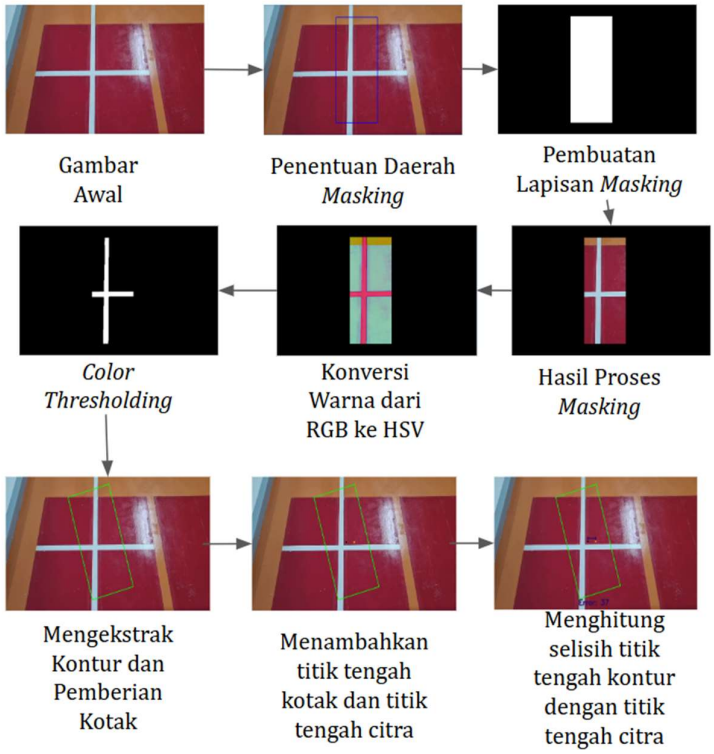


Gambar 8. Hasil output program python untuk *inverse kinematic*

Hasil *output* pada Gambar 8 terdapat kecepatan pada masing-masing motor dengan satuan rotasi per detik, serta sudut theta dari sumbu x positif pada Kerangka Koordinat Lokal. *Input* yang dimasukan berupa kecepatan vx, vy, dan wz. Di mana vx merupakan kecepatan horizontal, vy merupakan kecepatan vertikal, serta wz merupakan arah putar pada Kerangka Koordinat Lokal. Nilai vx dan vy sesuai dengan sistem koordinat kartesian dengan aturan tangan kanan. Nilai wz positif ketika arah putar robot berlawanan arah jarum jam, dan nilai wz negatif ketika arah putar robot searah jarum jam.

3.2 Pengolahan Citra pada Robot *Line Follower*

Robot harus mengikuti garis di lantai, sehingga dibuat algoritma *line follower* agar robot tersebut tidak menyimpang dari jalur yang telah ditetapkan. Robot menggunakan kamera sebagai input dari posisi robot terhadap garis. Gambar yang diperoleh dari kamera diproses dengan langkah-langkah seperti pada Gambar 8.



Gambar 9. Diagram alir proses *line follower*

Proses pertama adalah *masking*, yang bertujuan memperkecil area gambar yang diproses untuk mengurangi waktu komputasi dan menghindari gangguan dari objek lain berwarna sama (putih) seperti lantai, pantulan lampu pada lantai, dan dinding. Kemudian gambar diubah dari model RGB ke mode gambar HSV, dimana dengan proses ini gambar yang tadinya menunjukkan komposisi warna merah, hijau, dan biru, berubah menjadi representasi warna berdasarkan rona, saturasi, dan nilai kecerahan. Konversi ini memudahkan pendeteksian warna tertentu, seperti garis putih. Dapat dilihat pada Gambar 8 dimana garis yang tadinya berwarna putih berubah menjadi warna merah. Sedangkan warna merah berubah menjadi warna hijau. Setelah warna putih dipertegas dengan mengubah gambar menjadi model HSV, gambar diubah menjadi hitam putih. Pada gambar hitam putih tersebut, warna putih merepresentasikan warna putih pada *frame* gambar, sedangkan warna hitam adalah warna lain selain warna putih. Kemudian dari gambar hitam putih, dilakukan ekstrak kontur dari piksel-piksel putih pada gambar. Algoritma ekstrak kontur bekerja dengan menghasilkan persegi dengan luas terkecil yang dapat mencakup seluruh piksel putih pada *frame* gambar. Titik pusat dari persegi tersebut menjadi titik yang menentukan posisi robot terhadap garis.

Setelah titik pusat dari kontur diperoleh, ditentukan titik pusat atau *offset* yang menjadi acuan bagi robot untuk mengikuti garis. Dengan titik pusat kontur dan titik acuan yang telah diperoleh, dapat ditentukan besar simpangan dari robot atau *error* yang kemudian dapat dimasukkan ke dalam sistem kendali robot. *Error* berupa selisih posisi titik pusat kontur dan titik tengah acuan pada sumbu x robot.

3.3 Pendeteksian Persimpangan Garis

Pada area yang dilewati terdapat persimpangan yang dapat memengaruhi fungsional *line follower* pada robot. Untuk dapat menuju ke titik akhir yang telah ditentukan robot memerlukan kemampuan untuk memilih jalur yang harus diambil pada saat dihadapkan dengan persimpangan. Untuk melakukan proses ini, dilakukan dengan melihat perubahan lebar dari kontur garis. Dapat dilihat pada Gambar 9, bahwa terdapat perbedaan lebar kontur yang diekstraksi. Perubahan lebar kontur ini digunakan untuk menentukan apakah robot berada pada persimpangan atau tidak. Besarnya perubahan yang menentukan persimpangan tersebut berkaitan dengan lebar area *masking*.

(a) **(b)**

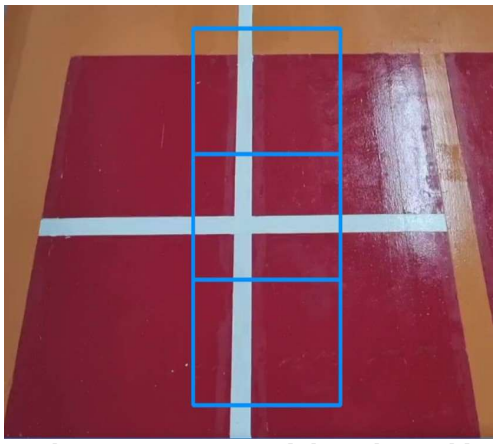
Gambar 10. (a) Kamera mendeteksi persimpangan dan (b) Kamera tidak mendeteksi persimpangan.

Saat pendeteksian persimpangan dilakukan dengan *frame rate* yang tinggi, perubahan lebar kontur terlalu bertahap. Padahal persimpangan ditentukan dari perubahan lebar kontur yang mendadak. Akibatnya ada kemungkinan dimana persimpangan tidak terdeteksi. Untuk

mengatasi hal ini, dapat dibuat jeda antar setiap iterasi pada looping program. Pada robot ini digunakan jeda sebesar 50 milidetik sehingga *sampling time* menjadi 180 milidetik.

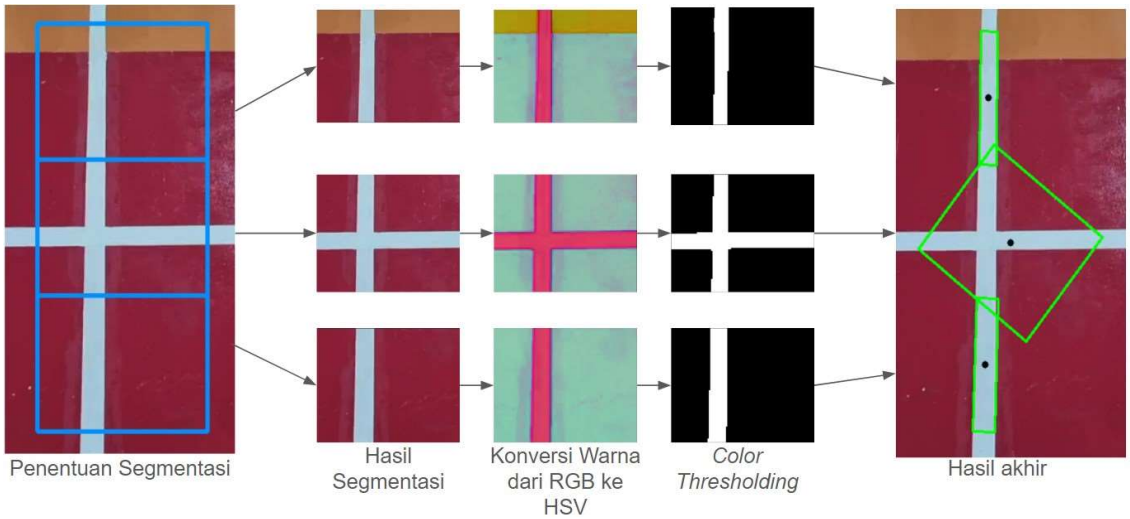
3.4 Segmentasi Daerah Masking

Pengolahan gambar dapat dilakukan dengan mudah pada saat tidak ada persimpangan. Namun, saat terdapat persimpangan, garis yang terdeteksi akan dipengaruhi oleh garis horizontal. Maka ini menjadi tantangan karena meski posisi robot berada pada garis, *error* robot akan membesar yang menyebabkan ketidaksesuaian pendeteksian posisi. Segmentasi dilakukan untuk memperhalus gerakan robot *mecanum* ketika melewati persimpangan. Pada Gambar 10, perubahan lebar dari kontur garis menyebabkan titik pusat dari kontur melenceng setiap ada persimpangan walaupun posisi robot masih sesuai dengan garis. Segmentasi dapat mengurangi pergeseran titik pusat dengan mengambil posisi rata-rata dari titik pusat masing-masing pada setiap segmen.



Gambar 11. Segmentasi daerah masking

Pada Gambar 11 dapat dilihat terdapat tiga buah daerah *masking*.

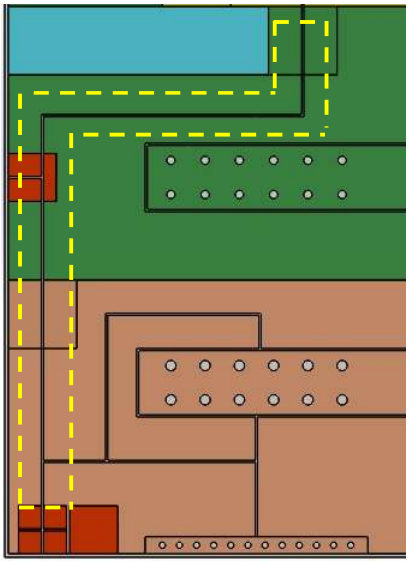


Gambar 12. Diagram alir proses segmentasi.

Proses segmentasi dapat dilihat pada Gambar 12 Setelah dilakukan segmentasi, ketiga segmen diproses dengan proses yang sama dengan proses *line follower* hingga memperoleh kontur dan titik pusat. Nilai posisi titik pusat dari ketiga segmen kemudian dirata-ratakan.

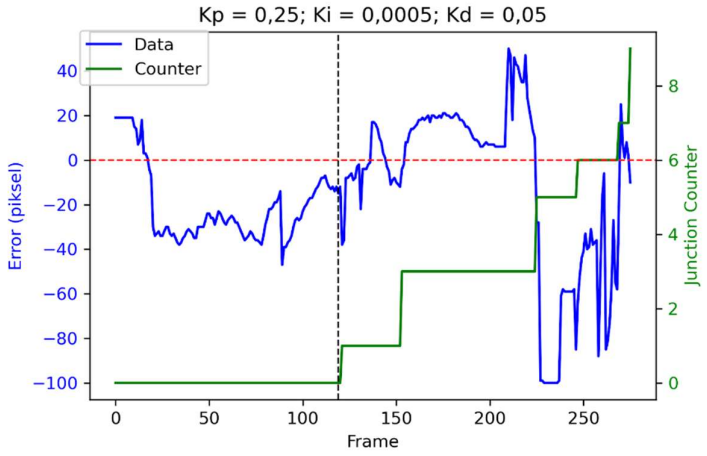
3.5 Hasil Pengujian

Pada fungsi *line follower* telah diperoleh nilai *error* yang merupakan selisih antara titik pusat garis dan titik tengah acuan (*offset*). Nilai *error* tersebut dapat dikalikan dengan konstanta pada parameter PID dan hasilnya dijadikan kecepatan pada sumbu x robot. Kemudian kecepatan sumbu y robot dapat ditentukan sesuai kebutuhan. Gambar 13 menunjukkan area lintasan pengujian robot *line follower* yang merupakan bagian arena dalam kontes robot Indonesia tahun 2024 divisi Abu Robocon. Adapun lintasan yang akan dilewati robot adalah yang telah diberi tanda garis kuning putus-putus.



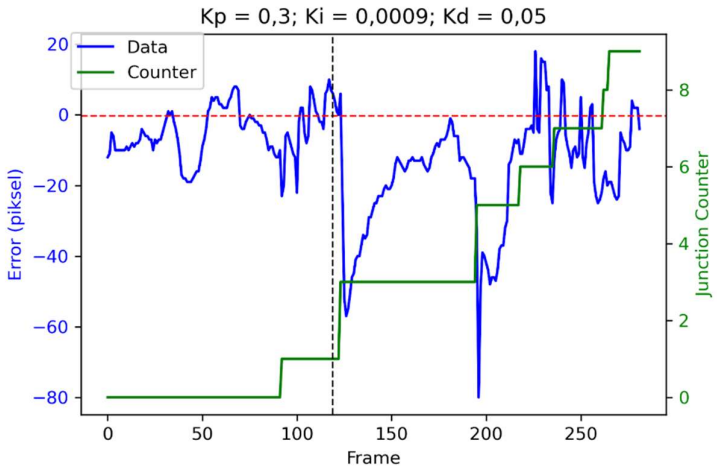
Gambar 13. Area pengujian lintasan robot *line follower*

Pada pengujian pertama, Kontansta PID yang didapatkan secara *trial* and *error* ialah $K_p = 0{,}25$, $K_i = 0{,}0005$ dan $K_d = 0{,}05$. Hasil penerapan kontstanta PID ini terlihat pada Gambar 14. Gambar 14 menunjukkan dua buah grafik, yaitu grafik *error* ialah perubahan nilai selisih antara titik acuan dan titik pusat garis terhadap perubahan *frame* dan grafik *counter* ialah perubahan nilai *counter* persimpangan terhadap perubahan *frame*. Perubahan *frame* pada proses pengolahan citra dilakukan setiap 180 milidetik.



Gambar 14. Data grafik *error* dan counter pada pengujian 1.

Pada Gambar 14, pergerakan robot terlihat berosilasi, hal ini disebabkan dari getaran robot saat bergerak yang mengakibatkan kamera yang ditempelkan pada bagian robot ikut bergetar. Selain itu, pergerakan robot yang tidak halus mengakibatkan kamera menangkap gambar secara tidak stabil. Dampak osilasi ini mengakibatkan kecepatan vx pada robot juga berisolasi, namun robot tetap dapat mengikuti garis lintasan. Pada Gambar 14, terdapat fluktuasi *error* yang cukup tinggi. Hal ini disebabkan karena adanya *counter* persimpangan dan robot harus berbelok. Pada arena, lintasan berbelok sebesar 90 derajat. Saat terjadi proses belok, robot mengalami nilai *error* yang cukup besar yang mengakibatkan pendeteksian kamera menjauhi lintasan *line follower*. Namun nilai *error* kembali berkurang setelah robot kembali berhasil mendeteksi lintasan. Nilai *error* rata-rata yang pada pengujian pertama ini didapatkan sebesar 26,891 %.



Gambar 15. Data grafik *error* dan counter pada pengujian 2.

Pengujian kedua dilakukan dengan perubahan pada parameter PID yang diperoleh secara *trial* and *error*. Parameter tersebut ialah $K_p = 0.3$, $K_i = 0.0009$ dan $K_d = 0.05$. Gambar 15 menunjukkan grafik *error* dan grafik *counter* dengan pengujian kondisi yang sama dengan parameter sebelumnya. Dari hasil pengujian kedua, terlihat robot mengikuti garis dengan menjaga supaya nilai *error* tetap mendekati nilai nol, namun robot tidak dapat menuju kondisi stabil dikarenakan getaran dan gerakan roda robot mecanum. Nilai *error* rata-rata yang diperoleh dengan parameter ini adalah 13,507% yang dapat disimpulkan jauh lebih baik dibandingkan dengan dengan pengujian dengan parameter PID yang pertama. Oleh karena itu, dengan pengaturan parameter konstanta PID, nilai *error* dapat diminimalisir sehingga robot dapat mengikuti garis dengan baik.

4. KESIMPULAN

Setelah melakukan pengujian dan analisis pada sistem *line follower* robot *mecanum*, diperoleh kesimpulan sebagai berikut. Sistem *line follower* menggunakan kamera pada robot mecanum telah berhasil dibuat dengan memanfaatkan teknik masking untuk memperkecil area yang dikomputasi, diikuti dengan pemisahan garis dan bukan garis menggunakan *color thresholding*. Garis yang terdeteksi dianalisis untuk menentukan titik tengahnya, yang dibandingkan dengan titik tengah acuan untuk menghitung *error*. *Error* ini dikalikan dengan konstanta PID yang disesuaikan dan digunakan untuk mengatur kecepatan pada sumbu x, memastikan robot mengikuti garis dengan akurasi yang baik. Sistem ini juga mampu

mengenali persimpangan dengan menganalisis perubahan lebar kontur garis. Penentuan persimpangan dilakukan berdasarkan perubahan mendadak pada lebar kontur saat robot bergerak dari area non-persimpangan ke area persimpangan. Dengan kemampuan mengikuti garis dan mengenali persimpangan secara efisien, sistem ini memberikan fleksibilitas dan akurasi dalam navigasi robot mecanum. Paramater PID yang didapatkan menghasilkan *error* sebesar 13,507%. Terdapat pengembangan yang dapat dilakukan dari hasil penelitian ini, diantaranya pengaturan kecepatan setiap motor sehingga robot dapat bergerak secara halus, penentuan parameter PID dengan berbagai metode sehingga mendapatkan nilai yang optimal, dan penggunaan *stabilizer* untuk kamera sehingga mengurangi osilasi saat dilakukan pengolahan citra.

DAFTAR RUJUKAN

Adnan, M. F. F., Ahmad, K. A., Boudville, R., Hussain, Z., & Husin, N. I. (2020). Modelling and Simulation of Omnidirectional Mobile Robot with Line Follower and Obstacles Avoidance. *2020 10th IEEE International Conference on Control System, Computing and Engineering (ICCSCE)*, (pp. 174–179). https://doi.org/10.1109/ICCSCE50387.2020.9204922

David, D. (2016). Kendali Logika Fuzzy Pada Robot Line Follower. *Creative Information Technology Journal*, *3*, 15. https://doi.org/10.24076/citec.2015v3i1.62

Deswal, M., & Sharma, N. (2014). *A Simplified Review on Fast HSV Image Color and Texture Detection and Image Conversion Algorithm*. https://api.semanticscholar.org/CorpusID:212604677

Erbay, O., Doğan, A., & Devecioğlu, E. (2024). *Line Follower Robot with PID Control*.

Fahmizal, F., Priyatmoko, A., & Mayub, A. (2022). Implementasi Kinematika Trajectory Lingkaran pada Robot Roda Mecanum. *Jurnal Listrik Instrumentasi dan Elektronika Terapan (JuLIET)*. https://api.semanticscholar.org/CorpusID:250931639

Gomes, M. V, Bássora, L. A., Morandin, O., & Vivaldini, K. C. T. (2016). PID control applied on a line-follower AGV using a RGB camera. *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, (pp. 194–198). https://doi.org/10.1109/ITSC.2016.7795553

Khade, K., Naik, R., & Patil, A. (2017). Design of all color line follower sensor with auto calibration ability. *2017 7th International Symposium on Embedded Computing and System Design (ISED)*, (pp. 1–5). https://doi.org/10.1109/ISED.2017.8303912

Latif, A., Widodo, H. A., Rahim, R., & Kunal, K. (2020). *Implementation of Line Follower Robot based Microcontroller ATMega32A*. https://api.semanticscholar.org/CorpusID:214403516

Manavaalan, G., Karon, S., Krishna, R., Raj, M. L., Priya, P. H., & Orlando, M. F. (2023). Development and Control of a Four-Wheel Drive Holonomic Mobile Robot. *2023 7th International Conference on Computer Applications in Electrical Engineering-Recent Advances (CERA)*, (pp. 1–6). https://doi.org/10.1109/CERA59325.2023.10455377

Miftahul, H. (2016). Pengontrolan Kecepatan Mobile Robot Line Follower Dengan Sistem Kendali PID. *TELKA*, *2*(2), 150–159.

Nugraha, M. B., Ardianto, P. R., & Darlis, D. (2015). Design and implementation of RFID line-follower robot system with color detection capability using fuzzy logic. *2015 International Conference on Control, Electronics, Renewable Energy and Communications (ICCEREC)*, 75–78. https://doi.org/10.1109/ICCEREC.2015.7337058

Ogata, K. (2010). *Modern Control Engineering* (5/Ed). Prentice Hall.

Risah Prayogi, Y., Lega Wibisono, C., Ahmad Hifdhul Abror, dan, & Sunan Ampel, N. (2019). *Prosiding Seminar Nasional Teknologi dan Sains (SnasTekS)*.

Shirmohammadi, S., & Baghbani, F. (2024). Design and Implementation of a Line Follower Robot. *2024 10th International Conference on Artificial Intelligence and Robotics (QICAR)*, (pp. 268–271). https://doi.org/10.1109/QICAR61538.2024.10496637

Soans, R. V., Ranjith, Hegde, A., Singh, C., & Kumar, A. (2017). Object tracking robot using adaptive color thresholding. *2017 2nd International Conference on Communication and Electronics Systems (ICCES)*, (pp. 790–793). https://doi.org/10.1109/CESYS.2017.8321192

Supriadi, & Rizal, A. (2017). Implementasi Fuzzy Logic Controller Pada Robot Line Follower. *Prosiding Seminar Nasional Teknologi IV*, (pp. 59–64).