PERANCANGAN SELF DRIVING DENGAN METODE KONTROL PD PADA SISTEM TRACKING AUTONOMOUS CAR

Muhammad Taufiqurrahman*), Sumardi, and Munawar Agus Riyadi

Departemen Teknik Elektro, Fakultas Teknik, Universitas Diponegoro,
Jl. Prof. Sudharto, SH, Kampus UNDIP Tembalang, Semarang 50275, Indonesia

*)E-mail : taufiqurr.muhammad@gmail.com

Abstrak

Self-driving merupakan sebuah sistem yang dapat mempermudah pekerjaan manusia dalam hal mengemudi. Sistem self-driving akan menggerakkan gas throttle, gearshift, rem, dan steer secara otomatis. Dalam perkembangannya sistem self-driving memiliki beberapa metode diantaranya dengan menggunakan radar, lidar, GPS, atau pun kamera. Penelitian ini merancang sistem self-driving pada prototype autonomous car dengan skala 1:10 dengan menggunakan kamera sebagai sensor vision dan sensor ultrasonik untuk pendeteksi halangan untuk menghindari benturan. Sistem dirancang untuk dapat mendeteksi street mark dengan memproses gambar dari kamera menggunakan metode filter warna HSV. Sistem self-driving berjalan dengan melakukan tracking pada street mark dengan menggunakan kontrol PID dengan nilai ki=0 untuk mengurangi osilasi. Pada penelitian ini telah berhasil dirancang sistem self-driving yang dapat mengikuti street mark dari lintasan yang disediakan. Parameter kontrol PD dari self-driving yang sesuai dengan karakteristik untuk self-driving pada pengujian secara trial and error dengan nilai kp=1 dan kd=3 pada lintasan lurus yang menghasilkan performa dengan rising time=0,899 detik dan nilai kp=1 dan kd=2 pada lintasan menikung yang menghasilkan performa dengan rising time=1,381 detik. Variabel color filtering dengan metode HSV untuk mendeteksi street mark yang berwarna putih adalah dengan nilai threshold “hue” 0-179, “saturation” 0-30, dan “value” 200-255.

Kata kunci : Filter warna HSV, Metode kontrol PD, Mobil self-driving, Tracking street mark.

Abstract

Self-driving is a system that make human activity easily on driving a car. Self-driving system can control gas throttle, gearshift, engine breaker, and steer automatically to follow the track. Some self-driving developers use radar, LIDAR, GPS, or Camera for built this system. Present research is design self-driving system with 1:10 prototype scaled. Self-driving system built by camera as a vision sensor and ultrasonic sensor as a range sensor for avoid collision. Self-driving system is design for detect the street mark with HSV color filtering method on real time captured image from camera. This research use PID control method to control the stability with Ki=0 to reduce oscillation for the sustainable self-driving system. Self-driving system was built and to be able to follow the street mark on linear track and bend track. PD control parameter for self-driving with trial and error test are obtained on linear track had value Kp = 1 and Kd = 3 with rising time=0,899 second and on bend track had value of Kp = 1 and Kd = 2 with rising time=0,899 second. HSV color filtering variable for street mark detection had value 0-179 of “hue” threshold, 0-30 of “saturation” threshold, and 200-255 of “value” threshold.

Keywords: HSV color filtering, PD control system, Self driving car, Street mark tracking.

1. Pendahuluan

Transportasi adalah salah satu bagian penting bagi kehidupan manusia. Salah satu transportasi pribadi yang banyak digunakan yaitu mobil. Seiring dengan perkembangan jaman, setiap perusahaan pembuat mobil selalu mengeluarkan produk terbarunya yang lebih canggih dan menyesuaikan perkembangan jaman. Setiap vendor mobil saling bersaing untuk meningkatkan kualitas dari segi disain, interior, multimedia, kenyamanan, keamanan, hingga otomatisasi kontrol yang dapat membantu penggunanya.

Penggunaan kontrol otomatis pada mobil ini mulai populer banyak dikembangkan oleh vendor-vendor mobil. Kontrol otomatis yang dikembangkan kedepannya sangat beragam jenisnya, diantaranya *Automatic Park Assist* [1], *Adaptive Cruise Control* (ACC) [1], *Highway Driving Assistant* [1], *Autonomous Highway Driving* [1], *traffic jam Assist* [1], *Auto Pilot* [2], *Lane Departure Warning*

System [2], *Lane Keeping Assist System* [2], *automatic breaking system* (ABS), *Parallel Parking Assist* [2] dan *Self Driving System*[3].

Self driving merupakan salah satu jenis kontrol mobil otomatis yang dapat memudahkan manusia dalam berkendara tanpa campur tangan pengemudi. *Self driving* berguna ketika pengemudi mengalami kondisi-kondisi tertentu yang harus melepaskan kemudi seperti menganggkat telpon, mengambil sesuatu di dashboard, dan lain sebagainya. Berdasarkan makalah dari Morgan Stanley Research tentang *autonomous driving* hingga tahun 2016 ini masih bersifat pasif [4]. Pengembangan penelitian *self-driving* ini pun diprediksikan oleh Morgan Stanley Research telah lengkap dan memungkinkan untuk diimplementasikan pada tahun 2022 [4]. Pengembangan penelitian tentang *autonomous driving car* masih sangat banyak dan luas. Untuk itu, penulis berkeinginan untuk melakukan penelitian tentang *self-driving* dengan skala *prototype*.

Berdasarkan penelitian sebelumnya tentang *self-driving* yang memberikan desain model yang mengkombinasikan beberapa sensor [5] dan penelitian *self-driving* dengan menggunakan kontrol *adaptive*-PD [6] untuk itu pada penelitian ini merancang dan membuat sebuah sistem *self-driving* pada *prototype autonomous car* yang dapat bergerak mengikuti jalur secara otomatis dengan menggunakan kamera sebagai sensor *vision* untuk mendeteksi *street mark* dan sensor jarak ultrasonik untuk mendeteksi adanya halangan sebagai pengaman agar tidak terjadi tabrakan terhadap mobil.

2. Metode

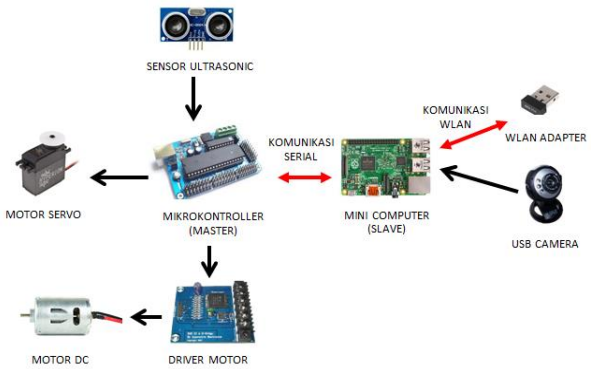
Sistem *self-driving* yang dirancang pada penelitian ini menggunakan *prototype autonomous car* dengan skala 1:10 yang dapat bergerak mengikuti jalan secara otomatis. *Prototype* tersebut menggunakan sensor kamera sebagai sensor *vision* dan sensor jarak ultrasonik untuk mendeteksi halangan di depan mobil. Gambar yang diambil oleh kamera kemudian diproses menggunakan *mini computer* raspberry pi 2 dengan metode HSV *color filtering*.

2.1. Perancangan Perangkat Keras

Perangkat keras yang digunakan merupakan *prototype autonomous car* dengan kit RC model *touring* dengan skala 1:10. Sistem penggerak/aktuator yang digunakan yaitu motor DC untuk kontrol kecepatan dan motor servo untuk kontrol *steering* roda depan. Sensor yang digunakan pada *self-driving* ini adalah kamera dan sensor jarak ultrasonik. sementara kontroler yang digunakan adalah *mini computer* raspberry pi 2 dan mikrokontroler ATMega 16. *Prototype* yang digunakan pada penelitian ini dapat dilihat seperti Gambar 1.

Gambar 1. Prototype autonomous car

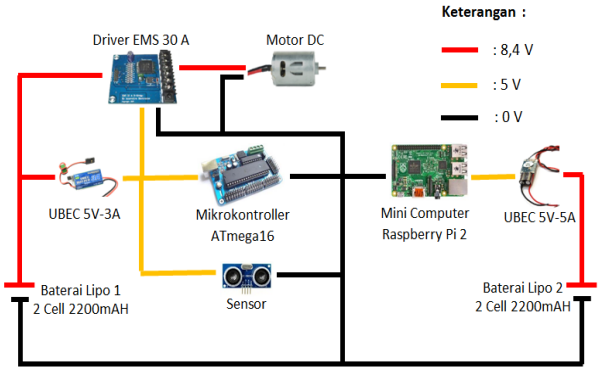
Self-driving dirancang untuk dapat medeteksi letak dari *street mark* melalui hasil proses gambar yang diambil oleh kamera. Kamera diperintahkan sebagai *input* gambar oleh *mini computer* raspberry pi. Hasil proses dari koordinat deteksi *street mark* dari gambar dikirimkan ke mikrokontroler melalui komunikasi serial. Mikrokontroler bertugas untuk membaca data koordinat yang dikirimkan raspberry pi dan membaca sensor jarak ultrasonik untuk kemudian diproses dengan metode kontrol PD. Hasil dari kontrol PD diteruskan menjadi sinyal kontrol motor servo sebagai *steering* roda depan dan hasil baca jarak di proses untuk menjalankan motor dc. Proses pengolahan citra dapat dimonitor oleh komputer melalui jaringan *wireless terminal adapter*. Hubungan antar *hardware* berdasarkan sistem kerja dari setiap *hardware* yang digunakan pada penelitian ini dapat dilihat seperti pada Gambar 2.



Gambar 2. Diagram hubungan antar *hardware*

Prototype autonomous car dalam perancangan ini menggunakan dua buah sumber tegangan yang berupa baterai lipo 2 *cell* dengan kapasitas 2200mAH. Kedua baterai digunakan untuk memisahkan tegangan *mini computer* raspberry pi agar terhindar dari komponen induktor yang besar seperti motor. Hal tersebut dilakukan karena *mini computer* raspberry pi membutuhkan tegangan yang stabil agar tidak terjadi reset yang dapat membuat program *image processing* berhenti. Rangkaian

catu daya dari masing-masing *hardware* ditunjukan seperti pada Gambar 3.



Gambar 3. Rangkaian catu daya

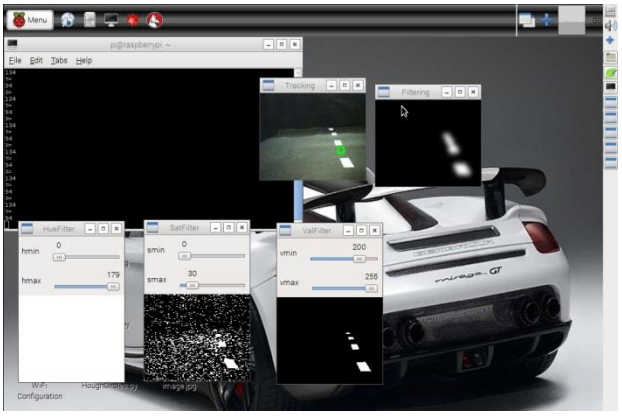
2.2. Perancangan Perangkat Lunak

Perangkat lunak (*software*) yang digunakan untuk program pada mikrokontroler atmega 16 adalah bahasa pemrograman C dan pada *mini computer* raspberry pi adalah python dengan *library image processing* openCV.

Mikrokontroler sebagai *master* selalu membaca data sensor ultrasonik dan membaca data dari komunikasi serial. data koordinat hasil komunikasi diproses menjadi nilai *error* yang kemudian diumpan balik ke aktuator servo dengan penambahan kontrol PD terhadap setiap perubahan. Jika hasil perhitungan 0 maka posisi *steering* lurus, jika hasil perhitungan positif maka *steering* belok ke kiri, dan jika nilai hasil perhitungan negatif maka *steering* belok ke kanan. Sementara jika sensor jarak mendeteksi halangan maka motor dc tidak berputar dan jika sensor jarak tidak mendeteksi halangan maka motor akan diperintahkan untuk berputar menjalankan mobil.

Mini computer raspberry pi bertindak sebagai *slave* yang memiliki tugas untuk melakukan proses gambar yang diambil dari kamera hingga menghasilkan data koordinat objek. *Mini computer* raspberry pi melakukan operasi *color filtering* dengan metode hsv, melakukan perhitungan koordinat objek, dan mengirim data ke mikrokontroler melalui komunikasi serial. proses tersebut diperintah dengan menggunakan program dengan bahasa pemrograman python. Sementara untuk menjalankan perintah pengolahan citra digital menggunakan *library* openCV.

Gambar 4 menunjukan tampilan *interface* yang didesain pada perancangan *self-driving* ini. *Interface* dirancang untuk mempermudah proses *thresholding* untuk menentukan nilai variabel minimal dan maksimal dari komponen *filtering hue, saturation*, dan *value*. *Window* yang dirancang antara lain "HueFilter", "SatFilter", "ValFilter", "Filtering", dan "Tracking".



Gambar 4. Tampilan *interface*

Windows "HueFilter", "SatFilter", dan "ValFilter", menampilkan *trackbar* untuk mengatur variabel minimal dan maksimal dari *threshold* masing-masing komponen *filtering*. *Windows* tersebut pun menampilkan hasil *filter* dari setiap *filter hue, saturation*, ataupun *value*. *Window* "Filtering" menampilkan hasil jumlah dari ketiga proses *hue, saturation, dan value* menjadi hasil *filter* akhir HSV yang menentukan objek terdeteksi atau tidak.

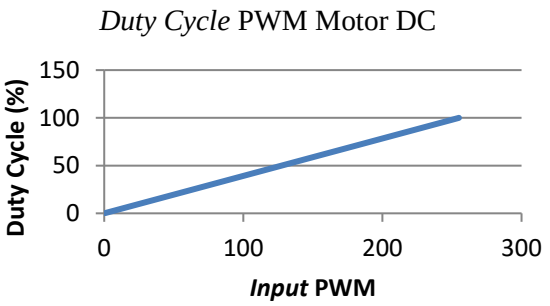
Setelah objek dapat terdeteksi dari *filter* HSV, selanjutnya dilakukan perhitungan koordinat dari objek yang terdeteksi. Data koordinat tersebut kemudian dikirimkan kepada mikrokontroler melalui komunikasi serial. Sementara *Window* "Tracking" menampilkan gambar asli yang diambil oleh kamera dengan penambahan bentuk persegi di titik pusat koordinat dari objek yang dideteksi.

3. Hasil dan Analisa

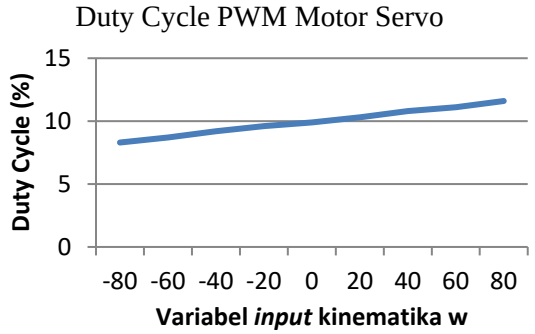
Pengujian dilakukan untuk mengetahui kondisi dari *prototype autonomous car* yang dirancang. Pengujian juga dilakukan untuk menjelaskan kestabilan dari kemampuan sistem *self-driving* dalam mengikuti lintasan yang telah diberikan.

3.1. Pengujian Perangkat Keras (*Hardware*)

Pengujian *hardware* yang dilakukan pada *prototype autonomous car* ini meliputi pengujian motor dc, pengujian motor servo, dan pengujian sensor jarak. Pengujian motor dc dan motor servo dilakukan dengan membandingkan sinyal pwm yang dikeluarkan mikrokontroler terhadap perubahan variabel v dan w. Perbandingan *dutycycle* dari pin PWM terhadap variasi nilai *input* v motor dc terlihat seperti Gambar 5. Perbandingan *dutycycle* dari pin PWM terhadap variasi nilai *input* variabel w motor servo terlihat seperti Gambar 6.

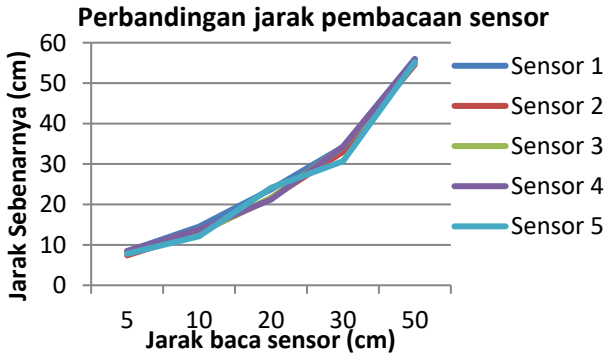


Gambar 5. Grafik pulsa PWM motor DC terhadap variasi *input* v



Gambar 6. Grafik pulsa PWM motor servo terhadap variasi *input* w

Pengujian sensor jarak ultrasonik dilakukan dengan mengukur jarak pembacaan sensor dengan jarak sesungguhnya. Sensor HC-SR04 membaca dengan nilai 0 pada jarak 4cm dari ujung sensor sehingga perhitungan *error* dikurangi 4cm. Nilai *error* rata-rata dari pengujian pengukuran sensor jarak sebesar 1,28. Grafik perbandingan pembacaan sensor jarak terhadap jarak sebenarnya dari kelima sensor yang digunakan terlihat seperti pada Gambar 7.



Gambar 7. Grafik perbandingan pembacaan sensor jarak Ultrasonik HC-SR04

3.2. Pengujian *Color Filtering*

Pengujian *color filtering* dilakukan untuk mengecek proses pengolahan citra terhadap kesesuaian dengan ruang warna dari metode HSV. Pada pengujian ini dilakukan dua metode *thresholding* yaitu minimal ke maksimal dan maksimal ke minimal. Pengujian ini dilakukan untuk membedakan warna putih dan hitam terhadap *threshold hue*, *saturation*, dan *value* yang bernilai minimal ataupun maksimal.

Kalibrasi minimal ke maksimal dilakukan dengan memberikan nilai minimal 0 dan nilai maksimal 0. Selanjutnya nilai batas maksimal digeser dengan penambahan hingga terjadi perubahan yang membedakan hasil *filter* dari objek berwarna hitam dan objek berwarna putih. Sementara kalibrasi maksimal ke minimal dilakukan dengan memberikan nilai minimal dan maksimal 179 untuk *hue*, 255 untuk *saturation* dan *value*. Selanjutnya nilai minimal digeser dengan pengurangan hingga terjadi perubahan yang membedakan hasil *filter* dari objek berwarna hitam dan objek berwarna putih.

Tabel 1. Pengujian Threshold Hue

Kalibrasi	*Hue min*	*Hue max*	Objek Putih	Objek Hitam
Minimal ke maksimal	0	74	Sebagian	Terdeteksi
Maksimal ke Minimal	78	179	Terdeteksi	Sebagian

Tabel 2. Pengujian Threshold Saturation

Kalibrasi	*Sat min*	*Sat max*	Objek Putih	Objek Hitam
Minimal ke maksimal	0	30	Terdeteksi	Tidak
Maksimal ke Minimal	25	255	Tidak	Terdeteksi

Tabel 3. Pengujian Threshold Value

Kalibrasi	*Val min*	*Val max*	Objek Putih	Objek Hitam
Minimal ke maksimal	0	203	Tidak	Terdeteksi
Maksimal ke Minimal	200	255	Terdeteksi	Tidak

Berdasarkan data dari Tabel 1 tentang kalibrasi threshold hue mendeteksi warna hitam pada nilai minimal dan mendeteksi putih dengan nilai maksimal meski hasil filtering pada komponen hue tidak sempurna. Karena warna hitam maupun putih dapat dihasilkan dari berapa pun nilai hue.

Berdasarkan data pengujian color filtering komponen saturation seperti pada Tabel 2, threshold saturation mendeteksi warna putih pada nilai minimal dan mendeteksi warna hitam pada nilai maksimal. Sementara pada hasil pengujian color filtering komponen value seperti pada Tabel 3 bahwa threshold value mendeteksi

objek berwarna putih pada nilai maksimal dan mendeteksi objek berwarna hitam pada nilai minimal.

3.3. Pengujian Iluminasi

Pengujian iluminasi dilakukan untuk mengetahui pengaruh dari perbedaan intensitas cahaya yang diberikan. Pada pengujian ini diberika dua kondisi yaitu kondisi penerangan tinggi dan penerangan rendah. Kondisi penerangan tinggi diberikan penerangan langsung dari lampu SL tanpa adanya halangan. Sedangkan kondisi penerangan tinggi dengan penerangan yang sama namun diberikan penghalang intensitas langsung sehingga pada objek timbul bayangan yang membuat objek deteksi lebih gelap.

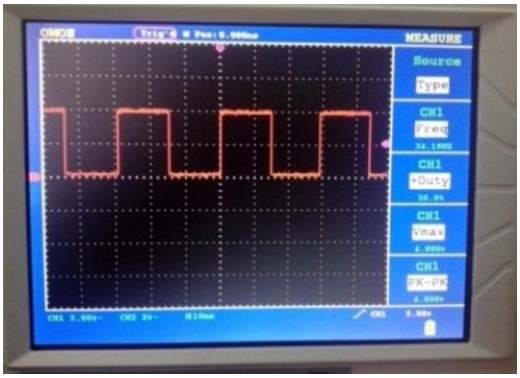
Tabel 4. Pengujian kinematika *prototype autonomous car*

Threshold Saturation	*Threshold Value*	Kondisi Penerangan	Deteksi *Street mark*
0-6	200-255	Tinggi	Terdeteksi
0-6	200-255	Rendah	Tidak Terdeteksi
0-30	130-255	Rendah	Terdeteksi
0-30	130-255	Tinggi	Disertai banyak noise di sekitarnya
0-30	200-255	Rendah	Terdeteksi
0-30	200-255	Tinggi	Terdeteksi

Berdasarkan Tabel 4 dapat dilihat dengan nilai *threshold* yang sama dengan pemberian kondisi penerangan yang berbeda dapat mempengaruhi hasil deteksi dari proses *color filtering* yang dilakukan. Namun dengan nilai *threshold saturation* 0-30 dan *threshold value* 200-255 sistem mampu beradaptasi baik pada kondisi penerangan tinggi maupun rendah.

3.4. Pengujian kecepatan komunikasi data

Pengujian kecepatan komunikasi data dilakukan untuk menunjang sebagai data waktu terhadap nilai *error* yang diterima komputer dalam pengujian kestabilan. Perhitungan waktu pengiriman data dihitung dari *looping* perintah kirim data serial. setelah kode program pengiriman data kemudian disisipkan untuk mengubah kondisi PORTB.0 dari logika *high* ke *low* dan dari logika *low* ke *high*. Sehingga dengan berubahnya logika PORTB.0 menghasilkan pulsa yang dapat dihitung frekuensinya dengan menggunakan osiloskop. Hasil pengukuran sinyal PORTB.0 dapat dilihat seperti pada Gambar 8.



Gambar 8. Hasil pengukuran pulsa dari pengiriman komunikasi serial

Berdasarkan pengujian pulsa dari logika perintah pengiriman data serial mikrokontroler ke komputer seperti pada Gambar 8 didapatkan nilai frekuensi yang terbaca pada osiloskop yaitu 34,19 Hz. Berdasarkan nilai frekuensi yang terukur pada osiloskop kemudian dilakukan perhitungan untuk mendapatkan lebar pulsa dari satu gelombang pulsa. Perhitungan dilakukan dengan menggunakan rumus persamaan sebagai berikut.

$$\text{lebar pulsa} = \frac{1}{\text{frekuensi}}$$
$$= \frac{1}{34{,}19}$$
$$= 0.02924 \text{ s} = 29{,}24 \text{ ms}$$

Berdasarkan perhitungan tersebut, pulsa dari PORTB.0 memiliki lebar pulsa dengan waktu 29,24 ms. Dalam satu gelombang pulsa terjadi dua kali perubahan logika yaitu dari kondisi *low* ke *high* dan *high* ke *low*. Maka dalam satu gelombang pulsa terjadi dua kali pengiriman data serial. Maka waktu dari setiap pengiriman data serial mikrokontroler ke komputer adalah setengah dari waktu dari satu gelombang PORTB.0. maka nilai lebar pulsa dari satu gelombang pulsa PORTB.0 dibagi 2 dan menghasilkan waktu dari setiap pengiriman data serial yaitu 14,62 ms.

3.5. Pengujian Kinematika dan *free run*

Pengujian kinematika dilakukan untuk mengetahui kondisi *prototype autonomous car* yang diuji dari respon berdasarkan *input* perintah v dan w. pengujian dilakukan selama 1 detik dengan variasi nilai v untuk mengatur kecepatan dan w untuk mengatur *steering*. Dari variasi tersebut menghasilkan perubahan pada *prototype autonomous car* yang kemudian dilakukan pengukuran perubahan dari titik acuan kepada titik akhir dari sumbu x, sumbu y, dan sudut rotasi. Hasil pengujian kinematika ini dapat dilihat seperti pada Tabel 5.

Tabel 5. Pengujian kinematika *prototype autonomous car*

Pengujian	v	w	Jarak Perubahan translasi (cm)		Perubahan sudut rotasi (cm)
			Sumbu y	Sumbu x	
1	50	0	13,8	0	0
2	70	0	56,7	+ 2,5	+ 10°
3	50	-50	11,4	+ 2,3	+ 10°
4	50	50	14,4	- 3,6	- 10°
5	70	-50	56,6	+15,5	+ 25
6	70	50	57,9	- 25,5	- 30°

Pengujian *freerun* dilakukan untuk mengetahui kondisi *prototype autonomous car* yang diuji dari respon setelah mendeteksi adanya halangan. pengujian dilakukan variasi nilai v untuk mengatur kecepatan dan menentukan titik deteksi halangan yang menyebabkan motor berhenti putar. Akan tetapi *prototype autonomous car* masih bergerak akibat momentum yang dialami dan terjadi *free run*. Pengujian dilakukan dengan mengukur jarak *free run* terhadap variasi nilai pwm kecepatan yang diberikan. Hasil pengujian pengukuran jarak *free run* ini dapat dilihat seperti pada Tabel 6.

Tabel 6. Pengujian *free run* dari sistem *self-driving*

Variasi Kecepatan PWM	Jarak *free run* dari titik deteksi (cm)
40	4,5
45	17
50	27
55	43
60	58

3.6. Pengujian Kontrol Kestabilan PD

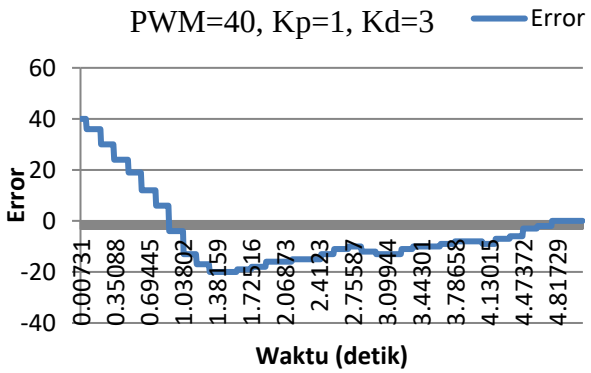
Pengujian kontrol kestabilan dilakukan dengan menggunakan metode PD *trial and error*. Karakteristik kontrol yang ingin dicapai yaitu memiliki *rising time* yang paling cepat, tidak berosilasi dalam waktu dekat, dan memiliki toleransi *error* yang kecil.

Pengujian kestabilan ini dilakukan pada dua kondisi. Kondisi pertama pada lintasan A dengan kondisi jalan lurus sementara kondisi kedua pada lintasan B dengan kondisi menikung.

Pengujian kestabilan pada lintasan A dilakukan dengan 8 kali percobaan untuk mendapatkan variasi perubahan terhadap variabel kecepatan PWM, nilai konstanta proporsional, konstanta integral, dan konstanta derifatif. Pada pengujian ini didapatkan hasil yang paling baik yang sesuai dengan karakteristik kestabilan untuk *self-driving* dengan nilai PWM=40, Kp=1, Ki=0, dan Kd=3 yang dapat dilihat seperti pada Gambar 9. Sementara hasil pengujian kestabilan *self-driving* pada lintasan A secara lengkap disajikan seperti pada Tabel 7.

Tabel 7. Hasil pengujian kestabilan dari kontrol PD *trial and error* pada lintasan A

Pengujian	Kestabilan
1. PWM=40, Kp=1, Ki=0, Kd=0	Lambat dalam mencapai *set point* dan berosilasi pada *error* negatif
2. PWM=40, Kp=2, Ki=0, Kd=0	Osilasi dengan *error* yang semakin tinggi (tidak stabil)
3. PWM=40, Kp=1, Ki=0, Kd=1	Tidak mampu mencapai *set point*
4. PWM=40, Kp=1, Ki=0, Kd=2	*Error* kecil, tr cukup lama
5. PWM=40, Kp=1, Ki=0, Kd=3	Tr cukup rendah, *error* kecil, *error steady state* yg cukup lama
6. PWM=50, Kp=1, Ki=0, Kd=1	Berosilasi tetapi nilai *error* tidak semakin mengecil
7. PWM=50, Kp=1, Ki=0, Kd=2	error osilasi semakin besar (tidak stabil)
8. PWM=50, Kp=1, Ki=0, Kd=3	*Overshoot* besar, *error* osilasi besar.

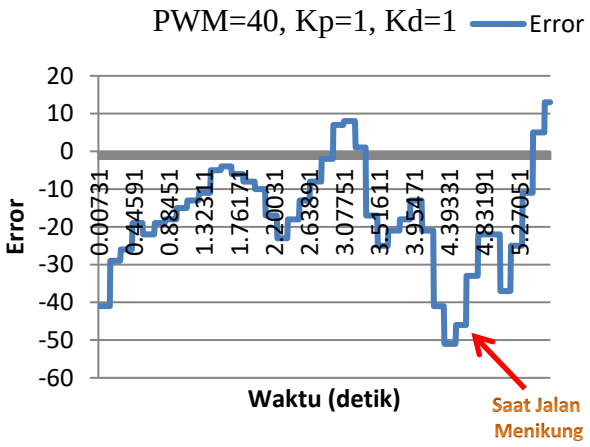


Gambar 9. Grafik kestabilan pada lintasan A

Pengujian kestabilan pada lintasan B dilakukan dengan 4 kali percobaan untuk mendapatkan variasi perubahan terhadap variabel kecepatan PWM, nilai konstanta proporsional, konstanta integral, dan konstanta derifatif. Pada pengujian ini didapatkan hasil yang paling baik yang sesuai dengan karakteristik kestabilan untuk *self-driving* dengan nilai PWM=45, Kp=1, Ki=0, dan Kd=2 yang dapat dilihat seperti pada Gambar 10. Sementara hasil pengujian kestabilan *self-driving* pada lintasan A secara lengkap disajikan seperti pada Tabel 8.

Tabel 8. Hasil pengujian kestabilan dari kontrol PD *trial and error* pada lintasan B

Pengujian	Kestabilan
1. PWM=50, KP=1, KI=0, KD=0	Error *max* sangat besar pada saat tikungan. Terlalu cepat dalam menikung.
2. PWM=40, KP=1, KI=0, KD=0	*Error* maks cukup kecil, tr cukup singkat, cukup overshoot kecil.
3. PWM=40, KP=1, KI=0, KD=1	*Error* maks kecil, tr cukup lama, overshoot kecil.
4. PWM=45, KP=1, KI=0, KD=2	*Error* maks kecil, tr singkat, overshoot kecil.



Gambar 10. Grafik kestabilan pada lintasan B

4. Kesimpulan

Berdasarkan pengujian dan analisis yang telah dilakukan, maka dapat disimpulkan bahwa sistem *self-driving* telah berhasil dirancang sesuai dengan spesifikasi perancangan sistem *self-driving* dan dapat berjalan mengikuti *street mark* dengan durasi maksimal selama 38,6 menit dengan catu daya 2 baterai lipo 2 cell 2200 mAH. *Threshold* yang digunakan dari hasil kalibrasi pada proses *color filtering* mendapatkan nilai *hue* maks=179, *hue* min=0, *saturation* maks=30, *saturation* min=0, *value* maks=255, dan *value* min=200 yang dapat mendeteksi *street mark* secara jelas baik pada pengujian iluminasi terang dan redup. Periode pengolahan data mikrokontroler ATmega 16 sebagai *master* dan waktu pengiriman data *error* dari mikrokontroler ke komputer saat melakukan pengujian kestabilan sebesar 14,62 ms. Periode proses *mini computer* raspberry pi sebagai pemroses gambar sebesar 138 ms. Sehingga hal ini membuktikan bahwa periode pemrosesan data pada sistem *self-driving* pada perancangan ini memenuhi spesifikasi yang dibutuhkan yaitu kurang dari 200 ms. Parameter kontrol PD dengan metode *trial and error* yang paling stabil pada pengujian kestabilan *self-driving* di lintasan lurus (lintasan A) adalah dengan nilai pwm = 40 (8 bit) dengan nilai *dutycycle* (+) 15,625 %, Kp=1 dan Kd=3 dengan *rising time* 0,899 detik dan *overshoot* sebesar 17. Parameter kontrol PD dengan metode *trial and error* yang paling stabil pada pengujian kestabilan *self-driving* di lintasan menikung (lintasan B) adalah dengan nilai pwm = 45 (8 bit) dengan nilai *dutycycle* (+) 17,57 %, Kp=1 dan Kd=2 dengan *rising time* 1,381 detik dan *overshoot* sebesar 9.

Referensi

[1]. S. P. Tao Jiang, U. Ayyer, A. Tolani, and S. Husain, "Self - Driving Cars : Disruptive or Incremental ?", Sutardja Center for Entrepreneurship & Tehnology, California, 2015.

[2]. A. Forrest, M. Konca, "Autonomous Cars and Society," Worcester, 2007.

[3]. ___,"Google Self-Driving Car Project Monthly Report Google Self-Driving Car Project Monthly Report", California, 2015.

[4]. R. Shanker, A. Jonas, S. Devitt, K. Huberty, S. Flannery, W. Greene, B. Swinbume, G. Locraft, A. Wood, K. Weiss, J. Moore, A. Schenker, P. Jain, Y. Ying, S. Kakiuchi, R. Hoshino, and A. Humphrey, "Autonomous Cars Self-Driving the New Auto Industry Paradigm", New York, 2013.

[5]. H. Cho, Y. Seo, B. V. K. V. Kumar, and R. R. Rajkumar, "A Multi-Sensor Fusion System for Moving Object Detection and Tracking in Urban Driving Environments," IEEE International Conference on Robotics & Automation, Hongkong, 2014.

[6]. P. Zhao, J. Chen, Y. Song, X. Tao, T. Xu, and T. Mei, "Design of a Control System for an Autonomous Vehicle Based on Adaptive-PD," International Journal of Advanced Robotic Systems, vol. 9, no. 44, INTECH, Rijeka, 2012.

[7]. ___, H. Torque servo, "MG996R," no. 6 V, pp. 7–8.

[8]. ___, "with 16K Bytes Programmable ATmega16 (L)."

[9]. ___, "Ultrasonic Ranging Module HC - SR04," pp. 3–5.

[10]. _,_, "Datasheet M-tech Camera Minicam 5mp." .

[11]. D. Putra, Pengolahan Citra Digital. Yogyakarta: Andi, 2010.

[12]. __,"Physics of Color," Global Beads Inc., pp. 2–6.

[13]. __,"The OpenCV Reference Manual," 2016.

[14]. ___, "Raspberry Pi 2 , Model B," p. 6.

[15]. A. Bejo, C&AVR Rahasia Kemudahan Bahasa C dalam Mikrokontroler ATMega8535. Yogyakarta: Graha Ilmu, 2008.

[16]. I. Setiawan, Kontrol PD untuk Proses Industri. Jakarta: Elex Media Komputindo, 2008.

[17]. E. Leksono, Teknik Kontrol Automatik Alih Bahasa karangan Katsuhiko Ogata, Jilid 1 ce. Jakarta: Erlangga, 1995.