Jurnal Ilmiah Setrum

Volume 11, No.1, Juni 2022 **p-ISSN : 2301-4652 / e-ISSN : 2503-068X**

SISTEM DETEKSI OBJEK MENGGUNAKAN METODE HSV DAN BINOCULAR DISPARITY PADA TURTLEBOT3

Anggoro Suryo Pramudyo[1], Adam Dwinanto Angkoso[1], Fadil Muhammad[1]

[1]Jurusan Teknik Elektro, Fakultas Teknik, Universitas Sultan Ageng Tirtayasa, Cilegon, Banten.

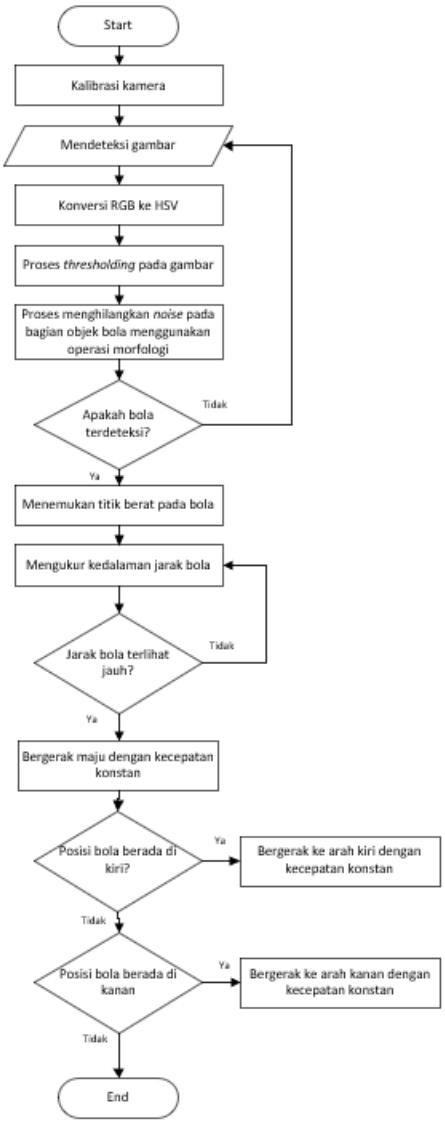
Informasi Artikel

Naskah Diterima : 6 Juni 2022
Direvisi : 21 Juni 2022
Disetujui : 22 Juni 2022

doi:10.36055/setrum.v11i1.15821

***Korespodensi Penulis :**
adamdwinanto1@gmail.com

Graphical abstract



Abstract

The development of technology is one of the factors that help humans in doing their daily work. Turtlebot3 is a mobile robot that can move from one place to another using wheels. The problem with turtlebot3 is that both cameras cannot be used to detect and measure the depth of objects, especially spherical objects. One way to detect objects is to use color selection. The color selection method is HSV, and other way to measure distance is to use left and right vision. The method used to measure distance is using binocular disparity which is applied to a stereo camera. The research was carried out to test detect objects using the HSV method and measuring object distances using the binocular disparity method applied to stereo cameras. The research was conducted at different lighting conditions, for 106 lux the highest percent error value at distance of 30cm is 1.65% and for 53 lux it has the highest percent error value at distance of 120cm, which is 4.48%.
Keywords : Stereo camera, Mobile robot, Turtlebot3, HSV, Binocular disparity

Abstrak

Perkembangan teknologi menjadi salah satu faktor yang membantu manusia dalam mengerjakan pekerjaan sehari-hari. Turtlebot3 merupakan *mobile robot* yang dapat berpindah tempat ke tempat lainnya dengan menggunakan roda. Masalah pada turtlebot3 yaitu kedua kamera yang tidak dapat digunakan untuk mendeteksi dan mengukur jarak kedalaman objek, khususnya pada objek bola. Salah satu untuk mendeteksi objek yaitu dengan menggunakan penyeleksian warna. Metode penyeleksian warna yaitu HSV, salah satu untuk mengukur jarak yaitu menggunakan penglihatan kiri dan kanan. Metode yang digunakan untuk mengukur jarak yaitu menggunakan *binocular disparity* yang diterapkan pada kamera stereo. Pengujian yang dilakukan untuk menguji mendeteksi objek menggunakan metode HSV dan pengukuran jarak objek menggunakan metode *binocular disparity* yang diterapkan pada kamera stereo. Pengujian dilakukan pada pencahayaan yang berbeda, untuk 106 *lux* memiliki nilai persen kesalahan tertinggi pada jarak 30cm yaitu 1,65% dan untuk 53 *lux* memiliki nilai persen kesalahan tertinggi pada jarak 120cm yaitu 4,48%.
Kata kunci: Kamera stereo, *Mobile robot*, Turtlebot3, HSV, *Binocular disparity*

1. PENDAHULUAN

Perkembangan teknologi robotika semakin berkembang pesat, dalam bidang robotika dapat memanfaatkan teknologi untuk melakukan pekerjaan dalam segala bidang seperti pekerjaan rumah tangga, pekerjaan pada restoran, dan dalam bidang lainnya. Teknologi robotika bisa digunakan untuk edukasi agar dapat mengembangkannya lebih jauh. Perkembangan teknologi robotika menghasilkan suatu *event* atau acara yang diselenggarakan untuk meningkatkan kreativitas dan inovasi tentang teknologi yang dibuatnya, yaitu *event* KRI atau Kontes Robot Indonesia.

KRI (Kontes Robot Indonesia) adalah kompetisi robot yang diselenggarakan oleh Kementerian Pendidikan dan Kebudayaan yang terbagi dalam kategori atau divisi yang berbeda [1]. KRI diadakan bertujuan agar mahasiswa dapat memberikan sebuah dukungan untuk kemajuan robotika di Indonesia dan sebagai dorongan agar dapat berkembang pada bidang robotika. KRSBI Beroda (Kontes Robot Sepak Bola Beroda) adalah salah satu kategori atau divisi pada perlombaan Kontes Robot Indonesia atau KRI.

Robot adalah perangkat yang mampu menyelesaikan masalah fisik dan perangkat dapat diprogram sesuai dengan kebutuhan. Adapun robot disesuaikan dengan kebutuhan, robot yang sederhana digunakan untuk pekerjaan yang mudah dan tidak kompleks, sedangkan robot yang dirancang spesifik digunakan untuk pekerjaan yang rumit dan memiliki kecerdasan buatan agar menyesuaikan dengan lingkungan yang ada [2]. Bentuk robot bermacam-macam, salah satunya adalah *Mobile robot*. *Mobile robot* adalah robot yang dapat berpindah dari satu tempat ke tempat lainnya, biasanya *mobile robot* menggunakan kaki atau roda [3][4]. Salah satu robot yang disebut sebagai *mobile robot* adalah robot turtlebot.

Robot turtlebot adalah robot yang biasa digunakan untuk tujuan edukasi dan menggunakan platform ROS (*Robot Operating System*). Salah satu seri yang ada yaitu turtlebot3. Robot turtlebot3 adalah generasi dari seri turtlebot. Robot turtlebot3 bentuknya kecil, *programmable*, basis yang digunakan yaitu ROS (*Robot Operating System*). Pembuatan turtlebot3 untuk mengurangi ukuran dari seri turtlebot sebelumnya dan lebih murah tanpa harus mengurangi fungsi dan kualitasnya. Turtlebot3 menggunakan algoritma SLAM (*Simultaneous Localization and Mapping*) untuk membuat sebuah peta sekitar [5][6]. Turtlebot3 dapat menggunakan kendali *remote* pada laptop, *joypad* ataupun berbasis Android. ROS atau *robot operating system* adalah sebuah *framework* yang digunakan untuk berbagai jenis robot, seperti *humanoid*, *mobile robot*, dan lain-lain [7][8][9]. ROS dikembangkan pada tahun 2007 oleh *Standford Artificial Intelligence Laboratory*. ROS di masukan ke dalam turtlebot3 yang bertujuan dapat memudahkan dalam mendeteksi objek menggunakan *library* OpenCV.

OpenCV adalah *library* komputer *vision* yang berbasis *open source* BSD dengan termasuk ratusan algoritma dari komputer *vision* tersebut. OpenCV biasa digunakan secara *real-time* [10][11][12]. OpenCV dapat menggunakan bahasa pemrograman seperti bahasa C, C++, *Python* dan Java. OpenCV biasa juga digunakan untuk *Face Recognition*, Pendeteksi benda dari jarak tertentu secara *real-time*.

Object Tracking adalah sebuah program di mana objek dapat dideteksi secara *real-time* atau berdasarkan data yang berada pada jarak pandang dari suatu alat yang digunakan menggunakan kamera, CCTV dan yang dapat menangkap dan merekam gambar [13][14]. *Object Tracking* dapat diartikan sebagai proses pelacakan dari satu objek ke objek lainnya menggunakan citra yang ada. *Object Tracking* memiliki 3 tahapan yaitu menganalisis video, mendeteksi objek yang bergerak, menambahkan *frame* pada citra yang didapat menandakan bahwa objek yang dideteksi sesuai dengan data yang ada.

Penelitian ini berfokus pada sistem deteksi pada robot turtlebot3 menggunakan raspberry pi 3 yang akan diterapkan pada robot sepak bola KRSBI Beroda ke depannya. Pencahayaan pada ruang dapat berpengaruh pada hasil keluaran pada gambar yang akan dihasilkan dan untuk menyesuaikan dengan keadaan yang ada dengan menggunakan HSV untuk mengubah parameter yang sesuai. Robot turtlebot3 menggunakan dua kamera atau *stereo vision*. *Binocular disparity* dapat diimplementasikan pada *stereo vision* untuk mengukur jarak pada objek [15]. Bahasa pemrograman yang digunakan adalah bahasa C dengan menggunakan *library* OpenCV. Perangkat yang digunakan yaitu raspberry pi 3 sebagai perangkat keras, kamera *raspi cam* , kamera *webcam* dan mikrokontroler OpenCR yang digunakan untuk mengendalikan motor servo dynamixels.

Tujuan dari penelitian sistem deteksi objek adalah untuk mendeteksi bola yang akan digunakan pada robot sepak bola, dan memiliki kinerja dalam melacak objek bola secara langsung ataupun *real-time*, menggunakan kamera untuk mengukur jarak bola menggunakan *binocular disparity* dan mendeteksi melalui seleksi bentuk warna HSV.

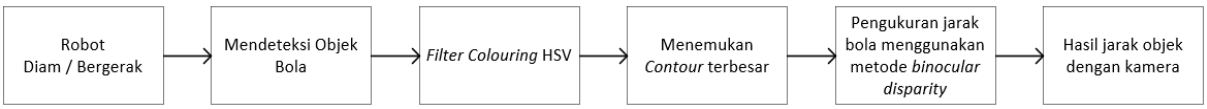
2. METODE PENELITIAN

2.1 Metode Penelitian

Metode dalam penelitian memiliki beberapa tahapan yang akan diterapkan dalam penelitian sebagai berikut :

1. Studi literatur, studi literatur menggunakan beberapa referensi jurnal, buku -buku dan beberapa artikel yang terkait dengan pengolahan citra, metode HSV, *object tracking*, operasi morfologi, robot turtlebot dan sebagainya yang terkait dengan penelitian ini.
2. Perancangan sistem, perancangan sistem pada penelitian yaitu perangkat keras dan perangkat lunak. Perangkat keras yang digunakan untuk mendeteksi objek menggunakan raspberry pi sebagai mikrokontroler dan raspicam sebagai kamera yang digunakan, sedangkan perangkat lunak yang digunakan ialah VMware Workstation sebagai *virtual machine* untuk menghubungkan antara *host* dengan robot.
3. Pengujian sistem deteksi, pengujian sistem deteksi menggunakan *color filter HSV* dalam keadaan diam dan bergerak. Keadaan diam ketika robot mendeteksi objek seperti bola. Keadaan bergerak untuk mengukur jarak antara robot dengan benda yang dideteksi menggunakan *raspi cam* dan *webcam* dengan menggunakan metode *binocular disparity*.
4. Analisa pengujian dan penelitian, mengandung analisa hasil dan pembahasan yang terkait dengan penelitian yang dikerjakan beserta dengan kesimpulan dari penelitian.

Tahapan metodologi penelitian di atas dapat dilihat pada diagram blok sebagai berikut :

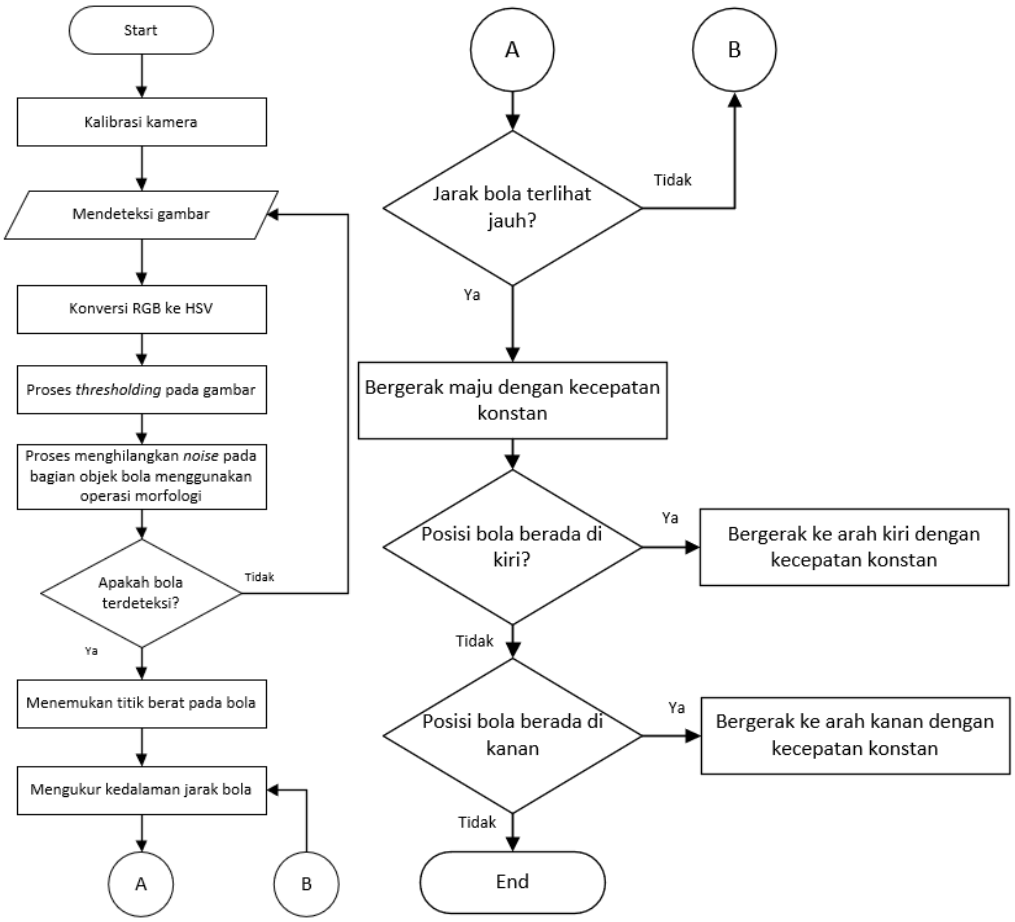


Gambar 1 Diagram Blok

Gambar 1 merupakan tahapan untuk mengolah citra menjadi objek yang dapat dideteksi dan didapatkan hasil dari jarak objek dengan mengukur menggunakan *stereo vision*. Langkah awal yang

dilakukan yaitu pendeteksian objek menggunakan metode *filter coloring* HSV. Langkah selanjutnya yaitu menemukan *contour* terbesar pada objek yang dideteksi sehingga pengukuran dapat dilakukan menggunakan metode *binocular disparity*. Langkah terakhir yaitu didapatkan hasil yang diinginkan berupa jarak yang dihasilkan. Gambar 2 merupakan diagram alir untuk proses dari mulai proses kalibrasi kamera hingga didapatkan hasil dari pengukuran jarak. Penyeleksi warna yang digunakan yaitu menggunakan metode HSV. Metode HSV merupakan metode yang menggunakan tiga parameter yaitu *hue, saturation,* dan *value*. Ketiga parameter ini yang akan digunakan untuk menyeleksi warna pada objek yang ingin di deteksi. Parameter yang didapat akan diaplikasikan ke dalam robot turtlebot yang akan digunakan.

2.2 Diagram Alir Penelitian



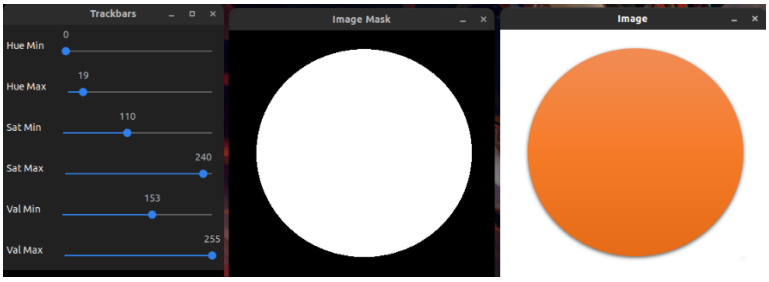
Gambar 2 Diagram Alir Penelitian

Desain penelitian ini menerapkan *color filter* HSV menggunakan kamera *stereo vision* dan mendapatkan posisi objek, mengirim perintah ke perangkat OpenCR untuk menggerakkan servo mengarah pada objek yang dideteksi. Pertama yang dilakukan yaitu kalibrasi kamera untuk menentukan parameter dari kedua kamera. Langkah selanjutnya yaitu menentukan keadaan robot yang sebenarnya. Langkah selanjutnya mengambil data gambar melalui kamera yang sudah dikalibrasi. Langkah selanjutnya yaitu mengubah gambar yang memiliki warna RGB menjadi warna HSV kemudian nilai *hue, saturation,* dan *value* diatur dalam program menggunakan *trackbars*. Langkah selanjutnya proses *thresholding* pada gambar untuk memisahkan antara objek dengan latar belakang dengan menggunakan citra biner. Selanjutnya proses menghilangkan *noise* menggunakan operasi morfologi. Operasi

morfologi terdapat dilasi, erosi, *opening*, dan *closing* yang berguna untuk menghilangkan *noise* yang ada pada objek. Kemudian apabila objek bola terdeteksi maka pengukuran akan dilakukan menggunakan *binocular disparity* untuk mengukur ke dalam objek dengan kamera *stereo vision*.

2.3 *Color filter* HSV

Color filter HSV merupakan metode mengubah RGB (*Red, Green, Blue*) menjadi HSV (*Hue, Saturation, Value*) menyeleksi objek melalui warna HSV agar mudah dideteksi menggunakan citra biner. *Color filter* HSV mendeteksi warna yang lebih cerah, gambar akan berbentuk citra biner dengan nilai antara 0 (hitam) dan 1 (putih) setelah melakukan proses *color filter* HSV, hasil yang diberikan pada Gambar 2.



Gambar 3 *Color Filter* HSV

Gambar 3 terdapat *trackbars* yang digunakan untuk menyesuaikan *hue, saturation* dan *value* sesuai dengan nilai yang di masukan. Ketika selesai menyesuaikan, di masukan ke dalam variabel yang digunakan pada program yang dibuat. Editor yang dipakai pada penelitian ini menggunakan *visual studio code* milik Microsoft. Berikut ini adalah cara konversi RGB menjadi HSV sebagai berikut [16].

$$R' = \frac{R}{255} \tag{1}$$

$$G' = \frac{G}{255} \tag{2}$$

$$B' = \frac{B}{255} \tag{3}$$

$$Cmax = \max(R', G', B') \tag{4}$$

$$Cmin = \min(R', G', B') \tag{5}$$

$$\Delta = Cmax - Cmin \tag{6}$$

$$H = \begin{cases} 0° & \Delta = 0 \\ 60° \times \left(\frac{G' - B'}{\Delta} \bmod 6\right), & Cmax = R' \\ 60° \times \left(\frac{B' - R'}{\Delta} + 2\right), & Cmax = G' \\ 60° \times \left(\frac{R' - G'}{\Delta} + 4\right), & Cmax = B' \end{cases} \tag{7}$$

$$S = \begin{cases} 0 & , Cmax = 0 \\ \frac{\Delta}{Cmax} & , Cmax \neq 0 \end{cases} \tag{8}$$

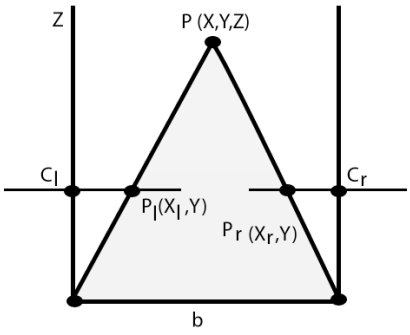
$$V = Cmax \tag{9}$$

Persamaan 1 sampai dengan Persamaan 9 merupakan cara konversi nilai RGB menjadi nilai HSV. Ketika nilai RGB sudah didapatkan di masukan ke dalam Persamaan 7 sampai dengan 9 hingga didapatkan nilai C*max* dan C*min*.

2.4 Metode Pengukuran Jarak

Metode pengukuran jarak yang menggunakan kamera stereo dengan pemanfaatan posisi dari objek kedua kamera dengan waktu *real-time* [16]. Metode ini disebut dengan *binocular disparity*. Metode *binocular disparity* merupakan metode menggunakan dua buah kamera yang dipasang secara paralel dengan *baseline* yang diketahui panjangnya [15]. Gambar 4 menjelaskan P_l dan P_r merupakan hasil pantulan pada bidang proyeksi, C_l dan C_r merupakan *principal point*. Nilai *principal point* didapatkan pada saat kalibrasi dari kedua kamera, sedangkan b merupakan jarak antar pusat pantulan. Maka didapatkan persamaan untuk mencari nilai Z.

$$Z = \frac{f \times B}{d} \tag{10}$$

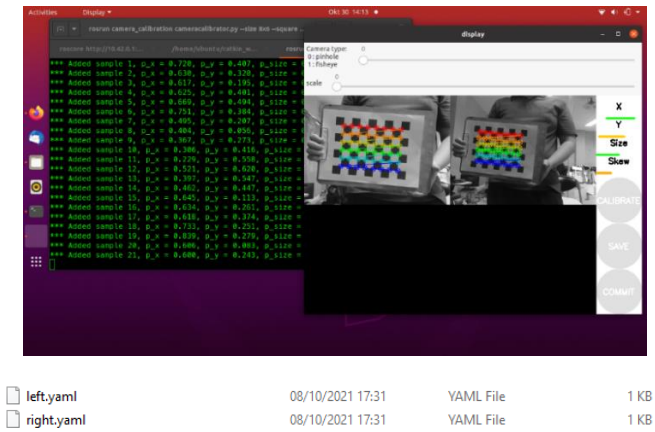


Gambar 4 Ilustrasi Persamaan Pengukuran Objek Menggunakan Dua Kamera

Z merupakan kedalaman yang diukur, f adalah *focal length* dari dua buah kamera, B merupakan *baseline* antara kedua kamera dan d merupakan *disparity*. *Disparity* didapatkan ketika titik P_l dan P_r diambil selisih dari kedua titik tersebut, kemudian di masukan dalam Persamaan 10.

2.5 Kalibrasi *stereo camera* pada turtlebot3

Kalibrasi kamera pada turtlebot3 menggunakan 2 kamera atau disebut dengan *stereo camera*. Kalibrasi digunakan untuk menyamakan kedudukan antara dua kamera yang berbeda. Kalibrasi untuk mencari kedudukan antara x, y dari dua kamera. Ketika kalibrasi berlangsung, objek yang digunakan yaitu pola papan catur. Pola papan catur banyak digunakan karena mudah dan akurasi kecocokan yang tinggi dan gambar harus diambil atau ditangkap dari beberapa sudut dan jarak pandang. Kalibrasi kamera bertujuan untuk mengurangi *error* pada saat pengukuran kedalaman bola. Berikut ini adalah kalibrasi pada dua kamera yang digunakan pada turtlebot3.



Gambar 5 Kalibrasi Dua Kamera Pada Turtlebot3

Gambar di atas merupakan ketika mencari sudut dan jarak pandang yang berbeda untuk mencari posisi yang cocok untuk kedua kamera, setelah selesai kalibrasi dua kamera, *file* akan dibentuk dalam bentuk yaml kemudian *file* yaml dapat digunakan untuk kedua kamera pada saat menjalankan perintah *stereo_usb_camera.launch*. Kamera dapat digunakan sesuai dengan nilai kalibrasi yang dilakukan.

3. HASIL DAN PEMBAHASAN

3.1 Pengujian kedalaman objek dengan berdasarkan nilai pencahayaan

3.1.1 Pengujian kedalaman objek dengan pencahayaan 106 lux

Pengujian dengan intensitas 106 *lux* pada ruangan. Robot dalam keadaan diam mendeteksi bola dengan menggunakan metode HSV atau *hue*, *saturation* dan *value*. Bola ditempatkan pada titik-titik tertentu untuk mengukur jarak ke dalam. Berikut ini adalah hasil dari pengukuran jarak kedalaman dengan menggunakan dua kamera pada turtlebot3. Hasil dari pengukuran kedalaman menggunakan 106 *lux* terdapat pada Tabel 1. Mencari persentase kesalahan pada jarak dapat menggunakan rumus sebagai berikut.

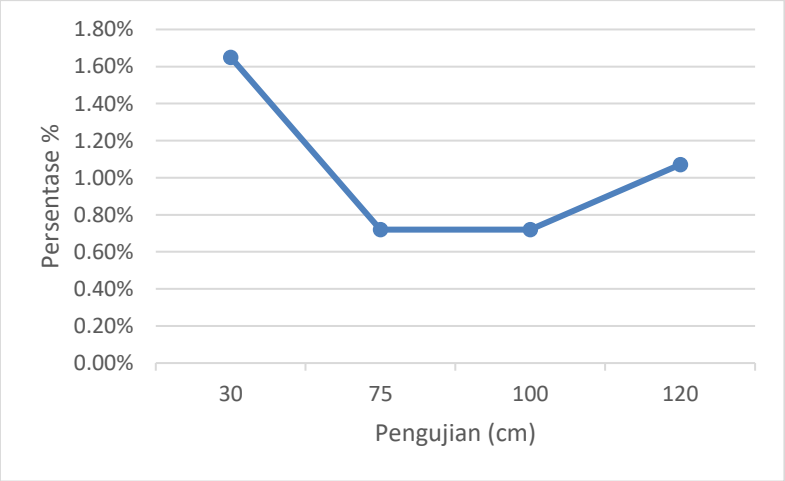
$$Error(\%) = \frac{|\text{jarak sebenarnya} - \text{jarak komputasi}|}{\text{jarak sebenarnya}} \times 100\% \quad (11)$$

Berikut ini adalah hasil dari pengukuran jarak kedalaman dengan menggunakan dua kamera pada turtlebot3.

Tabel 1 Hasil Pengukuran Kedalaman Pada Jarak Yang Ditentukan Dengan Pencahayaan 106 lux

Jarak Komputasi	Jarak Sebenarnya (cm)	Persentase *Error* (%)
30,394	30	1,65
70,5455	75	0,72
100,727	100	0,72
118.,714	120	1,07

Tabel 1 merupakan hasil dari pengujian yang dilakukan menggunakan pencahayaan 106 *lux* memiliki perbedaan pada setiap jarak yang diukur. Tampilan grafik yang didapatkan terdapat pada Gambar 5.



Gambar 6 Grafik Pengujian Penglihatan Kamera Pada Ruangan Dengan Intensitas 106 lux

Gambar 6 merupakan tampilan hasil berbentuk grafik. Hasil pada ruangan 106 *lux* memiliki tingkat kesalahan terbesar 1,65% pada jarak 30cm. Pencahayaan pada ruangan dapat mempengaruhi tingkat pengukuran jarak kedalaman pada bola.

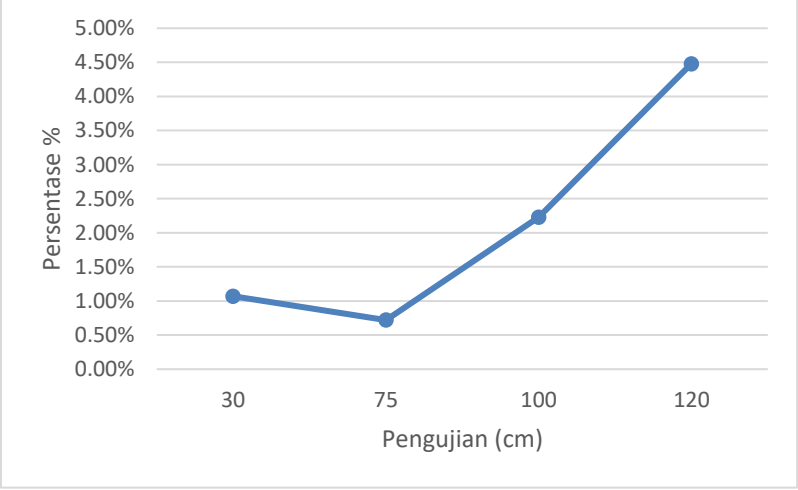
3.1.2 Pengujian kedalaman objek dengan pencahayaan 53 lux

Pengujian yang dilakukan pada pencahayaan suatu ruangan dengan nilai 53 *lux*. Pengujian yang dilakukan sama seperti pengujian sebelumnya, hanya saja berbeda nilai *lux* yang digunakan. Robot turtebot3 ditempat pada ruangan yang sama dengan pencahayaan yang berbeda nilainya. Nilai yang dihasilkan pada pengujian dengan pencahayaan 53 *lux* memiliki perbedaan nilai dengan pengujian dengan tingkat pencahayaan nya 106 *lux*. Parameter yang digunakan pada HSV sama seperti pengujian sebelumnya untuk membedakan ketika objek pada nilai *lux* yang berbeda. Menghitung persentase kesalahan pada pengukuran menggunakan Persamaan 11. Berikut ini adalah hasil pengujian dengan pencahayaan 53 lux pada ruangan yang sama.

Tabel 2 Hasil Pengukuran Kedalaman Pada Jarak Yang Ditentukan Dengan Pencahayaan 53 Lux

Jarak Komputasi	Jarak Sebenarnya (cm)	Persentase *Error* (%)
29,6786	30	1,07
75,5455	75	0,72
97,767	100	2,23
114,621	120	4,48

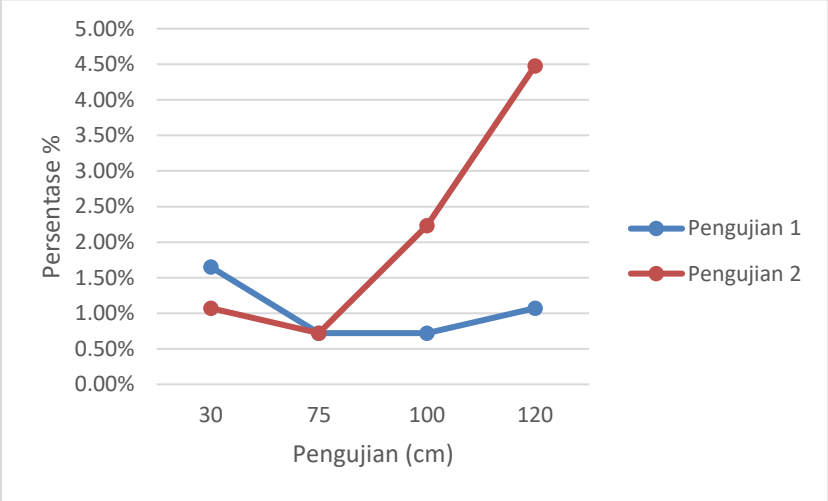
Tabel 2 merupakan hasil yang didapatkan pada ruangan dengan pencahayaan 53 *lux* memiliki tingkat kesalahan terbesar 4,48% pada jarak 120cm. Gambar 6 merupakan grafik hasil dari pencahayaan 53 *lux*. Hasil di bawah dari nilai *lux* memiliki persen kesalahan yang besar pada titik tertentu dikarenakan objek yang tidak terlihat dengan jelas



Gambar 7 Grafik Pengujian Penglihatan Kamera Pada Ruangan Dengan Intensitas 53 Lux

Gambar 7 terjadi penaikan kesalahan pada pendeteksian yang dipengaruhi oleh faktor pencahayaan yang digunakan pada ruangan, sehingga objek tidak terlalu terlihat jelas oleh kamera yang digunakan

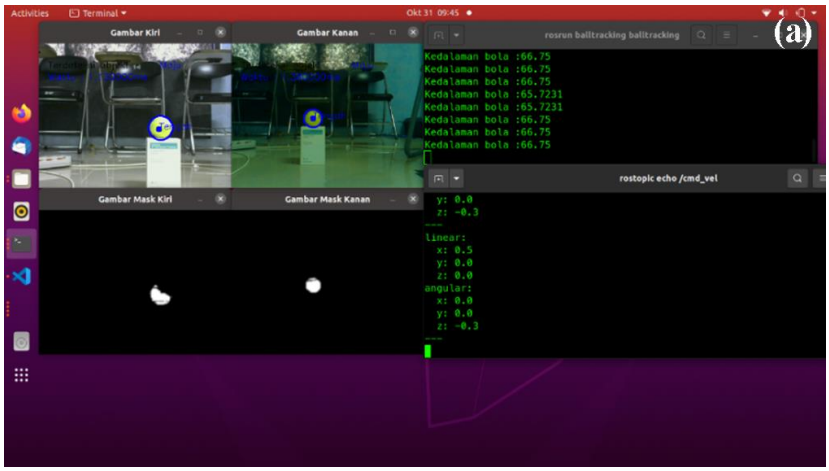
Perbandingan antara pengujian pertama dan pengujian kedua memiliki tingkat persentase kesalahan yang beragam sesuai dengan jarak yang dideteksi. Pengukuran mengalami peningkatan persentase kesalahan pada jarak 120cm menggunakan 53 *lux*. Berdasarkan penelitian [17] ketika pada jarak 40 cm memiliki tingkat akurasi 99,25%, ketika pada jarak 180cm tingkat akurasi menurun 95,33% hingga *error* yang dihasilkan menjadi 4,66%. Penelitian ini membuktikan bahwa ketika objek jauh akurasi yang didapat akan menurun sedangkan ketika objek dekat akurasi yang didapat mendekati tingkat keberhasilan yang tinggi. Perbedaan *lux* pada ruangan dapat mempengaruhi pendeteksian pada objek bola, untuk menyesuaikan dengan keadaan ruangan perlu mengatur HSV agar dapat memperkecil objek yang terdeteksi [18]. Sistem yang digunakan pada turtlebot3 yaitu mendeteksi *contour* terbesar untuk mendeteksi objek bola, ketika HSV tidak diatur maka benda yang memiliki warna sama seperti objek bola dapat terdeteksi dan menggunakan operasi morfologi seperti *opening*, dan *closing* untuk memperkecil objek bola dari gangguan *noise* yang tidak perlu. Gambar 8 merupakan gambar grafik perbedaan antara persentase kesalahan pengujian pertama dan kedua.

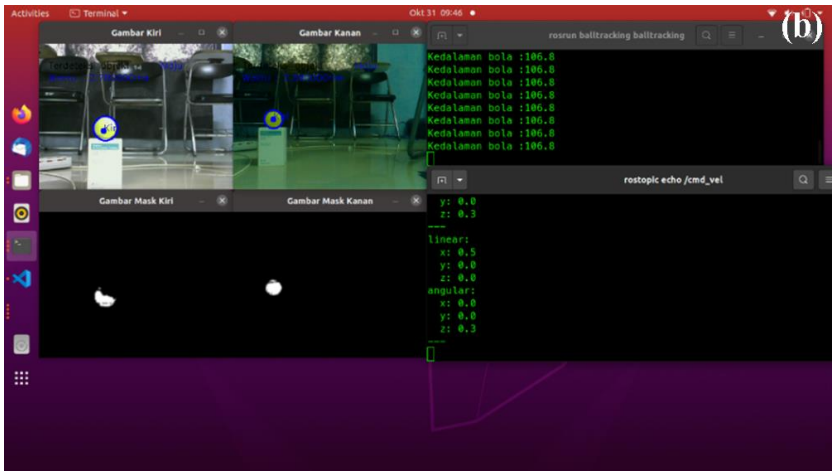


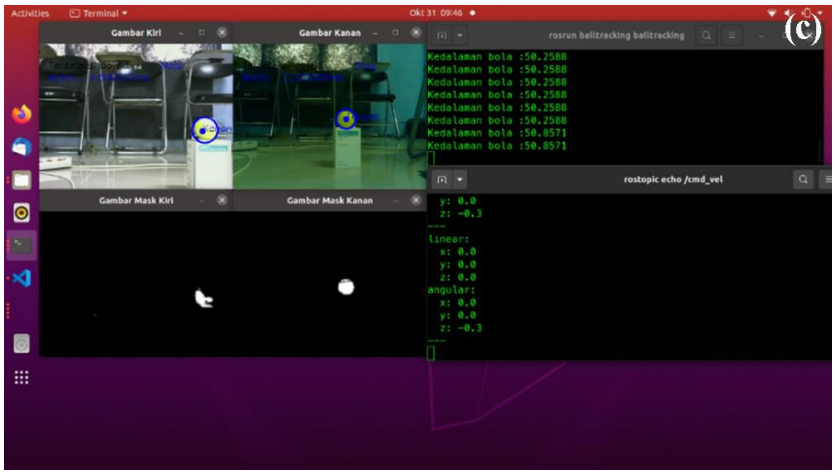
Gambar 8 Grafik Perbandingan Persentase Kesalahan Pada Pengujian Pertama Dan Kedua

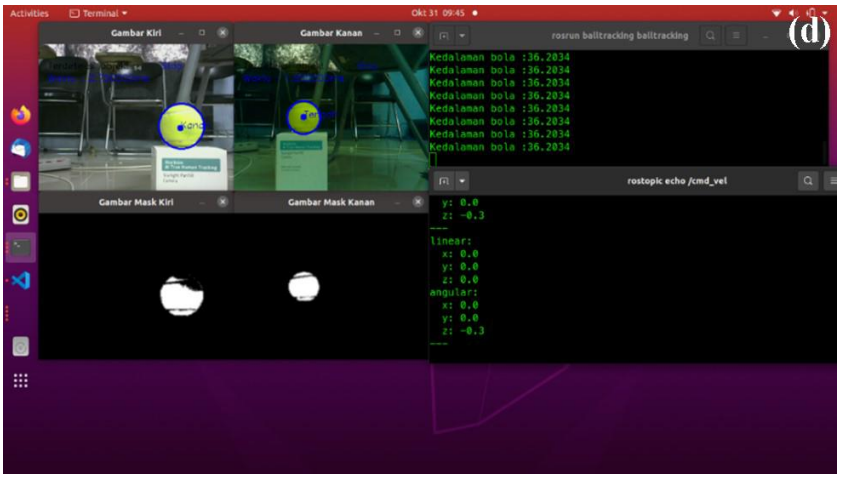
3.2 Pengujian Ketika Robot Keadaan Berjalan

Pengujian yang dilakukan untuk berjalan menggunakan metode yang diterapkan. Metode yang digunakan adalah untuk pendeteksian objek bola menggunakan metode warna HSV dan untuk pengukuran objek bola menggunakan metode *binocular disparity* pada kamera stereo. Turtlebot3 akan bergerak ketika bola berada jauh dari jarak pandang kedua kamera dengan nilai > 51cm , ketika turtlebot3 mendekat ke arah bola dengan nilai <= 51cm maka raspberry pi akan memberikan perintah pada opencr untuk berhenti. Nilai yang digunakan agar tidak terlalu dekat dengan robot. Turtlebot3 akan bergerak ke arah kiri ketika bola berada pada jarak pandang atau daerah sebelah kiri dengan kecepatan konstan 0.2 dan untuk mengarah kanan menggunakan nilai -0.2. Turtlebot3 dapat di gerakan menggunakan *library geometry_msgs* yang sudah ada pada *library* ROS. *Geometry_msgs* adalah perintah untuk menggerakkan robot secara *linear* atau *angular*. Berikut ini adalah turtlebot3 dapat berjalan ketika mendeteksi bola dengan jarak yang sudah ditentukan tertera pada Gambar 9.





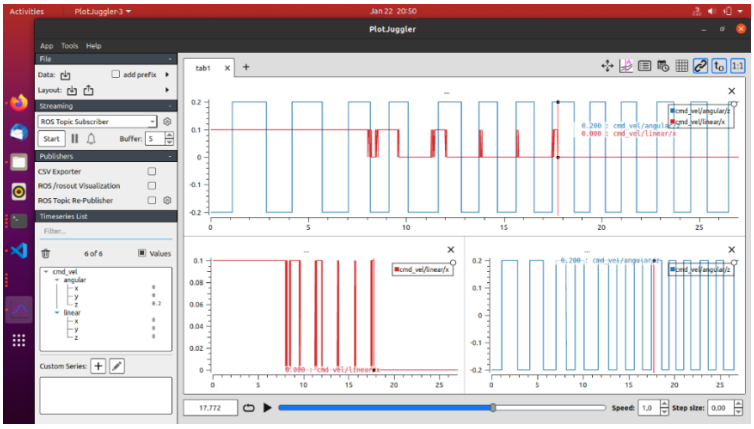




Gambar 9 Turtlebot3 Bergerak (a) Maju (b) Kiri (c) Kanan (d) Berhenti

Gambar 9 pengujian yang dilakukan merupakan pengujian yang dilakukan untuk robot berjalan sesuai dengan penglihatan dari kamera stereo. Gambar 9a turtlebot3 akan bergerak ketika bola berada jauh dari jarak pandang kedua kamera dengan nilai > 51cm sehingga turtlebot3 akan bergerak sesuai dengan objek yang dideteksi. Gambar 9b turtlebot3 akan bergerak ke arah kiri ketika bola berada pada daerah sebelah kiri sehingga robot akan bergerak ke arah kiri dengan kecepatan secara konstan. Gambar 9c turtlebot3 akan bergerak ke arah kanan ketika bola berada pada daerah sebelah kanan sehingga robot akan bergerak ke arah kanan dengan kecepatan secara konstan. Gambar 9d turtlebot3 ketika berhenti dengan nilai yang ditentukan pada *listing code*.

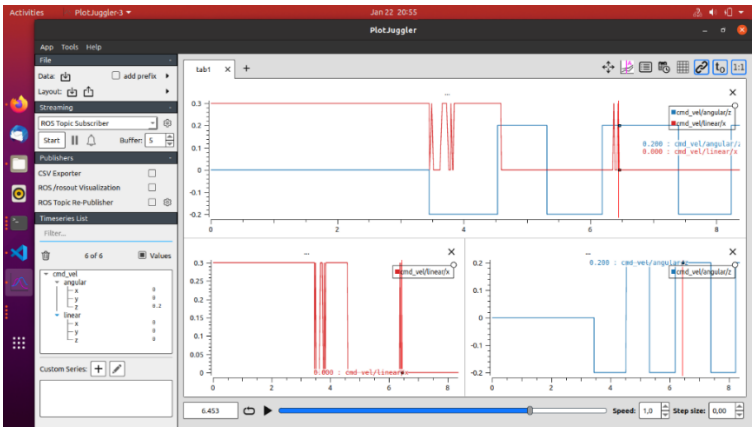
Robot turtlebot3 memiliki tingkatan kecepatan laju yang dapat diubah sesuai dengan kebutuhan dari *user* yang ingin diberikan. Maksimal kecepatan yang didapat pada *servo* yaitu 0,5m/s. Kecepatan laju pada robot memiliki 3 pengujian dengan jarak 0cm hingga 125cm yaitu 0,1m/s; 0,3m/s; dan 0,5m/s. Berikut ini adalah salah satu data yang didapat tertera pada Gambar 10.



Gambar 10 Hasil Kecepatan Laju Robot Menggunakan 0,1m/s

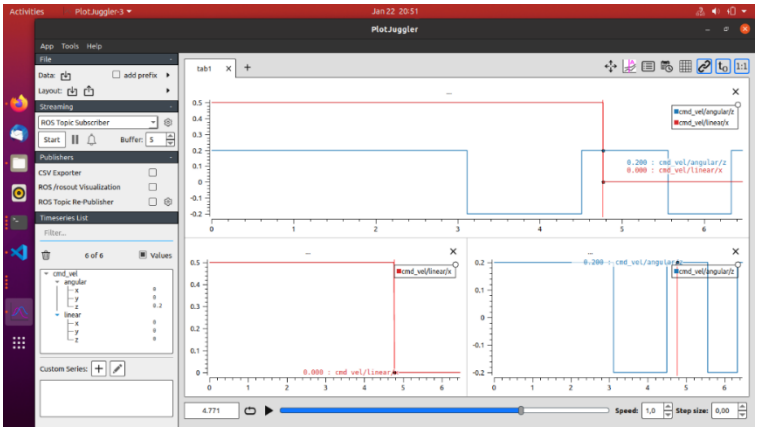
Gambar 10 merupakan hasil kecepatan laju robot menggunakan 0,1m/s terlihat lebih lama dengan waktu 17,7 detik dengan jarak awal hingga 125cm. Faktor yang mempengaruhi yaitu robot masih berbelok ke kanan dan kiri yang membuat pergerakan robot melambat, dalam hal ini ketika objek masuk ke wilayah kanan atau kiri robot akan menyesuaikan posisi objek. Ketika robot sudah mencapai

titik posisi objek maka robot akan diperintahkan untuk berhenti. Berikut ini adalah salah satu data yang didapat tertera pada Gambar 11.



Gambar 11 Hasil Kecepatan Laju Robot Menggunakan 0,3m/s

Gambar 11 merupakan hasil kecepatan laju robot menggunakan 0,3m/s terlihat lebih cepat dibandingkan dengan menggunakan kecepatan 0,1m/s dan waktu yang dibutuhkan yaitu 6,4 detik dengan jarak awal hingga 125cm. Pengujian ini robot fokus mengikuti objek sehingga lebih cepat dan memperkecil pergerakan ke arah kiri dan kanan dibandingkan dengan kecepatan laju menggunakan 0,1m/s. Berikut ini adalah salah satu data yang didapat tertera pada Gambar 12.



Gambar 12 Hasil Kecepatan Laju Robot Menggunakan 0,5m/s

Gambar 12 merupakan hasil kecepatan laju robot menggunakan 0,5m/s terlihat lebih cepat dibandingkan dengan menggunakan kecepatan 0,1m/s dan kecepatan 0,3m/s waktu yang dibutuhkan yaitu 4.7 detik dengan jarak awal hingga 125cm. Kecepatan 0,5m/s adalah batas maksimal pergerakan *servo* yang digunakan. Pengujian ini robot fokus mengikuti objek sehingga lebih cepat dibandingkan dengan kecepatan laju menggunakan 0,1m/s dan 0,3m/s. Tabel 4.3 hasil kecepatan dari 3 pengujian yang dilakukan.

Tabel 3 Hasil Kecepatan *Servo* dengan Jarak 0-125cm

Kecepatan	Waktu Perhitungan	Waktu Sebenarnya
0,1m/s	12,5 detik	17,7 detik
0,3m/s	4,2 detik	6,4 detik
0,5m/s	2,5 detik	4,7 detik

Tabel 3 perhitungan kecepatan menggunakan Persamaan 3.16 dengan kecepatan dan jarak yang sudah diketahui, sedangkan hasil sebenarnya dengan melakukan pengujian yang dijalankan. Faktor yang mempengaruhi robot yang bergerak lebih lambat yaitu robot masih belok ke kanan dan ke kiri dikarenakan robot menyesuaikan posisi objek berada pada wilayah kiri atau wilayah kanan yang berakibat robot bergerak lebih lambat, dan bagian kamera masih bergerak ketika permukaan yang tidak rata. Perhitungan waktu ketika jarak lebih jauh yaitu dari 0 sampai dengan 450cm yang tertera pada Tabel 4.

Tabel 4 Perkiraan Waktu dengan Jarak 0-450cm

Kecepatan	Waktu
0,1m/s	45 detik
0,3m/s	15 detik
0,5m/s	9 detik

Tabel 4 merupakan hasil perhitungan yang dilakukan ketika menggunakan arena 450cm atau 4.5 meter. Penggunaan kecepatan 0,5m/s dapat mencapai titik lebih cepat dibandingkan dengan 0,1 dan 0,3m/s. Kecepatan pada Tabel 4 merupakan hanya sebagai perhitungan apabila tidak terjadinya gangguan yang dapat menambah waktu seperti kamera bergetar dan permukaan yang tidak rata yang berakibat robot bergerak ke kiri dan ke kanan.

4. KESIMPULAN

4.1 Kesimpulan

Kesimpulan saat pengujian dengan menggunakan metode HSV sudah dapat mendeteksi objek bola, pengukuran objek bola sudah dapat dilakukan menggunakan metode *binocular disparity* dengan menggunakan kamera stereo. Perbedaan pada intensitas cahaya berpengaruh pada kedua pengujian. Parameter nilai HSV dari kedua pengujian bernilai sama. Pengujian pertama memiliki persentase kesalahan terbesar yaitu 1,65% pada jarak 30cm sedangkan untuk pengujian kedua memiliki persentase kesalahan terbesar yaitu 4,48% pada jarak 120cm. Pendeteksian dan pengukuran menggunakan *lux* yang lebih rendah lebih besar persen kesalahannya dibandingkan dengan *lux* yang lebih besar. Kecepatan laju robot yang ideal yaitu menggunakan kecepatan 0,5m/s.

4.2 Saran

Saran untuk pengembangan berikutnya yaitu lebih baik mengimplementasikan menggunakan laptop dibandingkan dengan *mini pc* dikarenakan spesifikasi yang terbatas, selanjutnya menggunakan kamera dengan spesifikasi yang sama agar mengurangi persentase kesalahan pada pengukuran jarak, dan pencahayaan yang diharapkan yaitu 300 sampai dengan 500 *lux* agar tidak menimbulkan *noise*.

REFERENSI

[1] Nasional, P. P., R. Indonesia, "*Pelaksanaan Kontes Robot Indonesia Daring*," 2020.
[2] Ben-Ari, M., F. Mondada, "*Robots and Their Applications*," Elem. Robot., pp. 1–20, 2018, doi: 10.1007/978-3-319-62533-1_1.
[3] Pandey, A., "*Mobile Robot Navigation and Obstacle Avoidance Techniques: A Review*," Int. Robot. Autom. J., vol. 2, no. 3, 2017, doi: 10.15406/iratj.2017.02.00023.
[4] Kadena, E., H. P. D. Nguyen, L. Ruiz, "*Mobile robots: An overview of data and security*," ICISSP 2021 - Proc. 7th Int. Conf. Inf. Syst. Secur. Priv., no. Icissp, pp. 291–299, 2021, doi: 10.5220/0010174602910299.
[5] Amsters, R., P. Slaets, "*Turtlebot 3 as a robotics education platform*," Adv. Intell. Syst. Comput., vol. 1023, no. January, pp. 170–181, 2020, doi: 10.1007/978-3-030-26945-6_16.
[6] Raje, S. D., "*Evaluation of ROS and Gazebo Simulation Environment using TurtleBot3 robot*," no. April, 2020.

[7] Jalil, A., "*Robot Operating System (Ros) Dan Gazebo Sebagai Media Pembelajaran Robot Interaktif*," Ilk. J. Ilm., vol. 10, no. 3, pp. 284–289, 2018, doi: 10.33096/ilkom.v10i3.365.284-289.

[8] Spina, M., S. Gualtieri, F. De Rango, "*Integrating ROS and Gazebo Tools with a Network Security Module to Support Secure Autonomous Robot Coordination*," no. Simultech, pp. 369–377, 2021, doi: 10.5220/0010566303690377.

[9] Starry, Z. N., A. N. Jati, F. C. Hasibuan, "*SIMULASI OPENSLAM MENGGUNAKAN TURTLEBOT3 PADA ROS*," vol. 8, no. 5, pp. 6201–6211, 2021.

[10] Mordvintsev, A., K. Abid, "*OpenCV-Python Tutorials Documentation*," OpenCV Python Doc., p. 269, 2017, [Online]. Available: https://media.readthedocs.org/pdf/opencv-python-tutroals/latest/opencv-python-tutroals.pdf.

[11] Guennouni, S., A. Ahaitouf, A. Mansouri, "*Multiple object detection using OpenCV on an embedded platform*," Colloq. Inf. Sci. Technol. Cist, vol. 2015-Janua, no. January, pp. 374–377, 2015, doi: 10.1109/CIST.2014.7016649.

[12] Tripathi, A., T. V. Ajay Kumar, T. K. Dhansetty, J. Selva Kumar, "*Real time object detection using CNN*," Int. J. Eng. Technol., vol. 7, no. 2, pp. 33–36, 2018, doi: 10.14419/ijet.v7i2.24.11994.

[13] Mushawwir, L. A., I. Supriana, "*Deteksi dan Tracking Objek untuk Sistem Pengawasan Citra Bergerak*," Konf. Nas. Inform. 2015 Deteksi, vol. 2354–645X/, no. October, pp. 1–10, 2015.

[14] Chandan, G., A. Jain, H. Jain, Mohana, "*Real Time Object Detection and Tracking Using Deep Learning and OpenCV*," Proc. Int. Conf. Inven. Res. Comput. Appl. ICIRCA 2018, no. July, pp. 1305–1308, 2018, doi: 10.1109/ICIRCA.2018.8597266.

[15] Mansour, M., P. Davidson, O. Stepanov, R. Piché, "*Relative importance of binocular disparity and motion parallax for depth estimation: A computer vision approach*," Remote Sens., vol. 11, no. 17, 2019, doi: 10.3390/rs11171990.

[16] Marzuqi, I., G. P. Arinata, Z. M. A. Putra, A. Khumaidi, "*Segmentasi dan Estimasi Jarak Bola dengan Robot Menggunakan Stereo Vision*," pp. 140–144, 2017.

[17] SOLAK, S., E. D. BOLAT, "*A new hybrid stereovision-based distance-estimation approach for mobile robot platforms*," Comput. Electr. Eng., vol. 67, pp. 672–689, 2018, doi: 10.1016/j.compeleceng.2017.10.022.

[18] Muharom, S., S. Asnawi, A. Bachri, "*Robot Pengikut Target Berdasarkan Bentuk dan Warna Menggunakan Metode HSV Untuk Aplikasi Assistant Robot*," JE-Unisla, vol. 6, no. 1, 2021, [Online]. Available: http://jurnalteknik.unisla.ac.id/index.php/elektronika/article/view/571.