Sensor Penglihatan Pelacakan Objek pada Navigasi Robot Bergerak

Ilham Akbar Al asy' ari[1], Agung Nugraha Jati[2], Casi Setianingsih[3]
Prodi S1 Sistem Komputer, Fakultas Teknik Elektro, Universitas Telkom
ilhamasy@student.telkomuniversity.ac.id[1], agungnj@telkomuniversity.ac.id[2],
setiacasie@telkomuniversity.ac.id[3]

Abstrak – *Perkembangan mobile robot navigation sangat signifikan seperti Autonomoius Mobile Robot. Salah satu robot software platform yang popular yaitu ROS(Robot Operating System). ROS mempermudahkan dalam menyusun sistem pada robot yang mana pada dasarnya begitu rumit. ROS mempunyai alat bantu yang memudahkan kita, serta banyak komunikasi didalamnya. Dalam penelitian ini yang mana melakukan tracking pada target yang berupa ar_track_alvar yang berupa augmented reality yang divisulisasikan dengan alat bantu ROS. Penelitian ini membahas tentang menentukan kecepatan sudut untuk mencari marker, menentukan kecepatan linear untuk melakukan tracking, dan jarak minimal antar Robot dan Target.*

Kata kunci*: ROS, vision based sensor, camera calibration, ar_track_alvar*

1. Latar Belakang

Pada perkembangan teknologi, banyak penilitian – penelitian membuat robot layaknya seperti makhluk hidup, yang mana bisa bergerak, mengenali suatu objek dan nantinya dapat membantu pekerjaan manusia. Tujuan penelitian robot saat ini adalah untuk melakukan berbagai macam tugas fisik baik robot yang dikontrol manusia ataupun robot yang diprogram untuk melakukan berbagai tugas. Penelitian yang banyak dilakukan dalam bidang robot yaitu pembuatan *autonomous robot*[1].

Mobile Robot adalah suatu robot yang mana bergerak dari suatu titik ke titik lain dengan misi tertentu. *Mobile Robot* secara luas dimanfaatkan didalam berbagai kehidupan manusia, seperti robot untuk pelayanan, robot untuk militer, robot untuk transportasi, robot untuk didalam air, dan lain sebagainya[2]. Penelitian dalam bidang robot ini sudah semakin berkembang karena adanya penelitian pada robot otonom yang dapat bergerak sendiri karena memiliki kecerdasan buatan untuk dapat menghindari tabrakan, berkomunikasi dan berkoordinasi dalam melakukan pengambilan keputusan[1].

Untuk membuat perangkat lunak pada system robot tentu akan sulit terutama karena skala dan ruang lingkup robotika akan terus berkembang mengikuti perkembangan jaman. Oleh karna itu untuk memudahkan untuk merancang system robot dalam penelitian ini menggunakan *robot software platform* yang bernama ROS(Robot Operating System). Didalam ROS terdapat banyak *tools*, lapisan komunikasi, dan simulasi untuk perancangan system robot[3].

2. Metode Penelitian

Mobile Robot adalah suatu robot yang mana bergerak dari suatu titik ke titik lain dengan misi tertentu. *Mobile Robot* secara luas dimanfaatkan didalam berbagai kehidupan manusia, seperti robot untuk pelayanan, robot untuk militer, robot untuk transportasi, robot untuk didalam air, dan lain sebagainya[2]. Perkembangan teknologi tentang mobile robot yaitu Single Robot atau sering dikenal dengan autonomous Robot, adalah konstruksi robot yang ciri khasnya adalah mempunyai actuator berupa roda untuk menggerakkan keseluruhan badan robot tersebut,

sehingga robot tersebut dapat melakukan perpindahan posisi dari satu titik ke titik yang lain[4]. Single robot dapat mengelesaikan tugasnya dengan tersendiri dapat berupa titik *end* atau menyelesaikan sebuah masalah sehingga mencapai titik tujuan.

ROS adalah suatu *meta operating system* yang menyediakan service dalam cakupan abstraksi *hardware*, mengkontrol *low level device*, implementasi fungsi yang sering digunakan pada system robot, pengiriman *message* antar proses dan manajemen *package*. Selain itu menyediakan *tools* dan *library* untuk memperoleh, membangun, menulis dan menjalan kode di beberapa komputer.

Istilah yang sering muncul di ROS diantaranya *node, package, message*. Node adalah unit yang paling terkecil dalam *executable processors* atau bisa dikatakan sebagai *single excutable program*, setiap node akan mengirim dan menerima data dengan menggunakan *message communication. Package* adalah suatu kumpulan dari node itu sendiri. *Message* adalah data yang dikirimkan atau diterima melalui *message* antara *node*, dan memiliki bervariasi tipe data diantaranya *integer*, *floating point*, dan *boolean*[3].

2.1 Desain Sistem Pergerakan Robot

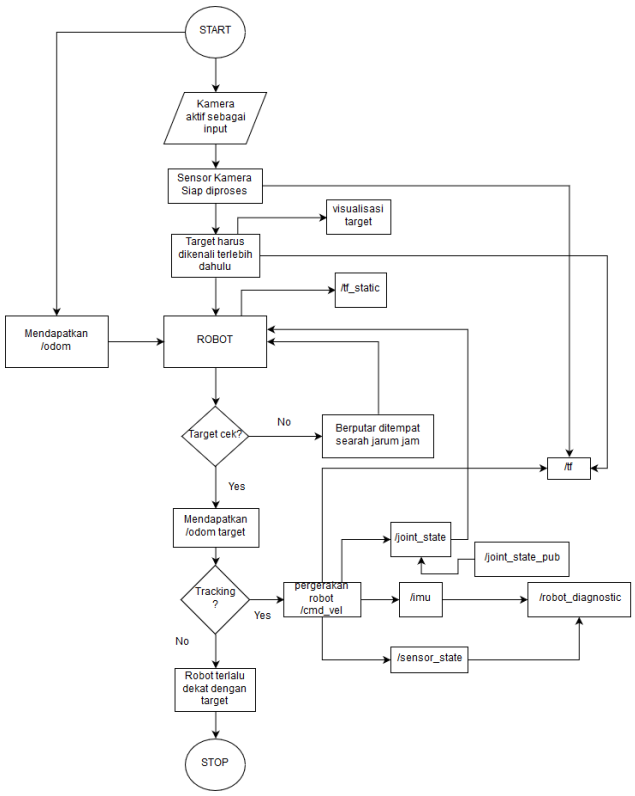


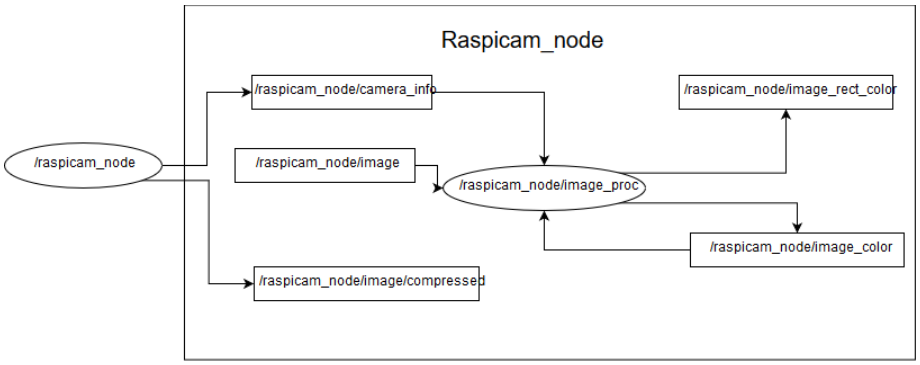
Figure 2. 1 Diagram Alir dari gambaran umum pergerakan *mobile robot*

Didalam diagram alir tersebut, dijelaskan bahwa bermulai dari mendapatkan data sensor dari kamera yang sudah siap diproses yang langkah – langkahnya sudah dijelaskan sebelumnya. Inisialisasi target harus dikenali terlebih dahulu dan target dan robot divisualisasikan melalui Rviz. Sementara itu Robot mendapatkan odometri, proses pergerakan baru dimulai. Bila tidak ada target ke detect robot itu akan berputar ditempat sampai menemukan target. Bila menemukan target dan sudah mendapatkan odometri dari target, robot itu pertama – tama langsung diam menghadap ke target lalu mulai jalan mendekati target dan kelebihan dari sistem ini bila dalam kondisi dinamis dalam arti setelah robot berputar dan menemukan

ISBN: 978-623-7036-34-0

target lalu robot memulai bergerak, target tersebut bisa dipindah pindahkan sehingga robot akan terus mengikutinya sampai dengan robot sangat mendekat ke target, untuk kelemahan dari sistem ini yaitu tidak bisa menghindari halangan.

2.2 Paket *Raspicam_node*



Gambar 1. Proses yang terjadi di dalam paket Raspicam_node yang mana sudah di kalibrasi dan sudah siap untuk diproses selanjutnya

Paket ini dibutuhkan dalam proses mendeteksi target dan dapat mengetahui jarak targetnya dengan menggunakan kamera. Namun butuh proses kamera harus dikalibrasi terlebih dahulu dengan menggunakan papan catur, selanjutnya dilakukan konfigurasi kamera seperti diatur kecerahan, kontras, dan lain sebagainya. Setelah itu gambar yang tadi dikonfigurasi harus diperbaiki, sehingga mengalami distorsi selain itu kalibrasi kamera tidak cukup untuk hasil yang optimal untuk mengukur jarak antara kamera dengan objek dan dibutuhkan bila memasuki ar_track_alvar dan gambar harus dikompress terlebih dahulu seupaya mempercepat komunikasi antar node. Namun apa yang kita inginkan didalam perbaikan gambar di dalam image_proc node adalah dengan tipe gambar mentah. Karenanya hasil kompres gambar harus ditransformasi menjadi gambar mentah

2.3 Paket ar_track_alvar

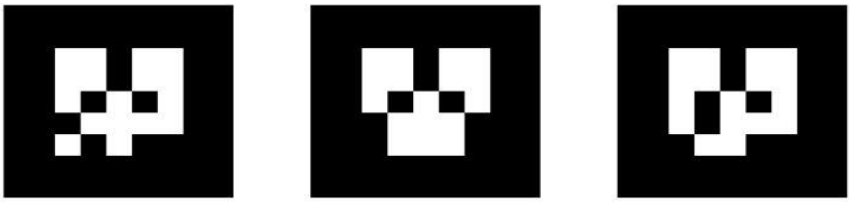
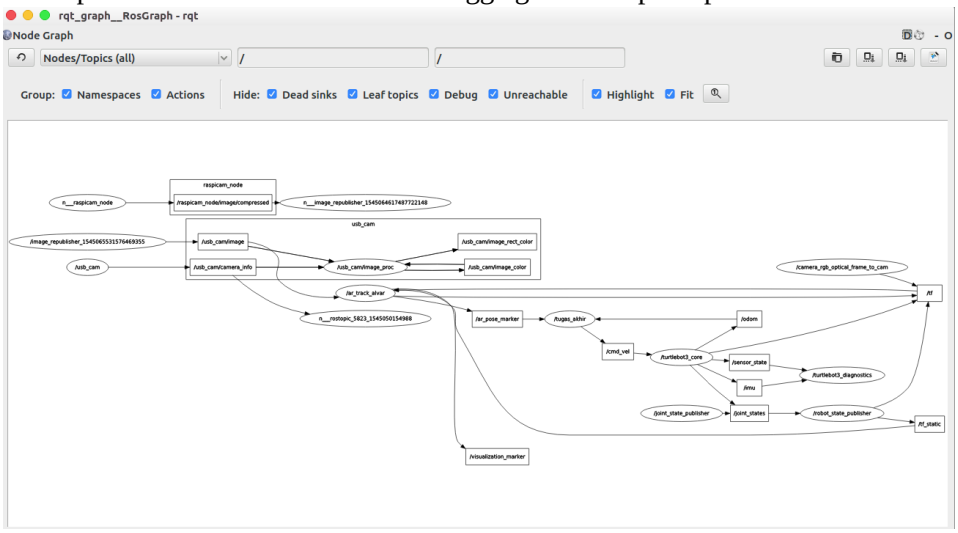


Figure 2. 2 Kind of ar_track_alvar

Paket ini akan menjadi target yang sudah dikenali oleh system robot. Paket ar_track_alvar semacam seperti *barcode* yang terdiri dari 17 jenis marker yang sudah teridentifikasi. Cara kerja paket ar_track_alvar dengan cara mempublikasikan topik /visualization_marker untuk memberi warna ar_track_alvar ke penanda target, dan topik /ar_pose_marker untuk mengirim posisi data target dan untuk proses lanjutan untuk robot pemindahan sistem, topik / tf untuk visualisasi x, y, z koordinat yang nanti nya akan terlihat di alat bantu ROS yaitu *Rviz*. Ada fungsi utama paket ini adalah menghasilkan *augmented reality* (AR) sebagai tanda dari berbagai ukuran, resolusi, dan pengkodean data, mengidentifikasi dan melacak posisi dari *marker*.

ISBN: 978-623-7036-34-0

Alvar sebagai marker atau target yang mana terdapat parameter seperti marker_size yaitu lebar dalam centimeter dari satu sisi perbatasan marker persegi hitam, max_new_marker_error yaitu ambang batas menentukan kapan marker dapat dideteksi dibawah ketidakpastian, max_track_error yaitu ambang batas yang menentukan berapa banyak kesalahan pelacakan dapat diamati sebelum tag dianggap menghilang, output_frame yaitu nama frame yang di publish lokasi cartesian dari tag AR akan relative terhadap topic /base_footprint, camera_image yaitu nama topik yang menyediakan frame kamera untuk mendeteksi tag AR, ini dapat berupa warna tetapi harus merupakan gambar yang sudah di perbaiki melalui image_proc node, camera_info yang menyediakan parameter kalibrasi kamera sehingga gambar dapat diperbaiki.



Gambar 2. 1 visualisasi paket utama pada rqt_graph salah satu alat bantu dama ROS

2.4 Paket Utama

Sebelum proses ini berjalan, robot harus mendapatkan odometri dirinya sendiri dengan menggunakan library tf.transformation yaitu mengubah dari quarternion[x,y,z,w] diubah menjadi euler [x,y,z][5][6]. Tetapi didalam mendapatkan koordinat z atau sebagai teta harus dalam rentang $0 < teta < 2\pi$. Dapat diasumsikan teta sama dengan 3 dan memiliki rotasi perubahan teta seperti ini (3.3, 3.14, -3.14, -3.1, -3) dan memiliki satu perubahan dalam rentang 3.14 sampai -3.14 untuk medeteksi perubahan ini dapat dihitung dengan (teta – teta_terakhir) dan jika hasilnya lebih dari 5 maka disini terjadi pengurangan sehingga (teta – teta_terakhir) – 2π dan berlaku sebaliknya. setelah itu pada fungsi Mencari_Marker(), bila target tidak ditemukan maka robot akan berputar dengan kecepatan 0.4 rad/s searah jarum jam dan bila menemukan odometri dari marker atau target mendapatkanya dengan cara sama seperti odometri pada robot, robot akan berhenti. Pada fungsi Merubah_Arah_robot() robot akan berputar berlawan sampai menghadap target Setelah mendapatkan odometri robot akan terus berputar bila terdeteksi maka robot akan langusng menghadap ke target dan menerima topic /ar_pose_marker bila tidak robot akan terus berputar. Selanjutnya robot akan berjalan yang terus mengikuti robot sampai jarak kondisi robot dengan target berjarak 10 cm. Dalam skema pergerakan robot akan mempublish topik /cmd_vel untuk

ISBN: 978-623-7036-34-0

keceptan dan topic /joint_state untuk sendi pada pergerakan roda yang mana berhubungan dengan /odom mengirim data sebagai kontrol umpan balik.

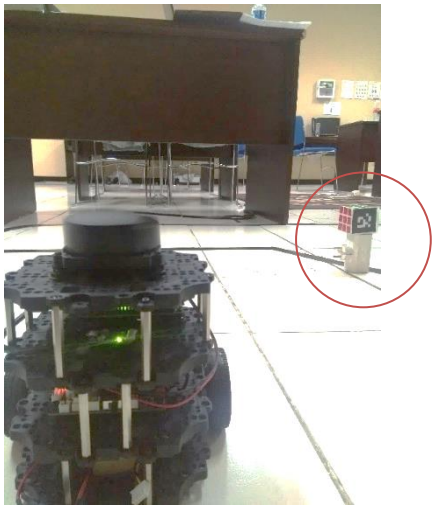


Figure 3. 1 Initial state robot spinning in place searching the object

3. Hasil dan Analisis

3.1 Peforma pengujian pada pengaruh kecepatan mencari target

Dalam pengujian ini, menentukan kecepatan yang optimal dalam mencari Target yang mana merupakan tahap pertama dalam *mobile robot.* Pengujian ini akan diletakkan robot dengan sudut 90 derajat dari arah target lalu robot akan berputar dengan kecepatan tertentu. Capaian hasil pengujian ini yaitu robot berputar sampai menemukan dan mendeteksi targetnya otomatis robot akan diam menghadap target.

Tabel 3. 1 Hasil pengujian kecepatan sudut dalam perputaran mencari taget terhadap jarak dan waktu

Kecepatan(rad/s)	Deteksi?	Jarak(cm)	Rata – Rata Waktu(s)
0,2	V	30	9
0,3	V	30	5,4
0,4	V	30	10,6
0,6	X	30	Error
0,2	V	90	11,4
0,3	V	90	9,6
0,4	X	90	Error
0,6	X	90	Error

3.2 Peforma pengujian mobile robot melakukan tracking dengan target dinamis

Pengujian ini dilakukan untuk menentukan kecepatan dan jarak minimal ke target sehingga robot akan berhenti. Target akan diletakkan sejauh 30 cm didepan robot yang akan berjalan sampai 60 cm kedepan. Pengujian dengan berbagai jarak, sudut antara target dan robot yaitu sebesar 90 derajat yang bermulai dengan mencari marker terlebih dahulu dengan kecepatan 0.3 rad/s.

ISBN: 978-623-7036-34-0

Tabel 3. 2 Hasil pengujian jarak minimal antar target terhadap waktu pada kecepatan tracking

Kecepatan Tracking (m/s)	Deteksi?	Waktu(s)	Jarak minimal antar target sensor kamera(m)	Jarak minimal antar target sebenarnya (m)	Persentase Jarak minimal Error
0.1	✓	23	0.15	0.138	8 %
0.2	-	Error	0.15	-	-
0.1	✓	22	0.10	0.095	5 %
0.2	-	Error	0.10	-	-
0.1	-	Error	0.05	-	-

3.3 Peforma pengujian mobile robot melakukan tracking dengan target statis

Pengujian ini dilakukan dengan cara meletakkan robot dengan berbagai sudut dan dengan berbagai jarak target. Proses berjalannya sistem ini bermulai dari mencari marker sampai dengan menghampiri target yang hanya diam berbeda dari pengujian yang sebelumnya. Bila pengujian jarak antar targetnya sebesar 30 cm maka target diletakkan 10 cm didepannya (dalam pengujian sebelumnya jarak minimal antar target dan robot sebesar 10 cm) maka total jarak nya sebesar 40 cm sehingga capaian dari pengujian ini robot harus tepat bergerak sebesar 30 cm, berlaku kelipatan selanjutnya.

Tabel 3. 3 Hasil pengujian jarak antar target pada sudut yang telah ditentukan dalam waktu

Sudut Robot (derajat)	Jarak antar Target yang ditentukan(cm)	Jarak sebenarnya(cm) (rata-rata)	Selisih Jarak (cm)	Waktu(s) (rata-rata)	Akurasi Deteksi
90	15	14,6	0,4	16,2	100 %
90	20	19,4	0,6	17,6	100 %
90	25	24,4	0,6	19,4	100 %
90	30	29,4	0,6	21,4	100 %
90	45	43,8	1,2	23,2	100 %
90	50	48,33	1,67	36,5	60 %
90	55	52,5	2,5	39	40 %
90	60	-	-	-	0 %
90	65	-	-	-	0 %
90	70	-	-	-	0 %
90	75	-	-	-	0 %

3.4 Analisis

Pengujian Peforma Pengujian pada pengaruh kecepatan mencari target, Dalam rentang 0,4 rad/s sampai 0,6 rad/s robot akan terlalu cepat berputar sehingga dalam kecepatan 0,4 rad/s pada jarak 90 cm yang didapatkan waktunya sangat lama, sedangkan dalam kecepatan 0,6 rad/s pada jarak 30 cm dan 90 cm itu tidak dapat terdeteksi dan terus berputar ditempat dengan kecepatan demikian. Dalam rentang 0,2 rad/s sampai 0,3 rad/s robot tidak mengalami error dan hasilnya juga tidak signifikan diantara keduanya. Tapi jika di uji keduanya pada jarak 30 cm dan 90 cm, hasil pada kecepatan 0,3 rad/s lebih baik dibandingkan 0,2 rad/s. Hal ini dibuktikan pada persamaan (2), bahwa misal dikatakan robot bergerak dari sudut 90 derajat kearah target dan langsung terdeteksi oleh sensor kamera hasilnya melalui perhitungan sebesar 5,23 detik yg mana

ISBN: 978-623-7036-34-0

hasil ini tidak terlalu berbeda dengan pengujian jarak target 30 cm. Maka dari itu dalam pengujian ini kecepatan perputaran mencari target yang paling optimal yaitu 0,3 rad/s dengan jarak 30 cm yang mana menjadi ambang batas optimal untuk kecepatan terhadap waktu yang baik.

Pada Pengujian Peforma pengujian *mobile robot* melakukan *tracking* dengan target yang dinamis Terlihat bahwa pada kecepatan tracking 0,2 rad/s robot akan berjalan cepat sehingga waktu pemrosesan sensor pada kamera tidak terdeteksi jadi terdapat error atau robot akan selalu jalan tanpa berhenti sehingga dapat diambil kesimpulan bahwa pada 0,1 m/s merupakan paling baik dibandingkan 0,2 m/s. Yang kedua yaitu jarak minimal antar target dengan sensor kamera. Pada jarak 0,05 m terdapat error dikarenakan jarak robot dan target terlalu dekat sehingga pada target terlihat sebagian lalu tidak bisa terdeteksi. Pada jarak 0,15 m robot akan tepat diam dalam waktu tracking yaitu 23 sekon sedangkan pada jarak 0,1 m yaitu 22 sekon. Perbedaan ini sangat tipis oleh karnanya dengan mempertimbangkan error antara selisih jarak sebenernya dan jarak yang sudah ditentukan. Pada jarak 0,15 m jarak yang sebenarnya dalam arti robot tepat diam didepan target sebesar 0,138 m sehingga persentase error yang didapatkan yaitu 8 %. Sedangkan pada jarak 0,1 m dan jarak yang sebenarnya sebesar 0,095 m sehingga persentase error yang didapatkan yaitu 6 %. Error ini disebabkan oleh perbedaan persepsi antara kamera dan dilingkungan sebenarnya yang dikarenakan pada saat proses kalibrasi kamera dan proses konfigurasi kamera.

Pada pengujian pada pengujian peforma pengujian *mobile robot* melakukan *tracking* dengan target yang statis Diatas jarak 60 cm semua percobaan error tidak bisa terdeteksi hal ini dikarenakan beberapa faktor yaitu intensitas cahaya, target terlalu jauh sehingga target terlalu kecil lalu tidak bisa terbaca atau jumlah tinggi piksel pada target yang kurang, pengaruh spesifikasi kamera. Intensitas cahaya berkemungkinan sangat kecil karena dalam percobaan ini menggunakan 57 lux yang mana percobaan sebelumnya perfomansi yang cukup bagus. Dalam penelitian ini menggunakan ar_track_alvar suatu library dari ROS yang mana terdapat ukuran marker sebesar 8 cm x 8 cm. Kamera yang digunakan pada penelitian ini menggunakan *focal length* sebesar 3,04 mm. Sebelumnya *Focal Length* merupakan jarak dalam millimeter antara bagian tengah elemen optik lensa dengan gambar yang terbentuk pada sensor. Semakin besar *Focal Length* semakin sempit sudut pandang dan semakin sempit perspektif terhadap objek tetapi bisa menangkap kamera dengan jarak jauh, sedangkan semakin pendek *focal length* semakin lebar sudut pandang dan semakin lebar perspektif terhadap objek tetapi tidak bisa menangkap jarak jauh. Faktor spesifikasi kamera ini berkemungkinan besar dengan tidak terdeteksinya target dengan jarak diatas 60 cm. Spesifikasi kamera dengan sebagai berikut resolusi 640 x 480, *focal length* 3,04 mm, *fixed to focus* 1 m sampai tak terhingga, *sensor height* 3,42 mm. Dalam persamaan (3) menghitung jarak antar kamera dengan target. Misal pada jarak 30 cm tertangkap sebuah tinggi target dalam gambar sebesar 151 piksel, dengan ukuran tinggi target sebesar 8 cm. Hasil dari perhitungannya didapatkan sebesar 30,13 cm yang mana selisih antar jarak yang ditentukan dengan jarak perhitungan sebesar 0,13 cm. Sedangkan pada jarak 60 cm tertangkap sebuah tinggi target dalam gambar sebesar 64 piksel dengan spesifikasi kamera dan ukuran target yang sama didapatkan hasil perhitungan sebesar 71,1 cm, bila dihitung selisihnya sebesar 11,1 cm. Hal ini disebabkan karena piksel yang tertangkap sangat kecil sehingga kamera sebagai pendeteksi target tidak berfungsi dengan baik. Selanjutnya bisa dibuktikan dengan persamaan (4) mencari ukuran target sebenarnya yang mana ukuran target sebesar 8 cm. Misal dalam jarak 30 cm hasil perhitungan target sebenarnya sebesar 7,96 cm dan bila dihitung selisih antara perhitungan dan ukuran target sebenarnya yaitu 0,04 cm. Dalam hal ini masih terdeteksi oleh ar_track_alvar karena selisih yang begitu kecil. Dalam jarak 60 cm hasil perhitungan target sebenarnya yang didapat yaitu 6,43 cm dan bila dihitung selisih antara perhitungan dan ukuran target sebenarnya yaitu 1,57 cm. Dalam hal ini tidak bisa terdeteksi oleh ar_track_alvar dikarenakan parameter max_track_error sebesar 0,2.

ISBN: 978-623-7036-34-0

3.5 Persamaan

Untuk mencari target dan sudah diterdeteksi robot akan berputar balik arah menghadap ke target.

$$arahteta = \arctan\frac{y_{marker}}{x_{marker}} \dots\dots\dots (1)$$

Pada analisis pengujian perputaran robot dan waktu yang diperlukan bila robot sudah tepat terdeteksi

$$Waktu\ tepat\ mendapat\ target(s) = \frac{derajat\ yang\ ditentukan(°)}{kecepatan\ sudut\ \left(\frac{°}{s}\right)} \dots\dots (2)$$

Pada analisis pengujian menentukan jarak minimum yang terlacak oleh kamera

$$D_r = \frac{f_l\ x\ h_o\ x\ h_r}{h_i\ x\ h_s} \dots . (3)$$

$Keterangan$:
$D_r =$ $Jarak\ antar\ kamera\ dengan\ target(mm)$
$f_l =$ $Focal\ Length(mm)$
$h_o =$ $Tinggi\ objek\ sebenarnya(mm)$
$h_r =$ $Tinggi\ resolusi\ pada\ kamera\ (piksel)$
$h_i =$ $Tinggi\ gambar\ (piksel)$
$h_s =$ $Sensor\ Height\ (mm)$

4. Kesimpulan

Robot Operating System (ROS) merupakan suatu platform untuk merancang sistem robot dan memudahkan dalam merancangnya karena dalam proses pembuatannya terdapat alat bantu yang dapat membantu dalam kesulitan.

Alur dari proses ini bermulai dari berputar untuk mencari target dengan menggunakan sensor kamera, bila sudah mendeteksi target robot akan berjalan mengikuti target. Target dapat diam atau bergerak. Sampai robot berhenti pada jarak maksimal antara target

Dalam skema pengujian, didapatkan parameter tertentu agar robot dapat bekerja optimal, seperti ukuran marker 8 x 8 untuk jarak 30 cm sedangkan 10 x 10 untuk jarak diatas 30 cm, kecepatan sudut dalam mencari marker sebesar 0.3 rad/s, kecepatan linier dalam proses tracking sebesar 0.1 m/s, robot akan berhenti pada jarak maksimal pada target sebesar 10 cm. Jangkauan sensor kamera pada robot sampai 60° terhadap target. Derajat kemiringan marker dalam posisi robot sudah menghadap target sampai dengan 45°

Daftar Pustaka

[1] L. Vig, J. A. Adams, and S. Member, “Multi-Robot Coalition Formation,” vol. 22, no. 4, pp. 637–649, 2006.
[2] T. Dewi, P. Risma, Y. Oktarina, and M. T. Roseno, “Neural Network Controller Design for a Mobile Robot Navigation ; a Case Study,” no. September, pp. 19–21, 2017.
[3] D. L. Yoonseok Pyo, Hancheol Cho, Leon Jung, *ROS Robot Programming (English)*. 2017.
[4] R. Illah, *Autonomous Mobile Robots*. .
[5] D. R. Wilkins, “ON QUATERNIONS , OR ON A NEW SYSTEM OF IMAGINARIES IN ALGEBRA,” 2000.
[6] N. Wangsaputra and S. Teknik, “Ilustrasi Penggunaan Quaternion untuk Penanggulangan Gimbal Lock,” 2016.

ISBN: 978-623-7036-34-0