

Implementasi Robot Patroli Sederhana Berbasis Computer Vision dan Deep Learning

Ivan Seya Ananda
Jurusan Informatika
Universitas Islam Indonesia
Yogyakarta, Indonesia
19523232@students.uii.ac.id

Chandra Kusuma Dewa
Jurusan Informatika
Universitas Islam Indonesia
Yogyakarta, Indonesia
Chandra.kusuma@uui.ac.id

Abstract—Kegiatan patroli membutuhkan tingkat konsentrasi dan konsumsi energi yang tinggi, sedangkan makhluk hidup memiliki banyak limitasi dalam melakukan aktivitas yang bersifat repetitif. Makhluk hidup terutama manusia terdapat batasan tingkat konsumsi energi dan persepsi kecerdasan yang berbeda-beda. Robot patroli merupakan salah satu teknologi yang dikembangkan pada aktivitas patroli. Oleh karena itu, penelitian ini bertujuan untuk mensimulasikan robot patroli yang dibangun melalui *framework* atau perangkat lunak Robot Operating System (ROS) yang mampu membangun struktur robot dan lingkungan uji coba simulasi sederhana. Dengan ROS robot akan diuji berbasis *computer vision* dan *deep learning*, pada implementasinya robot menggunakan model You Only Look Once (YOLO) versi kelima atau YOLOv5. Simulasi akan diujikan terhadap jarak dan skenario. Terdapat tiga skenario yang dilakukan meliputi objek tunggal, objek ganda dan objek berjalan pada model manusia dan mobil. Hasil dari pengujian model manusia diam dan bergerak mendapatkan akurasi di rentang 80% hingga 90% pada jarak 1 – 10 meter dan dalam kondisi terhalang objek lain, sedangkan pada objek model mobil mendapatkan akurasi di rentang 50% hingga 80% pada jarak 1 – 6 meter. Namun, pada objek mobil berwarna tidak netral seperti biru kerap mengalami salah identifikasi objek dan melabeli objek dengan model kelas lain.

Keywords—*Robot Patroli, Computer Vision, Deep Learning, YOLO, ROS*

I. PENDAHULUAN

A. Latar Belakang

Patroli merupakan “aktivitas untuk berkeliling atau menyusuri area pada interval reguler untuk berbagai macam kepentingan umumnya seperti observasi atau pemeliharaan keamanan”[1]. Patroli memiliki berbagai macam target untuk pemantauan dan pengawasan lingkungan, memperoleh informasi, mencari objek dan mendeteksi adanya anomali, yang membutuhkan keterlibatan secara berkala pada setiap poin infrastrukturnya[1]. Patroli merupakan hal yang umum dilakukan di seluruh dunia sebagai contoh dalam menekan tingkat keamanan. Prevalensi saat ini Index keamanan di Indonesia sendiri berkisar 46.63% pada Oktober 2022 dan berdasarkan survei data tingkat keamanan di Indonesia meningkat sebanyak 12.60% dari 34.03% pada 3 tahun terakhir yang disebabkan oleh peningkatan pengawasan patroli saat pandemi Covid-19[2]. Namun, menurut survei [3] tingkat kekhawatiran kriminalitas di Indonesia terdapat 71.43% terhadap pencurian, penganiayaan dan perusakan properti pribadi.

Kemampuan patroli masih dipengaruhi berbagai macam limitasi: keterbatasan kerja makhluk hidup[4], titik buta pada kamera pengintai[4], dan rintangan pada lingkungan[5]. Salah satu masalah yang paling sering ditemui adalah keterbatasan kerja makhluk hidup. Makhluk hidup memiliki banyak limitasi apabila ditemukan kepada pekerjaan yang melibatkan keterlibatan secara berkala[6]. Contohnya, pada manusia yang memiliki pengaruh emosi dan psikologi dalam menjalankan patroli sehingga dapat menciptakan berbagai tingkah laku yang tidak sesuai[6]. Tidak hanya itu kepercayaan dan pengetahuan yang dimiliki oleh makhluk hidup tidaklah sama[7]. Oleh karena itu, perlu adanya pendekatan umum untuk menentukan proses berpikir apa yang mengarah pada keterbatasan tindakan makhluk hidup sehingga dapat meningkatkan aktivitas berpatroli[6].

Saat ini berbagai macam pendekatan teknologi telah dikaji dan diterapkan dalam menggantikan peran makhluk hidup dalam bertugas patroli. Teknologi yang telah digunakan hingga saat ini diantaranya adalah kamera pengintai[4], drone patrolling[8], dan robotic patrolling[7]. Robotic patrolling merupakan salah satu pendekatan yang dapat menyamakan proses berpikir manusia dan menyamakan kemampuan ilmu pengetahuan manusia[9]. Tidak hanya itu robot memiliki kemampuan untuk memantau lingkungan, mengunjungi dan berulang kali melakukan pengamatan lokal untuk mengidentifikasi ancaman berkelanjutan yang dapat terjadi yang dapat dilakukan secara berkala dalam waktu yang tidak terbatas[9].

Dari hasil pemaparan data dan fakta tentang pengaruh teknologi terkait kegiatan berpatroli, maka penelitian ini bertujuan untuk mengujikan kemampuan robot patroli sebagai alternatif keterbatasan makhluk hidup dalam menjalankan kegiatan patroli. Oleh karena itu, dengan mengembangkan hal tersebut, diharapkan dapat mengurangi kekhawatiran masyarakat terkait pencurian, penganiayaan dan perusakan fasilitas pribadi. Penelitian ini akan merancang sebuah simulasi dengan robot melalui sebuah perangkat lunak yang dapat digunakan untuk mengevaluasi kinerja patroli robot dan pengetahuan apa yang bisa atau harus dilakukan untuk mengoptimalkan pengembangan robot patroli.

B. Analisis Kajian

Seiring berkembangnya urbanisasi dan digitalisasi, penelitian dan pengembangan terkait robot patroli telah beberapa kali dilakukan. Beberapa penelitian tersebut

menggunakan pendalaman deep learning terutama penggunaan arsitektur computer vision yang dapat memproses satu atau sekumpulan gambar pada mesin. Sesuai Tabel I merupakan ringkasan dari kajian penelitian terkait.

TABEL I. TABEL ANALISIS KAJIAN

NO	1	2
Penulis	Sai Siddartha Maram, Tanuj Vishnoi, Sachin Pandey [12]	Yonggang Zhang, Wenqiang Li, and Feng Shen [13]
Judul Penelitian	Neural Network and ROS based Threat Detection and Patrolling Assistance	Object Detection Technology of Substation Patrol Robot Based on mYOLO Algorithms
Metode Penelitian	Computer vision with YOLOv2	mYOLO, Regression Algorithm: Convolution Neural Network
Kekurangan	Akurasi pendeteksian pada kondisi kurang cahaya dan objek yang jauh masih dibawah 50%	Tidak diterangkan
Kelebihan	Akurasi tepat saat objek berada pada jarak portrait dan dapat mengenali kondisi outdoor dengan baik	Penelitian dilakukan dengan tiga jenis kamera yang berbeda dari sudut pandang yang berbeda sehingga akurasi

II. METODOLOGI PENELITIAN

A. Spesifikasi Lingkungan Simulasi

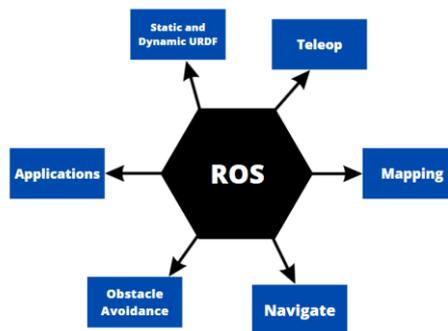
Pada pembuatan simulasi, penulis menggunakan simulasi "yolo_nav" yang dibangun oleh Andrii Paydin. Sebelum melakukan penelitian. Terdapat beberapa spesifikasi yang digunakan peneliti untuk menjalankan simulasi yang dicantumkan dibawah ini:

Ubuntu \geq 20.04 LTS
 Rosdistro \geq Noetic
 YOLOv5
 CUDA \geq 11.3.1
 Gitpython \geq 3.1
 Matplotlib \geq 3.3
 numpy \geq 1.18.5
 opencv-python \geq 4.1.1
 Pillow \geq 7.1.2
 PyYAML \geq 5.3.1
 Requests \geq 2.23
 scipy \geq 1.4.1
 torch \geq 1.7.0
 torchvision \geq 0.8.1
 tqdm \geq 4.64

B. Lingkungan ROS

Robot Operating System (ROS) merupakan framework yang menyediakan berbagai macam tools dan libraries untuk menulis dan menjalankan software robot, ROS memiliki berbagai macam fitur untuk membantu pengembang dalam melakukan seperti penyampaian pesan, komputasi terdistribusi, penggunaan kembali kode dan penerapan algoritma canggih untuk aplikasi robotik[14]. ROS merupakan pondasi dasar dalam menjalankan perangkat robot melalui komputer. Melalui terminal pada ubuntu

beberapa package ROS harus diinstal untuk membuat dunia yang akan disimulasikan. Langkah-langkah yang harus disiapkan untuk membangun robot supaya dapat memiliki purwarupa seperti indra-indra yang dimiliki makhluk hidup pada umumnya.



Gambar 1. Langkah membangun ROS, Source: [10]

Sesuai pada skema Gambar 1. sebagai bentuk upaya bagaimana robot dapat menyamai cara berpikir manusia dan membantu manusia dalam melakukan tindakan yang repetitif seperti patroli perlu adanya manipulasi yang dibentuk dari robot agar robot memiliki kemampuan bertindak layaknya makhluk hidup. Berikut penjelasan dari ilustrasi langkah-langkah dalam membangun ROS:

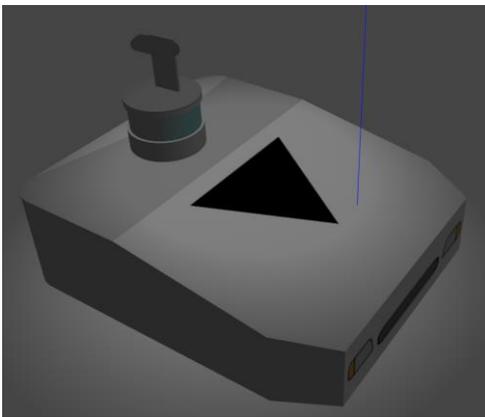
- **Static and Dynamic URDF:** Tahapan yang pertama adalah membangun tampilan fisik dari robot, hal ini membantu ROS memahami struktur fisik robot sebenarnya dari berbagai tautan pada robot dan dimana semua sensor dan aktuator berada.
- **Teleop:** Tahapan yang kedua adalah membangun fungsi gerak pada robot, hal ini membuat bagaimana robot dapat bergerak menggunakan sensor. Berfungsi terhadap layanan robot yang digunakan untuk mengaktifkan berbagai sensor dan memerintahkan aktuator menggunakan nilai sensor.
- **Mapping:** Tahapan yang ketiga adalah membangun fungsi perasa pada robot, mempelajari cara agar robot dapat memahami dan merasakan lingkungannya menggunakan LiDAR.
- **Navigate:** Dari tahapan yang sebelumnya pada tahapan yang keempat merupakan robot yang sudah mulai memahami dunianya. Sistem memberikan perintah kepada robot menggunakan algoritma perencanaan dan pengoptimalan jalur yang berbeda untuk membuatnya berpindah dari titik A ke titik B, hal ini juga akan mencakup konfigurasi peta, biaya dan perencanaan yang berbeda.
- **Obstacle Avoidance:** Tahapan kelima setelah robot melakukan navigasi dari titik A ke titik B pastinya memiliki berbagai rintangan yang telah dibentuk pada lingkungan melalui SLAM. Oleh karena itu, kita membutuhkan algoritma penghindaran rintangan yang baik dan perencanaan jalur di sekitar rintangan.
- **Applications:** Tahapan terakhir merupakan pengaplikasian dari lima tahapan sebelumnya yang memungkinkan robot dapat melakukan tugas yang diberikan.

C. Static and Dynamic URDF

URDF adalah singkatan dari Unified Robot Description Format. URDF merupakan format file berbasis XML yang digunakan untuk menggambarkan properti fisik dan kinematik robot dengan cara yang dapat dibaca oleh mesin. URDF banyak digunakan dalam bidang robotika untuk berbagai aplikasi seperti simulasi robot, perencanaan gerak dan kontrol.

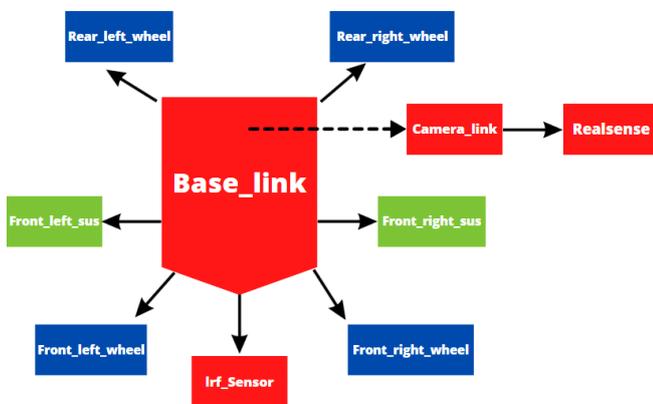
File URDF dapat menyertakan informasi tentang tautan robot, sensor dan properti fisik lainnya seperti massa, inersia dan geometri. File URDF biasanya digunakan dengan perangkat lunak simulasi robot seperti Gazebo, ROS dan kerangka kerja pengembang robot lainnya.

Pada penelitian ini peneliti menggunakan robot bernama `yn_robot` yang dibangun pada file `"yolo_nav"`. Bentuk pada `yn_robot` ini terdapat pada Gambar 2.



Gambar 2. Robot '`yn_robot`'

Semua file URDF tersimpan menjadi satu di file XACRO. XACRO adalah XML macro language untuk membangun dan membaca file XML pada URDF. File-file URDF yang terdapat pada XACRO biasanya berisi additional features, variable names dan macros yang dapat dimuat pada Gazebo. Penjelasan terkait XACRO pada `yn_robot` dijelaskan secara singkat melalui skema pada Gambar 3.



Gambar 3. File XACRO `yn_robot`

D. Teleop

Setelah membangun bentuk fisik dari robot menggunakan `yn_robot`. Selanjutnya adalah membangun fungsi gerak dari aktuator agar robot dapat bergerak sesuai dengan sensor pada controller yang bisa kita perintahkan dari jendela "controller" atau yang disebut dengan teleoperating.

Pada kontroler pertama terdapat 'joint_state_controller' yang menerbitkan gerak sendi robot pada kecepatan 50Hz. Kemudian pada kontroler kedua adalah 'diff_drive_controller' yang mengendalikan pergerakan robot menggunakan differential drive system. Ada beberapa parameter didalamnya seperti mengontrol sendi roda kiri dan kanan, jarak antar roda, radius dari roda dan juga memiliki batas kecepatan dan percepatan untuk gerakan linier(x) dan sudut(z).

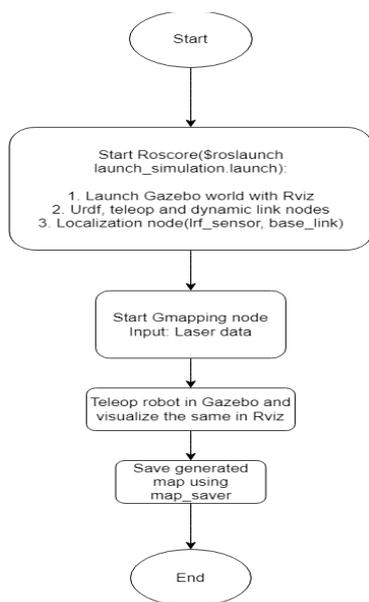
Pada kontroler ketiga adalah 'camera_stay_controller' yang berfungsi untuk mengontrol posisi sambungan penahan pada kamera menggunakan pengontrol posisi berbasis algoritme kontrol PID(Proportional-Integral-Derivative). Kontroler-kontroler tersebut berfungsi untuk mengendalikan pergerakan dan posisi dari gerak sendi robot dan kamera.

E. Mapping

Gmapping adalah salah satu dari paket ROS yang menyediakan kemampuan Simultaneous Localization and Mapping(SLAM) untuk mobile robot. SLAM adalah proses membuat peta lingkungan yang tidak diketahui sekaligus melokalkan robot di dalam lingkungan tersebut. Gmapping menggunakan sensor LiDAR, yaitu menggunakan data dari laser range finder dan sensor odometri robot untuk membuat peta lingkungan 2D, serta memperkirakan pose robot (posisi dan orientasi) dalam lingkungan tersebut.

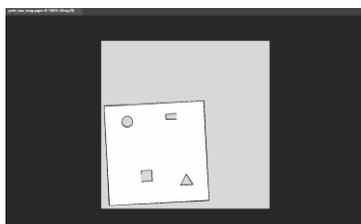
Untuk menggunakan Gmapping, sensor robot harus terkalibrasi dengan baik dan gerakan robot harus semulus mungkin. Paket tersebut membutuhkan robot untuk bergerak melalui lingkungan dengan cara mengambil sampel pada lingkungan dari berbagai sudut dan jarak. Setelah robot mengumpulkan cukup data, Gmapping dapat membuat peta lingkungan dan melokalkan robot di dalam peta tersebut.

Runtutan secara singkat proses dari pembentukan dan pembelajaran map oleh robot yn_robot dijelaskan pada diagram alur Gambar 4.



Gambar 4. Diagram alur proses gmapping

Setelah seluruh map sudah ditelusuri dan dipelajari oleh robot maka file akan tersimpan oleh map_saver kedalam format .pgm yang nantinya akan diimport ke dalam jendela konsol gui.



Gambar 5. Yolo_nav_map.pgm

F. Navigate

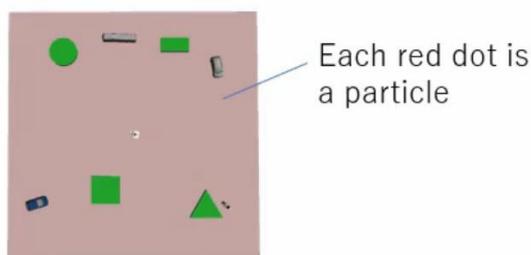
Setelah robot sudah memahami kondisi lingkungannya, selanjutnya merupakan bentuk perintah robot pada sistem controller agar robot dapat mengerti dan bergerak sesuai arahan konsol gui. Pada navigasi, robot akan belajar bagaimana mengkalkulasi target koordinat map menggunakan kamera yang berada pada robot yaitu menghitung koordinat objek dalam koordinat sistem kamera, terdapat beberapa tahapan agar robot dapat mengkalkulasi posisi objek didalam map.

- Tahapan pertama adalah mengkalkulasi koordinat objek sistem koordinat kamera. Pada tahapan ini menggunakan pendekatan transformasi perspektif proyeksi.
- Tahapan kedua adalah mengkalkulasi koordinat objek pada sistem koordinat robot menggunakan rotasi perputaran pada kamera

- Tahapan ketiga adalah mengkalkulasi koordinat objek pada map menggunakan posisi dan orientasi pada robot

G. Obstacle Avoidance

Setelah robot telah memahami bagaimana cara menavigasikan dari titik koordinat pusat ke titik lainnya, tentu saja ada beberapa halang rintang yang terdapat pada suatu lingkungan. Selanjutnya adalah bagaimana robot dapat melacak kondisi yang berada pada map. Oleh karena itu, diperlukan algoritma partikel filter. Partikel filter menggunakan bobot partikel untuk mendeteksi posisi dan orientasi robot. Hal pertama yang harus dilakukan adalah menginisialisasi terlebih dahulu posisi partikel yang tersebar seragam di seluruh peta seperti pada Gambar 6.



Gambar 6. Area partikel filter source:[11]

Perhitungan posisi robot terdiri dari tiga langkah yaitu prediksi, pembaruan dan sampel ulang:

- **Prediksi**
Selama langkah prediksi diasumsikan robot bisa menggunakan gerakan model untuk setiap partikel, sampel dan noise pada setiap aksi model.
- **Pembaruan**
Pembaruan menghitung masing-masing cara partikel diperoleh. Setiap bobot partikel memiliki kesamaan untuk dapat terbaca oleh sensor dari asumsi partikel.
- **Sampel ulang**
Beberapa set partikel baru dipilih dan dihasilkan melalui nilai sensor berdasarkan partikel yang tersisa dari bobot model.

Dalam kondisi raw posisi(x,y,z) dan orientasi(x,y,z,w) robot dapat diperoleh menggunakan topik "amcl_pose". Orientasi pada amcl dihasilkan oleh quaternion. Oleh karena itu, paternian akan dikonversi menjadi sudut euler untuk mengkalkulasi posisi target pada map.

Quaternion sering digunakan untuk perhitungan yang melibatkan rotasi tiga dimensi. Quaternion umumnya diwakili pada bentuk :

$$q_0 + q_1i + q_2j + q_3k$$

Penjelasan pada variabel q0, q1, q2 dan q3 merupakan bilangan real sedangkan i, j dan k merupakan dasar quaternion. Setiap bilangan real merepresentasikan kuadran-kuadran.

H. Pengumpulan Data

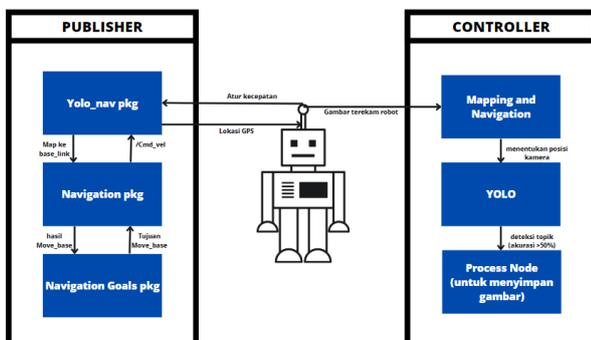
You Only Look Once (YOLO) merupakan real time object detection system dan image segmentation model[12]. YOLO menggunakan algoritma computer vision yang digunakan untuk mendeteksi objek, mensegmentasi gambar, dan melokalisasi gambar pada gambar dan video secara real time. Algoritma ini dikembangkan oleh Joseph Redmon, Santosh Divvala, Ross Girshick dan Ali Farhadi yang dikembangkan pada tahun 2015-2016[15]. YOLO dirancang agar mudah untuk dipelajari bagi pemula dan data yang diambil berasal dari data-data hasil dunia nyata. Adapun dataset yang digunakan merupakan dataset dari ultralytics COCO128.

Dataset ini berisi 128 gambar pertama dari COCO2017 untuk setiap kelasnya. Dataset menggunakan 128 gambar yang sama untuk dilatih dan diuji. Hal ini dimaksudkan untuk memvalidasi alur pelatihan dengan mendemonstrasikan overfitting pada kumpulan data kecil sebelum melatih kumpulan data yang lebih besar, dataset ini berisikan 80 kelas.

III. HASIL DAN PEMBAHASAN

A. Pengujian Model

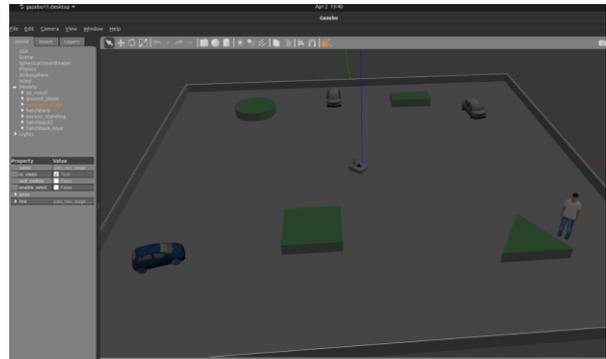
Setelah semua persiapan dan percobaan dilakukan maka langkah selanjutnya adalah pengujian YOLO terkait model yang telah diatur pada ROS. Untuk melakukan penelitian maka kita perlu mengkompilasi repositori ROS yang akan digunakan pada OS linux. Pada penelitian kali ini akan digunakan repositori yolo_nav yang memiliki robot navigasi yn_robot yang telah terancang kamera dengan konfigurasi YOLO. Rangkaian proses pengkompilasian repositori ROS terdapat pada ilustrasi activity diagram Gambar 7.



Gambar 7. Activity diagram deteksi objek robot dengan YOLO

Dijelaskan pada diagram bahwa pada publisher terdapat paket navigasi dan telah terintegrasi langsung dengan robot melalui paket yolo_nav yang berfungsi untuk menerbitkan posisi tujuan robot yang akan dinavigasikan. Tujuan dari deteksi objek ini penelitian dapat menambahkan program robot yang akan bisa menuju ke navigasi objek tujuan yang telah terdeteksi oleh YOLO melalui kamera robot yang telah memiliki sensor yang dikonfigurasi pada sensor mapping dan penelitian ini dapat memproyeksikan hasil akurasi deteksi gambar pada YOLO yang dijelaskan pada diagram pada controller gambar/UI. Setelah robot berhasil mengidentifikasi objek apabila objek yang terdeteksi memiliki akurasi lebih dari 50% maka objek akan terdata dan tersimpan dalam format ".jpg".

Pertama-tama yang harus ditemukan adalah paket Navigation Goals pkg paket ini mendefinisikan koordinat x y theta dalam file peluncuran robot dan akan menavigasikan robot sesuai dengan arah sensor yang telah ditentukan oleh dideteksi oleh YOLO.

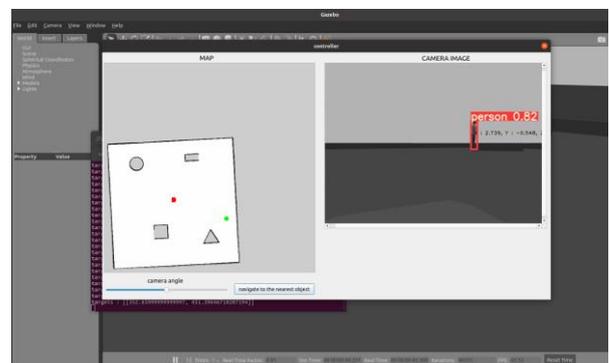


Gambar 8.. Tampilan dunia melalui gazebo

Sesuai pada Gambar 8. robot memiliki lokasi dari titik koordinat semula yaitu di titik tengah dari map dengan koordinat [0,0,0] yang berjarak 5 meter keseluruhan objek dengan sumbu Z dan objek yang akan deteksi berada jarak masing-masing yang telah ditentukan oleh navigasi

Dalam bagian ini akan ditampilkan koordinat target pada peta menggunakan kamera yang telah terpasang pada robot. Prosedur ini akan melibatkan tiga langkah yang telah dijelaskan pada bagian metodologi. Langkah pertama adalah menghitung objek koordinat didalam koordinat kamera sistemnya dalam hal ini penelitian akan menggunakan perspektif Teknik Transformasi Perspektif Proyeksi. Langkah kedua adalah menghitung objek koordinat dalam koordinat robot sistemnya penelitian akan menggunakan informasi dari angka sudut rotasi kamera. Langkah terakhir adalah menghitung objek koordinat dalam sistem koordinat peta sistemnya penelitian akan menemukan posisi robot dalam orientasi peta menggunakan amcl.

Setelah membuka tampilan dari dunia yang akan diuji, selanjutnya membuka pada terminal baru untuk memberi perintah kepada arsitektur YOLO agar dapat bekerja di dalam dunia yang sudah dibentuk.



Gambar 9. Panel 'Controller' GUI

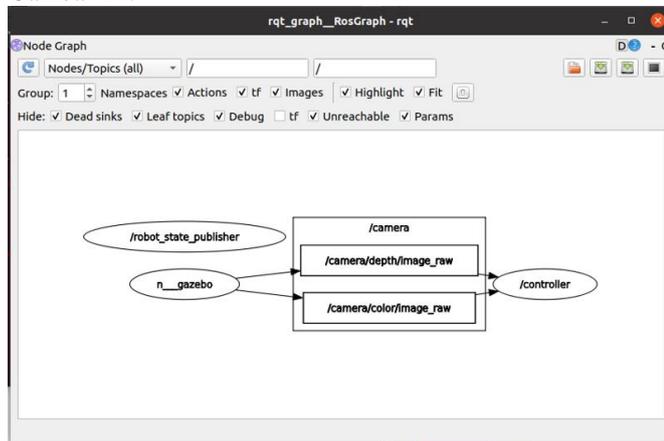
Sesuai pada Gambar 9 merupakan tampilan dari controller konsol GUI yang di perintah dari terminal pada kotak bagian kiri terdapat map yang sudah teridentifikasi dari proses mapping dan memuat koordinat navigasi map dan robot kemudian pada kotak bagian kanan terdapat kamera untuk deteksi objek menggunakan YOLO. Sesuai dengan yang digambarkan pada activity diagram target tujuan pada publisher sebagai lingkaran hijau yang menandakan lokasi model terdeteksi yang dikalkulasikan menggunakan quaternion sehingga mengkonversi target posisi menjadi koordinat robot untuk menavigasikan robot sesuai dengan target yang terbit pada map.

Setelah robot patroli menerima semua koordinat yang terbaca oleh map, dengan menggunakan controller robot dapat di navigasikan ke objek terdekat yang telah terdeteksi oleh sensor.



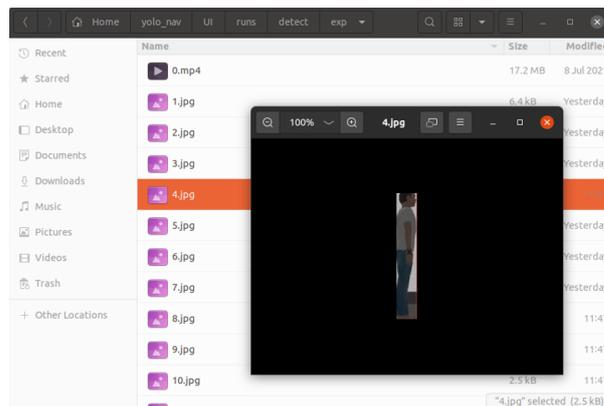
Gambar 10. Tampilan 'controller' GUI pendeteksian objek pada ROS

Proses dari pendeteksian objek dalam ROS dijelaskan dengan men-debug node-node yang berjalan pada terminal dapat di debugging dengan rqt_graph yang dapat dilihat pada Gambar 11.



Gambar 11. Debug proses penelitian

Setelah objek berhasil terdeteksi maka proses selanjutnya controller akan memproses gambar yang terdeteksi dengan YOLO ke dalam pengambilan gambar yang akan di export ke dalam folder runs/detect/exp dalam bentuk format ".jpg" seperti yang tertampil pada Gambar 12.



Gambar 12. Hasil simpan gambar deteksi objek

Pada Gambar 12 hasil dari deteksi objek yang tersimpan dan terdeteksi dari YOLO memiliki akurasi diatas 50% pada setiap kelasnya gambar akan dipotong dengan ukuran berskala sesuai dengan hyperparameter dari objek.

$$\text{hyp}[\text{'obj'}] *= (\text{imgsz} / 640) ** 2 * 3. / \text{nl}$$

Keterangan:

- hyp['obj'] : Skala ke ukuran gambar dan layer
- imgsz : Ukuran gambar yang dihitung dari besaran kotak
- nl : Jumlah layer

B. Analisis Hasil

Setelah robot patroli berhasil untuk melakukan implementasi patroli sederhana yang telah dijalankan pada simulasi, maka analisis hasil pun dapat dijalankan. Tujuan dari analisis hasil ini untuk mengukur kinerja simulasi yang telah dibuat apakah analisis dari simulasi robot patroli yang telah diteliti dapat menjawab rumusan masalah yang telah dipaparkan. Untuk itu akan dijalankan beberapa skenario yang dapat menunjukkan batasan kemampuan dari simulasi robot patroli menggunakan YOLO.

Skenario 1. YOLO akan mendeteksi objek tunggal dengan model manusia atau mobil dengan jarak koordinat yang berbeda

TABEL II. SKENARIO 1 PENGUJIAN

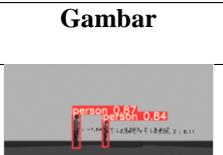
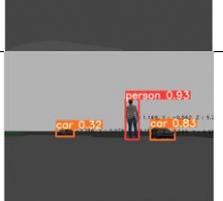
Deskripsi	Jarak (sumbu z)	Gambar
Model manusia berbaju putih tampak samping	Z : 6,897	
Model manusia berbaju putih tampak samping	Z : 4,714	
Model manusia berbaju putih tampak samping	Z : 2,454	

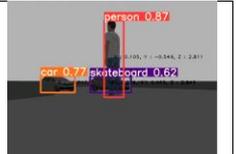
Model mobil berwarna biru tampak samping	Z : 5,646	
Model mobil berwarna biru tampak samping	Z : 4,089	
Model mobil berwarna biru tampak samping	Z : 3,055	
Model mobil berwarna putih tampak samping	Z : 2,581	
Model mobil berwarna biru tampak depan	Z : 5,810	

Sesuai dengan hasil pada Tabel II. Pada model manusia YOLO memiliki akurasi yang lebih baik dari jarak 7 meter hingga 2 meter akurasi yang dihasilkan dapat mencapai rata-rata 80% hingga 90%. Kemudian, pada model mobil berwarna hasil akurasi yang dihasilkan cenderung tidak terlalu akurat pada jarak diatas 5 meter deteksi mengidentifikasi model dengan kelas lain. Sedangkan untuk mobil yang berwarna putih memiliki akurasi yang lebih baik dengan akurasi diatas 50% dari berbagai sudut dan jarak.

Skenario 2. YOLO akan mendeteksi objek ganda dengan model manusia atau mobil dengan jarak koordinat yang berbeda

TABEL III. SKENARIO 2 PENGUJIAN

Deskripsi	Jarak (sumbu z)	Gambar
Model manusia berbaju putih	Z1 : 8,11 Z2 : 6,417	
Model manusia berbaju putih dan mobil berwarna biru	Z1 : 6,976 Z2 : 5,733	
Model manusia berbaju putih dan mobil berwarna putih	Z1 : 5,245 Z2 : 7,34 Z3 : 9,164	

Model manusia berbaju putih, mobil berwarna biru dan mobil berwarna putih	Z1 : 2,811 Z2 : 2,643 Z3 : 4,163	
Model manusia berbaju putih dan truk ambulans berwarna putih	Z1 : 4,381 Z2 : 8,822 Z3 : 5,689	

Sesuai dengan hasil pada Tabel III. YOLO berhasil mendeteksi seluruh kelas model yang berada pada jangkauan jarak tangkap kamera. Pada model manusia semua teridentifikasi dengan baik dengan akurasi diatas 50% pada semua jarak yang di uji cobakan. Kemudian, pada mobil berwarna biru terdapat kesalahan deteksi model yang terhalangi model manusia sehingga teridentifikasi sebagai papan seluncur. Sedangkan untuk model lainnya seperti mobil berwarna putih dan truk ambulans tidak mengalami kesalahan identifikasi dengan akurasi diatas 50%.

Skenario 3. YOLO akan mendeteksi model manusia yang bergerak dengan interupsi dari objek-objek sekitarnya

TABEL IV. SKENARIO 3 PENGUJIAN

Deskripsi	Jarak (sumbu z)	Gambar
Model manusia berjalan dan mobil berwarna biru	Z1 : 7,996 Z2 : 5,462	
Model manusia berjalan dibelakang mobil	Z1 : 8,202 Z2 : 5,465	
Model manusia berjalan tampak setengah badan dibelakang box	Z : 5,645	
Model manusia berjalan tampak kepala dibelakang box	Z : 5,381	

Sesuai dengan hasil pada Tabel IV. YOLO tidak memiliki masalah dalam mengidentifikasi model manusia berjalan meskipun model hanya tampak setengah badan maupun seperempat badan. Seluruh uji coba mengindikasikan akurasi diatas 50% pada model manusia berjalan.

IV. KESIMPULAN DAN SARAN

A. Kesimpulan

Berdasarkan hasil-hasil yang telah didapatkan, penelitian ini telah berhasil membangun dan mensimulasikan robot untuk melakukan kegiatan berpatroli umumnya yang diimplementasikan dengan computer vision dan deep learning menggunakan YOLOv5. Sehingga robot memiliki kemampuan seperti layaknya makhluk hidup yang memiliki kemampuan berupa navigasi, identifikasi dan koordinasi.

- Penelitian ini mampu meimplementasi robot patroli sebagai alternatif dari keterbatasan kerja makhluk hidup. Dengan adanya sensor dan aktuator yang dibentuk menggunakan algoritma pemrograman sehingga dapat mengenal kondisi lingkungannya serta penggunaan computer vision untuk memahami kondisi lingkungannya. Dengan data yang semakin banyak diperoleh sehingga dapat memiliki kemampuan cerdas satu tingkat dibawah atau hampir menyamai makhluk hidup. Namun, memiliki daya energi yang terbilang hampir tidak terbatas dan konsisten dalam rangka melakukan pekerjaan yang berkala.
- Penelitian ini menghasilkan deteksi objek yang akurat apabila harus mengenal kondisi model berupa manusia. Akurasi yang diberikan pada model berupa manusia bisa mencapai 80% hingga 90% dalam kondisi anomali terhalang aktivitas lain maupun dalam jarak 1 – 10 meter jauhnya. Namun, dalam pendeteksian objek lain seperti model mobil terutama pada mobil berwarna tidak netral seperti biru. Akurasi YOLOv5 masih cenderung salah mengidentifikasi kelas objek, tetapi memiliki akurasi yang cukup akurat pada jarak 1—6 meter dengan akurasi 50% hingga 80%.

B. Saran

Penelitian ini menggunakan simulasi ruang sederhana dan model-model buatan. Peneliti berharap penelitian ini dapat dikembangkan oleh peneliti lain di masa mendatang dengan beberapa saran seperti:

- Hasil deteksi objek dapat disimpan tidak hanya sekedar hasil deteksi gambar objek. Namun, bisa memuat beberapa informasi seperti waktu, tanggal, dan koordinat pengambilan gambar yang menjadi satu file dengan gambar.
- Penelitian dapat dikembangkan dan dibandingkan dengan versi arsitektur YOLO terbaru dari yang digunakan peneliti saat ini.
- Robot patroli bisa dikembangkan lebih dari satu robot patroli dengan menggunakan metode commander and follower pada ROS. Sehingga, terdapat lebih dari satu robot yang terdistribusi dalam lingkup patroli.

REFERENCES

- [1] Portugal, D., & Rocha, R. (2011). A survey on multi-robot patrolling algorithms. *IFIP Advances in Information and Communication Technology*, 349 AICT, 139–146.
- [2] Asia: Current Crime Index by City. (2022). Retrieved October 20, 2022, from https://www.numbeo.com/crime/region_rankings_current.js?region=142I. S. Jacobs and C. P. Bean, "Fine particles, thin films and exchange anisotropy," in *Magnetism*, vol. III, G. T. Rado and H. Suhl, Eds. New York: Academic, 1963, pp. 271–350.
- [3] Crime in Indonesia. Safety in Indonesia. (2019). Retrieved October 20, 2022, from <https://www.numbeo.com/crime/in/Indonesia-Indonesia> R. Nicole, "Title of paper with only first word capitalized," J. Name Stand. Abbrev., in press.
- [4] Lee, H.-T., Lin, W.-C., & Huang, C.-H. (2011). Indoor Surveillance Security Robot with a Self Propelled Patrolling Vehicle. *Journal of Robotics*, 2011, 1–9.
- [5] Rostami, S. M. H., Sangaiah, A. K., Wang, J., & Liu, X. (2019). Obstacle avoidance of mobile robots using modified artificial potential field algorithm. *Eurasip Journal on Wireless Communications and Networking*, 2019(1), 1–19. <https://doi.org/10.1186/S13638-019->
- [6] Hiatt, L. M., Harrison, A. M., & Trafton, J. G. (2011). Accommodating Human Variability in Human-Robot Teams through Theory of Mind. Twenty-Second International Conference on Artificial Intelligence. Retrieved from <https://www.aaai.org/ocs/index.php/IJCAI/IJCAI11/paper/view/3336>
- [7] Lopez, A., Paredes, R., Quiroz, D., Trovato, G., & Cuellar, F. (2017). Robotman: A security robot for human-robot interaction. 2017 18th International Conference on Advanced Robotics, ICAR 2017, 7–12. <https://doi.org/10.1109/ICAR.2017.8023489>
- [8] Nguyen, V. N., Jenssen, R., & Roverso, D. (2018). Automatic autonomous vision-based power line inspection: A review of current status and the potential role of deep learning. *International Journal of Electrical Power and Energy Systems*, 99(January), 107–120. <https://doi.org/10.1016/j.ijepes.2017.12.016>
- [9] Basilio, N. (2019). GROUP ROBOTICS (M GINI AND F AMIGONI, SECTION EDITORS) Recent Trends in Robotic Patrolling. *Current Robotics Reports*, 1, 3. <https://doi.org/10.1007/s43154-022-00078-5>
- [10] GitHub - Soft-illusion/Robotics_PicoDegree. (2022). Retrieved April 11, 2023, from https://github.com/Soft-illusion/Robotics_PicoDegree
- [11] Yuan, X., Yu, S., Zhang, S., Wang, G., Liu, S., Khoshelham, K., & Zlatanova, S. (2015). Quaternion-Based Unscented Kalman Filter for Accurate Indoor Heading Estimation Using Wearable Multi-Sensor System. *Sensors*, 15, 10872–10890. <https://doi.org/10.3390/s150510872>
- [12] Maram, S. S., Vishnoi, T., & Pandey, S. (2019). Neural network and ROS based threat detection and patrolling assistance. 2019 2nd International Conference on Advanced Computational and Communication Paradigms, ICACCP 2019. <https://doi.org/10.1109/ICACCP.2019.8883008>
- [13] Zhang, Y., Li, W., & Shen, F. (2020). Object Detection Technology of Substation Patrol Robot Based on mYOLO Algorithms. 2020 IEEE International Conference on Mechatronics and Automation, ICMA 2020, 594–598. <https://doi.org/10.1109/ICMA49215.2020.9233751>
- [14] Mastering ROS for Robotics Programming: Design, build, and simulate complex ... - Lentin Joseph, Jonathan Cacace - Google Buku. (n.d.). Retrieved December 30, 2022, from [https://books.google.co.id/books?hl=id&lr=&id=MulODwAAQBAJ&oi=fnd&pg=PP1&dq=robot+operating+system&ots=Cln5N5oVnP&sig=jicLvjkASxjhuZkgo86M8veFAE&redir_esc=y#v=onepage&q=robot operating system&f=false](https://books.google.co.id/books?hl=id&lr=&id=MulODwAAQBAJ&oi=fnd&pg=PP1&dq=robot+operating+system&ots=Cln5N5oVnP&sig=jicLvjkASxjhuZkgo86M8veFAE&redir_esc=y#v=onepage&q=robot%20operating%20system&f=false)
- [15] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *Proceedings of the IFFE conference on computer vision and pattern recognition* (pp. 779–788)