PENERAPANAN MODEL DETEKSI OBJEK UNTUK ROBOT MENGGUNAKAN MODEL SINGLE SHOT DETECTION DI LINGKUNGAN SIMULASI ROS

Disusun Oleh:

N a m a : Ilham Rizqyakbar

NIM : 19523066

PROGRAM STUDI INFORMATIKA – PROGRAM SARJANA

FAKULTAS TEKNOLOGI INDUSTRI

UNIVERSITAS ISLAM INDONESIA

2024

HALAMAN PENGESAHAN DOSEN PEMBIMBING

PENERAPANAN MODEL DETEKSI OBJEK UNTUK ROBOT MENGGUNAKAN MODEL SINGLE SHOT DETECTION DI LINGKUNGAN SIMULASI ROS

TUGAS AKHIR

Disusun Oleh:

N a m a	: Ilham Rizqyakbar
NIM	: 19523066

Yogyakarta, 1 Nopember 2024

Pembimbing,

Chandra Kusuma Dewa, S.Kom., M.Kom., Ph.D.

`

HALAMAN PENGESAHAN DOSEN PENGUJI

PENERAPANAN MODEL DETEKSI OBJEK UNTUK ROBOT MENGGUNAKAN MODEL SINGLE SHOT DETECTION DI LINGKUNGAN SIMULASI ROS

TUGAS AKHIR

Telah dipertahankan di depan sidang penguji sebagai salah satu syarat untuk memperoleh gelar Sarjana Komputer dari Program Studi Informatika – Program Sarjana di Fakultas Teknologi Industri Universitas Islam Indonesia

Yogyakarta, 4 November 2024

Tim Penguji

Chandra Kusuma Dewa, S.Kom., M.Cs., Ph.D.

Anggota 1

Andhik Budi Cahyono, S.T., M.T.

Anggota 2

Izzati Muhimmah, S.T., M.Sc., Ph.D.

Mengetahui,
Ketua Program Studi Informatika – Program Sarjana
Fakultas Teknologi Industri
Universitas Islam Indonesia

(Ir. Dhomas Hatta Fudholi, S.T., M.Eng., Ph.D.)

HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR

Yang bertanda tangan di bawah ini:

Nama : Ilham Rizqyakbar

NIM : 19523066

Tugas akhir dengan judul:

PENERAPANAN MODEL DETEKSI OBJEK UNTUK ROBOT MENGGUNAKAN MODEL SINGLE SHOT DETECTION DI LINGKUNGAN SIMULASI ROS

Menyatakan bahwa seluruh komponen dan isi dalam tugas akhir ini adalah hasil karya saya sendiri. Apabila di kemudian hari terbukti ada beberapa bagian dari karya ini adalah bukan hasil karya sendiri, tugas akhir yang diajukan sebagai hasil karya sendiri ini siap ditarik kembali dan siap menanggung risiko dan konsekuensi apapun.

Demikian surat pernyataan ini dibuat, semoga dapat dipergunakan sebagaimana mestinya.

Yogyakarta, 31 Oktober 2024

Ilham Rizqyakbar

`

HALAMAN PERSEMBAHAN

Saya menyusun laporan akhir ini sebagai bukti perjuangan yang telah saya lalui, penuh dengan rasa syukur yang saya dedikasikan kepada keluarga, khususnya orang tua dan kakak saya, yang sudah memberikan dukungan untuk saya selama perkuliah saya sampai hari ini. Doa dan cinta tanpa syarat yang mereka berikan sepanjang perjalanan studi saya telah menjadi kekuatan utama. Dengan penuh rasa terima kasih, saya sampaikan apresiasi mendalam atas dukungan dan dorongan yang selalu mereka berikan. Mereka adalah sumber inspirasi yang tak pernah membuat saya menyerah dalam menyelesaikan tugas akhir ini. Sekali lagi, terima kasih sebesar-besarnya atas segala pengorbanan yang telah membawa saya mencapai tahap akhir dalam studi ini.

`

HALAMAN MOTO

“May the dice roll in my favour.”

`

KATA PENGANTAR

Assalamu'alaikum Wr. Wb.

Puji dan syukur saya panjatkan kehadirat Allah SWT yang memberikan rahmat dan hidayah-Nya sehingga penulis menyelesaikan laporan penelitiandengan judul "Penerapan Model Deteksi Objek untuk Robot Menggunakan Model Single Shot Detection di Lingkungan Simulasi ROS" sebagai syarat kelulusan dalam jenjang sarjana informatika saya di Universitas Islam Indonesia

Penulis menerima bantuan dari berbagai pihak dalam menyelesaikan laporan penelitian ini, oleh karena itu penulis tidak lupa menyampaikan rasa terima kasih yang sebesar-besarnya kepada:

1. Orang tua dan keluarga, terutama kakak penulis, yang selalu memberikan dukungan untuk kelancaran dalam proses perkuliahan penulis.
2. Bapak Chandra Kusuma Dewa, S.Kom., M.Cs., Ph.D., selaku dosen pembimbing penulis yang telah memberikan arahan dalam menyelesaikan laporan penelitian.
3. Bapak Dhomas Hatta Fudholi, S.T., M.Eng., Ph.D., selaku dosen penulis dalam mata kuliah metodologi penelitian dan telah membantu dalam penulisan lapran penulis serta menilai draft laporan.
4. Teman-teman dan rekan di Badan Sistem Informasi Universitas Islam Indonesia yang memberikan dukungan kepada penulis.

Pada laporan akhir ini, penulis menyadari kemungkinan masih terdapat berbagai kekurangan yang perlu diperbaiki. Segala bentuk kritik dan saran akan penulis terima dengan senang hati. Penulis berharap laporan ini dapat memberikan manfaat bagi para pembacanya.

.

Yogyakarta, 4 November 2024

Ilham Rizqyakbar

`

SARI

Perkembangan industri automasi, khususnya dalam teknologi robotika, telah membawa revolusi dalam perekonomian Asia Tenggara, termasuk Indonesia, yang sedang bertransisi menuju negara maju pada tahun 2045. Meskipun Indonesia memiliki potensi besar, penggunaan teknologi otomasi dan robotika di negara ini masih tergolong rendah dibandingkan dengan negara tetangga. Salah satu inisiatif pemerintah adalah Kontes Robot Indonesia (KRI), yang bertujuan untuk meningkatkan kompetensi mahasiswa dalam bidang robotika. Dalam konteks ini, tim KRSBI-B dari Universitas Islam Indonesia menggunakan segmentasi warna untuk mendeteksi objek, namun metode ini memiliki keterbatasan, terutama dalam hal kalibrasi dan sensitivitas terhadap pencahayaan.

Untuk mengatasi kendala tersebut, penelitian ini bertujuan mengembangkan model deteksi objek berbasis computer vision yang dapat diterapkan pada robot dalam simulasi Gazebo. Penelitian ini mencakup pengembangan model deteksi, integrasi dengan simulasi Gazebo, dan kemampuan robot untuk mengambil tindakan berdasarkan hasil deteksi. Fokus penelitian ini terbatas pada pengembangan model deteksi objek, menggunakan Python 3.9.16, Robot Operating System (ROS), dan Ubuntu Linux, serta mendeteksi objek umum seperti bola dan gawang. Hasil dari penelitian model atas objek bola dan gawang mendapatkan akurasi 51.1% hingga 80.1% dan dapat mendeteksi sampai dengan jarak ~2.8 meter.

Kata kunci: ROS, Robot Sepak Bola, Computer Vision, SSD, Deteksi Objek.

`

GLOSARIUM

Classification loss	Metrik yang digunakan untuk mengukur seberapa baik model dalam mengklasifikasikan data ke dalam kategori yang benar
Localization loss	Metrik yang digunakan dalam konteks deteksi objek untuk mengukur seberapa akurat model dalam menentukan posisi dan ukuran objek dalam gambar
mAP	Metrik yang umum digunakan untuk mengevaluasi performa model deteksi objek.
Middleware	perangkat lunak yang berfungsi sebagai perantara antara berbagai aplikasi, layanan, atau sistem dalam suatu arsitektur perangkat lunak
Publish	Mengirim pesan pada sistem jaringan publish-subscribe.
Subscribe	Menerima pesan pada sistem jaringan publish-subscribe.
ROS	ROS merupakan framework *middleware* yang digunakan untuk mengembangkan perangkat lunak robotika
Topic	Sebuah jalur label yang digunakan untuk mengelompokkan pesan.
SDF	format file yang digunakan dalam simulasi robotika dan lingkungan 3D, khususnya dalam sistem simulasi seperti Gazebo

`

DAFTAR ISI

`

`

DAFTAR TABEL

`

DAFTAR GAMBAR

`

`

BAB I
PENDAHULUAN

1.1 Latar Belakang

Perkembangan industri automasi telah melakukan revolusi industri dan mengubah perekonomian di kawasan Asia Tenggara secara cepat melalui teknologi yang disruptif seperti otomasi menggunakan robot. Sebagai negara yang akan melakukan transisi menuju negara maju pada tahun 2045 nanti, Indonesia sudah seharusnya memanfaatkan peluang otomasi untuk terus meningkatkan daya saing. Robotika merupakan salah satu bukti konkret dari kemajuan teknologi di bidang automasi tersebut. International Federation of Robotics telah menempatkan Indonesia di antara yang terendah dibandingkan negara lain di Asia Tenggara dalam hal penggunaan teknologi otomasi dan robotika pada tahun 2019. Dibandingkan negara tetangga seperti Malaysia dengan perbandingan tiga puluh empat robot per 10.000 karyawan di tahun yang sama (Asian Robotics Review LLC, 2019).

Salah satu upaya yang dilakukan pemerintah Indonesia untuk melakukan pengembangan robotika adalah dengan diadakannya beberapa kompetisi robotika, seperti Kontes Robot Indonesia (KRI), oleh Kementerian Pendidikan, Kebudayaan, Riset, dan Teknologi setiap tahunnya. KRI yang diikuti seluruh mahasiswa pada perguruan tinggi di wilayah Indonesia yang tercatat pada Pangkalan Data Perguruan Tinggi (PDDikti), ditujukan untuk mencari kandidat-kandidat yang akan mewakili Indonesia dalam perlombaan Robocup, perlombaan tingkat Internasional, serta menjadi ajang pengembangan dan rancang inovasi mahasiswa dalam pengembangan robot (Puspresnas, 2024). KRI memiliki tujuh cabang perlombaan, salah satunya adalah Kontes Robot Indonesia Beroda (KRBI-B), dimana disini akan dikonteskan beberapa robot untuk bermain bola. Robot pada perlombaan ini memanfaatkan berbagai sensor sebagai alat bantu, termasuk kamera, sebagai alat visualisasi untuk mengenali lingkungan sekitar. Kamera membantu robot mengumpulkan informasi seperti jarak bola terhadap posisi robot, serta mendeteksi objek-objek seperti bola, gawang, dan penghalang yang berupa robot lawan maupun robot teman. Penggunaan sensor kamera ini menjadi kunci dalam mengatasi tantangan utama di lomba KRI, yaitu mendeteksi dan mengidentifikasi berbagai objek di lapangan.

Dalam melakukan tugas mendeteksi objek yang ada pada lingkungan lomba, tim KRSBI-B dari Universitas Islam Indonesia memanfaatkan segmentasi warna untuk hal tersebut.

`

Metode ini memanfaatkan warna dari sebuah objek serta teknik pada pemrosesan gambar, seperti dilasi dan erosi untuk mengektrak bentuk atau *contour*. Namun metode ini memiliki beberapa kelemahan, salah satu yang dialami tim KRSBIB Universitas Islam Indonesia adalah kalibrasi yang dilakukan untuk mendeteksi objek membutuhkan waktu yang cukup signifikan sehingga dapat menggangu proses persiapan saat perlombaan. Selain itu, berdasarkan penelitian dari Agus Darmawan, metode ini sangat dipengaruhi oleh tingkat pencahayaan sehingga sedikit perbedaan akan mengurangi efektivitas hasil deteksi objek yang dilakukan (Darmawan, 2018)

Untuk itulah dibutuhkan metode lain untuk melakukan tugas yang sama. Penggunaan metode *computer vision* telah menjadi standar dalam pengembangan robot pendeteksi objek di skala industri. Hal ini dikarenakan pengembangan dan penelitian yang dilakukan, baik oleh komunitas maupun akademisi, dalam merancang model yang lebih cepat serta berakurasi tinggi. Beberapa model memiliki kelebihan masing-masing bergantung pada arsitektur model tersebut. Model dengan arsitektur ringan seperti MobileNet dan YOLO banyak digunakan dalam aplikasi deteksi objek karena membutuhkan komputasi yang rendah. Di sisi lain, model seperti Faster R-CNN dan ResNet memiliki tingkat akurasi yang lebih tinggi, namun dengan trade-off pada waktu yang dibutuhkan untuk mendeteksi objek dalam sebuah gambar.

Selain itu penggunaan Robot Operating System juga akan digunakan dalam topik penelitian ini. Robot Operating System atau ROS adalah sebuah library perangkat lunak dan alat yang membantu untuk membangun sistem robotika. Dalam membangun sistem robotika dibutuhkan suatu sistem yang akan menjadi alat komunikasi antar robot, dan disinilah ROS berperang penting. Selain ROS, Gazebo, yang merupakan perangkat lunak simulasi berbasis ROS, juga akan digunakan dalam penelitian ini guna menjadi lingkungan pengujian robot yang sudah dikembangkan. Pada Gazebo inilah robot akan melakukan beberapa kegiatan, seperti melihat lingkungannya, berjalan, dan mendeteksi benda disekitarnya.

Sejalan dengan tren ini, fokus penelitian ini adalah mengembangkan model deteksi objek menggunakan *computer vision* untuk menggantikan metode segmentasi warna yang sebelumnya telah digunakan. Model yang dikembangkan akan diterapkan pada robot dalam simulasi Gazebo, kemudian diuji untuk mendeteksi objek, memperoleh informasi berdasarkan data dari kamera di dalam simulasi, serta melakukan perhitungan jarak. Dalam eksekusinya, dibuat sebuah environment menggunakan sistem operasi Ubuntu Linux dengan beberapa tools bantuan untuk melakukan kontrol dan komunikasi pada robot yang menggunakan Robot Operating System 1 (ROS 1) dan Python.

`

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah disajikan, rumusan masalah yang dapat diajukan adalah

a. Bagaimana mengembangkan sebuah model deteksi objek yang dapat diterapkan pada robot dalam simulasi Gazebo.
b. Bagaimana robot dapat melakukan sebuah aksi setelah mendapatkan informasi dari hasil pendeteksian objek yang telah dilakukan.

Rumusan masalah di atas mencakup beberapa aspek utama yang akan dijelaskan dan dipecahkan dalam penelitian, yaitu pengembangan model deteksi objek, integrasi model tersebut dengan simulasi Gazebo, dan pengembangan kemampuan aksi robot berdasarkan hasil deteksi objek dari kamera dalam simulasi.

1.3 Batasan Masalah

Batasan masalah yang terdapat pada penelitian ini adalah sebagai berikut:

a. Penelitian ini akan memusatkan perhatian pada pengembangan model deteksi objek.
b. Simulasi dalam penelitian ini menggunakan Python 3.9.16, ROS, dan Ubuntu Linux.
c. Penelitian ini akan membatasi jenis objek yang dapat dideteksi oleh model, yang dapat mencakup objek-objek umum seperti bola dan gawang

1.4 Tujuan Penelitian

Tujuan dilaksanakannya penelitian ini adalah Mengembangkan sistem robotika yang mampu melakukan deteksi objek dan mengambil tindakan berdasarkan informasi hasil pendeteksian objek dalam simulasi Gazebo. Adapun tujuan khusus dari penelitian ini, yaitu melakukan implementasi model deteksi objek pada suatu robot dalam lingkungan simulasi gazebo dan melakukan suatu pengukuran jarak antara objek dan robot, yang merupakan salah satu syarat kemampuan robot sepak bola pada perlombaan Kontes Robot Indonesia serta melakukan perbandingan dengan metode segmentasi warna yang pernah diterapkan sebelumnya.

1.5 Manfaat Penelitian

Manfaat dari dilaksanakannya penelitian ini adalah sebagai berikut:

`

a. Memberikan hasil atas model deteksi objek yang dapat diterapkan pada robot di simulasi Gazebo.
b. Memberikan wawasan mengenai bagaimana cara menerapakan model objek deteksi pada robot, khususnya menggunakan simulasi Gazebo.

1.6 Sistematika Penulisan

Sistematika penulisan pada laporan tugas akhir ini dibagi menjadi lima bagian. Secara garus besar, sistematika penulisan akhir ini dapat diuraikan sebagai berikut:

BAB I PENDAHULUAN

Bagian ini akan membahas mengenai latar belakang, rumusan masalah, batasan masalah, tujuan dan manfaat dari penelitian serta susunan penulisan tugas akhir.

BAB II TINJAUAN PUSTAKA

Bagian ini akan memberikan gambaran mengenai studi-studi sebelumnya yang telah dilaksanakan dan relevan dengan isu yang sedang dibahas, serta merujuk pada referensi yang telah digunakan**.**

BAB III METODOLOGI

Bagian ini menjelaskan metode yang digunakan mulai dari analisis masalah, perancangan sistem, implementasi sistemm, dan pengujian sistem.

BAB IV HASIL DAN PEMBAHASAN

Bagian ini membahas hasil yang telah didapatkan berdasarkan aktivitas yang telah dilakukan

BAB V KESIMPULAN DAN SARAN

Bagian ini akan dijelaskan kesimpulan yang ditarik berdasarkan temuan dari penelitian yang telah dilakukan, sekaligus memberikan rekomendasi atau saran yang dapat diterapkan dari hasil implementasi sistem, yang dapat menjadi masukan yang berharga untuk penelitian selanjutnya.

`

BAB II
TINJAUAN PUSTAKA

2.1 Landasan Teori

2.1.1 Gazebo

Gazebo adalah kumpulan perangkat lunak open source yang dirancang untuk menyederhanakan pengembangan aplikasi berkinerja tinggi. Gazebo dirancang untuk menciptakan lingkungan multi-robot dinamis 3D yang mampu mereplikasi dunia kompleks yang akan dihadapi oleh generasi robot berikutnya. Gazebo menawarkan lingkungan yang kaya untuk mengembangkan dan menguji sistem multi-robot dengan cepat dalam cara-cara baru dan menarik. Ini adalah alat yang efektif, dapat diskalakan, dan sederhana yang juga berpotensi membuka bidang penelitian robotika kepada komunitas yang lebih luas. Oleh karena itu, Gazebo sedang dipertimbangkan untuk digunakan dalam pengajaran sarjana, khususnya dalam pembelajaran robotika (P. Koenig & Howard, 2004).

Pengunaan Gazebo juga dapat menekan biaya pengunaan perangkat keras yang dibutuhkan di saat melakukan pengujian robot. Gazebo, yang merupakan *tools* dalam pengujian virtual, sangatlah efisien, tidak mahal, dan sangat independen dari pengunaan perangkat lunak yang harus disediakan dan dapat membuka peluang untuk penerapan konsep baru dan metode baru dengan robot yang sudah ada (Abbyasov, et al., 2020).

2.1.2 Robot Operating System

Robot Operating System atau ROS adalah sebuah sistem operasi robot yang bersifat open source. Di dalam ROS, terdapat berbagai *library* dan alat (tools) yang digunakan untuk mengembangkan perangkat lunak robot. ROS berfungsi sebagai middleware robot yang memiliki fleksibilitas untuk menghubungkan perangkat keras robot dengan sistem operasi komputer. Tujuan utama dari ROS adalah untuk menyederhanakan proses pengembangan perangkat lunak robot, menghindari keharusan membuat kode sumber dari awal, dan mendorong kolaborasi pengembangan bersama. ROS.

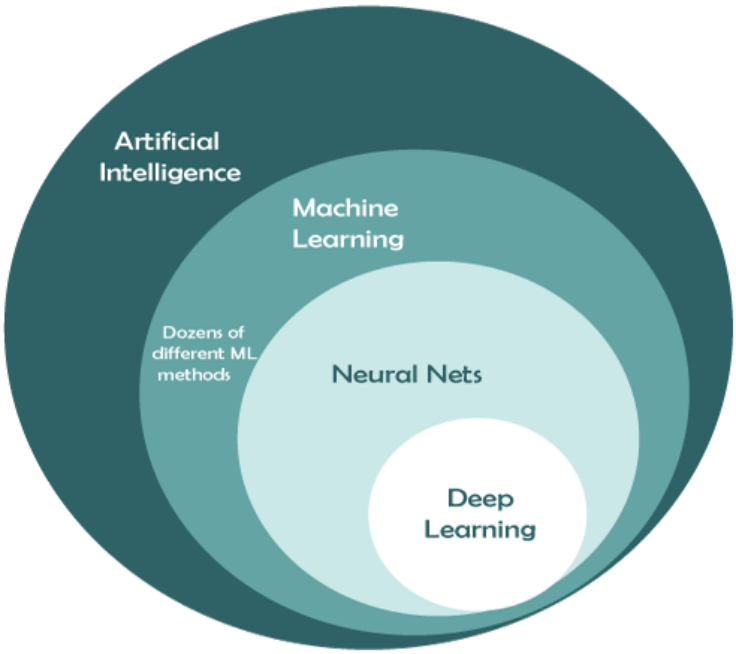
ROS menjadi *tools* yang cukup revolusioner di bidang robotika karena elah mentediakan beberapa alat, infrastruktur, dan metode untuk membuat suatu perangkat lunak robotika dengan libraries yang mereka sediakan. Penggunaan ROS dalam dunia robotika telah memudahkan

`

masalah yang dihadapi oleh para pengembang robot, yaitu komunikasi lancar antar komponen. Dengan penggunaan ROS, komponen-komponen ini dapat berkomunikasi dengan lancar berkat pengunaan konsep node yang disediakan ROS. Nod-node ini dapat berkomunilasi, baik menerima atau mengirim pesan, melalui suatu *topic* sebagai pesan.

2.1.3 Deep Leaning

Deep learning merupakan salah satu bagian dalam bidang *machine learning* yang merupakan metode inferensi data berdasarkan data-data dikenali oleh suatu model. Metode ini menjadi kunci dalam perkembangan *artifical intelligence* modern, dikarenakan kemampuan pengingkatan performa akurasi berdasarkan jumlah data yang dimiliki. Himpitan antara deep learning dan sub-bidang *artifical intelligence* lainnya dapat dilihat pada Gambar 2.1.



Gambar 2.1 Empat pekerjaan inti pada computer vision

Sumber: javapoint, Deep learning vs. Machine learning vs. Artificial Intelligence.

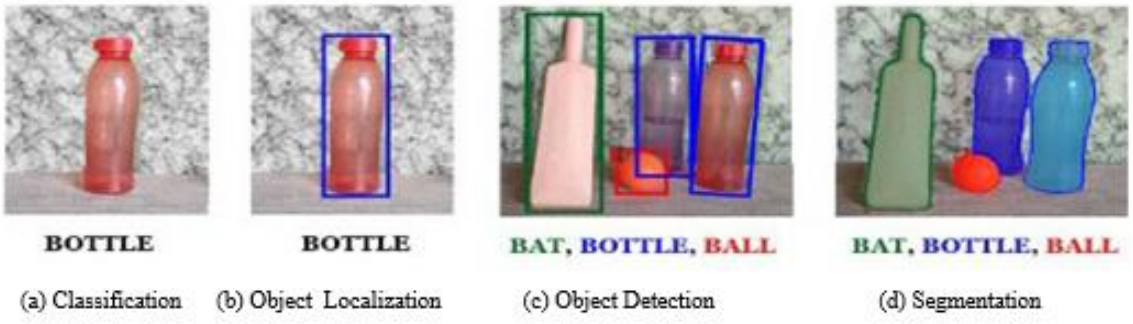
Kelebihan *deep learning* dibandingkan metode machine learning lainnya berada pada scaling yang dapat dilakukan *deep learning* sejalan dengan jumlah data yang dimiliki. Selain itu, deep learning dapat mengekstraksi fitur yang terdapat pada data lebih baik dibandingkan metode yang lain. Selain itu, dengan adanya algortima back propagation, metode ini dapat mengatur ulang lagi nilai-nilai *weight* berdasarkan hasil latih agar model memiliki performa jauh lebih baik.

`

2.1.4 Computer Vision

Computer vision merupakan salah satu disiplin ilmu komputer yang bertujuan untuk membuat komputer dapat melihat dan memahami dunia seperti layanya manusia (Matsuzaka & Yashiro, 2023). Proses dalam melihat lingkungan ini melibatkan pengembangan algoritma dan teknik untuk mengekstraksi fitur, pola, dan pengetahuan yang bermakna dari data visual, meniru kemampuan persepsi visual manusia (Pandey, 2023). Fitur yang dilakukan ekstraksi bisa berbagai macam, seperti garis, bentuk (contour), warna, dan lainnya sehingga dapat mengeneralisasi suatu objek dengan berbagai macam variasi. Hal ini tentu saja memudahkan robor sepak bola untuk mendeteksi objek tanpa perlu melakukan kalibrasi di tahap persiapan.

Setidaknya terdapat empat pekerjaan inti dalam computer vision, yaitu classification, object localization, object detection, dan segmentation (Alam, et al., 2022).



Gambar 2.2 Empat pekerjaan inti pada computer vision

Sumber: Leveraging Deep Learning for Computer Vision: A Review (2022)

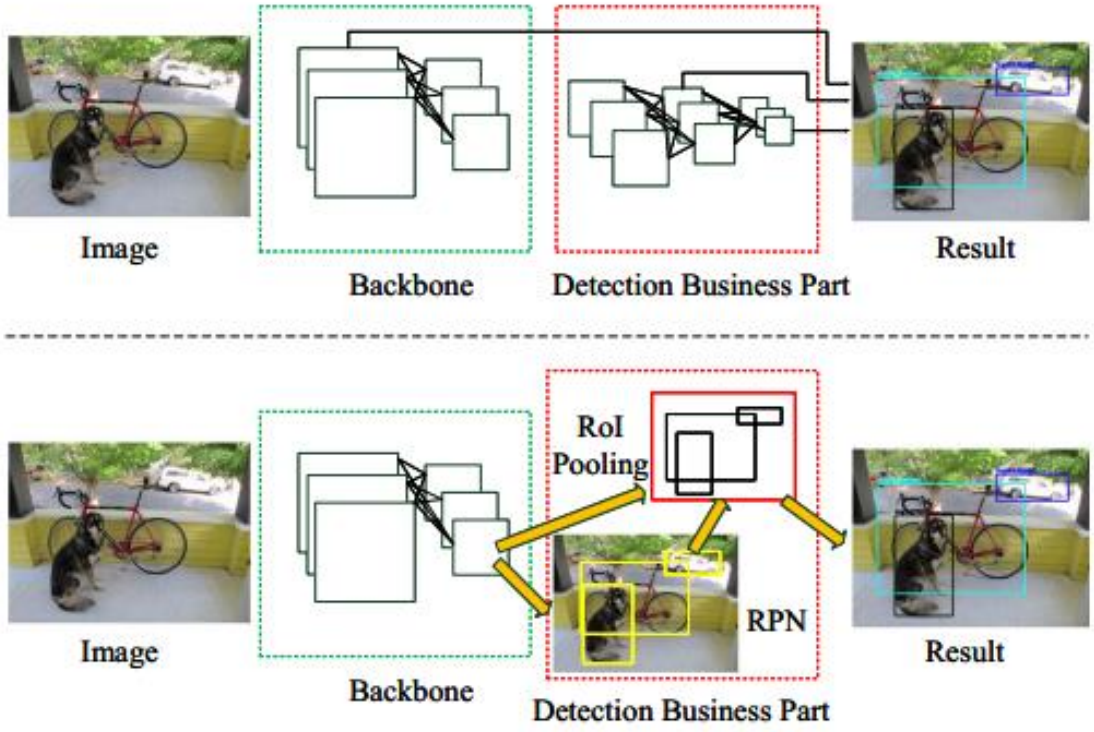
Keempat jenis pekerjaan ini memiliki peran dan kegunaannya masing-masing. Selain itu, bisa disebutkan bahwa jenis pekerjaan ini merupakan turunan atau peningkatan dari kenis pekerjaan lainnya. Sebagai contoh object detection merupakan penerapan lanjutan dari object localization dan classification dan segmentation merupakan metode yang jenis pekerjaan yang menerapkan object detection dan *segmentation mask*.

2.1.5 Object Detection

Object detection meruapakan salah satu sub-bidang dalam *computer vision* yang memiliki fokus tujuan untuk mendeteksi objek yang termasuk dalam target kelas tertentu dengan lokalisasi absolut dalam sebuah pemandangan realistis atau gambar input, serta memberikan setiap objek yang terdeteksi sebuah kelas yang termasuk dalam kelasnya (Sharma & Mir, 2020). Berdasarkan arsitektur dan cara kerja, model object detection terbagi menjadi dua jenis,

`

yaitu *one-stage* atau *proposal-free* model detector dan *two-stage* atau *proposal-model*. Pada metode two-stage, serangkaian kandidat objek akan di generate oleh *Selective Search* (Ujilings, van de Sande, Gevers, & Smeulders, 2013) atau *Edge Boxes* (Lawrence & Dollar, 2014) , kemudian lokasi objek tertentu dan label kategori yang sesuai ditentukan dengan menggunakan jaringan konvolusional (Zhang & Hong, 2019).



Gambar 2.3 Arsitektur *one-stage* (atas), arsitektur *two-stage* (bawah)
Sumber: Recent progresses on object detection: a brief review (2019)

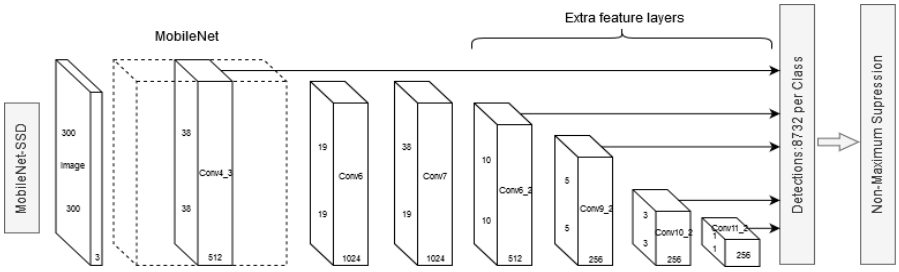
Perbedaan atas arsitektur inilah yang membedakan kedua metode tersebut. metode two-stage berfokus pada meningkatkan akurasi pada dataset evaluasi standar seperti PASCAL VOC dan MS COCO. Di lain sisi, metode one-stage berjalan lebih baik dalam hal kecepatan dan karena model ini memiliki parameter yang lebih sedikit, maka model one-stage memiliki size yang lebih kecil dibandingkan two-stage detector.

2.1.6 MobileNet SSD

MobileNet SSD atau Single Shot Detection merupakan salah satu model *one-stage* yang memiliki model *size* yang ringan dan memiliki kecepatan inferensi yang cepat. Model ini dibuat oleh para peneliti dari Google pada tahun 2016. Arsitektur ini memperkenalkan model deteksi

`

objek yang menggunakan jaringan saraf dalam tunggal yang menggabungkan *Region Proposed Network* (RPN) dan ekstraksi fitur (Palwankar & Kothari, 2022).



Gambar 2.4 Arsitektur SSD MobileNet V2

Model ini bekerja dengan membuat sekumpulan bounding box default dengan berbagai aspek rasio dan skala yang digunakan dan diterapkan pada peta fitur. Peta fitur dihitung dengan melewatkan gambar melalui jaringan klasifikasi gambar, sehingga ekstraksi fitur untuk kotak pembatas dilakukan dalam satu langkah. Skor dihasilkan untuk setiap kategori objek dalam setiap kotak default. Untuk lebih sesuai dengan kotak kebenaran dasar, penyesuaian offset dihitung untuk setiap kotak. Peta fitur yang berbeda dalam jaringan konvolusi sesuai dengan bidang reseptif yang berbeda dan digunakan untuk menangani objek pada skala yang berbeda secara alami. Karena semua perhitungan dilakukan dalam satu jaringan, kecepatan komputasi yang didapatkan cukup tinggi. Kelebihan lain dari model ini dibandingkan dengan YOLO adalah jumlah computing unit yang dibutuhkan lebih sedikit, ukuran yang lebih kecil, dan kecepatan dalam melakukan inferensi yang sedikit lebih cepat.

2.2 Kajian Pustaka

Penelitian sebelumnya yang serupa akan dijadikan sebagai bahan referensi dalam pembahasan saat ini. Tujuannya adalah untuk menunjukkan bahwa penelitian yang sedang dilaksanakan memiliki nilai kontribusi yang unik dalam perkembangan ilmu pengetahuan, serta untuk menghindari duplikasi penelitian yang memungkinkan.

`

Tabel 2.1 Analisis kajian penelitian yang sudah ada

No.	1	2	3
Penulis	**Wahyu Muldayani, Sumardi, Arizal Mujibtamala N.I, Muhammad Firza, Taufan Firmansyah** (Muldayani, Sumardi, N.I, Firza, & Firmansyah, 2023)	**Farhan Fadhillah Sanubari, Riky Dwi Puriyanto** (Sanubari & Puriyanto, 2022)	**Fadhli Dzil Ikram, Agus Khumaidi, M. Basuki Rahmat, Joko Endrasmono, Mat Syai'in, Dimas Pristovani, Riananda** (Ikram, et al., 2024)
Judul Penelitian	IMPLEMENTASI SISTEM OBJECT TRACKING UNTUK MENDETEKSI DUA OBJEK BERBASIS DEEP LEARNING	Deteksi Bola dan Gawang dengan Metode YOLO Menggunakan Kamera Omnidirectional pada Robot KRSBI-B	Deteksi Objek di Lapangan pada Robot SepakbolaBerodaMenggunakan Metode YOLOV5
Metode Penelitian	Computer Vision Model YOLOv5	Computer Vision Model YOLOv3 dan YOLOv3-Tiny	Computer Vision Model YOLOv5
Kelebihan	Memiliki nilai nilai precision dan recall lebih baik	Memiliki nilai akurasi sedikit lebih tinggi	Memiliki akurasi yang cukup tinggi
Kekurangan	Lebih memakan resource CPU dibanding SSD	Memiliki nilai loss classification yang lebih besar	Confidence_score pada metode ini masih kecil (0.467)
Perbedaan	Metode menggunakan YOLOv5	Metode menggunakan YOLOv3 dan v3-Tiny	Metode menggunakan YOLOv5 dan Pytorch

`

2.2.1 Robot Pendeteksi Objek

Penelitian yang dilakukan (Auliya, Dewi, Noer, & Oktarina, 2022) dan (Muldayani, Sumardi, N.I, Firza, & Firmansyah, 2023) akan dijadikan referensi dalam penelitian ini. Kedua penelitian tersebut menggunakan model object detection yang sama, YOLO, dengan versi yang berbeda dan fokus dalam kedua penelitian tersebut sejalan dengan penelitian yang dilakukan, yaitu menerapkan model deep learning pada robot. Perbedaan yang terdapat dari penelitian tersebut terdapat pada environment yang digunakan dalam melakukan percobaan. Pada kedua penelitian tersebut model digunakan pada robot yang tersedia, tidak menggunakan simulasi. Hasil pada kedua penelitian tersebut dalam melakukan objek deteksi dapat dilihat pada Gambar 2.5 dan Gambar 2.6.

Gambar 2.5 Hasil deteksi sistem object tracking mendeteksi dua objek

`

Gambar 2.6 Hasil deteksi pengolahan citra menggunakan metode YOLO

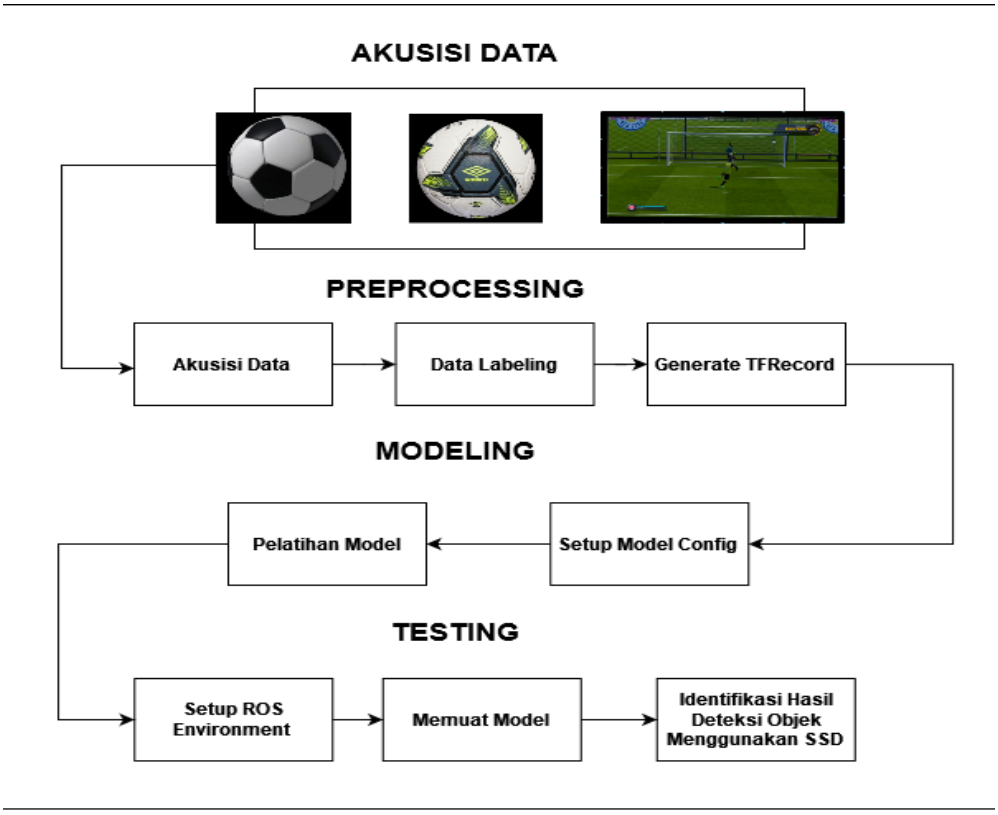
2.2.2 Simulasi Gazebo dan ROS

Dalam penggunaan Gazebo dan ROS, penelitian ini mengambil penelitian (Dwiyanto & Fath, 2020) sebagai referensi. Pada penelitian tersebut membahas mengenai transmisi data yang dilakukan pada *multiple machine* menggunakan ROS. Terdapat perbedaan pada penilitian tersebut dengan penelitian yang dilakukan, salah satunya adalah tujuan utama dalam penelitian yang diambil sebagai referensi adalah melakukan analisa sistem komunukasi data dan untuk penelitian yang sedang dilakukan berfokus pada pengunaan Gazebo untuk melakukan simulasi implementasi robot pendeteksi objek. Meskipun begitu, penelitian tersebut dapat digunakan sebagai referensi dalam melakukan transmisi data menggunakan ROS yang dilakukan disaat melakukan simulasi. Transmisi ini berupa melakukan konfigurasi pada file *launch* agar dapat menerima dan mengirim (*publish/subscribe*) pada sebuah *topic* dengan benar

`

BAB III
METODOLOGI

Dalam penelitian ini akan, terdapat tahapan atau alur yang akan dilaksanakan untuk mengetahui kinerja model deteksi objek dalam mendeteksikan dua objek utama, yaitu bola dan gawang, alur dalam penelitian ini dapat dilihat pada Gambar 3.1.



Gambar 3.1 Proses alur dalam pengembangan model deteksi objek

Tahapan ini meliputi akuisisi data, tahapan ini merupakan tahapan awal dimana data berupa gambar, baik bola dan gawang, dikumpulkan menjadi sebuah dataset, dan pembagian data menjadi training dan validation. Pada tahap kedua akan dilanjutkan dengan melakukan *preprocessing*, dimana pada tahapan ini akan dilakukan data labeling, memberikan keterangan kelas objek dan posisi objek pada gambar, serta melakukan konversi dataset menjadi sebuah kesatuan format yang dapat diterima oleh Tensorflow Object Detection API, yaitu tfrecord. Setelah itu, dilanjutkan pada tahap modeling, dimana data pada tahap sebelumnya akan digunakan untuk melatih model deteksi objek.Tahapan ini juga akan menguji kinerja model pada masa training serta melakukan pengujian menggunakan data validation yang sudah

`

disiapkan pada tahapan sebelumnya .Pada tahap terakhir dan akan menjadi inti pada penelitian ini, dimana model akan diimplementasikan kepada robot turtlebot pada lingkungan simulasi gazebo ROS, dan akan dilakukan pengujian untuk mengukur jarak dari robot ke objek, yang merupakan salah satu kebutuhan untuk robot sepak bola, dengan menggunakan beberapa skenario.

3.1 Akuisisi Data

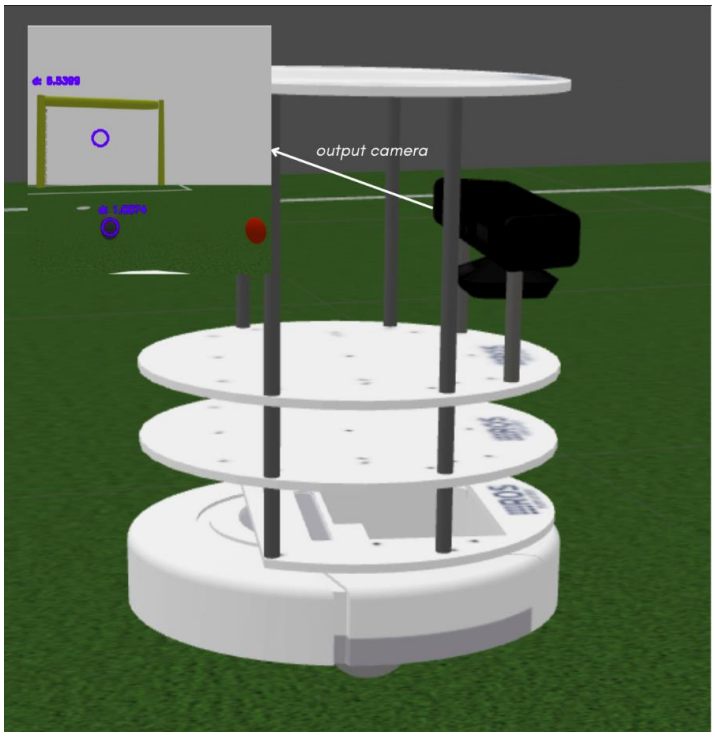
Tahap ini merupakan tahap awal dalam proses pembuatan model deteksi objek. Pada tahap ini dilakukan proses pengumpulan data-data, baik data primer maupun sekunder, yang akan digunakan sebagai dasar pelatihan untuk model deteksi objek. Beberapa dataset tersebut didapatkan melalui halaman web kaggle dan roboflow, dimana data bola didapatkan pada halaman website kaggle dan data gawang, dan bola, didapatkan pada halaman roboflow. Total data yang diperoleh dari kedua halaman tersebut berupa 35.895 gambar bola dan 268 gambar gawang. Gambar-gambar yang didapat baik dari halaman web akan dilakukan filter untuk mendapatkan dataset dengan kualitas yang baik untuk dilakukan training. Data-data ini memiliki variasi dalam perspektif objek terhadap kamera, maka dari itu lebih dari 65% data yang akan digunakan pada training akan mengambil data dengan perspektif yang mendekati dengan lingkungan simulasi, dimana kamera terpasang di robot dengan posisi tegak lurus.

Gambar 3.2 Contoh gambar yang digunakan dalam dataset (kiri), gambar yang tidak digunakan (kanan).

Selain gambar dari halaman website yang sudah disebutkan, gambar juga didapatkan pada kamera yang terpasang pada robot di gazebo. Kamera yang terpasang merupakan *kinect camera*

`

yang menghasilkan gambar berukuran 640 × 480 pixel dengan format gambar R8G8B8. Kamera terletak pada robot dengan posisi tegak lurus dengan *body* robot seperti terlihat pada Gambar 3.3.



Gambar 3.3 kinect camera pada robot dan output yang dihasilkan.

3.2 Data Preprocessing

Tahap berikutnya setelah melakukan akuisisi data dalam pembuatan model deteksi objek adalah melakukan labeling data. Labeling data merupakan proses manual memberikan kelas dan bounding box (kotak) kepada objek di gambar. Dalam melakukan proses labeling ini, maka akan digunakan aplikasi open source labelImg. Labeling ini bertujuan untuk memberitahu model bahwa data yang sedang dikonsumsi pada masa training memiliki objek yang berlabel, baik bola atau gawang, dan posisi objek di suatu gambar. Hasil dalam label ini merupakan sebuah file XML yang memuat informasi mengenai gambar dan objek yang ada pada gambar tersebut, seperti lokasi titik x dan y objek pada gambar dan path file gambar yang terlabeli. Dalam melakukan pelabelan objek pada gambar, maka sangat diutamakan bahwa objek yang terlabeli mencakup bagian besar yang ada di suatu gambar. maka jika terdapat beberapa objek pada suatu gambar yang dapat dilihat pada Gambar 3.4.

`

Gambar 3.4 Objek pada gambar yang terlabelkan sesuai kelas.

Selanjutnya maka akan dilakukan konversi dataset untuk mengubah data yang sudah dilabeli menjadi sebuah format yang dapat diterima disaat melakukan model *training*, yaitu mengubah gambar yang sudah dilabeli menjadi format .tfrecord agar model dapat dikonsumsi oleh Tensorflow Object Detection API.

3.3 Modelling

Setelah melakukan *preprocessing*, maka proses yang dilakukan selanjutnya adalah melakukan modeling yang akan dibagi menjadi dua tahap, yaitu konfigurasi model dan parameter tuning, dan model fitting dan model evaluation. Dalam penelitian ini akan digunakan model SSD MobileNet V2 300*, sebuah model *one-stage* yang memiliki size ringan dan memiliki benchmark yang cukup baik untuk pendeteksian objek, bahkan jika digunakan pada microprocessor.

3.3.1 Konfigurasi Model dan Parameter Tuning

Pada proses awal dalam pemodelan model deteksi objek akan dilakukan konfigurasi model SSD MobileNet V2 300* yang dilakukan pada file .config serta melakukan tuning pada beberapa parameter kritis yang perlu diatur agar model dapat menghasilkan deteksi objek yang optimal. Beberapa parameter yang dioptimalkan antara lain:

1. num_classes: parameter ini menentukan jumlah kelas objek yang akan diidentifikasi oleh model .
2. learning_rate: parameter ini menjadi faktor penting yang mempengaruhi kecepatan konvergensi model selama pelatihan.

`

3. batch_size: parameter ini menentukan jumlah sampel yang digunakan dalam satu iterasi pelatihan.
4. num_steps: mengindikasikan seberapa sering model melihat seluruh dataset. Pengaturan yang tepat dapat memastikan bahwa model telah melihat dataset dengan cukup baik untuk pembelajaran yang efektif.
5. max_number_of_boxes: parameter ini menentukan jumlah maksimum kotak pembatas objek yang dapat dihasilkan oleh model pada satu gambar.

3.3.2 Model Fitting dan Model Evaluation

Tahapan ini akan dilakukan training model yang sudah dikonfigurasikan dengan dataset yang sudah di-*preprocess* pada tahap sebelumnya. Pada proses ini akan sekaligus melakukan evaluasi model pada setiap 1000 steps-nya menggunakan data validasi yang juga sudah disiapkan pada tahap *preprocessing*. Hal ini tentunya dilakukan untuk mengetahui kinerja dari model deteksi objek yang sedang dilakukan training pada setiap iterasinya. Untuk mengetahui kinerja dari model, maka akan digunakan tiga buat *metric loss*, yaitu *classification loss*, untuk mengukur seberapa kesalahan model dalam mengidentifikasikan objek yang terdeteksi; *localization loss*, untuk mengukur kesalahan dalam pembuatan bounding box antara data yang terlabel dengan data prediksi hasil deteksi model; dan *total loss*, yang akan mengukur kinerja model secara keseluruhan, baik dalam mengidentifikasikan dan membuat bounding box pada objek tersebut di suatu gambar. Mendapatkan nilai-nilai ini akan didapatkan dengan beberapa perhitungan. Nilai dari *classification loss* akan didapatkan dari *cost function* yang digunakan dalam pelatihan model deteksi objek, yaitu *cross-entropy*. Persamaan untuk cross-entropy tersebut dapat dilihat seperti pada persamaan 3.1

$$\mathrm{H}(\mathrm{y}, \hat{\mathrm{y}}) = f(x) = -\sum_{i}^{n} (y_i \log(\hat{y}_i)) \qquad (3.1)$$

Keterangan:

y: label benar

ŷ: label hasil prediksi

Sedangkan untuk mendapatkan dari *localization loss* akan didapatkan menggunakan *smooth function* yang berdasar pada *Huber loss*. Persamaan untuk *smooth function* ini dapat dilihat pada persamaan 3.2 dan *Huber loss* dapat dilihat pada persamaan 3.3.

`

$$\mathrm{L}(\mathrm{y},\hat{\mathrm{y}}) = \sum_i^n \sum_j^m SmoothL1(y_{i,j} - \hat{y}_{i,j}) \quad (3.2)$$

$$\mathrm{SmoothL1}(\mathrm{x}) = \begin{cases} 0.5\,x^2\ if\ |x| < 1 \\ |x| - 0.5\ otherwise \end{cases} \quad (3.3)$$

Keterangan:

y: label benar

ŷ: label hasil prediksi

Setelah mendapatkan kedua nilai dari loss tersebut, maka kita dapat menghitung total loss dari model deteksi objek dengan cara menambahkan baik *classification loss* dengan *localization loss* seperti pada persamaan 3.4.

$$\text{Total Loss} = \mathrm{H}(\mathrm{y},\hat{\mathrm{y}}) + \mathrm{L}(\mathrm{y},\hat{\mathrm{y}}) \quad (3.4)$$

Selanjutnya untuk menguji kinerja model dalam melakukan deteksi menggunakan data validasi, maka dilakukan evaluasi model yang dilakukan bersamaan dengan masa training model pada tiap 1000 steps menggunakan data yang terpisah dengan data training yang digunakan. Berbeda dengan mengukur kinerja model pada masa training, pada masa evaluasi akan digunakan matrics Mean Average Precision (mAP) dan Recall dalam mengukur kinerja dari model yang telah dilakukan training. Dalam mengetahui nilai-nilai metrik tersebut, maka akan digunakan Tensorboard sebagai alat untuk memudahkan menganalisa kinerja model selama dilakukannya training dan evaluasi.

3.4 Testing

Setelah melakukan training model dan melakukan evaluasi atas kinerja model menggunakan dataset yang ada, makan akan dilanjutkan pada pengujian yang akan dilakukan pada lingkungan simulasi gazebo ROS. Dalam melakukan pengujian pada tahapan ini, maka akan terdapat beberapa hal yang harus dilakukan. Beberapa hal tersebut meliputi:

1. Menyiapkan lingkungan simulasi yang akan digunakan menggunakan gazebo ROS dengan membuat .world file .

`

2. Memasang aset model yang akan digunakan ke dalam .world file yang sudah dibuatkan, aset ini meliputi model robot yang akan digunakan serta model pendukung lainnya.
3. Membuat script yang akan melakukan pendeteksian objek, pengambilan data yang berasal dari model pada lingkungan simulasi, dan meng-outputkan informasi yang didapatkannya.

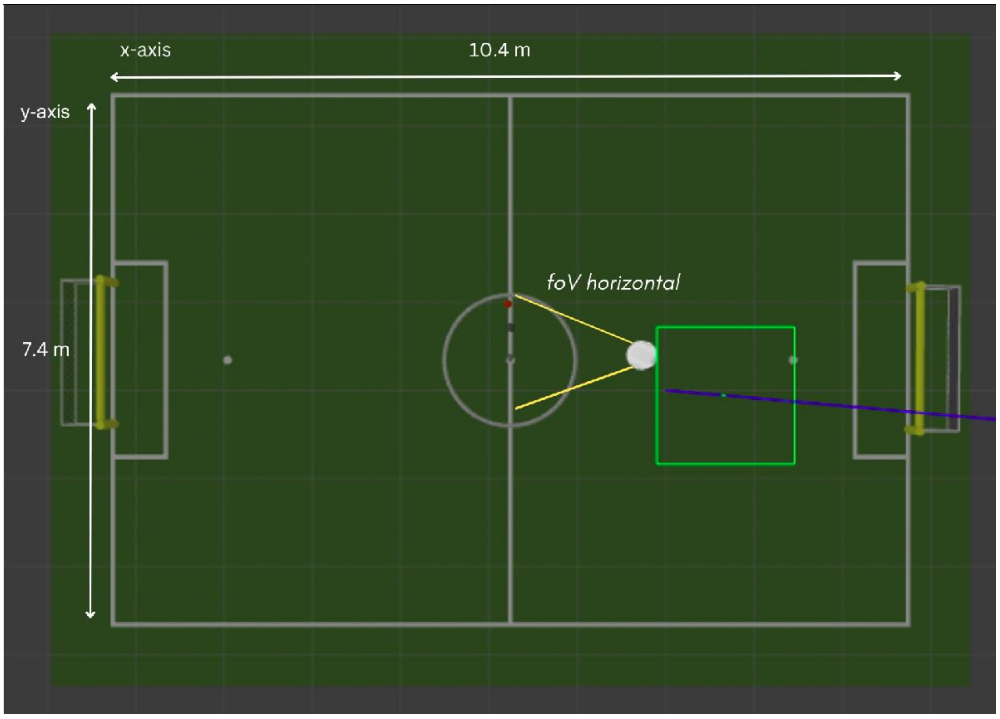
3.4.1 Menyiapakan Lingkungan Simulasi Gazebo ROS

Menyiapakan lingkungan simulasi pada Gazebo ROS akan menjadi tahap pertama dalam melakukan pengujian. Pada tahapan ini akan dibuat sebuah file .world yang akan menampung segala aset model pada gazebo ROS. File ini merupakan sebuah file berformat SDF atau Simulation Description Format, yaitu format berbentuk XML yang menggambarkan objek dan lingkungan untuk simulasi robot, visualisasi, dan kontrol (SDFormat, n.d.).

```
<?xml version="1.0" ?>
<sdf version="1.4">
  <world name="default">
    <include>
      <uri>model://ground_plane</uri>
    </include>
    <include>
      <uri>model://sun</uri>
    </include>
  </world>
</sdf>
```

Gambar 3.5 konfigurasi default file .world pada gazebo ROS

`



Gambar 3.6 Tampilan lingkungan simulasi yang akan digunakan.

Dalam menjalankan gazebo ROS untuk memuat file .world yang sudah disiapkan, maka akan digunakan command pada ROS, yaitu roslaunch. Command ini akan menjalankan file .launch, file init untuk menjalankan beberapa ROS package, untuk memuat dan memulai lingkungan simulasi serta aset yang ada di dalamnya. file launch dapat dilihat seperti pada Gambar 3.7.

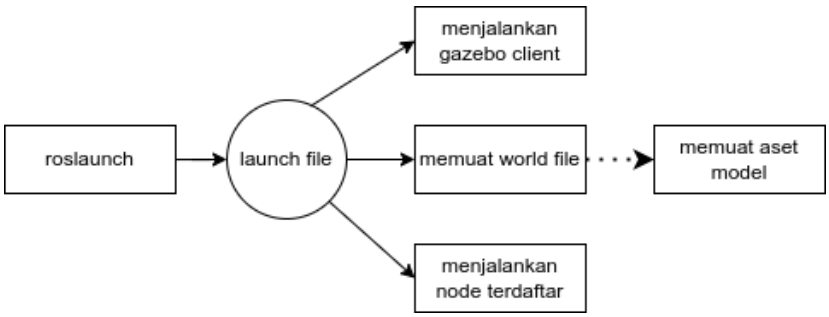
```
<?xml version="1.0"?>
<launch>
  <!-- arguments you can pass this launch file-->
  <!--   Note   that   'headless'   is   currently   non-functional.    See
gazebo_ros_pkgs
 <arg name="world_name" default="worlds/ta.world"/> <!-- load world file --
>

  <!-- set command arguments →

  <!-- node to start gazebo server-->
  <node
   name="gazebo"
   pkg="gazebo_ros"
   type="$(arg script_type)"
   respawn="$(arg respawn_gazebo)"
   output="$(arg output)"
   args="$(arg  command_arg1)  $(arg  command_arg2)  $(arg  command_arg3)  -e
$(arg physics) $(arg extra_gazebo_args) $(arg world_name)"
  required="$(arg server_required)" />
```

`

```
  <!-- node start gazebo client -->
  <group if="$(arg gui)">
    <node
    name="gazebo_gui"
    pkg="gazebo_ros"
    type="gzclient"
    respawn="false"
    output="$(arg output)"
    args="$(arg command_arg3)"
    required="$(arg gui_required)"
     />
  </group>
</launch>
```

Gambar 3.7 konfigurasi default file .world pada gazebo ROS

Sederhananya roslaunch akan menjalankan file .launch yang memuat beberapa konfigurasi untuk menjalankan gazebo, memuat file .world beserta asetnya, dan menjalankan beberapa node yang didefinisikan pada file .launch tersebut. Gambaran untuk alur tersebut dapat dilihat dengan jelas pada Gambar 3.8.

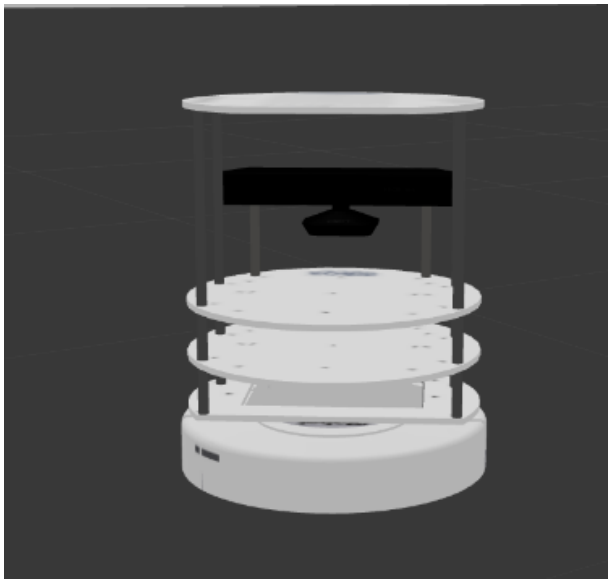


Gambar 3.8 alur yang terjadi saat menjalankan roslaunch

3.4.2 Aset Model SDF Gazebo ROS

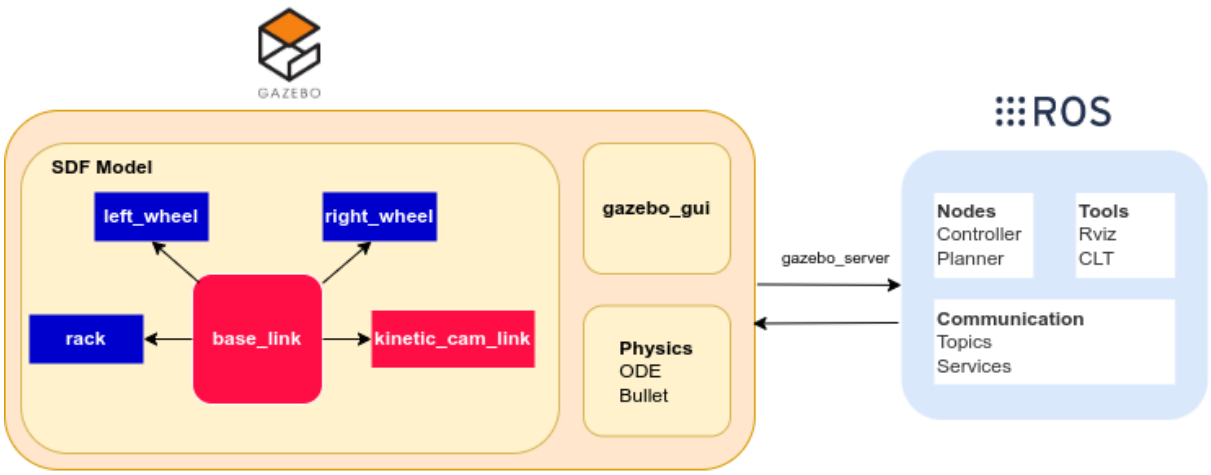
SDF, yang merupakan singkatan dari Simulation Description Format, merupakan file berjenis XML, Extended Markup Language, yang digunakan pada gazebo untuk mendeskripsikan properti fisik pada objek di gazebo. File SDF ini juga disertakan informasi mengenai sensor, posisi robot pada lingkungan simulasi, inersia, massa, dan geometri. Pada penelitian ini peneliti akan menggunakan model robot bernama turtlebot. Bentuk turtlebot pada gazebo akan terlihat seperti Gambar 3.9.

`



Gambar 3.9 model turtlebot

Seperti yang sebelumnya disebutkan beberapa model pada SDF memiliki sensor dan aktuator di dalamnya. Nilai pada sensor dan aktuator ini bisa didapatkan melalui nantinya bisa didapatkan melalui *publisher* pada ROS. Proses pengiriman data ini melalui beberapa hal, pertama setiap model yang memiliki sensor atau aktuator akan memiliki property *link* dalam file SDF model tersebut. Property ini akan membaca nilai sensor dan aktuator di lingkungan simulasi dan mengirimnya menuju gazebo_server, dimana gazebo_server ini terhubung dengan ROS disaat melakukan command roslaunch dan nantinya nilai pada sensor suatu model akan terbaca sebagai topic pada terminal di ROS. Penjelasan terkait komunikasi ini dijelaskan singkat melalui ilustrasi pada Gambar 3.10.



Gambar 3.10 alur komunikasi ros dan gazebo

`

3.4.3 Pembuatan Controller atau Script

Pembuatan script akan digunakan untuk menerima dan mengirim data dari dan ke gazebo melalui komunikasi jaringan ROS. Script yang dibuat pada penelitian ini berfungsi sebagai kontrol, khususnya dalam mengatasi gambar yang diterima melalui suatu topic publisher. Dalam script ini akan dilakukan preprocessing data, melakukan prediksi menggunakan model yang ada, dan membuat bounding box serta menampilkan informasi mengenai objek yang ada di sekitarnya melalui subscriber. Selain itu script ini juga akan memanipulasi pose robot di lingkungan simulasi agar robot dapat melakukan navigasi dan melihat sekelilingnya. Dalam melakukan perubahan atau manipulasi pose robot di lingkungannya, maka script akan melakukan publish message menuju gazebo melalui topic /gazebo/set_model_state agar pose robot dapat berubah. Proses menerima dan mengirim message ini lah yang akan menjadi dasar utama robot dalam mengambil informasi mengenai lingkungan sekitarnya.

Script-script ini akan dikumpulkan menjadi suatu package. Package ini merupakan package khusus yang di-*generate* menggunaakan suatu command pada ROS. Pembuatan package ini merupakan langkah yang wajib dilakukan dalam setiap pembuatan script yang akan dieksekusi oleh ROS. Package ini berperan penting dalam mengatur script agar dapat dijalankan dan memanfaatkan library yang disediakan oleh ROS secara optimal.

```
catkin_create_pkg nama_package dep1 dep2 dep..
```

Gambar 3.11 Command yang digunakan untuk membuat package pada ROS.

Package ini akan membuat satu folder, yang berisikan dua file konfigurasi. File ini masing-masing berupa file package xml yang digunakan untuk penyedia informasi untuk ROS, bahwa package ini merupakan suatu node yang dapat menerima dan meberi pesan kepada server ROS. Package ini berisi informasi-informasi speerti nama dari package, maintainer, build-tool yang digunakan untuk membuat package dan lain-lainnya seperti yang dapat dilihat pada Gambar 3.12.

```
<?xml version="1.0"?>
<package format="2">
  <name>robot_remote_control</name>
  <version>0.0.0</version>
  <maintainer email="irizqy@todo.todo">irizqy</maintainer>
  <license>TODO</license>
```

`

```
  <buildtool_depend>catkin</buildtool_depend>

  <export>

  </export>
</package>
```

Gambar 3.12 Informasi yang diberikan package.xml pada dan untuk ROS.

Package tambahan yang dihasilkan adalah package konfigurasi berformat CMake. Package ini berfungsi untuk mendeklarasikan dependencies atau library ROS yang akan digunakan, menentukan script yang termasuk dalam package, serta informasi lainnya. Setelah membuat package baru, konfigurasi pada package ini perlu diperbarui untuk memastikan ROS mengetahui lokasi script yang terkait dengan package tersebut, lihat Gambar 3.13.

```
cmake_minimum_required(VERSION 3.0.2)
project(robot_remote_control)

find_package(catkin REQUIRED)


add_service_files(
   FILES
   Service1.srv
   Service2.srv
)

add_action_files(
   FILES
   Action1.action
   Action2.action
)

${catkin_EXPORTED_TARGETS})

# setup python files to be recognize by ROS system
catkin_install_python(PROGRAMS
  src/remote_control.py
  DESTINATION ${CATKIN_PACKAGE_BIN_DESTINATION}
)
```

Gambar 3.13 Konfigurasi yang terdapat pada Cmakelist.txt

3.4.4 Perhitungan Jarak Objek

Sebagai penambahan pada tahap sebelumnya, pada penelitian ini robot akan melakukan perhitungan atas estimasi jarak antara robot dengan objek. Hal ini dilakukan untuk memenuhi beberapa kebutuhan pada dalam robot sepak bola, yaitu mendeteksi dan mengenali objek di sekitarnya, di sini bola dan gawang, dan menghitung jarak robot ke objek. Dalam

`

mengaproksimasi jarak antara posisi robot dan objek yang terdeteksi, maka akan digunakan persamaan 3.5.

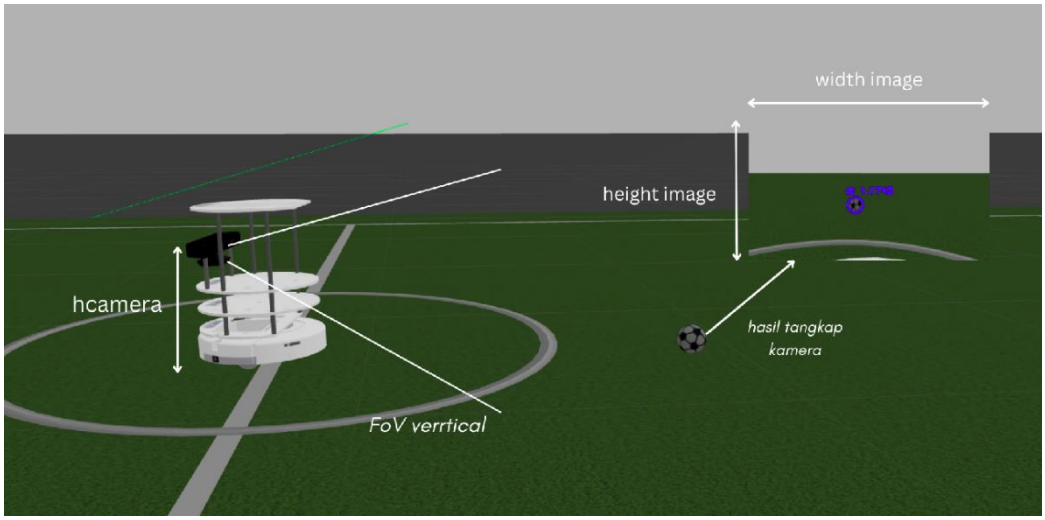
$$\mathrm{D} = \tan(\theta_y) \, . \, h_{camera} \qquad (3.5)$$

Dimana cara menemukan θ_y, yang merupakan sudut elevasi vertikal kamera, kita dapat menggunakan persamaan 3.6.

$$\theta_y = \frac{(y_{center} - \frac{h_{image}}{2})}{\frac{h_{image}}{2}} \, . \, \frac{FoV_{verikal}}{2} \qquad (3.6)$$

FoV merupakan field of view mengacu pada area maksimum yang dapat ditangkap oleh kamera dalam gambar. Biasanya diukur dalam derajat dan dapat bervariasi tergantung pada lensa yang digunakan pada kamera. Pada kinetic_camera yang digunakan pada penelitian ini, FoV vertikal bernilai 60° dan Fov horizontal bernilai 70°, nilai ini didapatkan dari deskripsi di file SDF pada model kinetic_camera. Dalam mengukur jarak menggunakan lingkungan simulasi terkadang kita memerlukan konversi unit, dari satu unit ke unit lain. Pada gazebo semua sumbu kutub, baik axis-x maupun axis-y, ditentukan dalam meter, maka jika robot bergerak dari titik x: 1.0 ke x: 2.0, robot telah berjalan sebesar 1 meter (Gazebo, 2024). Dalam percobaan ini akan dilakukan beberapa skenario uji coba menggunakan objek bola untuk menilai dari seberapa akurat aproksimasi perhitungan jarak dengan jarak asli pada peta dengan tujuh buah kategori, yaitu sangat dekat, dekat, cukup dekat, medium, cukup jauh, jauh, sangat jauh. Gambaran dari skema pengukuran dari robot ke objek dapat dilihat pada Gambar 3.11.

`



Gambar 3.14 Keterangan dalam mengukur jarak robot ke objek

`

BAB IV
HASIL DAN PEMBAHASAN

Dalam penelitian ini, data yang digunakan merupakan data sekunder yang berasal dari beberapa website, seperti kaggle dan roboflow. Total data yang diperoleh yaitu sebanyak 35.895 data bola dan 268 data gawang. Gambar yang akan digunakan untuk proses training pada tahap modeling berjumlah 1052 data atau sekitar ~87% dari total data yang akan digunakan dan sebanyak 152 data akan digunakan untuk evaluasi atau sebanyak ~13% data dari total data yang akan digunakan. Contoh dari gambar yang digunakan dapat dilihat pada Gambar 4.1.

Gambar 4.1 Gambar yang memuat objek bola dan gawang

Selanjutnya dilakukan preprocessing data untuk data yang sudah diperoleh dan dipisahkan menjadi data train dan data validation. Pada tahapan ini data dilakukan labeling, yaitu penandaan objek yang ada pada gambar. Proses melakukan labeling ini menggunakan aplikasi open source labelImg. Dalam melakukan labeling data, format anotasi yang digunakan adalah PascalVOC. PascalVOC merupakan sebuah standar anotasi dalam bidang computer vision, terkhususnya bidang object detection. Standar anotasi ini menghasilkan file xml dimana file tersebut terkandung berbagai informasi objek yang sudah dilabelkan pada gambar, seperti nama file, path file, posisi bounding box (xmin, ymin, xmax, ymax), nama kelas objek yang berkaitan dengan gambar tersebut serta beberapa informasi lainnya [10]. PascalVOC

`

merupakan format anotasi 1:1, dimana jumlah file xml yang ada akan merepresentasikan satu gambar yang ada. Selain itu perlu juga membuat suatu file berformat .pbtxt, yang merupakan suatu file untuk menyimpan metadata model mengenai nama kelas dari objek yang terdapat pada dataset yang ada. Contoh gambar yang sudah dilabelkan dapat dilihat pada Gambar 4.2, dimana terdapat bounding box berwarna hijau di sekitar objek yang ditandakan.

Gambar 4.2 Gambar yang sudah dilabelkan

Setelah itu, lakukan konversi data yang sudah dilabeli menjadi format tfrecord. Konversi ini bertujuan agar data yang telah diproses sebelumnya dapat digunakan oleh model TensorFlow Object Detection selama pelatihan, command dalam konversi ini dapat dilihat pada Gambar 4.3.

```
python generate_tfrecord.py -x #path_to_train_data -l #path_to_pbtxt_file -
o #path_for_output
```

Gambar 4.3 Perintah untuk melakukan konversi data ke tfrecord

4.1 Pelatihan dan Evaluasi Model

Pada fase pelatihan, data tfrecord yang telah dihasilkan sebelumnya akan digunakan oleh model selama n jumlah iterasi yang telah ditentukan. Seperti yang telah disebutkan sebelumnya, model Single Shot Detector (SSD) MobileNet V2 akan digunakan dalam penelitian ini untuk mendeteksi objek. Pelatihan model dilakukan pada laptop dengan CPU

`

Intel I7, RAM 16GB, dan GPU Nvidia GTX 1650 TI. Pelatihan akan dilakukan dengan iterasi 10000 steps dan memiliki parameter pada file config seperti pada Gambar 4.4.

```
model {
  ssd {
    num_classes: 6
    image_resizer {
  fixed_shape_resizer {
        height: 320
        width: 320
    }
  }

### parameter lain

train_config {
  batch_size: 8
  data_augmentation_options {
    random_horizontal_flip {
    }
  }
  data_augmentation_options {
    ssd_random_crop {
    }
  }
  sync_replicas: true
  optimizer {
    momentum_optimizer {
      learning_rate {
        cosine_decay_learning_rate {
          learning_rate_base: 0.001
          total_steps: 10000
          warmup_learning_rate: 0.001
          warmup_steps: 2000
        }
      }
      momentum_optimizer_value: 0.9
    }
    use_moving_average: false
  }
fine_tune_checkpoint: "PATH_TO_CHECKPOINT"
  num_steps: 10000
  startup_delay_steps: 0.0
  replicas_to_aggregate: 8
  max_number_of_boxes: 20
  unpad_groundtruth_tensors: false
  fine_tune_checkpoint_type: "detection"
  fine_tune_checkpoint_version: V2
}

### parameter lain
}
```

Gambar 4.4 Konfigurasi model SSD MobileNet v2

`

Teks bercetak tebal menunjukkan parameter yang telah disesuaikan dengan kebutuhan. Untuk num_classes, yang merupakan jumlah kelas dalam data latih, diubah menjadi enam karena penelitian ini berfokus pada deteksi enam objek yang biasa pada lingkungan robot sepak bola. Parameter batch_size diubah menjadi 8 data per langkah karena keterbatasan perangkat keras yang digunakan. Sementara itu, parameter learning_rate, baik base maupun warmup, diubah menjadi 0.001 agar model dapat dilatih dengan baik dan menghindari pelatihan yang terlalu cepat atau terlalu lambat dalam mempelajari data. Dua parameter terakhir yang diubah digunakan untuk mengkonfigurasi ulang jumlah iterasi yang dilakukan serta jumlah total bounding box yang dapat dihasilkan dalam satu data. Setelah melakukan pengubahan konfigurasi, maka akan dilanjutkan dengan pelatihan model menggunakan perintah seperti pada Gambar 4.5.

```
python model_main_tf2.py --model_dir=”PATH_TO_MODEL” --
pipeline_config_path=”PATH_TO_MODEL_CONFIG”
```

Gambar 4.5 Perintah untuk mmemulai melatih model

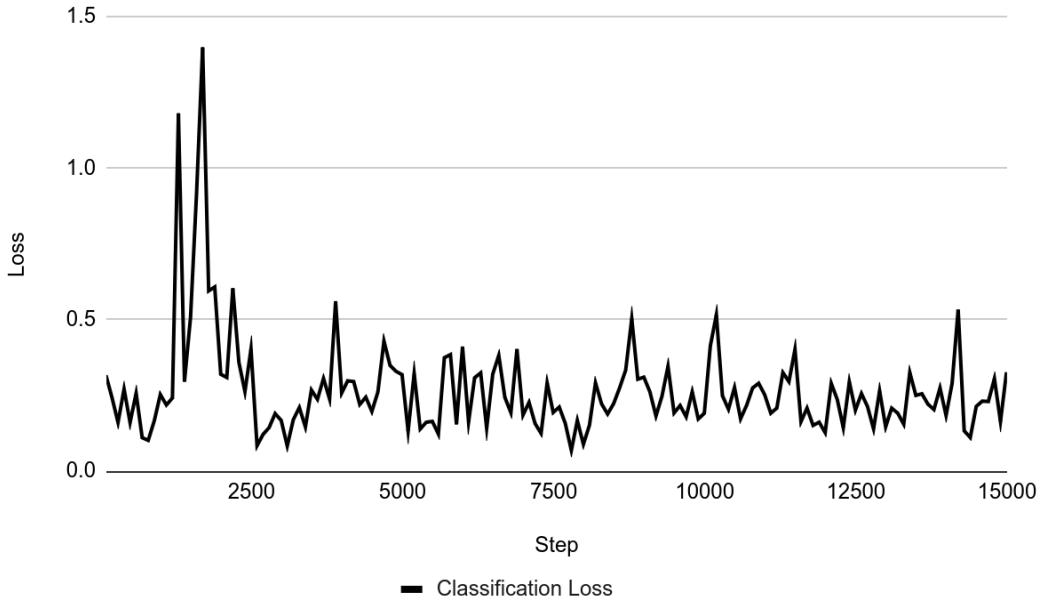
Tahap ini akan dilakukan evaluasi model yang telah dilatih yang akan dilakukan bersamaan dengan pelatihan model pada setiap 1000 stepnya. Oleh karena itu, perintah untuk melakukan evaluasi juga dieksekusi bersamaan dengan perintah pelatihan model. Perintah ini dapat dilihat pada Gambar 4.6.

```
python model_main_tf2.py python model_main_tf2.py --
model_dir=”PATH_TO_MODEL” --pipeline_config_path=”PATH_TO_MODEL_CONFIG” --
checkpoint_dir “PATH_FOR_CHECKPOINT_DIR”
```

Gambar 4.6 Perintah untuk melakukan evaluasi model

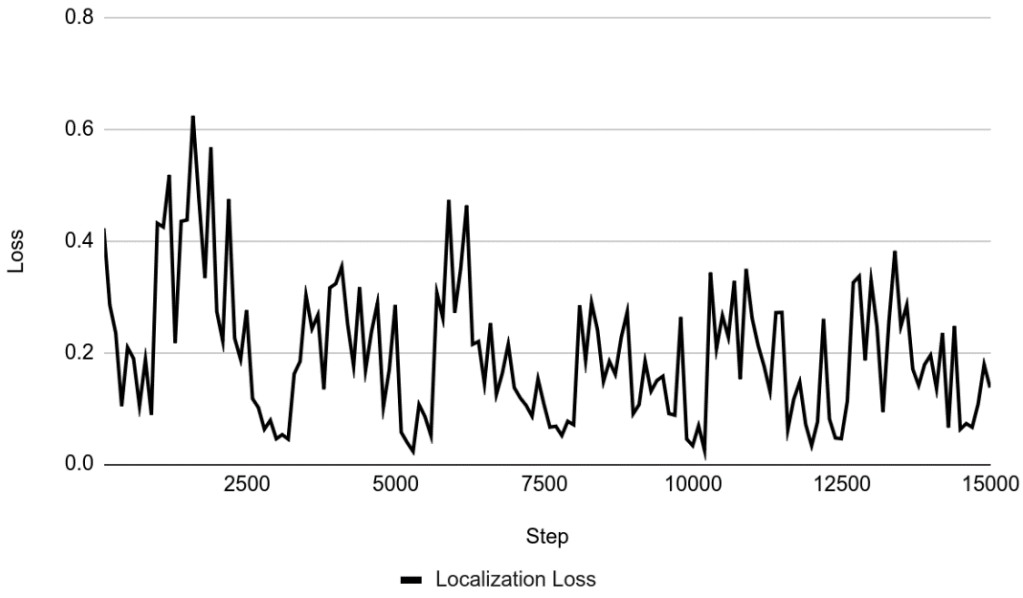
Proses pelatihan berlangsung selama 2 jam 45 menit, dengan rata-rata waktu pelatihan sekitar 23.5 menit untuk setiap 2000 langkah. Berdasarkan hasil pelatihan model menggunakan 39.418 data, diperoleh beberapa nilai loss, yaitu 0.038 atau 3.8% untuk classification loss, 0.091 atau 9.1% untuk localization loss, dan total loss sebesar 0.215 atau 21.5%. Classification loss mengukur seberapa baik model dapat mengklasifikasikan objek yang dideteksi ke dalam kelas yang benar, yaitu bola atau gawang. Dengan nilai yang relatif kecil ini, model menunjukkan kemampuan yang cukup baik dalam mengklasifikasikan objek yang ditemukan. Grafik untuk data classification loss ini dapat dilihat pada Gambar 4.7.

`



Gambar 4.7 Grafik classification loss selama pelatihan

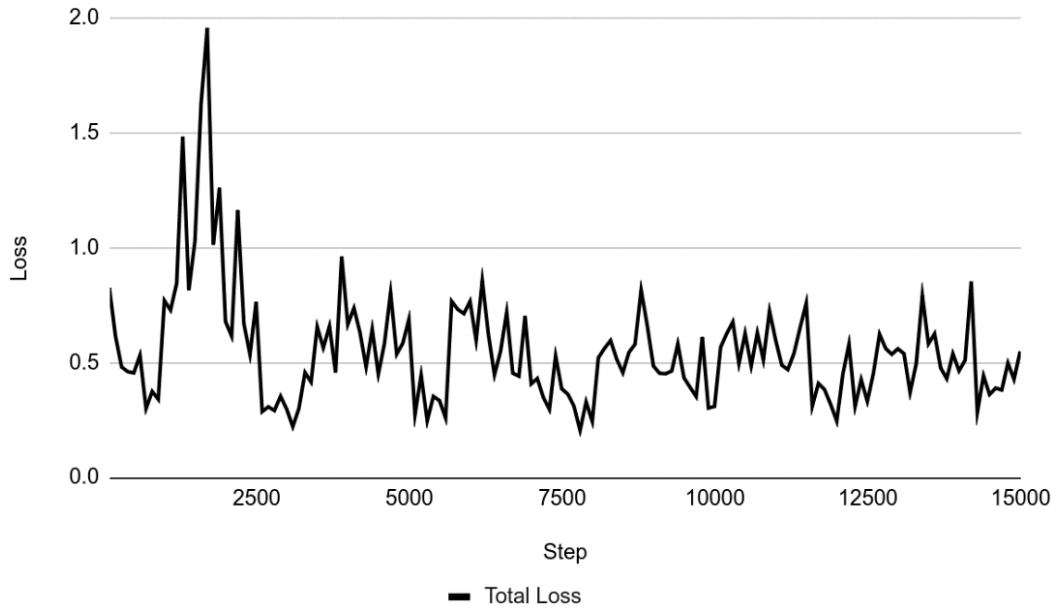
Localization loss mengukur akurasi model dalam menentukan lokasi objek dalam gambar. Meskipun nilai ini menunjukkan bahwa ada ruang untuk perbaikan dalam menentukan posisi objek, masih dalam batas yang wajar dalam hasil yang didapatkan. Grafik untuk nilai localization loss ini dapat dilihat pada Gambar 4.8



Gambar 4.8 Grafik localization loss selama pelatihan

`

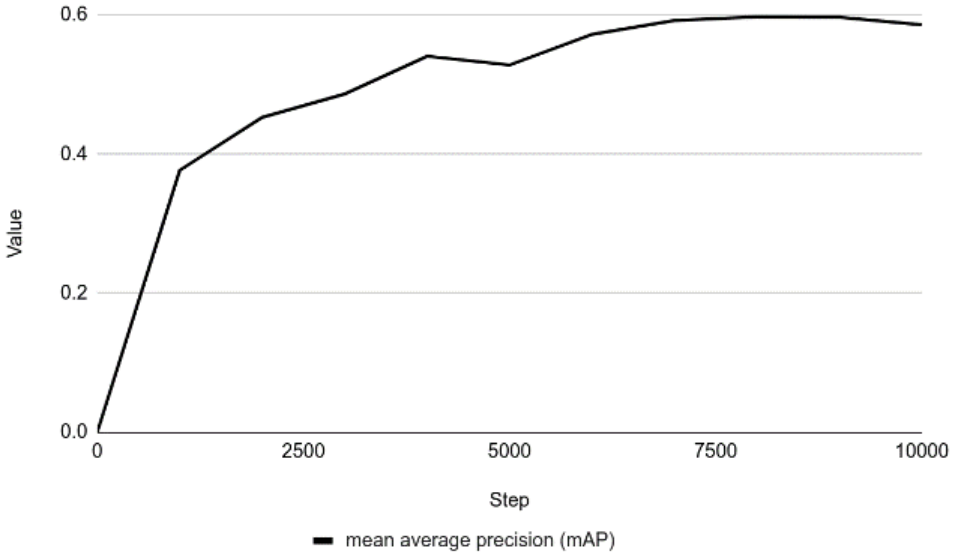
Total loss, yang merupakan gabungan dari classification loss dan localization loss, menunjukkan kinerja keseluruhan model dalam mendeteksi dan mengklasifikasikan objek. Dengan total loss sebesar 21.5%, model memiliki kinerja yang cukup baik, meskipun masih ada beberapa kesalahan yang perlu diperbaiki. Grafik total loss yang disajikan dalam Gambar 4.9 memberikan visualisasi tentang bagaimana nilai-nilai loss ini berubah selama proses pelatihan, membantu untuk lebih memahami kinerja model seiring waktu. Secara keseluruhan model cukup baik dalam melakukan deteksi objek secara keseluruhan meskipun nilai dari total loss ini terlihat cukup tinggi jika dibandingkan dengan individual loss yang ada.



Gambar 4.9 Grafik total loss selamat pelatihan

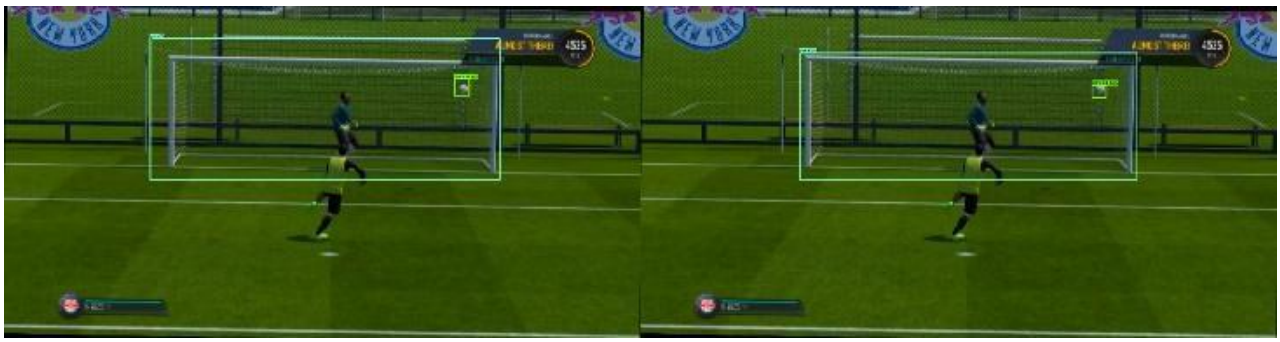
Secara keseluruhan model cukup baik dalam melakukan deteksi objek secara keseluruhan meskipun nilai dari total loss ini terlihat cukup tinggi jika dibandingkan dengan individual loss yang ada.

`



Gambar 4.10 Grafik mAP pada evaluasi yang dilakukan

Berdasarkan hasil evaluasi, dapat disimpulkan bahwa model yang dikembangkan menunjukkan kinerja yang cukup baik dalam mendeteksi dan mengklasifikasikan objek, dengan nilai mean average precision (mAP) sebesar 58.6%. Meskipun demikian, masih ada ruang untuk peningkatan lebih lanjut guna meningkatkan akurasi model. Selain itu, nilai recall sebesar 66% mengindikasikan bahwa model sudah cukup baik dalam mendeteksi objek, namun masih ada beberapa objek yang terlewatkan dalam proses deteksi. Oleh karena itu, pengembangan lebih lanjut diperlukan untuk meningkatkan kemampuan model dalam mengenali objek secara lebih akurat dan komprehensif. Secara keseluruhan, hasil ini memberikan dasar yang kuat untuk optimisasi lebih lanjut dan menunjukkan potensi yang baik untuk aplikasi di berbagai bidang. Salah satu hasil evaluasi dapat dilihat pada Gambar 4.11.



Gambar 4.11 Hasil prediksi model (kiri), data yang dilakukan label (kanan),

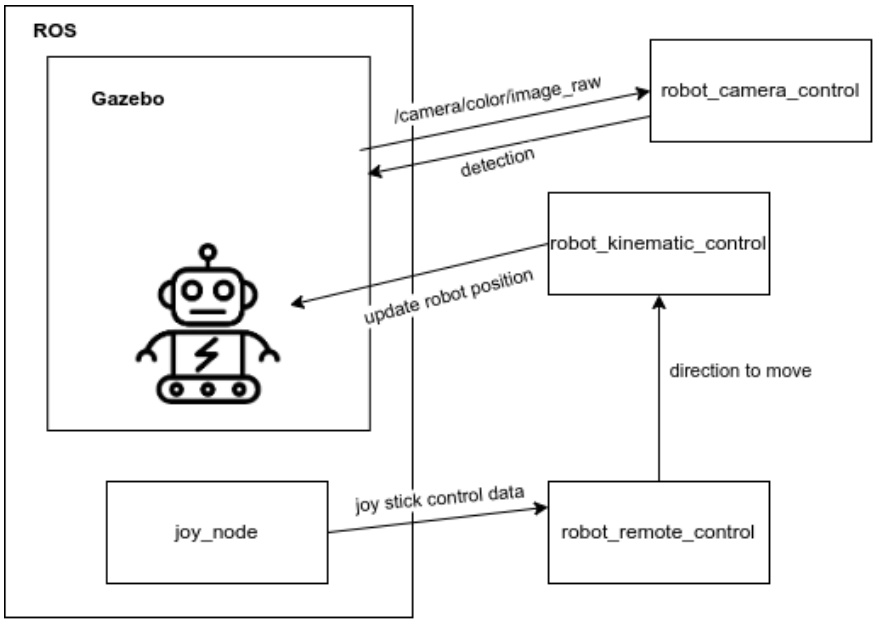
`

4.2 Pengujian di Simulasi Gazebo

Pengujian pada tahap ini akan dilakukan dengan menjalankan model deteksi objek sudah di train dan evaluasi pada tahap sebelumnya pada robot turtlebot di lingkungan simulasi gazebo ROS. Pengujian ini dilakukan menggunakan ROS dan sebuah komputer bersistem operasi linux Ubuntu 20.04. Pada penelitian ini, pengujian akan dilakukan dengan menggunakan beberapa package yang dibuat menggunakan ROS untuk melakukan beberapa kegiatan, package itu antara lain:

1. **robot_camera_control**: package ini digunakan untuk membuat model deteksi objek, mengambil data yang berasal dari robot, melakukan pendeteksian, dan menghitung jarak antara objek dan robot.
2. **robot_kinematic_control**: package ini digunakan dalam memanipulasi gerak dan posisi robot di lingkungan simulasi.
3. **robot_remote_control**: package ini digunakan sebagai memanipulasi data dari perangkat *joystick* dan mengirimkannya ke package robot_kinematic_control untuk menggerakan robot.

Adapun penggunaan package joy_node yang akan digunakan untuk memfasilitasi agar ROS dapat membaca pergerakkan yang berasal dari joystick. Ilustrasi mengenai proses aktivitas pengiriman data antar package ini dapat dilihat pada Gambar 4.12.

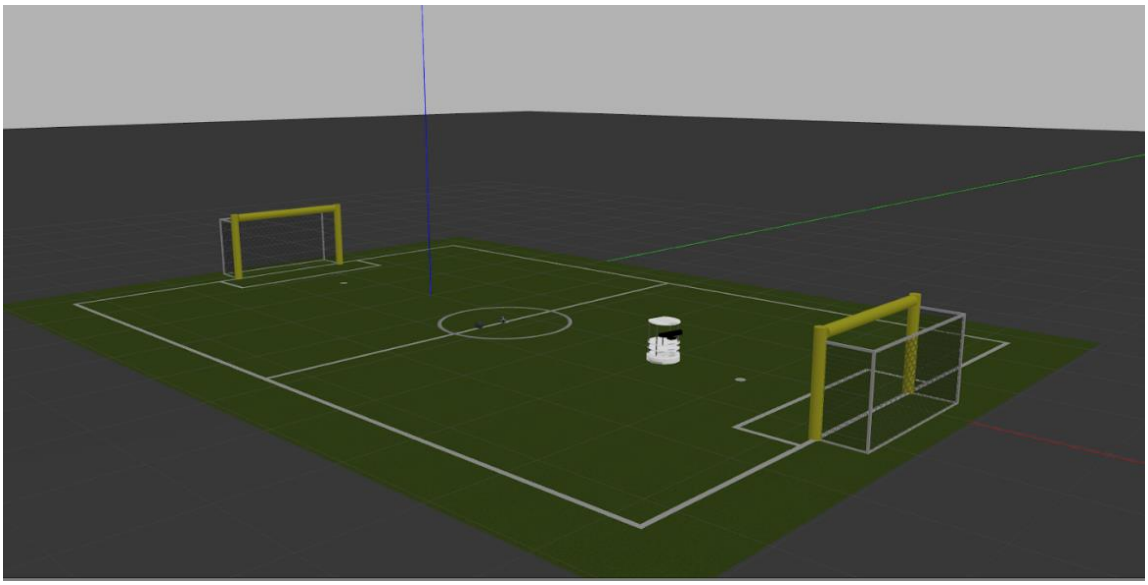


Gambar 4.12 Ilustrasi aliran data antar package

`

Ilustrasi di atas menggambarkan bagaimana proses data tertransmisi dari satu package ke package lain. Hal ini dapat dilakukan dengan menggunakan subscriber dan publisher yang tersedia oleh ROS. Robot pada gazebo akan mendapat data yang berasal dari suatu package dan perubahan ats state robot ini dapat dilakukan dengan melakukan publish data menuju topic /gazebo/set_model_state.

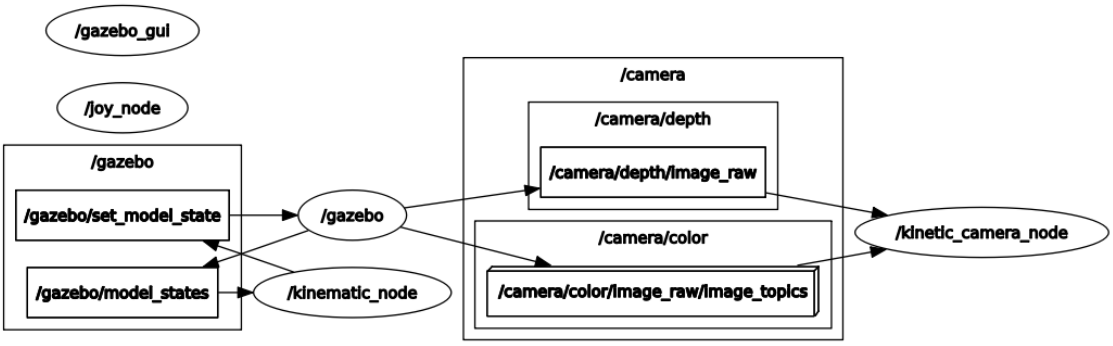
Dalam penelitian ini lingkungan pengujian akan mengambil lapangan pertandingan robot sepak bola dan terdapat beberapa model pendukung seperti bola dan gawang sesuai dengan ketentuan Robocup untuk middle size robot seperti pada Gambar 4.13.



Gambar 4.13 lingkungan simulasi pada simulasi gazebo

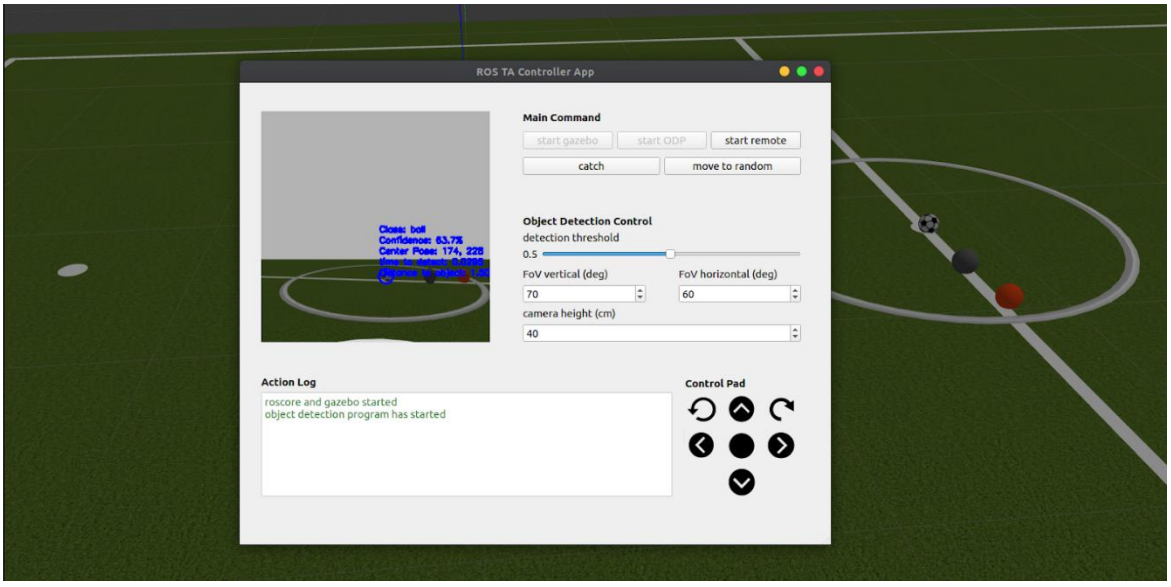
Setelah menampilkan lingkungan simulasi pada gazebo ROS, maka tahap berikutnya akan dijalankan package robot_camera_control yang akan digunakan untuk memuat model SSD MobileNet dan melakukan perhitungan atas jarak antara objek dengan objek. Setelah menjalankan package robot_camera_control, maka dijalankan juga beberapa package support yang akan digunakan untuk mengartur kinematika robot dalam melakukan navigasi di sekitar lingkungannya, hal ini dilakukan dengan menjalankan package robot_kinematic_control, robot_remote_control, dan joy_node. Setelah menjalankan semua package yang dibutuhkan telah berjalan, akan dilakukan pengecekan node yang bekerja pada pengujian pada ROS dan gazebo menggunakan library rqt seperti pada Gambar 4.14.

`



Gambar 4.14 Ilustrasi pada rqt atas node yang berjalan pada ROS

Setelah memastikan bahwa semua node yang dibutuhkan robot telah berjalan dengan aman, maka akan dilanjutkan dengan tahap selanjutnya yaitu melakukan pengujian pendeteksian dan melakukan perhitungan jarak menggunakan model SSD MobileNet v2. Dalam memudahkan melakukan pengujian tersebut, maka akan digunakan sebuah aplikasi graphical user interface (GUI) dalam membantu menangkap gambar, menjalankan beberapa program, melakukan kalibrasi parameter serta melakukan navigasi pada penelitian ini. Tampilan dari aplikasi ini dapat dilihat pada Gambar 4.15.



Gambar 4.15 Tampilan aplikasi GUI controller

`

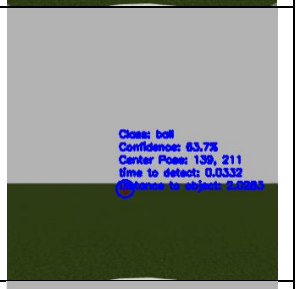
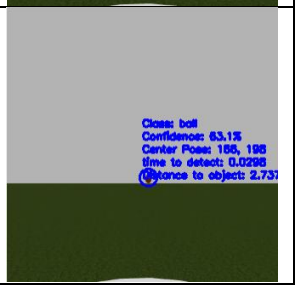
4.3 Hasil Pendeteksian dan Pengukuran

Setelah robot berhasil dijalankan pada lingkungan simulasi dan telah melakukan pengujian, maka hasil pun telah didapatkan. Tujuan dari hasil pengujian ini adalah melakukan perbandingan hasil antara penggunaan metode segmentasi warna dan metode deteksi objek dalam melakukan kegiatan yang dilakukan robot sepak bola. Oleh karena itu, dilakukan dua skenario, deteksi tunggal dan deteksi ganda, yang dapat menunjukkan kemampuan dan batasan dari metode ini dibanding metode sebelumnnya.

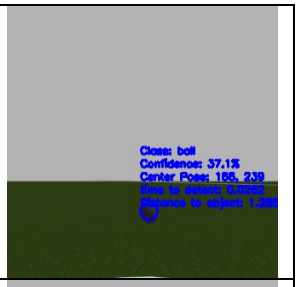
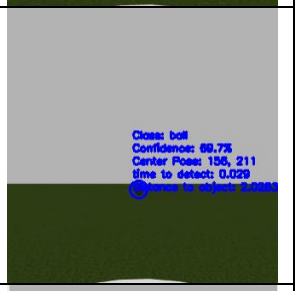
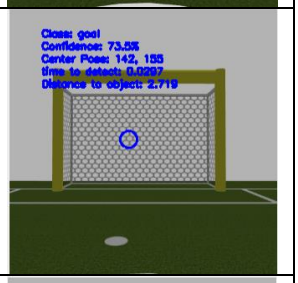
Skenario 1. Model akan mendeteksi objek tunggal dengan jarak yang berbeda.

Tabel 4.1 Hasil pengujian skenario 1

`

Deskripsi	Jarak Estimasi (m)	Jarak Asli (m)	Gambar
bola hitam putih tampak depan	1.15	1,17	
bola hitam putih tampak depan	2.25	2,29	
bola hitam putih tampak depan	2.66	2.92	
bola oren tampak depan	1.2	1,16	
bola oren tampak depan	2.02	1,77	
bola oren tampak depan	2.73	2.4	

`

bola abu tampak depan	1.26	1.21	
bola abu tampak depan	2.02	2.1	
gawang tampak depan	2.7	2.16	
gawang tampak depan	3	2.52	

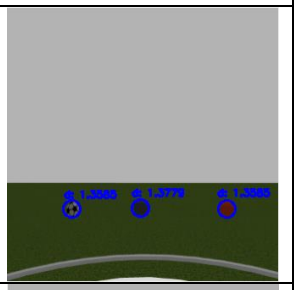
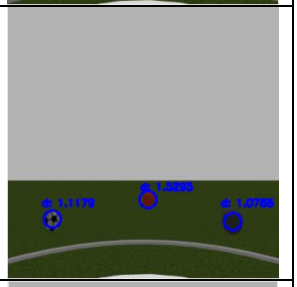
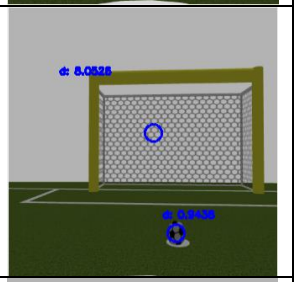
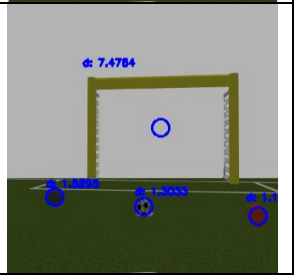
Berdasarkan hasil pada Tabel 4.1 dapat disimpulkan bahwa akurasi estimasi jarak bervariasi untuk setiap objek. Pada bola hitam putih tampak depan, akurasi baik pada jarak dekat, ditunjukkan pada jarak 1.15 m vs. 1.17 m dan 2.25 m vs. 2.29 m, tetapi menurun pada jarak jauh, ditunjukkan 2.66 m vs. 2.92 m. Untuk bola oren, akurasi juga menurun dengan jarak (1.2 m vs. 1.16 m, 2.02 m vs. 1.77 m, dan 2.73 m vs. 2.4 m). Bola abu menunjukkan akurasi baik (1.26 m vs. 1.21 m, 2.02 m vs. 2.1 m). Gawang tampak depan memiliki kesalahan lebih besar pada jarak jauh (2.7 m vs. 2.16 m, 3 m vs. 2.52 m). Secara keseluruhan, akurasi lebih baik pada jarak dekat dengan variasi kesalahan tergantung objek. Selain itu akurasi pada pendeteksian yang dilakukan pada juga bervariatif antara 51.1% hingga 80.1%. Hal ini bergantung pada jarak objek yang terlihat pada kamera, menunjukkan bahwa semakin terlihat dan membentuk sebuah objek, maka nilai akurasi akan semakin tinggi. Berdasarkan data pengujuran jarak yang ada, maka dapat dihitung galat yang didapatkan dalam pengukuran

`

dengan menggunakan *mean absolute error* (mae) dan didapatkan nilai sebesar 0.209 atau 20.9%

Skenario 2. Model akan mendeteksi objek ganda pada posisi yang berbeda.

Tabel 4.2 Hasil pengujian skenario 2

Deskripsi	Jarak Estimasi (m)	Jarak Asli (m)	Gambar
bola beberapa warna berjejer	bola_putih: 1.35 bola_oren:1.37 bole_silver: 1.35	1.37 1.36 1.43	
bola beberapa warna tersebar	bola_putih: 1.11 bola_oren: 1.52 bola_silver: 1.07	1.17 1,46 1.09	
bola di depan gawang	bola_putih: 0.94 gawang: 3.05	0.91 2.56	
beberapa bola di depan gawang	bola_putih: 1.52 bola_oren: 1.3 bola_silver: 1.13 gawang: 2.47	1.57 1.23 1.21 2.51	

Sesuai dengan hasil pada tabel, pada pengujian jarak estimasi dan jarak asli untuk beberapa skenario, terlihat variasi akurasi yang berbeda. Pada bola beberapa warna berjejer, estimasi

`

jarak cukup akurat dengan nilai mendekati jarak asli (1.35 m, 1.37 m, 1.35 m vs. 1.37 m, 1.36 m, 1.43 m). Untuk bola beberapa warna tersebar, akurasi estimasi lebih bervariasi dengan beberapa kesalahan kecil (1.11 m, 1.52 m, 1.07 m vs. 1.17 m, 1.46 m, 1.09 m). Pada bola di depan gawang, estimasi cukup akurat dengan kesalahan kecil pada jarak lebih pendek namun meningkat pada jarak lebih jauh (0.94 m, 3.05 m vs. 0.91 m, 2.56 m). Sementara itu, beberapa bola di depan gawang menunjukkan estimasi yang cukup baik dengan sedikit perbedaan (1.52 m, 1.3 m, 1.13 m, 2.47 m vs. 1.57 m, 1.23 m, 1.21 m, 2.51 m). Secara keseluruhan, estimasi jarak menunjukkan akurasi yang bervariasi tergantung pada distribusi dan posisi bola, dengan kecenderungan kesalahan lebih kecil pada jarak lebih pendek. Dalam skenario beberapa objek didapatkan galat pada pengukuran sebesar 0.084% atau sebesar 8.4%, Nilai nilai relatif kecil dibandingan dengan skenario sebelumnya.

`

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil yang didapatkan pada bab sebelumnya, penelitian ini berhasil mengembangkan dan menguji model pendeteksi objek SSD MobileNet v2 pada robot dalam lingkungan simulasi ROS. Dalam pengujian yang dilakukan, robot dapat mendeteksi berbagai objek dalam lingkungan simulasi berdasarkan beberapa fitur objek yang ada, serta berhasil menyelesaikan tugas yang diberikan seperti permainan sepak bola. Kemampuan untuk mengetahui jarak antara robot dan objek juga terbukti cukup baik. Dengan menggunakan pendekatan ini, robot dapat mengidentifikasi objek di sekitarnya tanpa terpengaruh oleh masalah eksternal yang sering dialami pada metode segmentasi warna, seperti pengaruh cahaya dan variasi warna objek.

Akurasi yang dihasilkan model pendeteksi objek juga cukup bagus, yaitu sebesar 50–81% dengan waktu yang relatif kecil yaitu 0.001 detik. Variasi interval pada model ini dipengaruhi jarak antar objek dan robot, sehingga semakin dekat robot maka akan semakin tinggi juga akurasi deteksi yang dilakukan dan sebaliknya. Pengukuran yang telah dilakukan juga sudah berhasil mengestimasikan jarak antara objek dan robot dengan galat antara 8–20%. Walaupun begitu, berdasarkan pengujian yang sudah dilakukan, terdapat kelemahan yang harus dipertimbangkan sebelum menerapkan model ini pada robot asli, yaitu kemampuan deteksi objek tidak akan bekerja pada jarak lebih dari 3 meter. Hal ini tentunya dapat diperbaiki dengan melakukan penambahan data objek-objek di gambar yang terlihat kecil.

5.2 Saran

Adapun saran untuk penelitian lain di masa mendatang terkait topik yang adalah berdasarkan penelitian yang sudah peneliti lakukan sebagai berikut:

1. Melakukan penambahan data dan evaluasi yang dekat dengan data uji coba dan membagi data tersebut menjadi tiga buah kategori untuk satu objeknya, yaitu: objek dekat, objek medium, dan objek jauh. Dengan hal ini maka kemungkinan besar model pnedeteksi objek akan dapat menangkap objek lebih akurat di semua ukuran objek tersebut.

`

2. Menerapakan model *single-shot* yang menerapkan Feature Pyramid Network (FPN) atau *Region Proposal Netowork* (RPN) untuk meningkatkan akurasi untuk tiap kelasnya.
3. Menggunakan sensor tambahan untuk model robot dalam pengujian gazebo sehingga pengujian dapat lebih dekat dengan pengunaan pada robot bola nantinya.

`

DAFTAR PUSTAKA

Abbyasov, B., Lavrenov, R., Zakiev, A., Yakovlev, K., Svinin, M., & Magid, E. (2020). Automatic tool for Gazebo world construction: from a grayscale image to a 3D solid model. *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 7226-7232.

Alam, E., Sufian, A., Das, A. K., Bhattacharya, A., Ali, M. F., & Rahman, M. H. (2022). Leveraging Deep Learning for Computer Vision: A Review. *TechRxiv*.

Asian Robotics Review LLC. (2019). *Automatian & The Future of Southeast Asia.* Asian Robotics Review.

Auliya, A., Dewi, T., Noer, M. N., & Oktarina, Y. (2022). Implementasi Pengolahan Citra Menggunakan Metode YOLO pada Security Robot dibidang Pertanian. *Journal of Applied Smart Electrical Network and Systems (JASENS)*, 44-46.

Darmawan, A. (2018). Sistem Pembacaan Bola MenggunakanOmnidirectionalCamera untuk Pergerakan Robot SepakBolaBeroda. *dspace*, 6.

Dwiyanto, D., & Fath, N. (2020). Analisa Sistem Komunikasi Data Pada Simulasi Robot Sepak Bola Beroda. *Jurnal Maestro*, 421-428.

Gazebo. (2024, 11 11). *Gazebo Sim*. Retrieved from gazebosim.org: https://gazebosim.org/api/sim/8/spherical_coordinates.html

Ikram, F. D., Khumaidi, A., Rahmat, M. B., Endrasmono, J., Syai'in, M., Pristovani, D., & Riananda. (2024). Deteksi Objek di Lapangan pada Robot Sepakbola Beroda Menggunakan Metode YOLOV5. *Jurnal Elkolind*, 609-611.

Lawrence, C., & Dollar, P. (2014). Edge Boxes: Locating Object Proposals from Edges. *Sprigerlink*, 391-405.

Matsuzaka, Y., & Yashiro, R. (2023). AI-Based Computer Vision Techniques and Expert Systems. *MDPI*, 289-302.

Muldayani, W., Sumardi, N.I, A. M., Firza, M., & Firmansyah, T. (2023). MPLEMENTASI SISTEM OBJECT TRACKING UNTUK MENDETEKSI. *Junal SIMETRIS*, 3-13.

P. Koenig, N., & Howard, A. (2004). Design and Use Paradigms for Gazebo, An. *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566*, 2149-2154 vol.3.

Palwankar, T., & Kothari, K. (2022). Real Time Object Detection using SSD and MobileNet. *International Journal for Research in Applied Science and Engineering Technology*.

`

Pandey, A. (2023). Computer Vision. *International Journal for Research in Applied Science & Engineering Technology (IJRASET)*.

Puspresnas. (2024). *Kontes Robot Indonesia*. Retrieved from https://kontesrobotindonesia.id

Sanubari, F. F., & Puriyanto, R. D. (2022). idDeteksi Bola dan Gawang dengan Metode YOLO Menggunakan Kamera Omnidirectional pada Robot KRSBI-B. *Buletin Ilmiah Sarjana Teknik Elektro*, 81-83.

SDFormat. (n.d.). Retrieved July 3, 2024, from http://sdformat.org/

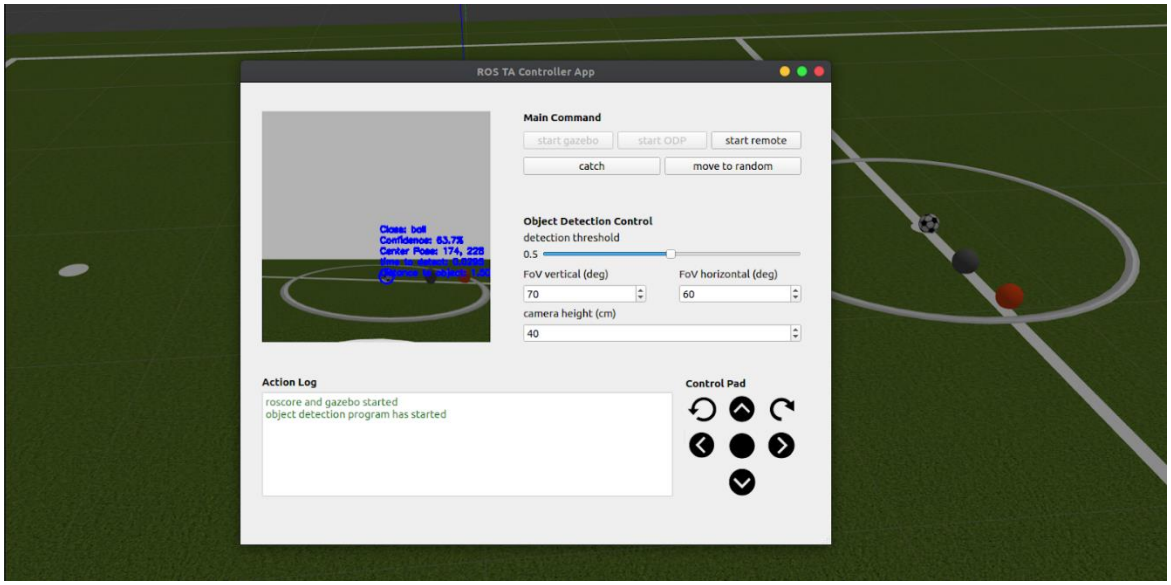
Sharma, V., & Mir, R. N. (2020). A comprehensive and systematic look up into deep learning based. *Comput. Sci. Rev*, 100301.

Ujilings, J., van de Sande, K., Gevers, T., & Smeulders, A. (2013). Selective Search for Object Recognition. *Sprigerlink, 104*, 154-171.

Zhang, H., & Hong, X. (2019). *Recent progresses on object detection: a brief review.* springer link.

`

LAMPIRAN



github url: https://github.com/ilham-2001/ros-ta

`