Mobile Robot Obstacle Detection and Avoidance with NAV-YOLO

Ndidiamaka Adiuku[1,]*, Nicolas P. Avdelidis [1], Gilbert Tang [2], Angelos Plastropoulos [1], and Yanis Diallo [2]

[1] Integrated Vehicle Health Management Centre (IVHM), School of Aerospace, Transport and Manufacturing, Cranfield University, Cranfield, UK

[2] Center for Robotics and Assembly, School of Aerospace, Transport and Manufacturing (SATM), Cranfield University, Cranfield, UK

Email: n.p.adiuku@cranfield.ac.uk (N.A.); np.avdel@cranfield.ac.uk (N.P.A.); g.tang@cranfield.ac.uk (G.T.); a.plastropoulos@cranfield.ac.uk (A.P.); yanis.diallo.254@cranfield.ac.uk (Y.D.)

*Corresponding author

Abstract—**Intelligent robotics is gaining significance in Maintenance, Repair, and Overhaul (MRO) hangar operations, where mobile robots navigate complex and dynamic environments for Aircraft visual inspection. Aircraft hangars are usually busy and changing, with objects of varying shapes and sizes presenting harsh obstacles and conditions that can lead to potential collisions and safety hazards. This makes Obstacle detection and avoidance critical for safe and efficient robot navigation tasks. Conventional methods have been applied with computational issues, while learning-based approaches are limited in detection accuracy. This paper proposes a vision-based navigation model that integrates a pre-trained Yolov5 object detection model into a Robot Operating System (ROS) navigation stack to optimise obstacle detection and avoidance in a complex environment. The experiment is validated and evaluated in ROS-Gazebo simulation and turtlebot3 waffle-pi robot platform. The results showed that the robot can increasingly detect and avoid obstacles without colliding while navigating through different checkpoints to the target location.**

Keywords—**autonomous navigation, object detection, obstacle avoidance, mobile robot, deep learning**

I. Introduction

The application of mobile robots for visual inspection has been adopted in Maintenance, Repair, and Overhaul (MRO) hangars, where a significant level of autonomy and automation is required to safely navigate environments with irregular object structures and varying complexities. This implies that the mobile agents must autonomously observe, detect, avoid, and make control decisions from multisensory information to make collision-free navigation in unstructured and dynamic scenes. One of the critical navigation challenges is detecting and avoiding obstacles in complex and changing environments, such as busy hangars, where obstacles of varying shapes and sizes can lead to potential collisions and safety hazards. Efficient mobile robot navigation requires robust perception capability [1], and most recent approaches use vision-based sensors like RGB cameras and LiDAR to gain an understanding of environmental features and improve object detection, avoidance, and navigation techniques [2]. Laser devices are more costly, and conventional algorithms that are based on this information source are usually complex. On the other hand, RGB cameras provide depth and feature-rich environmental information and are widely used for developing robot navigation models. However, extracting accurate obstacle information from these input devices and modelling it for a reliable and safe navigation experience is still widely studied.

Over the years, conventional vision-based navigation systems based on the Robot Operating System (ROS) framework have been widely used. ROS is an open-source robotic platform that was first released in 2007 [3] to provide classical support tools, packages, and libraries ready-to-use [4, 5] for developing robotic applications. The ROS navigation stack is an aspect of ROS ecosystem with a toolset and functionalities that equip robots with the capability to interact with their environment, avoid obstacles, and plan paths with reasonable intelligence [6]. This system's accuracy, reliability, and environment suitability depend on factors like obstacle detection, collision avoidance and path planning algorithms. The default ROS (Robot Operating System) navigation stack uses A* (A-star) global and Dynamic Windows Approach (DWA) local planning algorithms [7] where the latter employs a local search strategy that evaluates potential trajectories based on robot dynamics, environmental constraints, and safety margins. DWA has been widely used in different domains and has shown good results in robot local path planning and obstacle avoidance cases. However, it's challenged with scenarios that require complex navigation behaviours, accurate real-world obstacle identification and handling of dynamic obstacles. Also, achieving good performance with DWA requires handcrafted parameter tuning to suit changing scenarios which limits its capability in real-world application. Recent

Manuscript received July 21, 2023; revised September 7, 2023; accepted October 10, 2023; published March 22, 2024.

doi: 10.18178/ijmerr.13.2.219-226

advancements have applied deep learning [8] techniques with Convolutional Neural Networks (CNN) to learn deeper feature representations from environmental data for object classification and detection [9] to enhance intelligent navigation. The YOLOv5 (You Look Only Once) model is among the most widely adopted systems for object detection [10] but it struggles with detection accuracy in complex and changing environments, especially where varying object types and lighting conditions are critical factors. Different hybrid models have been proposed but are still limited in performance. The primary focus of this research is to design and develop an advanced object detection and obstacle avoidance system that transforms YOLOv5 detection output into laser scan data suitable for ROS navigation. This will create a robust and accurate real-time system that can effectively detect obstacles of varying shapes and sizes in different lighting conditions to ensure maximum safety and efficiency in the face of uncertainties. In our recent work, we developed a fused CNN [10] that uses image fusion from RGB (red, green, and blue) and infrared data to improve object visibility, detection accuracy, and feature extraction using the YOLOv5 object detection model [10]. The fuse-YOLO solution is integrated into the ROS navigation stack in this part of the project for efficient object recognition and avoidance.

This study presents a vision-based navigation model that employs a pre-trained YOLOv5 model and ROS navigation stack to improve obstacle detection and avoidance for safe navigation in a complex environment. This solution makes use of an RGB-D camera as the primary perception modality, providing rich environmental data through real-time visual feedback to enhance the robot's situational awareness in busy and cluttered hangar environments. By integrating the YOLOv5 object detection algorithm, the robot can respond proactively and accurately to varying and irregular obstacles. The YOLOv5 object detection model uses deep learning techniques to identify and classify objects [11] in the robot's field of view, enhancing robot perception detection and avoidance capabilities [12] The object detection node is created in the ROS framework to publish detected objects' positions and sizes and is transformed into laser_scan data representation that interacts with the DWA planner in the ROS navigation stack. This provides a full navigation pipeline for the robot to dynamically improve path planning and obstacle avoidance using sensor information. The contribution is to enhance the obstacle detection and avoidance problems in the complex aircraft hangar environment using coordinate transforms from camera frames to map frames to improve the accuracy and reliability of the obstacle detection and avoidance mechanism. The complete research process is shown in Fig. 1. First, the OpenCV AI Kit with Depth (OAK-D) camera [13] is embedded in the turtlebot3 waffle_pi robot to gather environmental information through the detection node created with a pre-trained YOLOv5 model [10]. The output with bounding box and depth information representing object size, position, and distance estimation is converted to interface with the costmap parameter node in the ROS navigation stack to detect obstacles accurately and plan a smooth obstacle avoidance path for the mobile robot. The detection and avoidance capabilities were benchmarked with YOLOv5 baseline variants and the default ROS navigation stack. The performance is evaluated, analysed, and validated with a real robot in an office environment. The overall contribution includes:

- The integration of the OAK-D stereo camera in Turtlebot3 waffle_pi robot to obtain environmental information.
- Create a ROS node that generates object detection results using pre-trained YOLOv5 weight.
- Develop an improved navigation algorithm by transforming object detection results to laser scan format for the navigation stack's local planner to improve obstacle detection and avoidance.
- Evaluate and validate the solution using baseline YOLOv5 versions and DWB local planner.

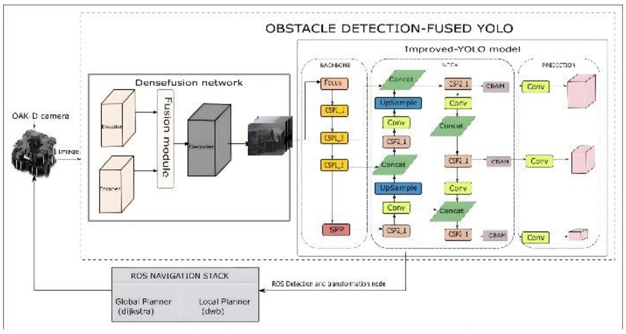

Fig. 1. Integrated YOLO and ROS navigation framework.

The remainder of this paper is organised as follows: Section II briefly reviews the related works for object detection and the ROS combination approach in mobile robot navigation. In Section III, the necessary background information on YOLOv5 and the ROS navigation stack was introduced, along with a detailed overview related to the proposed methodology. Section IV demonstrates the experimental and evaluation of the effectiveness of the proposed algorithm. This is followed by Section V, which dwells on the result and discusses the achieved result. Finally, Section VII presents the summary and future work of the paper.

II. Related Work

The requirement for obstacle detection and avoidance for mobile robots to safely navigate in complex scenarios is a crucial area of attention in robotics research. In real-world operations, accurately detecting static and dynamic obstacles is still a bottleneck. Conventional obstacle detection employs mathematical approaches, which limit detection efficiency and the extraction of detailed obstacle features. Recent deep learning-based obstacle avoidance developments have gained significance in research and application [14]. Through stacked convolutional layers, image feature representations are learned to realize the description of environmental-complicated objects [15]. For instance, Madawi *et al.* [2] proposed the use of Faster RCNN, which is a Region-Based Convolutional Neural Network (R-CNN) and multi-stage object detector model [2] improved to detect and avoid hazardous objects using a custom image dataset for an indoor mobile robot operational environment. Henke *et al.* [16] proposed a safe navigation system with YOLOv2 that recognizes objects and calculates their distances to the robot for a control decision. However, these algorithms did not take cognizance of the object's height and size, which are critical for quality detection and safety in a complex setting. To improve the detection capability, Hairol *et al.* [17] *proposed* an object detection and avoidance algorithm that can give control instructions to a mobile robot but is limited to objects with either square or rectangular shapes. This relates to traditional obstacle avoidance methods that are based on mapped environments and struggle with environmental changes and dynamic structures. To address this for real-world applications, hybrid solutions have been studied to detect and avoid irregular-shaped objects. Luo *et al.* [18] developed Contextual-YOLOV3 with improved environment perception capability to help identify even small objects and enhance obstacle avoidance techniques for an intelligent inspection task. Aghi *et al.* [19] used the ROS navigation system and CNN techniques to create a resilient navigation model that is power efficient, and Lee *et al.* [20] integrated a pre-trained deep convolutional neural network to detect and classify obstacle types for efficient obstacle detection and avoidance. In general, research on obstacle and avoidance for mobile robot navigation is still widely studied, and the introduction of object detection models to improve performance still requires further improvement. This paper proposes a new robust and accurate real-time object detection and avoidance method based on the YOLOv5 and ROS navigation stack that strikes a balance between accuracy and efficiency, enabling the robot to navigate smoothly and safely in dynamic environments.

III. Methodology

For a safe visual inspection with mobile robots, accurate perception, and an avoidance system is required to navigate autonomously in a complex environment. This paper proposes a combination of object detection and avoidance methods, as shown in Fig. 1, where a pre-trained and improved YOLOv5 model [10], trained with custom datasets, was integrated into the ROS framework for real-time obstacle recognition and avoidance towards improved navigation for aircraft visual inspection tasks. The study aims to compare the proposed method against the existing default navigation stack planners: the A* algorithm and the traditional Dynamic Windows Approach (DWA).

A. YOLOV5

YOLOv5 (You Only Look Once*)* is one of the releases of the YOLO family developed by Redmon *et al.* [21], which is a single-stage object detection model that can learn image features at different scales. As compared to other detection algorithms, YOLOv5 is widely used for its fast inference time and low computer resource demand, making it efficient for real-time applications [22]. It was built on the PyTorch framework and uses the CSPDarknet53 backbone structure to extract feature maps from the input image. The neck region architecture is designed with a path aggregation network to improve the detection of objects of different sizes, while the head structure consists of prediction layers that generate bounding box coordinates, class probability and object confidence score [23]. The model employs many techniques to improve detection accuracy, like the use of anchor boxes, post-processing techniques, optimizers, and mish activation functions. It performs significantly well in detecting different object structures and uses non-maximum suppression to suppress multiple detections of the same object. YOLOv5 contains small, medium, large, and extra-large types that are distinguished by their respective network structures. An improved YOLOv5s was applied in this context, which was developed by integrating the DENSEFUSE network [24] and the CBAM-based YOLOv5s technique [25] For a mobile robot that is navigating in a cluttered hangar environment with lightning variations, fused environment-specific thermal and RGB images and LLVIP datasets [26] were trained for efficient obstacle detection and avoidance capability. The package generates output in the form of a ROS topic named /yolov5/yolov5 which disseminates crucial information like bounding box coordinates and object detection probability value from the processed camera feed.

B. Ros Navigation Stack

The ROS navigation package is a configurable framework that houses modules that manage autonomous navigation system capabilities for mobile robots. This includes the move-base sub-package that provides functionalities for action servers, path planning, obstacle avoidance and motion control modules [27]. Different path-planning algorithms can be implemented, and navigation parameters can be configured based on the application scene. The ROS framework, by default, consists of A star global and DWA local planners. These are used as the baseline in this context to compare the navigation abilities of mobile robots through obstacle routes without collision. The Dynamic Windows Approach (DWA) is widely used for robot motion planning and obstacle avoidance [28]. It employs a local search strategy that allows the robot to select the best trajectory in real-time while avoiding collisions with detected obstacles. Comparing its capability with other ROS local planners such as potential field [29], DWA uses a velocity-based method to determine a collision-free path towards the objective against the way following the approach of the potential field. The robot can dynamically access and choose viable velocities that steer it towards the intended destination while ensuring obstacle avoidance [30]. The combination with the improved YOLOv5 object detection model enhances the mobile robot's obstacle detection and avoidance strength and overall navigation behaviour, which are critical for safe navigation.

C. The Proposed Method

Our goal is to develop a real-time object detection and avoidance system that can overcome object irregularities and light variations in a complex environment when carrying out an autonomous navigation task. The core idea is that it can pick out where the obstacles are and create a trajectory to avoid them accordingly. This will improve its ability to detect obstacles and augment the trajectories that it produces.

1) OAK-D camera

This paper uses the DepthAI OAK-D camera [13] as the primary vision-sensing device to capture images or video footage of the robot's environment. The camera is specifically designed for conducting experiments related to depth estimation, object detection, and tracking that effectively promote the obstacle detection and avoidance tasks required in this case study. Cameras are notable for high resolution and detailed information as compared to Light Detection and Ranging (LiDARs) especially when attempting to accurately track fast-moving objects in real-time.

The information from the Light Detection and Ranging (LiDAR) is used by default to update the local planner costmap, which is a 2D grid-based representation of the local environment around the Turtlebot in ROS navigation stack. The costmap assigns cost values to each cell in the grid based on the information received from the LaserScan message. Obstacles are assigned higher costs, while free spaces are given lower costs. To achieve this capability with a camera device, a mechanism is established to convert the objects detected within the camera's field of view into relevant scan-based information.

From the simulation perspective, the obstacle detection and avoidance system process begins with the integration of the OAK-D Depthai ROS repository developed for ROS into ROS navigation nodes as the source of data stream for object detection. The improved YOLOv5 model used in this work was developed to detect varieties of object structures in changing and light variation environments. These are setup in the ROS simulation environment [31] to acquires information from the environment as ROS image message using /camera/image_raw topic, which is converted to a suitable YOLOv5 readable format and resized using the OpenCV package. With the pretrained YOLOv5s package that includes model weights and configuration files, the ROS detector node process generates information to identify and describe objects in the environment. The detection output in the form of a ROS topic named /yolov5/yolov5, contains structural properties like the bounding box coordinates, class probabilities and object detection probability values that are transformed to generate a safe path to the target location [32].

2) YOLO's detected result to laser scan data

The detected object transformation starts by fetching the positions of the objects detected by the YOLOv5 package and injecting them into the local path planning algorithm. The DWA planner by default takes laser scan data as input from the LiDAR device to steer the velocity commands [33]. The detected YOLO result is converted to laser scan data and reprogrammed in the move-base package to improve DWA planner function as displayed in Fig. 2 workflow. The /yolov5 topic which is of vision_msgs/Detection2DArray of ROS object detection message type, generates the bounding boxes' pixel coordinates among other features and computes the center of the bounding boxes. To increase the robustness of the solution and ensure more accurate laser information, the left and right parts of the pixel coordinates of the obstacles were also computed. This provides accurate detection of varying shapes of obstacles within the 10m detection range of the camera device. The pixel coordinates are converted to camera coordinates with correspondence to the depth image captured by the RGB-D camera that provides the distance of each pixel from the camera's optical centre. This conversion is facilitated using camera intrinsic and extrinsic parameters provided by turtlbot3 Waffle Pi sensor_msgs/CameraInfo topic. This contains the focal lengths f_x and f_y and the principal points C_x and C_y that are used to compute the distance point of the objects as shown below in Eqs. (1) and (2).

$$Camera\ coordinate\ in\ x = \frac{(Pixel\ coordinate\ in\ x - C_x)}{f_x} \quad (1)$$

$$Camera\ coordinate\ in\ y = \frac{(Pixel\ coordinate\ in\ y - C_y)}{f_y} \quad (2)$$

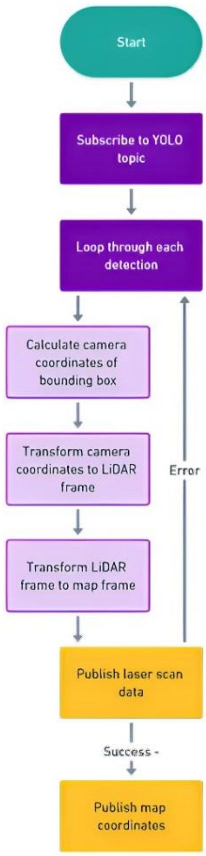

Fig. 2. YOLO object detection to scan data conversion workflow.

Once the camera coordinates were obtained, the ROS tf2 library which provides built-in tools to transform the camera data into scan data was employed to automate the transformation of data between frames. With the 3D coordinates of the detected objects now in the laser scan frame, the system can calculate the Euclidean distance from each object to the robot's centre. The Euclidean distance formula shown in Eq. (3) is applied to determine the straight-line distance between the detected object and the robot's centre. The LaserScan message is generated and published on a specific ROS topic to allow interface with other ROS nodes or applications.

$$distance = \sqrt{(X - x_o)^2 + (Y - y_o)^2} \tag{3}$$

where (X, Y) represent the transformed coordinate of the 3D point and (x_o, y_o) represent the origin of the robot's centre which in this case will be (0,0). To be sure of the consistency of the data converted, the LaserScan data corresponding to the position of the bounding boxes were also converted to map coordinates and printed. Then, the robot was placed in front of an obstacle of a known position. The map coordinates printed were slightly different by an avoidable few centimetres from the obstacle coordinates.

3) Obstacle avoidance

The process of detecting and transforming the object and generating distance values is repeated for all the pixels in the camera frame, essential for the DWA local planner to understand the immediate obstacles and free spaces surrounding the robot. The transformed /scan node that generates coordinate frames in suitable format for path planning and obstacle avoidance in the navigation stack is configured in the costmap file of the move-base package. The navigation stack capability is enhanced, and the ground robot from the experiment navigates through different checkpoints while avoiding detected obstacles with significant accuracy. The resulting solution is a lightweight system that can detect small to large environmental objects and respond in real-time to achieve a safe autonomous navigation experience. Performance metrics were defined for comparing the models, such as mean Average Precision (mAP), inference speed, time, distance covered and number of obstacles detected. Both accuracy and efficiency were considered to strike a balance between performance and real-time capabilities.

IV. Experiment

Aiming at a solution that will solve the navigation problem of obstacle detection and avoidance, the integrated solution was evaluated and validated through systematic experiments. These experiments were conducted on a computer running an Ubuntu 20.04 64-bit operating system, ROS noetic, and powered by an 8th generation Intel® CPU, with YOLO packages, OAK-D dependencies and ROS requirements adequately configured. To measure the solution's detection accuracy and collision avoidance efficiency, the robot was navigated from its initial position to a designated goal within the hangar setting. The average time taken for the robot to reach its destination was approximately 15 min. Throughout this process, the models' performance was observed in correctly identifying obstacles and instances where obstacles were not detected at all. We used common performance indicators for the object detection model and avoidance measures in the ROS framework, including the number of obstacles detected and avoided per distance and time, Intersection over Union (IOU), precision, and mean average precision.

We implemented our proposed model in the ROS_Gazebo simulation environment with the Cranfield DARTeC aircraft hangar virtual world model as shown in Fig. 3, and a total of fifteen obstacles, 12 static and 3 dynamic obstacles, were spawned in the environment at an increasing rate. The obstacles include a moving person and other aircraft hangar-related object types. The turtlebot3 robot navigation experience was tested in three different environments with varying complexities and 5 checkpoints each, to study the behaviour of the robot at different instances. The Adaptative Monte Carlo Localization (AMCL) pose of the robot was recorded using the rosbag tool. The AMCL is a localisation algorithm used in ROS to obtain more accurate information about the robot's pose in its environment. The robot poses were plotted to compare the path taken for the two methods and the total distance covered by the turrtlebot3 was computed using the Euclidean distance formula shown in equation 3 to determine the shortest path. Finally, the total navigation time for each of the methods and the number of collisions were recorded. The model was transferred to the real-world Turtlebot3 Waffle_pi robot, which showed good detection avoidance accuracy when used in an office environment, as shown in Fig. 4.

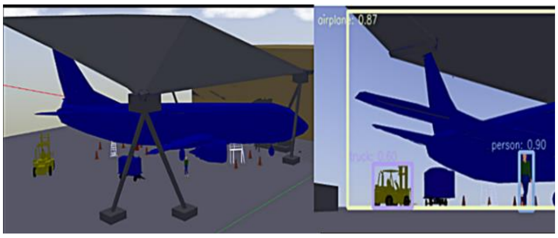

Fig. 3. Robot obstacle detection in a hangar environment.

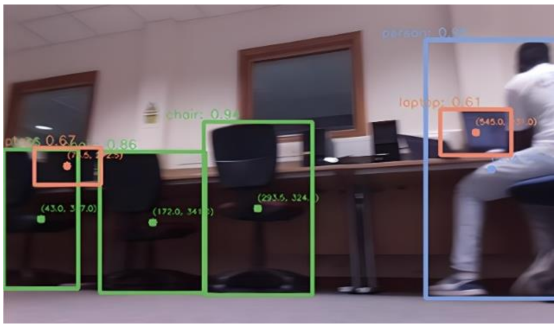

Fig. 4. Real-world object detection test result.

We compared the NAV-YOLO framework with the default ROS navigation stack DWA planner based on the number of obstacles detected and avoided per distance, distance covered and time to the target location. The detection accuracy was also evaluated based on YOLOv5 small and medium versions to observe the model's performance and generalizability to increased environmental complexities. To ensure robustness and reliability in the evaluation process, each experiment type is repeated five times. This repetition serves to mitigate any potential anomalies and to derive a more accurate and representative understanding of the system's behaviour across multiple instances. The performance evaluation results are detailed in Tables I, II, and III with visual representations in Figs. 5 and 6.

TABLE I. COMPARING TIME AND DISTANCE COVERED BY NAV-YOLO AND ROS DWA

Models	Time (min)	Distance Covered (m)	Number of Collisions
ROS NAV.Stack	16.37	278.9	2
NAV-YOLO	14.43	262.7	0

TABLE II. COMPARING OBSTACLES AVOIDED BY NAV-YOLO AND ROS DWA IN DIFFERENT ENVIRONMENTS

Models	Environ 1 with 5 Static Obstacles	Environ 2 with 10 obstacles (1 dynamic)	Environ 3 with 15 obstacles (3 dynamic)
ROS Navigation Stack	5	7	11
NAV-YOLO	5	10	14

TABLE III. COMPARING THE ACCURACY OF NAV-YOLO WITH YOLOV5 VARIANTS

Models	$mAP_{0.5}$ (%)	Precision (%)
YOLOv5m	75	74
YOLOv5s	70	77
NAV-YOLO	79	78

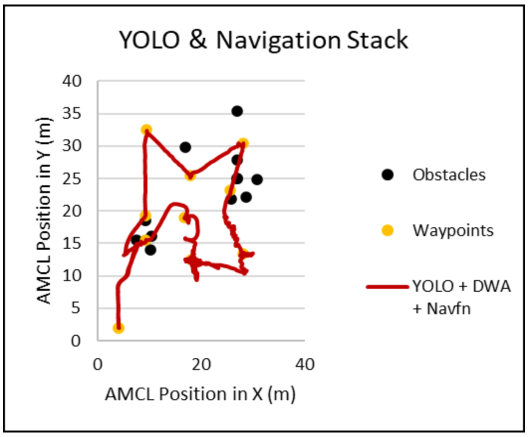

Fig. 5. NAV-YOLO trajectory.

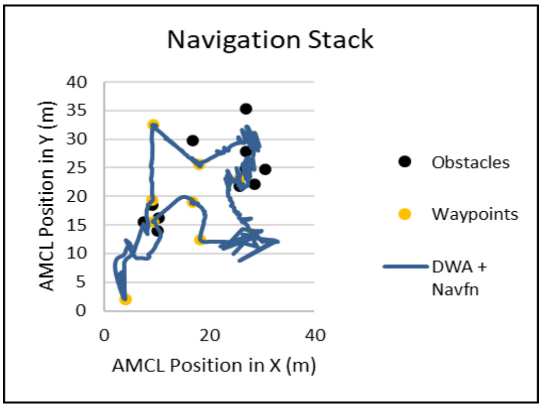

Fig. 6. ROS NAV trajectory.

V. RESULT AND DISCUSSION

A. Result

This section delves into the empirical evaluation of the object detection and avoidance models, highlighting their performance in accurately identifying obstacles of varying shapes and sizes for obstacle avoidance. Two variants of YOLOv5 are used: YOLOv5s and YOLOv5m, as benchmarks to compare the obstacle detection performance of NAV-YOLO in real-time. For the obstacle avoidance evaluation, efficiency was compared based on the ROS DWA planner and the new NAV-YOLO model.

After simulating each method, data shown in Table I below were obtained. The NAV-YOLO experiment was the fastest in terms of execution time, covered a reasonably short distance and the path followed seemed smoother without collision as shown in Fig. 5. The default algorithm experiment encountered an obstacle after reaching the second and fourth checkpoints and Fig. 6 showed a very rough path to the target location which contributes to the slightly longer average time of 16.37 min for 278.9 m distance covered. An essential aspect contributing to the extended execution time of the default algorithm lies in the occurrence of robot localization failures during a subset of tests. This phenomenon necessitates a period for the robot to rectify its position before resuming its intended path, consequently influencing the overall navigation time. This gives the proposed solution an edge as it shows better perception ability of its environment which improves its localization capability and overall navigation without collision. In Table II, three different environments were explored to evaluate the performance of the obstacle avoidance system for this work. The first experiment makes use of only static obstacles, in the second experiment, we increase the number of static obstacles plus one dynamic obstacle, and in the final experiment, we introduce three dynamic obstacles into the environment. It is important to note that the start position of the robot in all the experiments is the same. In environment 1, the robot detected all the static obstacles with both models but showed reduced detection ability for the ROS navigation stack in environments 2 and 3. These environments showed that our approach outperformed the baseline method by detecting almost all static and dynamic obstacles at different phases with augmented scenario complexity. The analysis of the accuracy of the detected objects, as shown in Table III brought out exciting findings. YOLOv5 versions showed good accuracy of 72.5% on average in detecting obstacles correctly, while NAV-YOLO performed significantly better with an accuracy of 79.2%, as visually presented in Fig. 3. The investigation is further conducted through real-world experiments on the Turtlebot3 robot, shedding light on its adaptability in navigating changing and complex environments. As shown in Fig. 4, NAV-YOLO displays an improved level of detection accuracy with a precision rate of 78% in an office environment without prior environmental experience.

B. Discussion

The successful integration of the ROS local planner and YOLO detector in both simulation and real-world states was proven, highlighting the effectiveness of our approach. The hangar model complexity was changed by increasing the number of obstacles per scene, as shown in Table II, to observe model efficiency and adaptability in complex settings. The ROS navigation stack with the default DWA planer [34] failed in some instances to detect and avoid obstacles, while the proposed approach showed a significant improvement for both static and dynamic environments in terms of accuracy of detection and avoidance in different environments. The baseline model's detection mechanism simplifies obstacle representation [35] and does not give detailed detected object information like the size, shape, and color of the object, and this impacts the avoidance technique, especially where dynamic obstacles are applicable.

Furthermore, the comparison of YOLOv5 variants and the proposed method shows a disparity in detection accuracy, as displayed in Fig. 5. The primary factor includes the architectural pattern that increases complexity from small to large versions of YOLOv5 models [11]. An increase in network structure increases detection accuracy and speed but requires more computational resources. The integrated pretrained YOLOv5s model [10] was developed with an improved YOLOv5m structure that balances computational intensity and performance to achieve better detection results. The overall combined solution shows improved detection avoidance capability for mobile robots compared to the performance of the ROS-based Dynamic Windows Approach planner. The notable limitation in this project is that the camera field of view can only capture objects in the forward direction making dynamic obstacle avoidance more challenging in more complex environments. The use of additional stereo cameras and fusion with LiDAR will be explored in the next project to mitigate this problem.

VI. Conclusion

This paper presents a vision-based obstacle detection and avoidance algorithm for robust and efficient navigation in aircraft hangars. The proposed approach successfully combines the YOLO detection model and ROS navigation stack to enhance robots' ability to detect and avoid obstacles accurately.

This combination harnesses the strengths of YOLOv5's object detection capabilities and the ROS DWA planner's real-time obstacle avoidance technique facilitating the real-time adaptation of the robot's trajectory to avoid collisions, showcasing the potential for dynamic obstacle avoidance. The ROS DWA planner uses real-time obstacle data from YOLOv5 to adjust and update the robot's planned path and this demonstrates its ability to effectively identify and classify obstacles of varying structures in different environments. The navigation solution was integrated into a real-world robotic platform (TurtleBot3), and the object detection and avoidance goals were significantly achieved. The proposed solution is a valuable solution for autonomous robot navigation and is applicable in diverse real-world scenarios.

With these promising results, there are still areas for further improvement. The integration of a more advanced planner algorithm can improve obstacle prediction with optimal obstacle avoidance navigation strategy to improve adaptability in unknown environments. For detection enhancement, object detection accuracy, especially in identifying intricate objects, remains a challenge and can be addressed through continuous model refinement and dataset augmentation. More performance metrics will be investigated and optimized to expand the effectiveness of this model in the real world.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

Each author made a great contribution to this work. Ndidiamaka conducted the research, simulation and experiments in the real world. Yanis contributed to the experiments. All authors contributed to the writing, reviewing and approval of the final version of this paper.

Acknowledgement

We would like to acknowledge Professor Nico Avdelidis and Dr. Gilbert Tang for their support and collaboration in this work.

References

[1] F. Rubio, F. Valero, and C. Llopis-Albert, "A review of mobile robots: Concepts, methods, theoretical framework, and applications," *Sage Journal*, 2019. doi: 10.1177/1729881419839596

[2] K. El Madawi, H. Rashed, A. El Sallab, O. Nasr, H. Kamel, and S. Yogamani, "RGB and LiDAR fusion based 3D semantic segmentation for autonomous driving," arxiv Reprint, arXiv:1906.00208, 2019.

[3] Robot Operating Systems (ROS): The fundamentals of ROS and its remarkable. [Online]. Available: https://www.ijraset.com/research-paper/robot-operating-systems-the-fundamentals-of-ros-and-its-remarkable

[4] D. Chikurtev, "Mobile robot simulation and navigation in ROS and Gazebo," in *Proc. 2020 International Conference Automatics and Informatics*, Oct. 2020. doi: 10.1109/ICAI50593.2020.9311330

[5] G. Chen, H. Yang, and I. M. Mitchell, "ROS-X-Habitat: Bridging the ROS Ecosystem with Embodied AI," in *Proc. 2022 19th Conference on Robots and Vision, CRV 2022*, Sept. 2021, pp. 24–31. doi: 10.1109/CRV55824.2022.00012

[6] K. Zheng, "ROS navigation tuning guide," *Studies in Computational Intelligence*, vol. 962, pp. 197–226, 2021. doi: 10.1007/978-3-030-75472-3_6/COVER

[7] P. Marin-Plaza, A. Hussein, D. Martin, and A. De La Escalera, "Research article global and local path planning study in a ROS-based research," *Journal of Advanced Transportation*, pp. 1–10, 2018. doi: 10.1155/2018/6392697

[8] I. Sarker, "Deep learning: A comprehensive overview on techniques, taxonomy, applications and research directions," *SN Comput Sci*., vol. 2, no. 6, Nov. 2021. doi: 10.1007/S42979-021-00815-1

[9] Z. Q. Zhao, P. Zheng, S. T. Xu, and X. Wu, "Object detection with deep learning: A review," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 11, 2019.

[10] N. Adiuku, N. Avdelidis, G. Tang, A. Plastropoulos, S. P. N. Adiuku, and S. Perinpanayagam, "CNN-fusion architecture with

visual and thermographic images for object detection," *SPIE*, vol. 12536, no. 12, pp. 180–189, June 2023. doi: 10.1117/12.2665984
[11] M. Horvat, L. Jelečević, and G. Gledec. A comparative study of YOLOv5 models performance for image localization and classification. [Online]. Available: https://www.researchgate.net/publication/363824867_A_comparative_study_of_YOLOv5_models_performance_for_image_localization_and_classification
[12] Z. Xu, X. Zhan, B. Chen, Y. Xiu, C. Yang, and K. Shimada. (Sept. 2022). A real-time dynamic obstacle tracking and mapping system for UAV navigation and collision avoidance with an RGB-D camera. [Online]. Available: https://arxiv.org/abs/2209.08258v2
[13] K. Egan. Review: Luxonis Oak-D AI camera with computer vision and image depth-make. [Online]. Available: https://makezine.com/article/maker-news/review-luxonis-oak-d-ai-camera-with-computer-vision-and-image-depth/
[14] H. Fujiyoshi, T. Hirakawa, and T. Yamashita, "Deep learning-based image recognition for autonomous driving," vol. 43, no. 4, *IATSS Research*, 2019. doi: 10.1016/j.iatssr.2019.11.008
[15] R. Sekhar, R. Dheekonda, S. K. Panda, N. Khan, M. Al-Hasan, and S. Anwar, "Object detection from a vehicle using deep learning network and future integration with multi-sensor fusion algorithm," *Collection of Open Conferences in Research Transpor*, vol. 2017, 133, 2017. doi: 10.4271/2017-01-0117
[16] D. Henke, *et al.*, "Mobile robot navigation using an object recognition software with RGBD images and the YOLO algorithm," *Applied Artificial Intelligence*, vol. 33, pp. 1290–1305, 2019. doi: 10.1080/08839514.2019.1684778
[17] H. Shah, Z. Kamis, M. Abdollah, A. Khamis, M. Aras, M. Baharon, and I Azni, "Vision-based obstacle avoidance for mobile robot using optical flow process," *International Journal of Innovative Technology and Exploring Engineering (IJITEE)*, vol. 8, no. 4S, February 2019.
[18] H. W. Luo, C. S. Zhang, F. C. Pan, and X. M. Ju, "Contextual-YOLOV3: Implement better small object detection based deep learning," in *Proc. 2019 International Conference on Machine Learning, Big Data and Business Intelligence*, MLBDBI 2019, Nov. 2019, pp. 134–141. doi: 10.1109/MLBDBI48998.2019.00032
[19] D. Aghi, V. Mazzia, and M. Chiaberge, "Local motion planner for autonomous navigation in vineyards with a RGB-D camera-based algorithm and deep learning synergy," *Machines*, vol. 8, no. 2, 2020. doi: 10.3390/machines8020027
[20] T. Lee, D. Yi, and D. Cho, "A monocular vision sensor-based obstacle detection algorithm for autonomous robots," *Sensors (Switzerland)*, vol. 16, no. 3, Mar. 2016. doi: 10.3390/S16030311
[21] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You Only Look Once: Unified, Real-Time Object Detection. [Online]. Available: http://pjreddie.com/yolo/
[22] M. Stancel and M. Hulic, "An Introduction to image classification and object detection using YOLO detector," *ICTERI PhD Symposium*, 2019.
[23] J. Zhang, Z. Ti, X. Lei, R. Arifando, S. Eto, and C. Wada, "Improved YOLOv5-Based lightweight object detection algorithm for people with visual impairment to detect buses," *Applied Sciences*, vol. 13, no. 9, 5802, May 2023. doi: 10.3390/APP13095802
[24] H. Li and X. Wu, "DenseFuse: A fusion approach to infrared and visible images," *IEEE Transactions on Image Processing*, vol. 28, no. 5, pp. 2614–2623, Apr. 2018. doi: 10.1109/TIP.2018.2887342
[25] L. Miao, N. Li, M. Zhou, H. Zhou Lize Miao, and H. Zhou, "CBAM-Yolov5: improved Yolov5 based on attention model for infrared ship detection," in *Proc. International Conference on Computer Graphics, Artificial Intelligence, and Data Processing*, Mar. 2022, vol. 12168, pp. 564–571. https://doi.org/10.1117/12.2631130
[26] X. Jia, C. Zhu, M. Li, W. Tang, and W. Zhou. LLVIP: A Visible-infrared Paired Dataset for Low-light Vision. [Online]. Available: https://bupt-ai-cz.github.io/
[27] Z. Li *et al.*, "A YOLO-GGCNN based grasping framework for mobile robots in unknown environments," *Expert Syst Appl*, vol. 225, 119993, Sept. 2023. doi: 10.1016/J.ESWA.2023.119993
[28] M. Kobayashi, "Local path planning: dynamic window approach with virtual manipulators considering dynamic obstacles," *IEEE Access*, 2022. doi: 10.1109/ACCESS.2022.3150036
[29] B. Kovács, G. Szayer, F. Tajti, M. Burdelis, and P. Korondi, "A novel potential field method for path planning of mobile robots by adapting animal motion attributes," *Rob Auton Syst*, vol. 82, pp. 24–34, Aug. 2016. doi: 10.1016/J.ROBOT.2016.04.007
[30] P. Huang, P. Huang, Z. Wang, X. Wu, J. Liu, and L. Zhu, "Deep-learning-based trunk perception with depth estimation and DWA for robust navigation of robotics in orchards," *Agronomy 2023*, vol. 13, vol. 13, no. 4, 1084, Apr. 2023. doi: 10.3390/AGRONOMY13041084
[31] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and N. Andrew. ROS: an open-source Robot Operating System. [Online]. Available: http://stair.stanford.edu
[32] K. Singh, D. Kapoor, K. Thakur, A. Sharma, and X. Gao, "Computer-vision based object detection and recognition for service robot in indoor environment," *Computers, Materials & Continua*, vol. 72, no. 1, pp. 197–213, Feb. 2022. doi: 10.32604/CMC.2022.022989
[33] C. Liu, B. Zheng, C. Wang, Y. Zhao, S. Fu, and H. Li, "CNN-based vision model for obstacle avoidance of mobile robot," in *Proc. MATEC Web Conf.*, 2017, vol. 139. doi: 10.1051/matecconf/201713900007
[34] D. Lima and G. Pereira, "Navigation of an autonomous car using vector fields and the dynamic window approach," *Journal of Control, Automation and Electrical Systems*, vol. 24, no. 1–2, pp. 106–116, Apr. 2013. doi: 10.1007/s40313-013-0006-5
[35] H. Yang and X. Teng, "Mobile robot path planning based on enhanced dynamic window approach and improved A* algorithm," *Journal of Robotics*, 2022. doi: 10.1155/2022/2183229

2024-03-22

Mobile robot obstacle detection and avoidance with NAV-YOLO

Adiuku, Ndidiamaka

EJournal Publishing

Adiuku N, Avdelidis NP, Tang G, et al., (2024) Mobile robot obstacle detection and avoidance with NAV-YOLO. International Journal of Mechanical Engineering and Robotics Research, Volume 13, Issue 2, March 2024, pp. 219-226

https://doi.org/10.18178/ijmerr.13.2.219-226