Journal of Robotics and Control (JRC)
Volume 2, Issue 1, January 2021
ISSN: 2715-5072 DOI: 10.18196/jrc.xxxx

Autonomous Person-Following Telepresence Robot using Monocular Camera and Deep Learning YOLO

Ahmad Amin Firdaus Bin Sakri
Faculty of Engineering & Built Environment, Universiti Sains Islam Malaysia,
Bandar Baru Nilai, Malaysia
Email: minfirdaus99@gmail.com

***Abstract*— A telepresence robot (TR) is a mobile robot with a teleconference system that can be moved and driven remotely by a remote user. It functions as an avatar for the remote user, allowing him/her to feel physically present in a local place and communicate with local individuals. For example, doctors may use the TR to remotely monitor and interact with Covid-19 positive patients in the hospital safely. In the area of TR, autonomous person-following is a well-needed feature to help the remote user navigate the environment. Thus, in this study, a person-following TR is developed using a deep learning algorithm and robot operating system (ROS). The deep learning algorithm based on You Only Look Once (YOLO-v4) is proposed for human identification and distance estimation using images captured from the camera. The algorithm has the advantage of fast computation that is suitable for our real-time application. For handling the communication framework, the robot operating system (ROS) is implemented on the TR. Finally, YOLOv4 and the ROS are integrated on a TR, namely Taariq, the mascot of USIM. Several experiments conducted show that the proposed TR can successfully follow a person.**

Keywords—Robot Operating System, You Only Look Once, Field of View

I. INTRODUCTION

People in the modern era were preoccupied with their daily tasks at home and in the office. Some of them were too preoccupied to complete their work because it required them to move from one location to another. A telepresence robot can be used to solve such issues and make long-distance communication feel more alive and real. Telepresence is the ability to engage directly with a physically real, remote world from a first-person POV there are no constraints on the location or size of the device used to carry out the user's commands at the remote site [1]. The telepresence robot is now widely used and continues to evolve, particularly during the Covid-19 pandemic, where physical movement and mass gathering are prohibited. Telepresence robots can be used in telemedicine to remotely check on patients in their homes, as well as to remotely check patients with highly contagious diseases such as Covid-19 [2]. It makes it possible for the student and teacher to attend class remotely for educational purposes. The use of telepresence robots in the workplace has been shown to provide benefits according to findings from a study where users can engage and involve in various interactions and participation in hub-and-satellite teams [3]. Usually, the telepresence robot is equipped with equipment and technologies related to Virtual Reality (VR) such as video cameras, microphones, and LCD screens to allow the user to sense and respond to the environment [4]. One of the autonomous techniques to control the telepresence robot is by using a human following technique that will follow a specific person within a set distance. Communication between the human and the robot is the most important aspect and a sensor such as an ultrasonic sensor, camera, and Laser Range Finder (LRF) is required to assure its success. Deep learning is a technique that can be implemented in the telepresence robot's control system to find and follow a specific person based on a visual tracking algorithm while avoiding obstacles. There are various types of deep learning methods that have different performances suitable for different applications such as YOLO, SSD, RCNN, R-FCN [5]. Object detection in YOLO is done as a regression problem and provides the identified photos' class probabilities [6]. It was implemented in many real-time application object detections by examining the entire image, and its predictions are influenced by the image's overall context during the test. It also produces predictions based on a single network evaluation, as opposed to models like RCNN, which require thousands of evaluations for a single image [7].

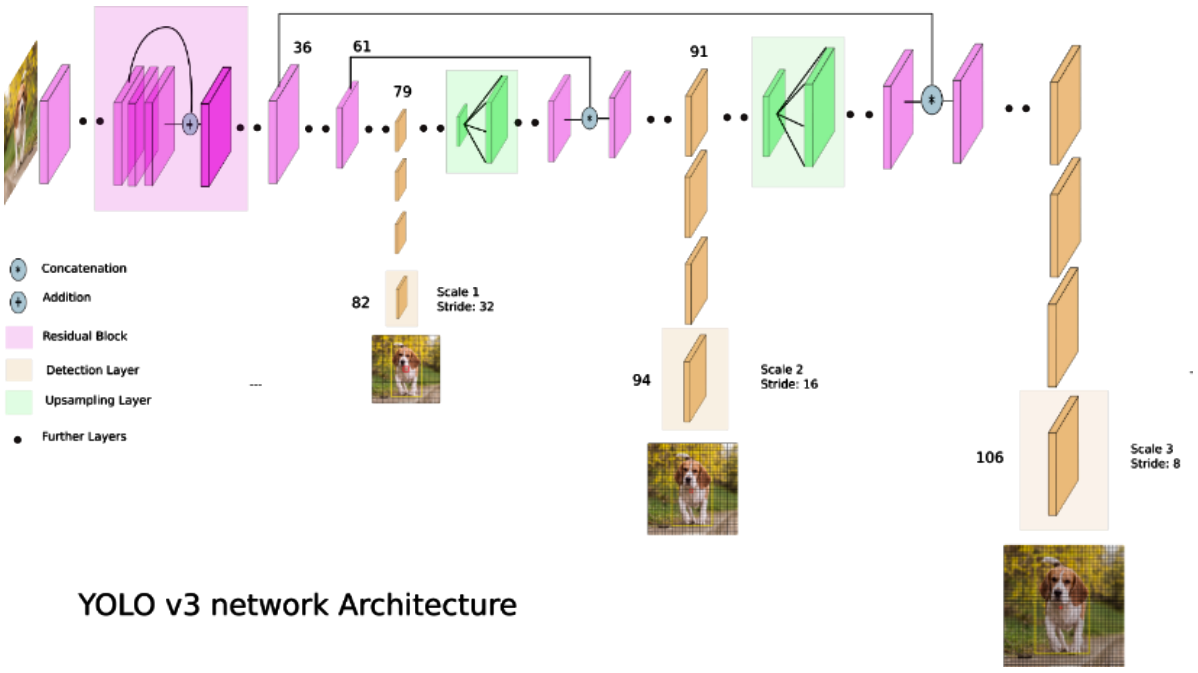

Figure 1: The architecture of the YOLOv3 algorithm [8].

Figure 1 shows the architecture of the YOLOv3 algorithm. The input image is divided into an SxS grid and if the object's center falls inside a grid cell, that cell is responsible for detecting the object. Every grid cell will manage to observe and predict five bounding boxes to determine the confidence score and class of each box. It determines the object class based on a probability distribution score of overall potential classes [9]. The likelihood that this bounding box contains a given sort of object is determined by combining the bounding box's confidence score with the class prediction [7]. There are many datasets for real-time objection detection available to train the YOLOv4 algorithm such as Open Images dataset, Stanford human actions dataset, and ImageNet dataset.

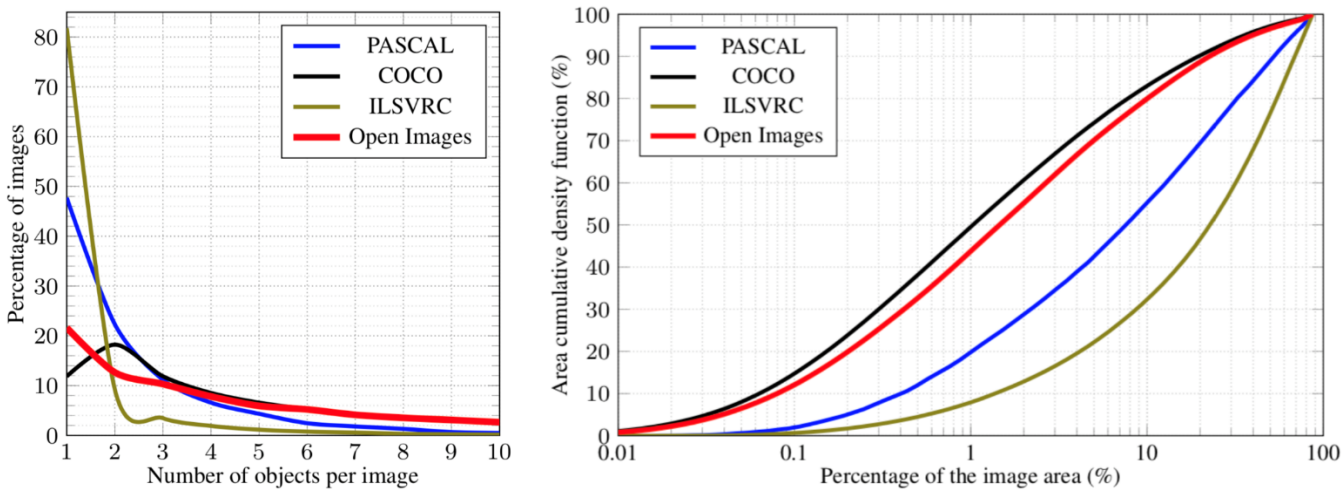

Figure 2: The number of objects per image and the area of each object for four different datasets [10].

According to Figure 2, the Open Images Train and Challenge sets, which include the majority of the data, show a rich and diverse complexity distribution similar to the COCO dataset. This is based on the number of objects per image and their area distribution. A person following robot required some sensors to detect and estimate the distance of the object. There are two types of image sensor technologies used in digital monocular cameras: CMOS and CCD. In terms of linearity, the CCD sensor is two orders of magnitude more linear than the CMOS sensor, which has non-linear effects that are only visible at lower intensities [11]. Image quality is also an essential parameter to choose a suitable camera according to the applications. It can be achieved by improving the SNR to avoid the image being disturbed by noise [12]. When compared to CCD cameras, the larger pixel size of CMOS sensors results in a larger full well capacity and high SNR. In CMOS, each pixel had an open loop output amplifier, and the offset and strength of each amplifier varied a lot due to wafer processing differences, making both dark and lit nonuniformities worse than in CCDs [13]. Dark

images are eliminated from recorded images to increase image uniformity in CMOS [14]. The image quality provided by CMOS cameras is slightly greater than that of CCD cameras, but a CCD camera should be used for high-precision measurements. The distance between the object and the robot is important in many fields of application for autonomous mobile robots. The information about the distance of surrounding objects was used for the robot's localization and navigation. An algorithm is required to estimate the distance of the object using the input image or real-time video from the camera. The motor's speed is determined by the distance between the robot and the target. Distance estimation will assist the robot in avoiding heavy collisions that can damage the robot's structure and injure humans. The main objective of this project is to develop a working model of an autonomous person-following telepresence robot. The robot should be able to navigate through an indoor environment by following a specific human target without the use of a controller. This paper is organized as follows. Section II describes the project methodology, which includes the system architecture, hardware, and software design. Section III describes the results of several experiments conducted with Taariq, and Section IV concludes the paper.

II. METHOD

A. *Architecture of The System*

The telepresence robot was designed with two modes of operation. It has both autonomous and manual mobility modes. The manual mode is implemented in this project as a backup for the user to take control of the robot when needed, as well as a safety measure to avoid an accident if the robot's autonomous mode fails.

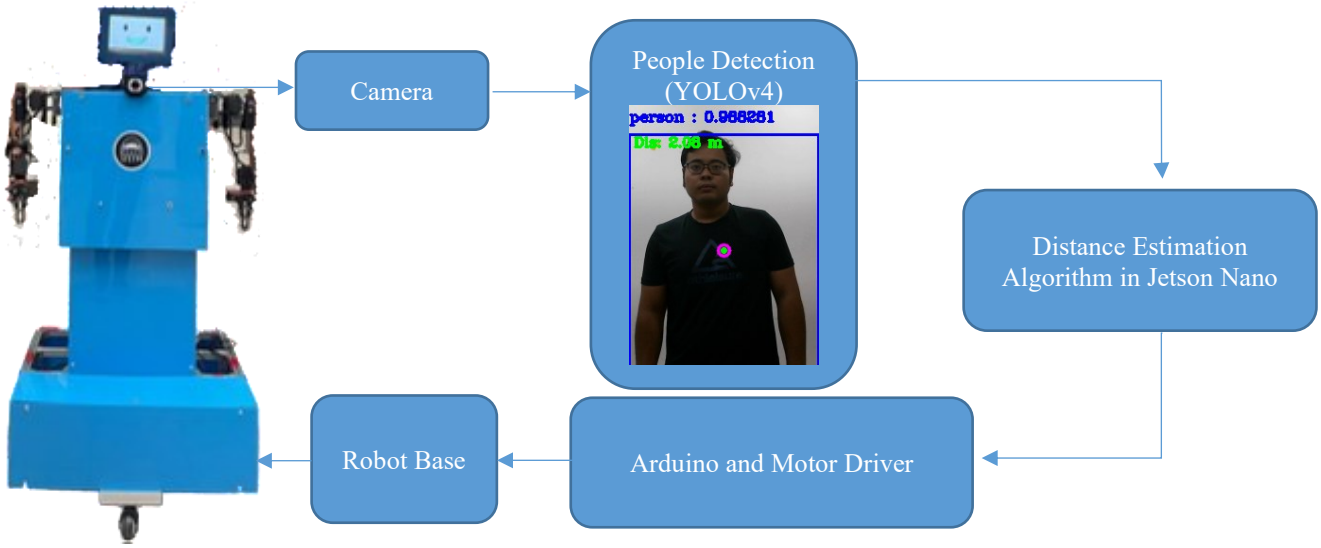

Figure 3: the overall flow of the system of the project.

Figure 3 shows the flow and connection of the system and components. the camera will act as a sensor, recording and capturing images from the environment to use as input images. The image is sent to the Jetson Nano which acts as a high-level controller where the image processing is done. The Jetson Nano at the robot is remotely controlled and monitored by the user using a Desktop over the same network using SSH and VNC Viewer software. The desktop will be turned into a control station where the user can monitor the output image processed by the Jetson Nano and switch to manual control mode to control the telepresence robot with a keyboard. Furthermore, the control station is critical for debugging and avoiding accidents if the autonomous system fails. The Jetson Nano will send its real-time display to the control station. The person-following program will process the image recorded by the webcam and publish direction data to the 'Direction' topic. The data from the Jetson are subscribed by the Arduino program using ROS Serial Communication. Arduino will read the input data and begin the robot's movement by sending the output data to the motor driver. The motor driver will control the speed and direction of both DC motors to create the differential motor mechanism. The differences in speed and rotation of the two-wheeled system allow the robot to rotate 360 degrees and keep moving while keeping the target person in the detection box. The body of the robot is made up of aluminum profile 20x20, acrylic sheets, and all other components such as DC Motor, Motor Driver, Arduino Mega (low-level controller), Jetson Nano (high-level controller), 12 V battery is stored in the base of the robot.

The monocular camera that has been used in this project is a webcam camera (Logitech C615). This monocular camera developed with autofocus features and 30 FPS makes this webcam camera suitable for use in this project's real-time application. The video recorded by this camera will stay sharp even if the person is moving around and close to the camera. The minimum distance this webcam's autofocus feature can work is about 9.906 cm. The camera has a field of vision

(FOV) of 74 degrees and produces videoframes with a typical resolution of 720P (1280720 pixels) hence it does produce higher-quality photos that will improve the accuracy of the YOLOv4 person detection algorithm by producing sharp video [15]. It also can reduce the time for image early-preprocessing for subsequent back-end procedures.

B. Software Design

i. Person Detection by using YOLO

Object detection necessitated the development of a suitable model that could be implemented in a real-time application. YOLO (You Only Look Once) was used to predict image classes and bounding boxes. The model is based on regression. The project used YOLOv4 pretrained model to detect and calculate human distance. This project does not require any training because the pre-trained model is used. The training process necessitates high processor speed and power, which can burden other processes, memory space, and systems [16].

Figure 4: The method of the YOLOv4 implementation.

Figure 4 shows the implementation of YOLOv4 in the project. All of the YOLO algorithms used for human detection and distance approximation are run on the GPU with the CUDA backend to achieve faster image processing speeds and better FPS. The majority of recent accurate models require powerful GPUs for training with large mini-batch sizes, and doing so with only a CPU is extremely slow and inefficient. On the MS COCO dataset, YOLO v4 produces state-of-the-art results in real-time, with 43.5 percent Average Precision (AP) operating at 65 FPS on a Tesla V100 [17].

The image of the person detected by YOLO is encapsulated by a boundary box drawn using OpenCV libraries method. The live streaming output from the camera is displayed on the Jetson Nano and it can be accessed by the control station using VNC Viewer. The size of the output frame is set to 640 x 480 pixels. The output frame is shown in Figure 5, Figure 6, and Figure 7.

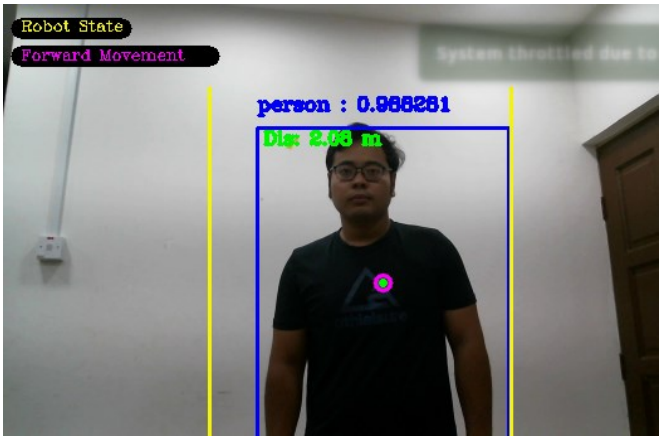

Figure 5: the output frame shows the center of the boundary box located in the center region of the camera.

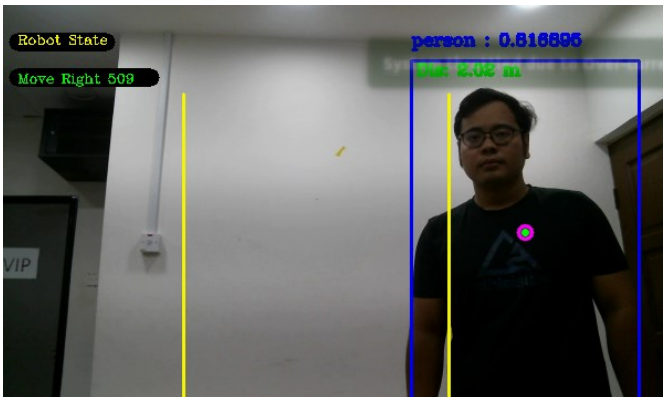

Figure 6: the output frame shows the center of the boundary box located in the right region of the camera.

Figure 7: the output frame shows the center of the boundary box located in the left region of the camera.

The camera output frame in Figure 5, Figure 6, and Figure 7 above shows the output from the camera with a boundary box detecting the person, as well as the distance estimation and accuracy values. The functions and methods from the OpenCV library successfully drew the boundary line that distinguishes the output video into three regions A center point was calculated by dividing the frame's width and height by two, and it was successfully displayed on the boundary box detecting the person. The “twist” message is published

according to the distance estimation values and the area where the center point lies.

ii. Distance Estimation using YOLOv4 and monocular camera

The autonomous person-following features required the robot to detect and measure the distance between the telepresence robot and the targeted person. Hence, an algorithm is used and combined with the YOLOv4 object detection.

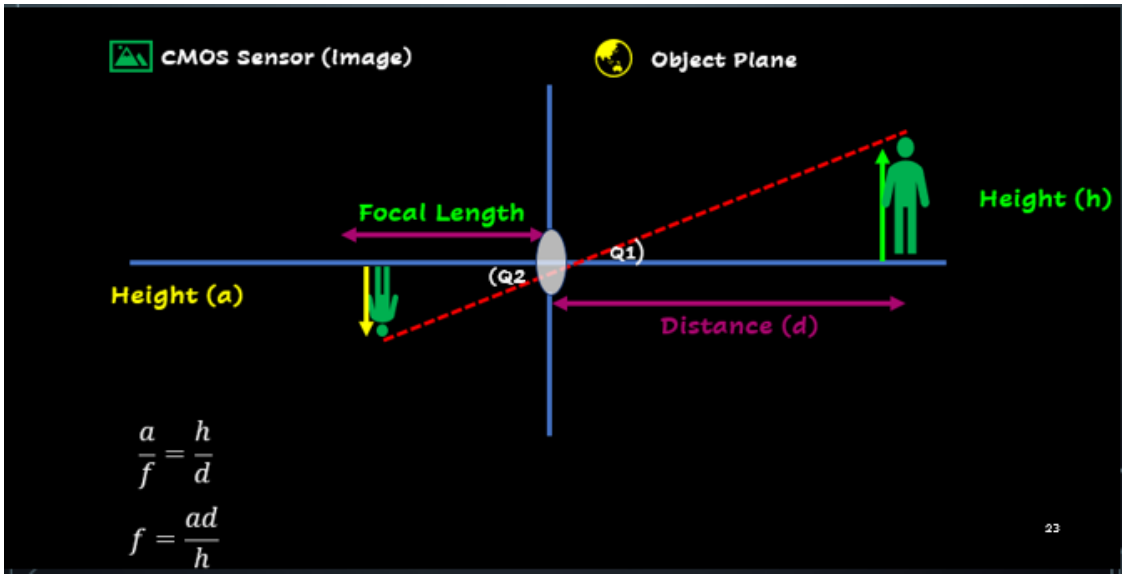

Figure 8: the diagram of the method for obtaining the focal length of the camera.

Figure 8 describes the basic concept of the method used in this project. On the left side of the plane, the image of the person captured by the camera lens is inverted. The angles at the left and right sides of the plane are equal to each other, a reference distance is required to calculate the focal length of the lens. This refer distance is measured carefully to obtain a precise focal length measurement. The width is calculated instead of the height of the person. The reference distance and the reference width of the person are set to 1.143 m and 0.406 m respectively. The derivation of the algorithm is calculated as below:

$$\tan\theta_1 = \tan\theta_2$$
$$\frac{a}{f} = \frac{h}{d}$$
$$f = \frac{ad}{h}$$

Once, the focal length of the lens is obtained, the new and unknown distance of the targeted person can be measured by deriving the previous focal length equation.

$$d^* = \frac{h^* f}{a^*}$$

Where d^* is the unknown distance that the robot needs to estimate its measurement. It is calculated by multiplying the real width of the person by the focal length and dividing the result by the width of the person in the image.

iii. Robot Operating System (ROS)

ROS is a piece of open-source software that provides a framework for hardware abstraction and low-level device control. It is a set of tools for developers to use to create and develop robot applications. This operating system includes low-level device drivers as well as high-level application blocks including visualization and vision tools. There is a master called "roscore" who will manage all communication between the programs [18]. In ROS, there are many packages, and each package is a directory for more nodes. A node is a program that communicates with other nodes using the topics and services protocols.

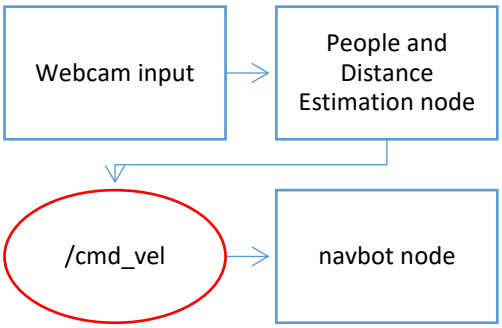

Figure 9: the architecture of the autonomous person following method where topic is represented by red circle

Figure 9 shows the architecture of the autonomous person-following feature using ROS. This ROS system enables peer-to-peer communication and standardizes the hardware environment. A node will publish a message to a certain topic name. Other nodes can subscribe to this topic to receive access to the published message. ROS allows data exchange between the Jetson Nano node (Python) and the Arduino (C++). Because the publisher and subscriber are independent and unaware of each other's existence, the master will take on the responsibility of exchanging messages between them [19]. There is a Python program running on

Jetson Nano to publish an integer value signal as a topic called “Direction” that indicates the position of the person in the image to Arduino. The object detection used the OpenCV package to draw the detection box and find the center of the object detected on the recorded video. The Arduino will subscribe to the “Direction” topic to learn the coordinates and distances of the person from the webcam. The Arduino compared the integer value subscribed from the "Direction" topic to control the speed of both DC motors forward, backward, right, left, or stop.

iv. OpenCV Library for Computer Vision

OpenCV is a free and open-source library for developing computer vision infrastructure that was used in this project [20]. The main function of OpenCV used is to capture the video output from the camera to be processed by the YOLOv4 algorithm and create a frame to be displayed on the screen [20].

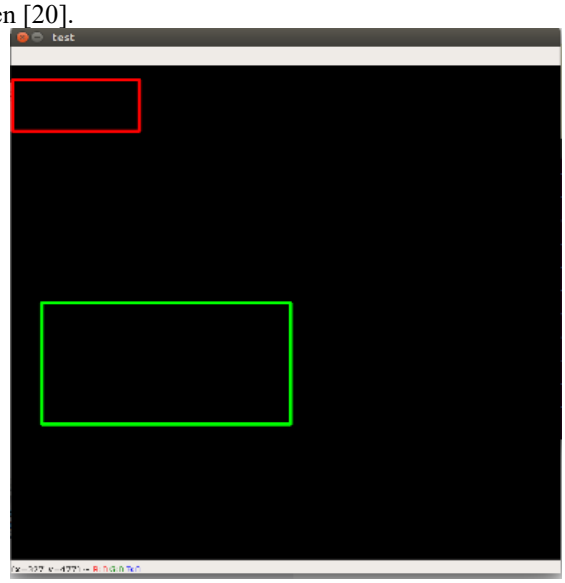

Figure 10: the boundary box created with the (cv2.rectangle()) method.

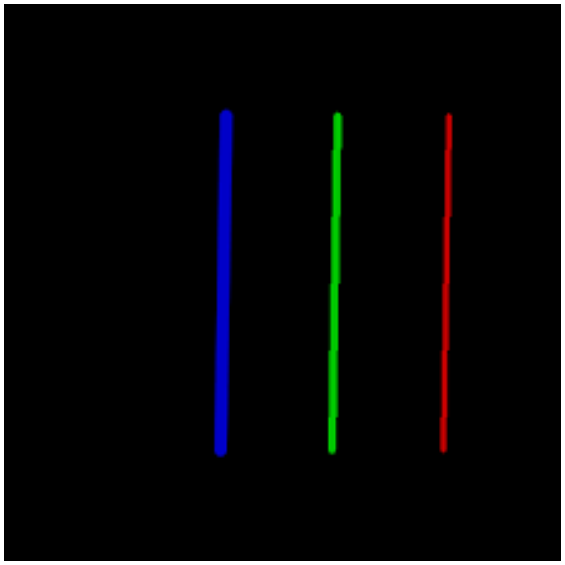

Figure 11: the line created with the (cv2.line()) method.

The library provides a useful function to draw the boundary box and center point for person detection in this project using the (cv2.rectangle()) method like in Figure 10 [21]. There is also two boundary line drawn using the OpenCV method (cv2.line()) shown in Figure 11 [22]. Both the center point and the boundary line are used to pass the position value of the person in the camera’s output. The boundary lines are drawn at point 140 from the left frame width and at point -140 from the right frame. The line is drawn to indicate the edge of the image's center area. If the center point exceeds the right boundary lines, it indicates that the person is in the right area of the robot, and vice versa if the center point exceeds the left boundary line. OpenCV also supports text printing on streaming video. The method (cv2.putText()) is used to print the action and direction that should be used by the DC motors to drive the robot back into the center area of the video. The version of OpenCV used is 4.5.3, which is installed in Ubuntu 18.04 and has been compiled and built with the CUDA backend. The CUDA backend is essential and must be enabled during the building process for the program and OpenCV functions to run on the GPU rather than the CPU. All of the hardware and software that have been stated in this chapter have been integrated to create a working system for the telepresence robot. The performance of this system has been tested and studied in the next section.

III. RESULT AND DISCUSSION

Several analyses have been discussed on the experiment done during the development of this telepresence robot. The result of this experiment has been divided into five modules to give a better understanding of how the final result of this project is successfully achieved.

A. Accuracy of The Person Detection Using YOLOv4

Figure 12: the values from the accuracy of the detection and distance estimation are displayed on the top of the boundary box.

Figure 12 illustrates the method of the experiment conducted to test the accuracy of the person detection using YOLOv4. The experiment is conducted by recording the accuracy of the person detection reading for every 0.2 meters

between 0.9 and 3.3 meters in an indoor environment with optimal lighting. The accuracy of object detection is important for determining the closeness of the object in the image to the object in the real world [51].

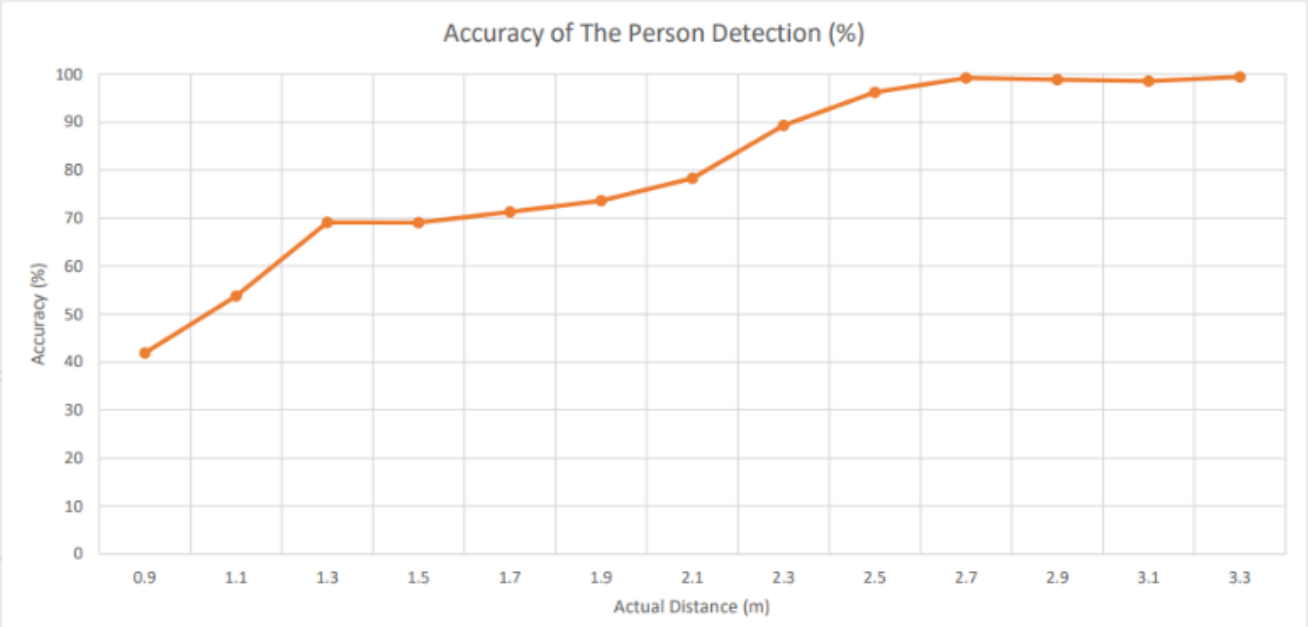

Figure 13: the graph for the accuracy of the person detection.

Figure 13 displays the YOLOv4's accuracy in detecting people increases as the distance between the camera and the person increase. The accuracy begins to increase gradually at 2.3 meters and remains constant above 98 percent starting at 2.7 meters. This is because as the person gets closer to the camera, only a small part of their body is captured in the boundary box, lowering the detection's accuracy. At 2.3 meters, the boundary box from YOLOv4 detects two-thirds of the person's body and at 2.7 meters, the boundary box detects almost the entire body. The boundary box is responsible for detecting the coordinate and probability of the object in each cell produced by image separation. It suppressed the boundary box with the low probability score and remain the boundary box with the highest probability score only. Furthermore, the accuracy of the detection for the range 1 to 3 meters is all above 50% percent which is good because the telepresence robot for this project is required to detect the person between this range. The accuracy of person detection is important to ensure that the robot follows the real person rather than other objects, especially in areas with poor lighting which can reduce accuracy.

B. Accuracy of The Distance Estimation Using Monocular Camera

The experiment is carried out in the same way as in Figure 12 to test the accuracy of the distance estimation algorithm using a monocular camera.

Table 1: the measurements between the actual distance and estimation distance.

Actual Distance (m)	Estimation Distance (m)
0.90	0.91
1.10	1.13
1.30	1.35
1.50	1.54
1.70	1.73
1.90	1.94
2.10	2.20
2.30	2.30
2.50	2.57
2.70	2.76
2.90	2.93
3.10	3.20

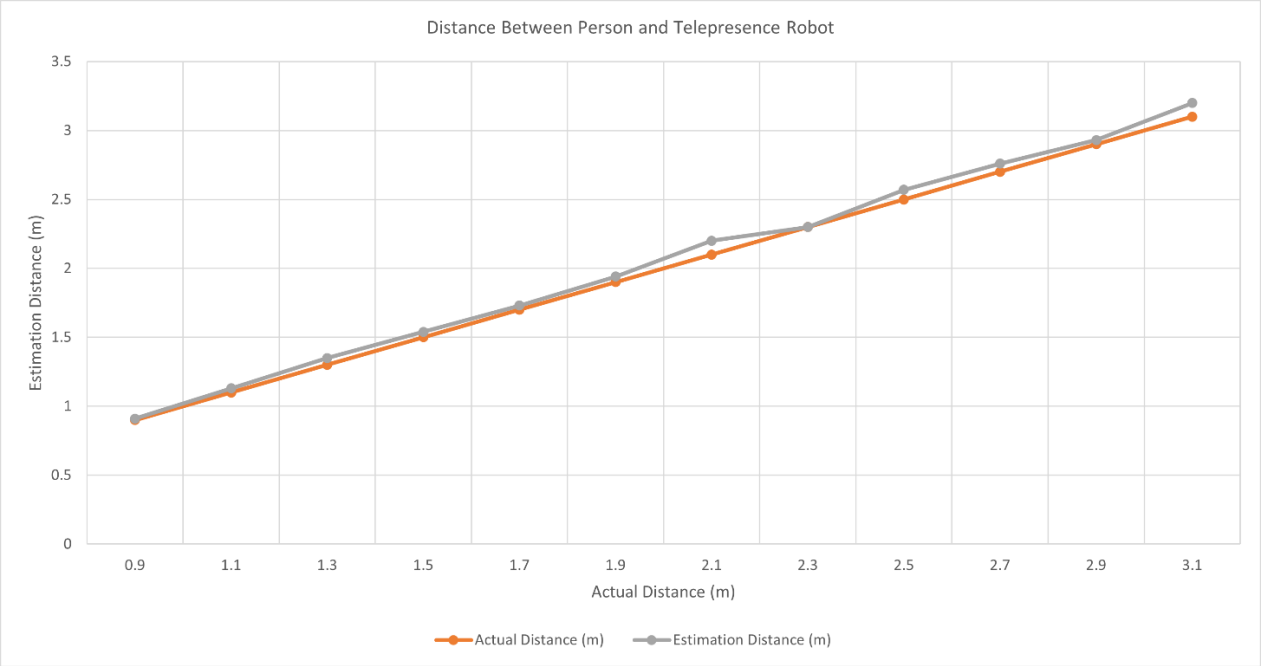

Figure 14: The comparison between the actual distance and estimation distance.

Table 2 and Figure 14 both show the comparison between the actual distance and estimation distance. The estimation distance produced by the algorithm is quite accurate with slight differences only. The greatest difference between the estimation and actual distance is 0.10 m for distances 3.10 m and 2.10 m. The accuracy of the algorithm is defined by the width of the boundary box detecting the person. The position of the hand is critical in determining its accuracy. The estimation distance becomes less accurate as the hand is raised wider.

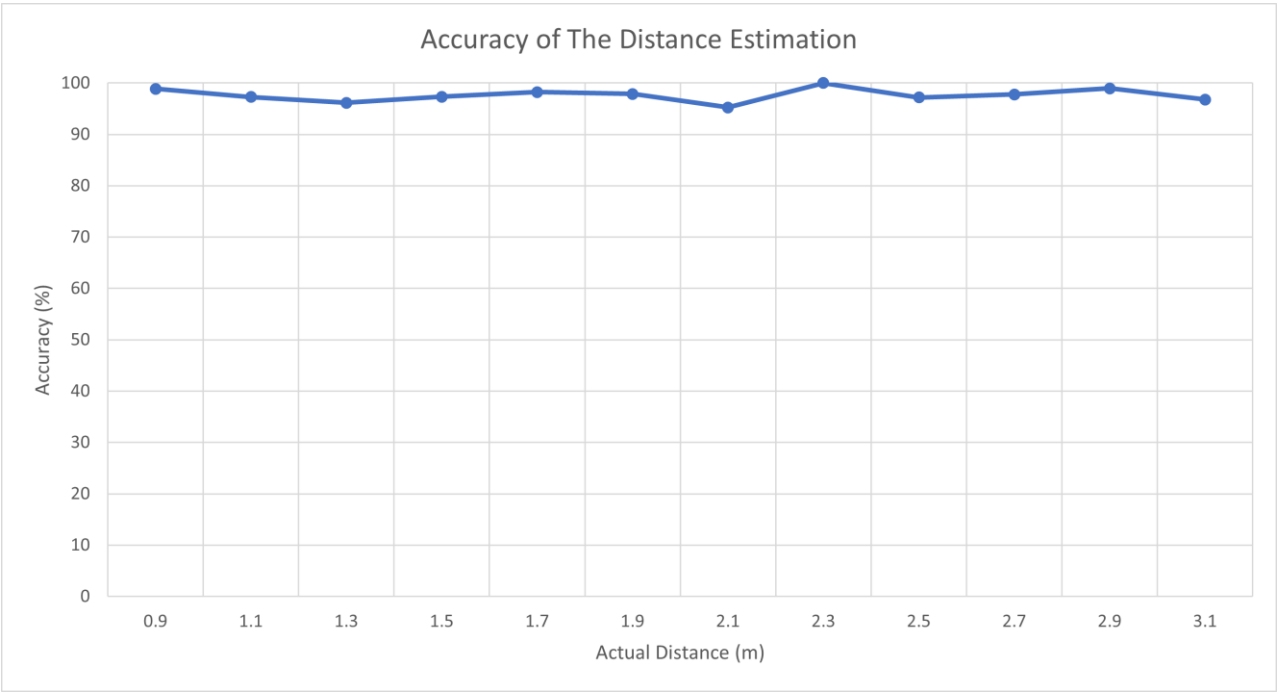

Figure 15: the accuracy of the distance estimation algorithm.

Figure 15 illustrates the accuracy of the distance estimation performed by the algorithm. All of the readings achieved an accuracy above 95%. Thus, this proves the algorithm is reliable and accurate enough to be used in the project. The framerate has also been increased by utilizing the CUDA backend and Nvidia GPU computation. The algorithm takes 6 seconds to estimate distance using CPU computation. By enabling and compiling the CUDA backend together with OpenCV, the delay time was reduced to one second. This gave enough time for the telepresence robot to follow the targeted person in a real-time application.

C. Movement of the telepresence robot

After all of the modules have been compiled and merged, the developed telepresence robot now has autonomous functionality. The telepresence robot is programmed to

follow a specific person in a range of 1 to 5 meters and to move backward when the distance is less than 0.5 meters when the person stands or walks in front of the robot as shown in Figure 16. The robot's movement can be clearly understood by referring to Table 3.

Table 2: the robot's movement based on the location of the center point and the distance estimation value.

Location of The Center Point	Distance Estimation Value, X (m)	Twist Message Value	Robot Movement
Center Region	X<= 0.5	twist.linear.x = −1 twist.angular.z = 0	Backward
Center Region	1<X<=5	twist.linear.x = 1 twist.angular.z = 0	Forward
Center Region	X>5	twist.linear.x = 0 twist.angular.z = 0	Stop
Right Region	X	twist.linear.x = 0 twist.angular.z =−1	Right
Left Region	X	twist.linear.x = 0 twist.angular.z =1	Left

The output frame from the camera is important because it allows the remote user to monitor the movement of the robot and is also used to troubleshoot problems with the robot's movement. Due to the fast computation using YOLOv4 and the CUDA backend, the robot can follow people and turn almost instantly with a one-second delay.

Figure 16: the telepresence robot successfully detects and follows the targeted person with the correct measurement.

IV. CONCLUSION

Autonomous person-following telepresence robots have numerous applications in a wide range of fields, including industry, medicine, and education as an avatar for the user to sense and interact with the telepresence robot's surrounding environment. This project was developed to solve the drawbacks of the manual control mechanisms used in most of today's telepresence robots. The project employed a deep learning technique to process the input images via the YOLO algorithm due to its fast computation in a real-time application, as well as ROS to standardize the environment of various hardware and allow the data exchange between them. The monocular camera used in this project is integrated with the person detection and distance estimation algorithms to provide computer vision to the telepresence robot, allowing the robot to operate in autonomous mode. There is a control station that allows the user to monitor the robot's movement of the telepresence robot. Two types of controllers have been used, with Jetson Nano handling high-level processes and Arduino handling low-level applications such as controlling the motor speed. The high accuracy of the person detection and distance estimation algorithms used in this project enables the telepresence robot to detect and estimate the distance between it successfully. Finally, the telepresence robot developed in this project has achieved all of the objectives where it can finally operate in autonomous person-following mode. This telepresence robot's hardware design and software architecture allow it to operate in real-time indoor applications, making it suitable for use as a USIM mascot at a variety of indoor events and occasions.

As future work, this project can be improved in the future by incorporating a more advanced depth camera that can operate in low light conditions while reducing color distortion effects and producing a more accurate distance estimation value. Furthermore, the number of people detected by this project can be increased in the future.

REFERENCES

[1] W. R. Sherman and A. B. Craig, "Introduction to Virtual Reality," *Understanding Virtual Reality*, pp. 4–58, 2019, doi: 10.1016/B978-0-12-800965-9.00001-5.

[2] M. Suomalainen, K. J. Mimnaugh, I. Becerra, E. Lozano, R. Murrieta-Cid, and S. M. LaValle, "Comfort and Sickness while Virtually Aboard an Autonomous Telepresence Robot," Sep. 2021, doi: 10.1007/978-3-030-90739-6_1.

[3] IEEE Robotics and Automation Society, Robotics Society of Japan, Han'guk Robot Hakhoe, Nanjing lin ye da xue, and Institute of Electrical and Electronics Engineers., *IEEE RO-MAN 2018 : RO-MAN2018 : the 27th IEEE International Symposium on Robot and Human Interactive Communication*.

[4] J. Du, H. M. Do, and W. Sheng, "Human–Robot Collaborative Control in a Virtual-Reality-Based Telepresence System," *International Journal of Social Robotics*, vol. 13, no. 6, pp. 1295–1306, 2021, doi: 10.1007/s12369-020-00718-w.

[5] C. E. Kim, M. Dar Oghaz, J. Fajtl, V. Argyriou, and P. Remagnino, "A Comparison of Embedded Deep Learning Methods for Person Detection."

[6] M. W. Adou, H. Xu, and G. Chen, "Insulator Faults Detection Based on Deep Learning."

[7] C. E. Kim, M. M. D. Oghaz, J. Fajtl, V. Argyriou, and P. Remagnino, "A Comparison of Embedded Deep Learning Methods for Person Detection," Dec. 2018, [Online]. Available: http://arxiv.org/abs/1812.03451

[8] B. Chethan Kumar, R. Punitha, and Mohana, "YOLOv3 and YOLOv4: Multiple object detection for surveillance applications," in *Proceedings of the 3rd International Conference on Smart Systems and Inventive Technology, ICSSIT 2020*, Aug. 2020, pp. 1316–1321. doi: 10.1109/ICSSIT48917.2020.9214094.

[9] S. Sambolek and M. Ivasic-Kos, "Automatic person detection in search and rescue operations using deep CNN detectors," *IEEE Access*, vol. 9, pp. 37905–37922, 2021, doi: 10.1109/ACCESS.2021.3063681.

[10] "Open Images V6 - Description." 2019. [Online]. Available: https://storage.googleapis.com/openimages/web/factsfigures.html

[11] J. Manin, S. A. Skeen, and L. M. Pickett, "Performance comparison of state-of-the-art high-speed video cameras for scientific applications," *Optical Engineering*, vol. 57, no. 12, p. 1, Dec. 2018, doi: 10.1117/1.oe.57.12.124105.

[12] L. Fan, F. Zhang, H. Fan, and C. Zhang, "Brief review of image denoising techniques", doi: 10.1186/s42492-019-0016-7.

[13] K. D. Stefanov *et al.*, "Monolithic Pinned Photodiode CMOS Image Sensor Using Reverse Substrate Bias", doi: 10.3390/s18010118.

[14] H. Cao, X. Gu, X. Wei, T. Yu, and H. Zhang, "remote sensing Lookup Table Approach for Radiometric Calibration of Miniaturized Multispectral Camera Mounted on an Unmanned Aerial Vehicle", doi: 10.3390/rs12244012.

[15] A. Fitwi, Y. Chen, H. Sun, and R. Harrod, "Estimating Interpersonal Distance and Crowd Density with a Single-Edge Camera," 2021, doi: 10.3390/computers10110143.

[16] M. U. Yaseen, A. Anjum, O. Rana, and R. Hill, "Cloud-based scalable object detection and classification in video streams," *Future Generation Computer Systems*, vol. 80, pp. 286–298, Mar. 2018, doi: 10.1016/J.FUTURE.2017.02.003.

[17] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," Apr. 2020, Accessed: Mar. 02, 2022. [Online]. Available: http://arxiv.org/abs/2004.10934

[18] L. Joseph, *Robot Operating System for Absolute Beginners*. Apress, 2018. doi: 10.1007/978-1-4842-3405-1.

[19] R. and C. A. Caiazza Gianluca and White, "Enhancing Security in ROS," in *Advanced Computing and Systems for Security: Volume Eight*, A. and S. K. and C. N. Chaki Rituparna and Cortesi, Ed. Singapore: Springer Singapore, 2019, pp. 3–15. doi: 10.1007/978-981-13-3702-4_1.

[20] S. Gollapudi, "OpenCV with Python," in *Learn Computer Vision Using OpenCV: With Deep Learning CNNs and RNNs*, Berkeley, CA: Apress, 2019, pp. 31–50. doi: 10.1007/978-1-4842-4261-2_2.

[21] H. Adusumalli, D. Kalyani, R. K. Sri, M. Pratapteja, and P. V. R. D. P. Rao, "Face Mask Detection Using OpenCV," in *Proceedings of the 3rd International Conference on Intelligent Communication Technologies and Virtual Mobile Networks, ICICV 2021*, Feb. 2021, pp. 1304–1309. doi: 10.1109/ICICV50876.2021.9388375.

[22] B. Jeeva, V. Sanjay, V. Purohit, D. O. Tauro, and J. Vinay, "Design and development of automated intelligent robot using opencv," in *Proceedings - 2018 International Conference on Design Innovations for 3Cs Compute Communicate Control, ICDI3C 2018*, Aug. 2018, pp. 92–96. doi: 10.1109/ICDI3C.2018.00028.