American Journal of Sciences and Engineering Research

E-ISSN -2348 – 703X, Volume 6, Issue 1, 2023

Development of an Intelligent Model for Visually Impaired Person Guidance

Wissam S. Mohammed [1], Dr. Thair A. Salih [2]

[1]*Technical Engineering College, Computer Engineering Technology, Northern Technical University, Mosul, Iraq.*

[2]*Technical Engineering College, Computer Engineering Technology, Northern Technical University, Mosul, Iraq.*

ABSTRACT: *In this work, the implementation of the development of an intelligent model for visually impaired person guidance using the Jetson Nano developer kit has been accomplished. The main goal of this work is to help visually impaired people to move safely indoors and outdoors. Implementing a prototype for real-time door recognition and classification in both indoor and outdoor contexts into three classes (open, semi-open, and closed). A Depth Camera called D435 RealSense has been used. It has been processed in two stages: The preparation of the dataset, which was freely gathered from the internet, came first. In the second stage, the model is trained and made ready to run in real-time on low-power computers. An augmentation method was used to increase the number of images in the new dataset from (3000) to (6985). The pre-training stage used the YOLOv5s algorithm, which was used for detection and classification; it was trained with a mean average precision (mAP@0.5) reaching (98%) and (mAP@0.5:0.95) being (84%). By utilizing the measuring tube, the distance between the object and the camera at different distances, starting at a maximum of (5) meters and ending at a minimum of (1.5) meters was determined and contrasted with the measurement provided by the camera. Finally, display the result on the screen with a notification voice to the person about the door state. Real-time frame rates for the PC ranged from (6) to (15) FPS, while those for the Jetson Nano were (0.9) to (2) FPS.*

Keywords: *deep learning; object detection; computer vision; distance measuring; depth camera; Jetson nano; real-time; Yolo; visually impaired persons.*

I. INTRODUCTION

In recent years, bad eyesight has become a serious global concern. Some basic complexity that an afflicted person encounters are difficulties with regular movement, recognizing besides detecting nearby objects, also correct indoor and outdoor navigation. Vision issues are more common in developing countries because they cannot afford the latest technologies, which are particularly costly.

The struggle primarily affects elderly persons in highly populated emerging countries [1]. Refractive troubles, glaucoma, retina abnormalities, and age-related eye illnesses are some of the most prevalent causes of vision impairment [2], [3]. In unfamiliar situations, navigation becomes a key worry for these persons. The people utilize services such as awareness campaigns, frequent rehabilitation programs, and social inclusion in addition to medical assistance. Numerous individuals use a white cane, the length of which is determined by the touch feeling. However, its utility is incomplete whereas traveling. Furthermore, its lack of elasticity causes cracks with repeated people frequently utilizing guiding dogs to help them walk. Dogs warn owners of potential

 Received-02-11-2022, Accepted- 01-01-2023

hazards in the path. However, in congested and complex environments, guide dogs might not always provide precise directions. Many people use GPS-enabled devices as aids in their daily lives. To find the desired places, this equipment serves as a navigation and orientation interface. They are good at finding specific areas, but they are not very good at avoiding and locating barriers [4].

Other approaches to obstacle identification, based on quick response codes and bar codes are utilized to distinguish numerous sorts of impediments in crowded situations [5]–[7]. Nevertheless, completing tasks, it required technologically advanced infrastructure and the assistance of a third party. Similarly, several gadgets exist to assist visually impaired users, although many of them use machine vision to detect barriers or sensory modules such as GPS and distance sensors. People with vision abnormalities can be enhanced, and navigation can be made easier and more successful [8], [9].

Effective sensor methodology incorporation with computer vision and image processing can aid in the development of a real-time robust, cost-effective, and dependable model to assist the user with visual deficiencies. As a result, they are made aware of any potential hazard. Deep learning, one of the most cutting-edge current technologies, has made it possible to provide an interactive solution to help a person with visual issues autonomously navigate in any environment. Based on the requirements, the system should be capable to evaluate data rapidly, calculating the distance that has a wide coverage area through superior obstruction identification (both static and dynamic), and operating indoors and outdoors.

II. PROPOSED METHOD

The requirements, architecture, and overall perspective of the system will be clearly illustrated. The main hardware components utilized in the project are Jetson Nano Developer Kit, the Intel RealSense depth camera D435, the screen, the power bank, and the headset.

Numerous strategies are developed from related research to assist the visually impaired, in the proposed system, the first aspect is door recognition, followed by classification into three states (open, semi, and closed), and voice feedback informs the person of the state of the door. The second aspect is determining the distance between the door and the person who suffers from impaired vision. The phases of the process are listed below:

- Prepare datasets.
- The YOLO algorithm for detection and classification employs Convolutional Neural Networks (CNN) and runs in real-time, informing the person about the door's state via headset.
- Distance detection with a camera depth.

2.1. System Requirements

The suggested objective of the system is to help improve the lifestyle of the person who suffers from impaired vision and safety navigation between rooms.

The accuracy and speed of the algorithm as well as voice information regarding the status of the door will be considered. Moreover, the precise distance between the person and the door.

Works with low-power computers and achieves low cost and time. The following are the system's primary goals when the Jetson Nano receives the live video (frame streaming) from the camera depth. Every frame is processed and displayed the result is on the screen. Then informs the person via voice about the door state. Figure (1) illustrate all these stages.

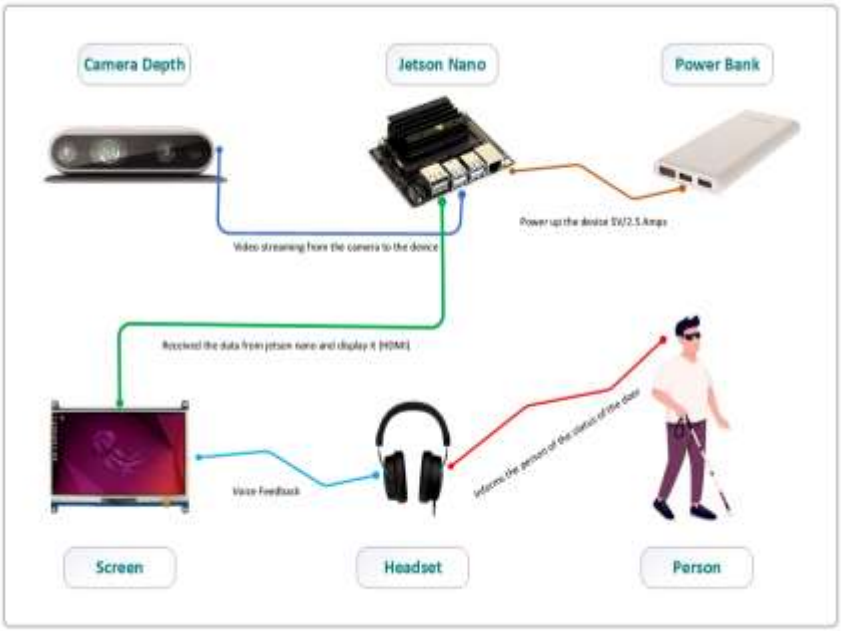

Figure (1): The Proposed System Schematic

The primary hardware of the proposed System equipment required for the proposed system is:

- Jetson Nano Developer Kit[10].
- Depth Camera D435[11].
- Screen with HDMI port.
- Power bank.
- Headset.

2.2. Software Design

The suggested method uses a depth camera to show a person with impaired vision whether a door is open, partially open, or closed. The collection of real-time video streams from a camera has been followed by a number of operations that will be processed frame by frame. In order to achieve a wider viewing angle and better accuracy when estimating the distance between the camera and the door, both frames (RGB and Depth) are first rotated 90 degrees clockwise. Next, the RGB frame is transferred to the detection and classification unit represented by the YOLOv5s algorithm once the depth frame has been extracted. When detecting and classifying the door state with a bounding box, the depth frame aligned with the bounding box and specified the center of the door with the depth frame. To achieve high precision in measuring the distance, the distance is measured from the center of the door to the camera.

The result for the door's status is displayed on the screen along with the bounding box with the percentage of classification and the distance between the camera and the door. Additionally, provide a notice voice describing the door's existence and class to inform the person with an impaired vision of the door's status. The software flowchart for the functional system is shown in Figure (2).

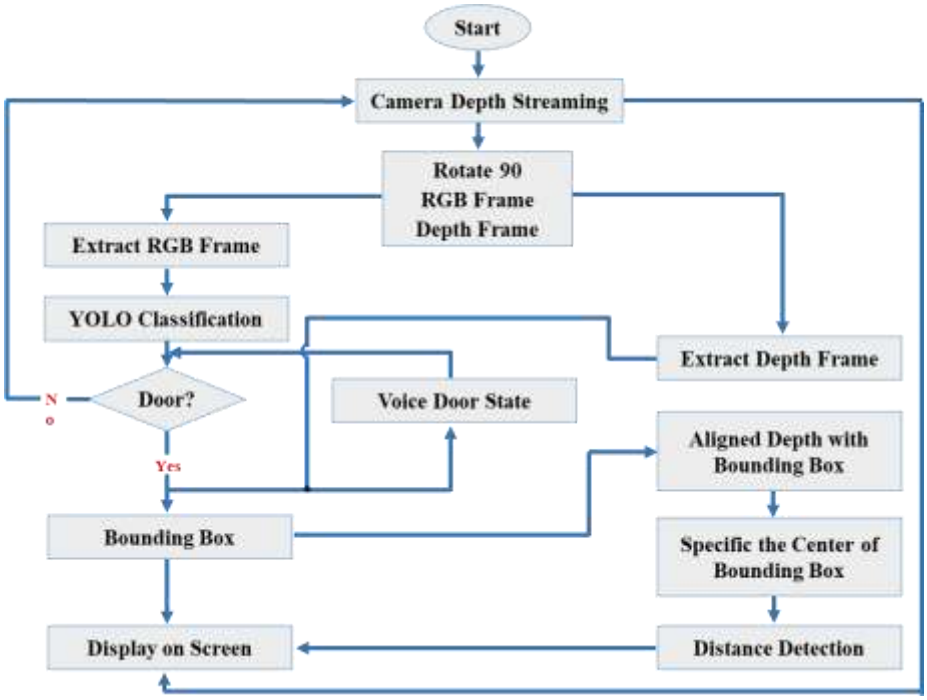

Figure (2): Flowchart of the Implemented System's Software

2.3. Data Preparation

The datasets are a group of data used to train and evaluate object detection systems. A collection of online data was utilized. The images were taken using a camera depth with model number D435. Since the camera's horizontal viewing angle of 86 degrees is greater than its vertical viewing angle of 57 degrees, as shown in figure (3), it was turned 90 degrees to ensure that the images capture the whole door area. The images were taken in public spaces and in within people's homes with multiple doors and different environments with various textures and sizes. Two principles of this dataset: one for door segmentation with 3000 images and the other for door classification with the same number of images. This dataset contains three classes of doors: 1000 images of closed doors, 1000 images of opened doors, and 1000 images of semi-open [12], [13]. Table (1) explain the details and specifications of the datasets.

Figure (3): Datasets Sample [13]

Description	
The sum of datasets	3000 images
The sum of masks	3000 images
Classes	Open door (1000 images)
	Semi-open door (1000 images)
	Closed door (1000 images)
Dimensions image (W x H)	480 x 640 pixels
Images format	PNG

Table (1): Description of Datasets

2.3.1. Data Annotation

For the dataset used in training the Yolo algorithm, another online tool called the MakeSense [14] has been used for labelling these datasets. Initially, all images have been uploaded to the site and labelled one by one, as seen in figure (4). The images labelled by 0 mean closed door, 1 for semi-open door, and 2 for open door. When all the images are finished, the labelled files for each image are exported with the same name (that contains a class number and the coordinates of the door in the image and which is in text format) to the computer for later use.

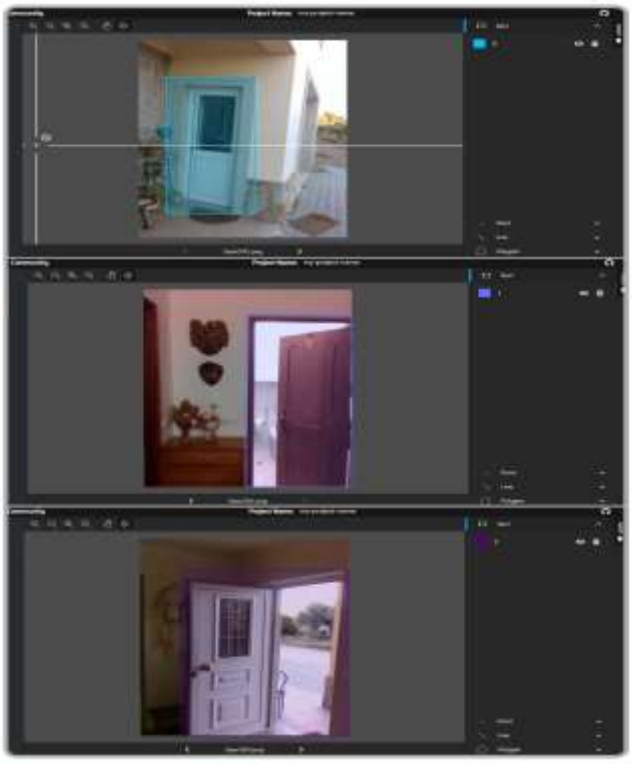

Figure (4): Data Labelling Using the MakeSense Tool [14]

2.3.2. Data augmentation

Data augmentation is a method that may be used to produce altered copies of the dataset's images in order to increase the size of a training dataset fictitiously. Neural network models can be improved by augmenting them

with more datasets, and augmentation techniques can create variations of the images they are trained on. Training deep-learning neural network models on more data can result in more skillful models [15].

Data augmentation for the YOLO algorithm has been used in this work to obtain the highest possible accuracy for the model. So, use the Roboflow appropriate tool to increase the number of datasets and arrangement them. A framework called Roboflow is used to create computer vision apps that enhance approaches for data collection, preparation, and model training. Users can easily upload their own original data and have access to public datasets. Roboflow supports a variety of annotation formats [16].

The following steps explain how to augmentation data by using the Roboflow online tool:

- **Upload:** All dataset images should be uploaded to the Roboflow website with annotations files.
- **Split dataset:** Used Train-Valid-Test technique for the split dataset. Figure (5) shows each group's number and percentage of images [17].

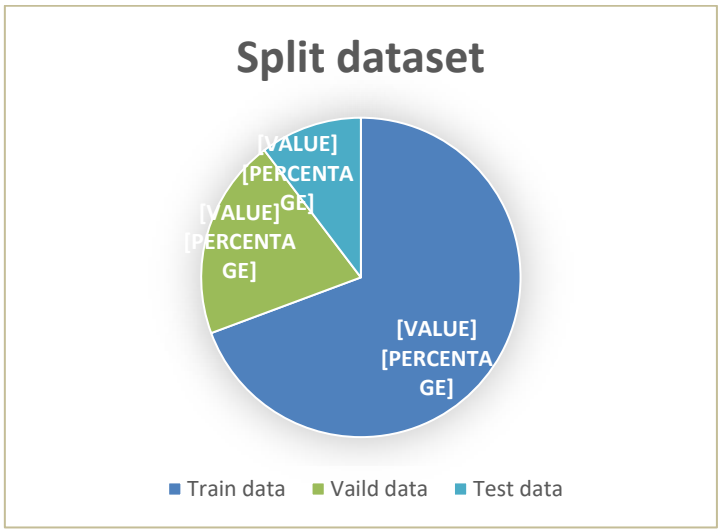

Figure (5): Split Dataset into (Train-Valid-Test)

- **Preprocessing:** Resize all images from 480*640 to 640*640 to deal with the Yolo algorithm that uses this size of images.
- **Augmentation:** Use various processes on the images for augmentation. Table (2) explains these processes.

Process	Description
Flip	Horizontal
Crop	0% min zoom – 20% max zoom
Saturation	Between -25% to +25%
Brightness	Between -25% to +25%
Blur	Up to 10px
Noise	Up to 10% pixels

Table (2): Augmentation Processes

Only use horizontal flip because it does not make sense to train the model on doors with the vertical flip. All processes of augmentation are finished randomly on the dataset. Figure (6) illustrates a sample of images with these processes. It should be noted that the augmentation in the dataset is done only on the training dataset.

Figure (6): Sample of the Dataset after Applying Augmentation

- **Generate:** The images for the new dataset version are being created with 6985 images with a new annotation for each image. The new split dataset is shown in figure (7), and a sample of a new version of the dataset illustrates in figure (8).

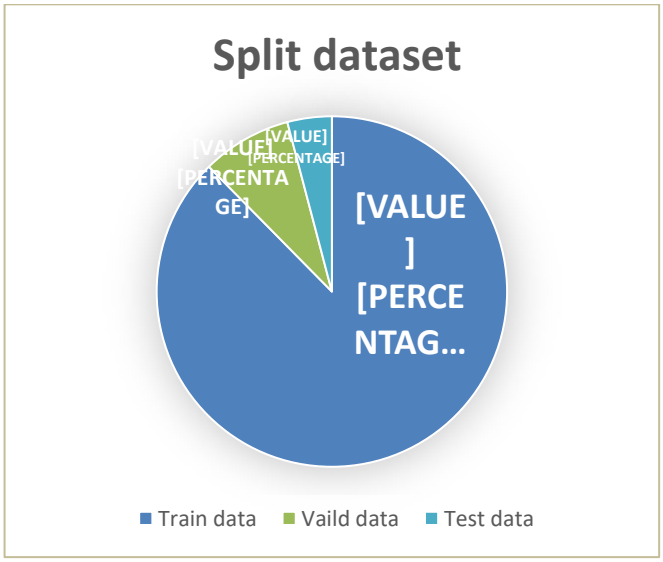

Figure (7): New Split Dataset

Figure (8): New Version of a Dataset

2.4. YOLOv5s Object Detection Training

The same elements, including input, backbone, and neck, are presented in the YOLOv5 network topology as in every previous YOLO series. The pre-training of YOLOv5 was based on the COCO dataset, a significant collection of images used for object detection and segmentation [18] [19].

The YOLOv5 Small version has been used because it is light and compatible with low-power computers such as Jetson Nano or Raspberry Pi. The Google Colab Pro was used to train the model with GPU (Tesla P100-PCIE-16GB) with CUDA support and 16GB memory.

First, installation and cloning of the YOLOv5 repository on Google Colab as illustrated in figure (9).

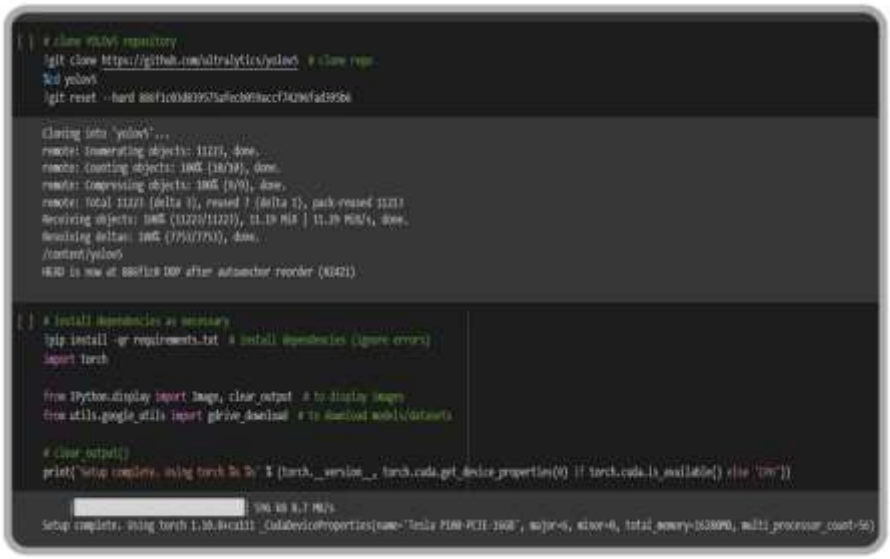

Figure (9): Installation and Cloning of the YOLOv5s Repository

After producing the new version of the dataset, it is uploaded to the Google Colab website by using a code got

from the Roboflow website, figure (10) shows that. The folder path specifies the train, valid, and test data (data.yaml) illustrated in figure (11).

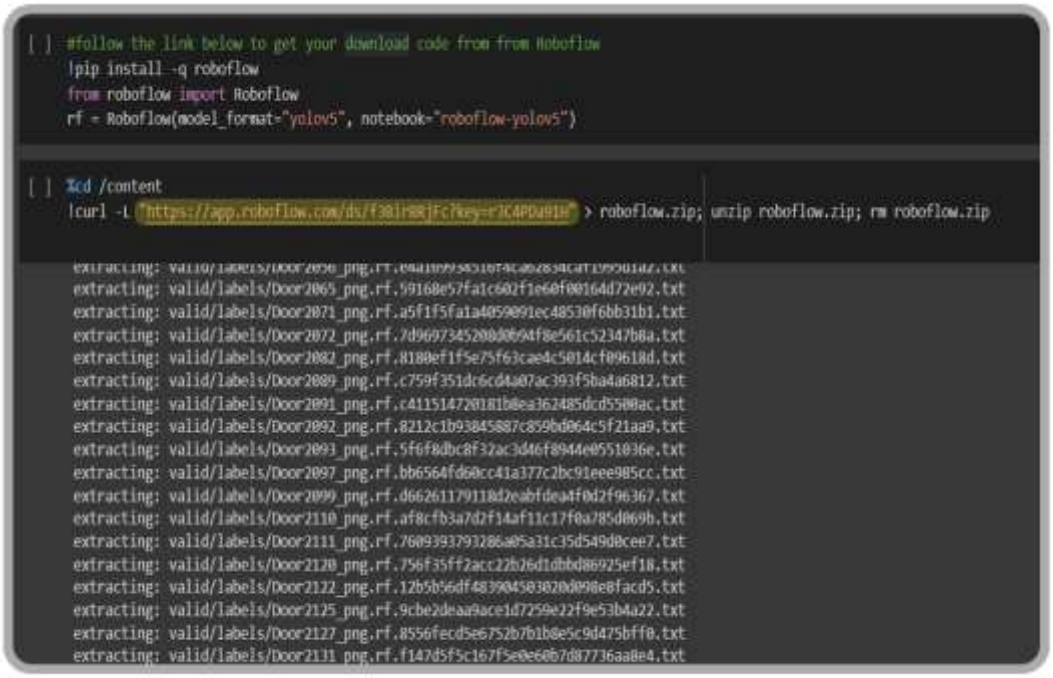

Figure (10): Upload New Version Dataset

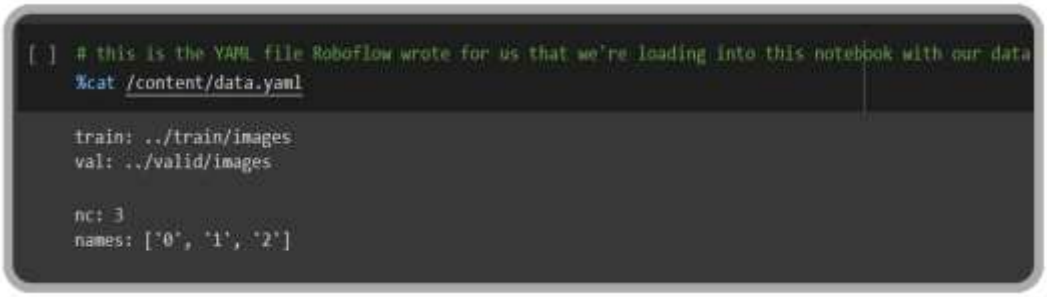

Figure (11): Define Paths of New Version Dataset

The YAML file for YOLOv5s is imported into the Google Colab, which contains the parameters of input, backbone, and neck. Tune the number of classes to three classes (0 = closed, 1 = semi-open, 2 = open door), see figure (12).

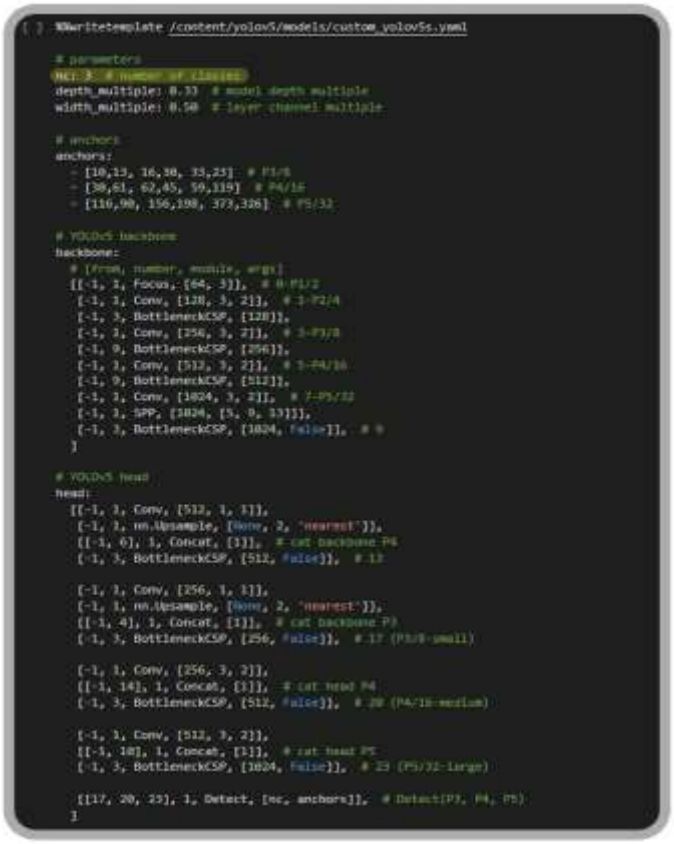

Figure (12): Change the Number of Classes and Save the Model as a Custom Version

Using the command line seen in Figure (13), the model will be trained by compiling file train.py along with its configurable arguments.

Figure (13): Preparation of the Training Process

Table (3) explain the configuration for the training model.

Term	Description	Configure
Img	Size of images	416 *416
Batch	Determine the batch size	16
Epochs	Number of training epochs	100
Data	Path of dataset	/content/data.yaml
Cfg	Specify model configuration	/models/custom_yolov5s.yaml
Weights	Provide a path for the weights	left empty (random weights)
Name	Name of the result folder	yolov5s_results
Cache	Cache images for training faster	

Table (3): Set Parameters of YOLOv5s Model for Training

Figure (14) illustrates the ground truth augmented training data to ensure that the label (annotations) matches the augmented dataset.

Figure (14): Ground Truth Augmented Training Data

2.5. Distance Measuring

Get the (RGB and Depth) frames, as shown in figure (15), Depth frame is grayscale using the infrared projector (z-axis in the image), through it and with the help of pyrealsense2 library can predict the distance in mm that divide by 1000 to convert to m [20].

Figure (15): The Difference between RGB and Depth Frame

III. RESULT and DISCUSSION

1.1. Training and Testing Results for YOLO5s Training Process

Images in various circumstances has been used because it is important to have a wide range of data to create a robust model. Open doors, semi-open doors, and closed doors in various environments.

Google Colab Pro, which offers access to strong GPUs with no configuration needed was used to train the model, the pre-trained COCO weights have been used. The door dataset has been added and the number of epochs has been adjusted to be trained in order to detect the classes. The model was trained on 6120 images of the new version dataset in 3 different classes (open door, semi open, and closed door) for 100 epochs takes about 1.44 hours (87 minutes).

For 1 epoch, the average time to perform the training process on 381 batches (6120 / 16 batch size) was 54 seconds and for evaluation on 19 batches (batches = batch_size // world_size * 2) where 3 seconds as explain in figure (16). The entire execution time for 100 epochs on the dataset comprising 6699 images was 1 hour, 27 minutes, and 14 seconds and the mAP of the last epoch is 98.5%.

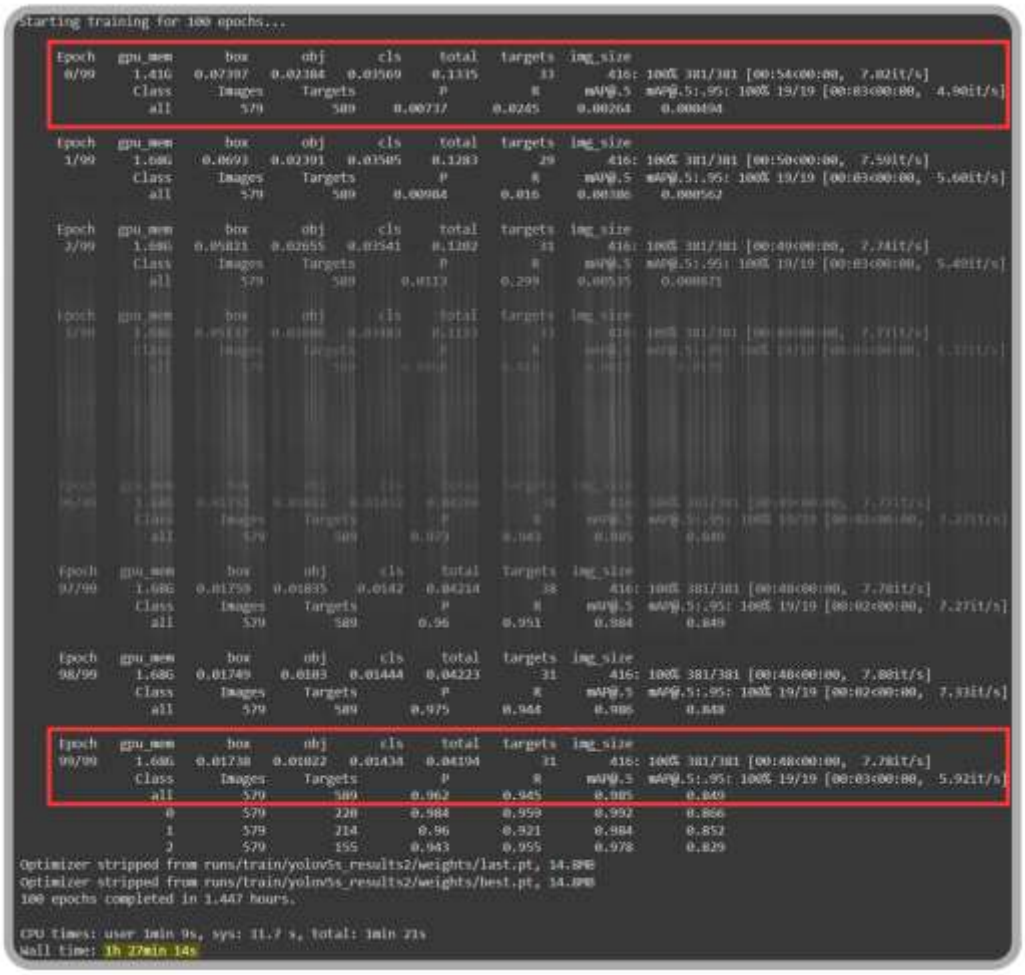

Figure (16): Training Development During 100 Epochs

Figure (17) shows the various performance measures for both the training and validation sets, and how the model has been improved.

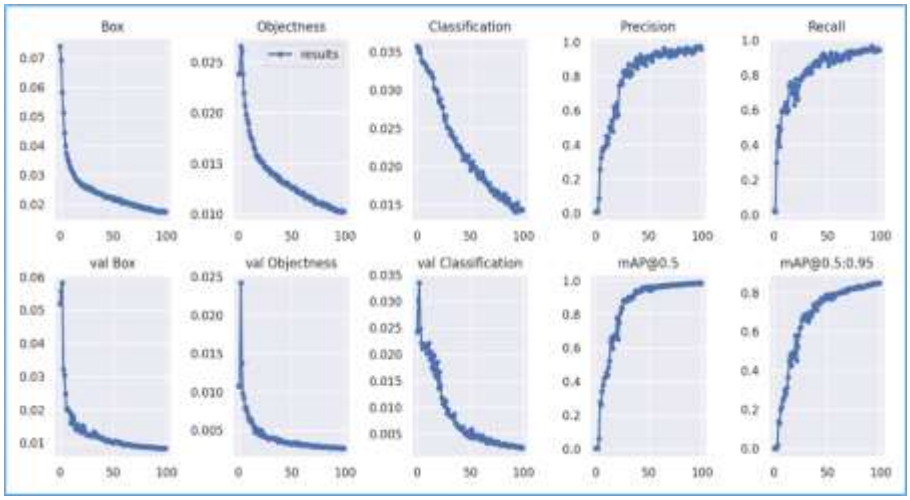

Figure (17):): Graph Showing the Evolution of YOLOv5 Training's Precision, Recall, and mAP

Figure (17) displays three various types of box loss, loss, objectness loss and classification loss. The box loss evaluates the algorithm's precision to determine an object's center and how completely the predicted bounding box encloses an object. The possibility that objects existent in a suggested zone of interest is basically measured by objectness. If the objectivity is high, an item is probably present in the image window. How successfully the classification loss indicates that the algorithm can determine the precise class of a given object. Predictions have been made for the new and untested images in the test dataset after training the model.
Examples in Figure (18) demonstrate how the algorithm may determine the status of the door with a high degree of certainty.

Figure (18): Images From the Test Dataset Displaying the Detection Performance for the Three Classes Open Door, Semi Open, and Close

The three examples of the door are depicted in the figure above, the doors that are marked with a yellow rectangle to indicate that they are in closed state (the condition that was previously determined to be specific

to the closed door state class and the classification accuracy values, respectively, is indicated by the number 0 and the values of 0.94 and 0.95 at the top left of the yellow rectangle), the doors that have a blue rectangle around them, indicating that they are in semi open state (the condition that was previously determined to be specific to the semi open door state class and the classification accuracy values , respectively, is indicated by the number 1 and the values of 0.89, 0.93, and 0.95 at the top left of the blue rectangle), and the doors that are marked with a purple rectangle to indicate that these doors are in open state (the condition that was previously determined to be specific to the open door state class and the classification accuracy values, respectively, is indicated by the number 2 and the values of 0.83, 0.93 and 0.94 at the top left of the purple rectangle). Table (4) below shows each door status's classification number.

Class number	Door status
0	Closed door
1	Semi open
2	Open door

Table (4): Summary of each Door's Categorization Number

The precision, Recall, and mAP values of the classification process for each case of the door's state are displayed in the table (5) below, and they indicate the high degree of precision.

For the state class of an open door, the accuracy was equal-to 0.96, recall was 0.95, and mAP was scored to 0.99, and for the state class of a semi-open door, the precision was equal to 0.96, recall was achieved 0.92, and mAP was 0.98, finally for the closed-door condition the precision, Recall and mAP values were 0.94, 0.95, and 0.97 respectively. The accuracy of all classes was 0.96 with mAP 0.98.

Classes	Precision	Recall	mAP@0.5	mAP@0.5:0.95
All classes	0.96	0.94	0.98	0.84
Closed Door	0.98	0.95	0.99	0.86
Semi-Open Door	0.96	0.92	0.98	0.85
Open Door	0.94	0.95	0.97	0.82

Table (5): Precision, Recall, and mAp of each Door's Categorization

1.2. Camera Depth Accuracy Rating

The camera's depth can be used to calculate the distance between the door and the camera. Realistic detection techniques were used to determine the actual distance between the door and the camera, and the findings have been compared to the expected value. The samples and the camera were both positioned 1.5 meters above the ground. The prediction error rate increases when the camera is moved farther away from a limit and decreases as we go closer.

As shown in table (6), the percent error was 6% when the true distance was 5 meters and the predicted distance was 5.3 meters, the ratio was 7.5% at a distance of 4 meters. The mistake rate decreases as distance decreases. The percent error starts to increase as the camera get closer to the door, rising to 13% at 1m. Figure (19) illustrates the comparison of the real and predicted values for the distance between the door and the camera.

The RMSE was found to be (0.10997). Besides, the SSE is evaluated to determine performance. The SSE value is brought into (0.03628) and the R-square is extremely 1 which leads to an efficiency of performance.

Truth distance	Predicted distance	Error percentage
5 m	5.3 m	6%
4 m	4.3 m	7.5%
3 m	3.1 m	3%
2 m	2 m	0%
1.5 m	1.7 m	13%

Table (6): Comparing the True Value to the Expected Value for the Distance Between the Camera and the Door

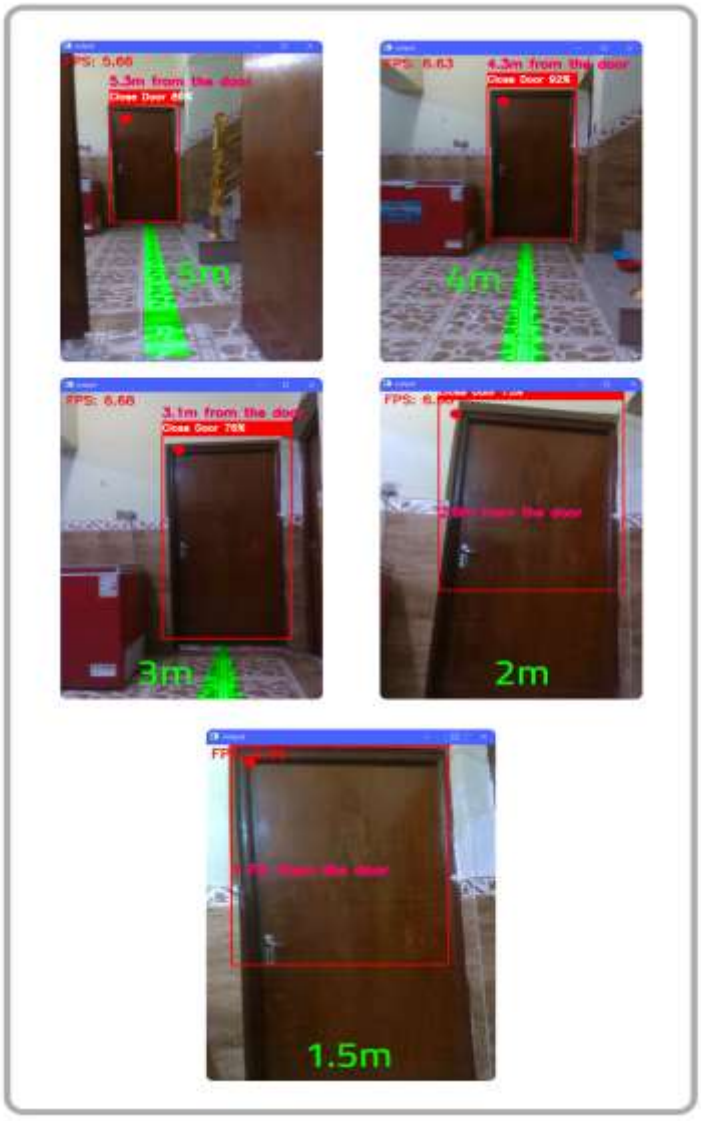

Figure (19): Evaluating the Precision of the Camera's Depth

1.3. System Implementation

A system architecture has been created for the door status detection in real time system, which consists of Jetson Nano Developer Kit, Depth Camera D435, Screen with HDMI port, and Power bank. Logical components and their interactions can be seen in figure (20) and (21). The necessary software libraries (OpenCV for reading and image processing, Pyrealsense2 library for dealing with the depth of the camera D435 is feasible, and PyTorch library for training the models in order to speed the implementation of research prototypes) and tools

are obtained.

The Jetson Nano Developer Kit HDMI cable was used to attach it to a display that supports HDMI, Headset, and the Depth Camera D435ra has been connected to the Jetson Nano Developer Kit through USB port. The Jetson Nano Developer Kit was then programmed using a keyboard and mouse that were connected to it via USB ports.

The Jetson Nano Developer Kit has been utilized as the system's processor, since it needs a 5V supply, it has been powered by the Power bank.

Figure (20): A Logical Overview of the Components Suggested System

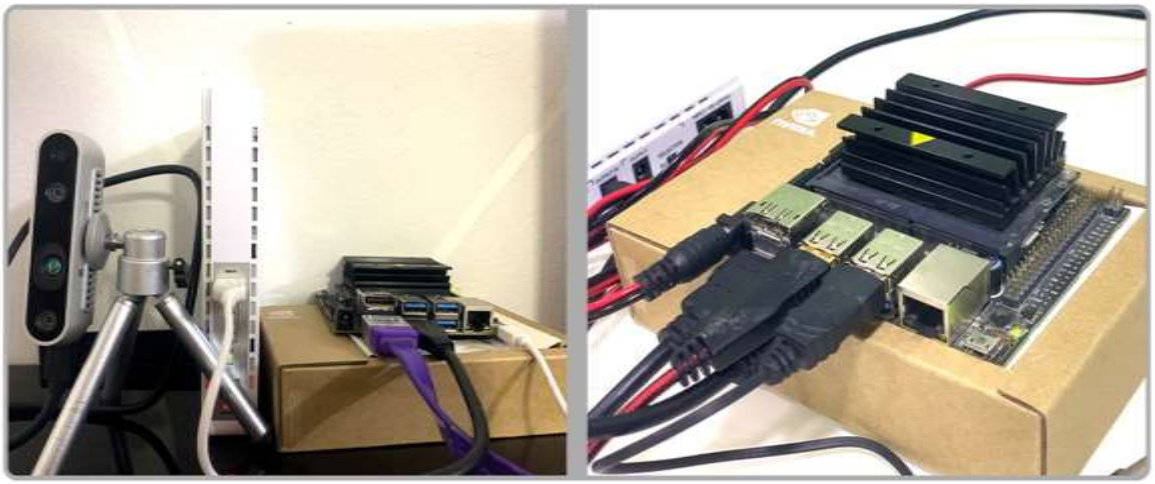

Figure (21): Jetson Nano with the Camera after Building for the System

1.4. System Testing in Real-World

YOLO has been connected to the Depth Camera and check to see if it preserves real-time performance, including how long it takes to display detections and retrieve images from the camera. Loading of "best.onnx " [21] file is done (the best checkpoints of the model are frozen once the training process is complete and saved under the name of "best.onnx "for use in the real-time detection process). For detection of door's state in real-time world, the video is captured in real-time from the Depth Camera that is connected to the Jetson Nano Developer Kit via USB ports, which will be processed by YOLO process as an individual frames. The camera continuously scans the area surrounding the blind individuals as they move, once a door is detected a real-time

audio feedback is provided to visually impaired users through a headset by the interaction system.
The visually impaired individual receives an audio message through the earpiece " opened door " when an open door is detected, alerting them to the situation, figure (22) shows the open-door state detection.
However, if the door is discovered to be in a semi open state, an alert audio message " semi-open door " is transmitted through the headset to the blind individual, and a voice message of " closed door" when the closed door's state is detecting.

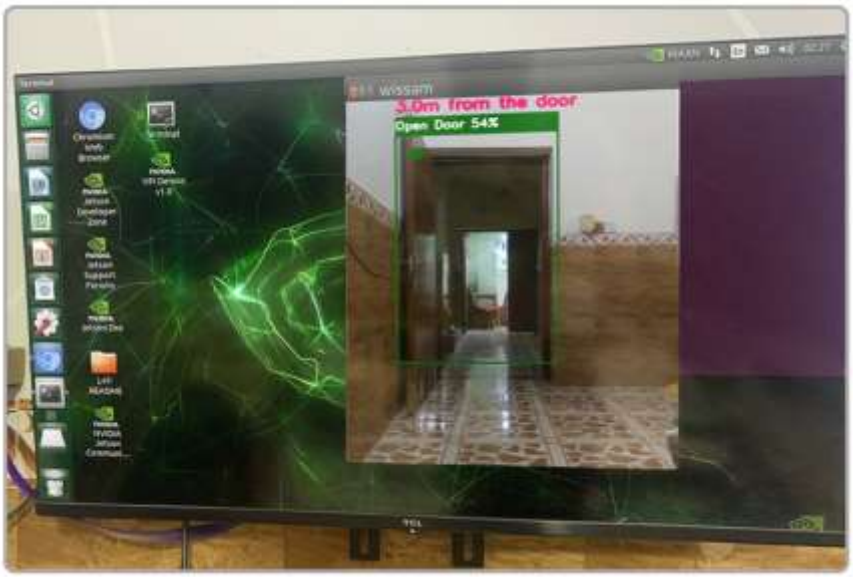

Figure (22): The Open Door State Detection by Suggested System

As seen in the figure above, a green rectangle around the door that the system has determined to be open is displayed on the HD screen that is attached to the Jetson Nano Developer Kit via the HDMI cable. The green rectangle is denoted in the python code as [0, 255, 0]. The number 54%, which indicates how accurately the system can assess the state of the door, which decreases as the distance between the door and the system camera gets longer and increases as this distance decreases. The 3m indicates how far the distance is between the door that was detected by the system and the camera.
Figure (23) below shows the second state of the door state detection which is the semi open state, the detection door is surrounded with a blue rectangle that is denoted in the python code as [0, 0, 255]. The percentage of 75% shows the system's ability to determine the condition of the door with increasing precision as the distance between the door and the system camera reduces and the value of 3.5 m indicates how much space there is between the camera and the door that the system recognized.

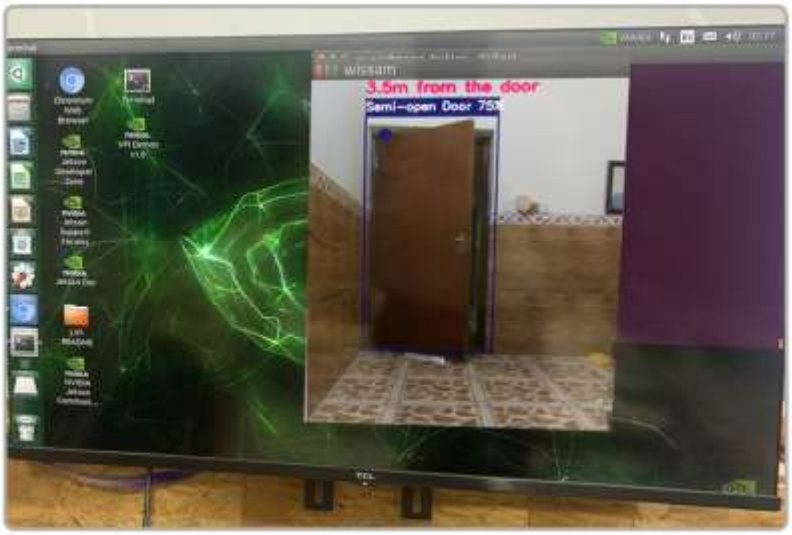

Figure (23): The Semi Open Door State Detection by Suggested System

Figure (24) illustrates how the suggested system determined that the door was in the closed state and drawing a red rectangle around it which is represented as [255, 0, 0] in the Python code.
The 54% shows how accurately the suggested system can identify the state of the closed door with the 3.2m distance between the camera and the system.

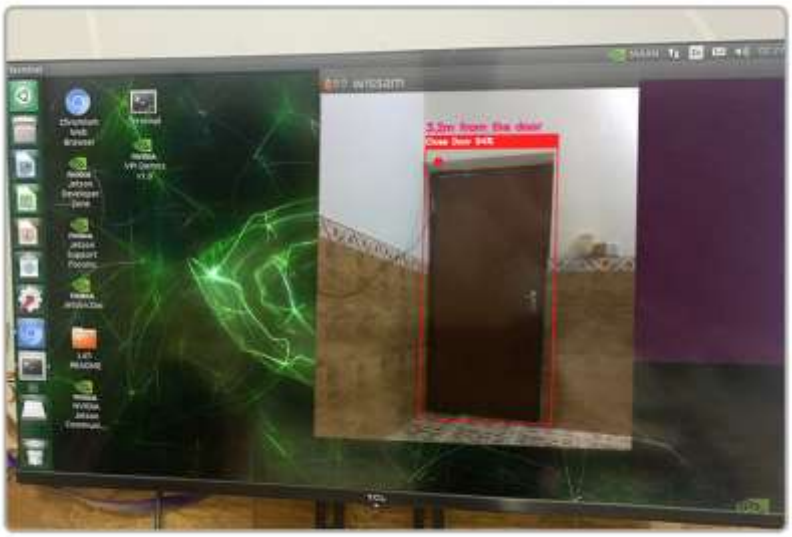

Figure (24): The Closed Door State Detection by Proposed System

III. CONCLUSION

In this work, the implementation of the development of an intelligent model for visually impaired person guidance using the Jetson Nano developer kit has been accomplished. By proposed door detection and classification to three class (open, semi-open, and closed) prototype implemented in real-time in indoor and outdoor environments. That makes it easier for those with visual impairments to navigate their surroundings safely. The suggested system was implemented on a personal computer, deployed on low-power computers such as a Jetson Nano, and equipped with a Realsense Camera Depth D435 camera. This camera was specifically used to predict distance by the depth feature provided by this camera.

The main objectives of the proposed system are based on two phases: the first was the preparation of

the dataset, which was collected from the internet freely. The second phase is training and preparing the model to operate in real-time on low-power computers.

An augmentation process was applied, and the new version of the dataset was increased from (3000) to (6985) images.

The pre-training YOLOv5s algorithm was used for detection and classification; it was trained with the new version dataset, with a mean average precision (mAP@0.5) reaching 98% and (mAP@0.5:0.95) being 84%. The algorithm was implemented in real-time with distance detection by using the depth of the camera with RMSE (0.10997), and the result was shown on the screen with a voice notification to the person about the status of the door.

In real-time, the PC achieved a frame rate of 6 - 15 FPS (frame per second), while the Jetson Nano achieved 0.9 - 2 FPS.

IV. REFERENCES

1. W. H. Organization, *World report on vision*. World Health Organization, 2019.
2. S. R. Flaxman *et al.*, “Global causes of blindness and distance vision impairment 1990–2020: a systematic review and meta-analysis,” *Lancet Glob Health*, vol. 5, no. 12, pp. e1221–e1234, 2017, doi: https://doi.org/10.1016/S2214-109X(17)30393-5. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2214109X17303935.
3. L. Thaler, S. Arnott, and M. Goodale, “Neural Correlates of Natural Human Echolocation in Early and Late Blind Echolocation Experts,” *PLoS One*, vol. 6, p. e20162, Sep. 2011, doi: 10.1371/journal.pone.0020162.
4. D. López-de-Ipiña, T. Lorido, and U. López, “BlindShopping: Enabling Accessible Shopping for Visually Impaired People through Mobile Technologies,” in *ICOST*, 2011.
5. S. Dambhare, “Smart stick for Blind: Obstacle Detection, Artificial vision and Real-time assistance via GPS,” 2011.
6. Z. O. Abu-Faraj, E. Jabbour, P. Ibrahim, and A. Ghaoui, “Design and development of a prototype rehabilitative shoes and spectacles for the blind,” in *2012 5th International Conference on BioMedical Engineering and Informatics*, 2012, pp. 795–799, doi: 10.1109/BMEI.2012.6513135.
7. P. Wawrzyniak and P. Korbel, “Wireless indoor positioning system for the visually impaired,” in *2013 Federated Conference on Computer Science and Information Systems*, 2013, pp. 907–910.
8. H. and M. S. Ray Camellia and Kumar Tripathy, “Assessment of Autistic Disorder Using Machine Learning Approach,” in *Intelligent Computing and Communication*, 2020, pp. 209–219.
9. H. K. Tripathy, P. K. Mallick, and S. Mishra, “Application and evaluation of classification model to detect autistic spectrum disorders in children,” *International Journal of Computer Applications in Technology*, vol. 65, no. 4, pp. 368–377, 2021, doi: 10.1504/IJCAT.2021.117286. [Online]. Available: https://www.inderscienceonline.com/doi/abs/10.1504/IJCAT.2021.117286.
10. “Jetson Nano Developer Kit NVIDIA Developer.” Mar. 2019.
11. “https://www.intelrealsense.com/depth-camera-d435.” .
12. opencv, “GitHub - opencv/cvat: Annotate better with CVAT, the industry-leading data engine for machine learning. Used and trusted by teams at any scale, for data of any scale.” Sep. 2022 [Online]. Available: https://github.com/opencv/cvat.
13. gasparramoa, “GitHub - gasparramoa/DeepDoors2: DeepDoors2 is a dataset for 2D/3D door classification, 2D door detection and 2D door segmentation.” 2021 [Online]. Available: https://github.com/gasparramoa/DeepDoors2.
14. “Make Sense.” [Online]. Available: https://www.makesense.ai/.
15. Jason Brownlee, “How to Configure Image Data Augmentation in Keras,” 2019. [Online]. Available: https://machinelearningmastery.com/how-to-configure-image-data-augmentation-when-training-deep-learning-neural-networks/.

16. “Roboflow: Give your software the power to see objects in images and video.” [Online]. Available: https://roboflow.com/.
17. S. Agrawal, “How to split data into three sets (train, validation, and test) And why? by Samarth Agrawal Towards Data Science.” May 2021 [Online]. Available: https://towardsdatascience.com/how-to-split-data-into-three-sets-train-validation-and-test-and-why-e50d22d3e54c.
18. “YOLOv5 Documentation.” [Online]. Available: https://docs.ultralytics.com/.
19. J. Yao, J. Qi, J. Zhang, H. Shao, J. Yang, and X. Li, “A Real-Time Detection Algorithm for Kiwifruit Defects Based on YOLOv5,” *Electronics (Basel)*, vol. 10, no. 14, 2021, doi: 10.3390/electronics10141711. [Online]. Available: https://www.mdpi.com/2079-9292/10/14/1711.
20. A. Grunnet-Jepsen, J. N. Sweetser, and J. Woodfill, “Best-known-methods for tuning intel® realsense™ d400 depth cameras for best performance,” *Intel Corporation: Satan Clara, CA, USA*, vol. 1, 2018.
21. onnx, “GitHub - onnx/onnxmltools: ONNXMLTools enables conversion of models to ONNX.” Sep. 2022.