Prototype of an Indoor Pathfinding Application with Obstacle Detection for the Visually Impaired

Ken Gorro, Lawrence Roble, Mike Albert Magana, Rey Paolo Buot,
Louis Severino Romano, Herbert Cando, Bonifacio Amper, Rhyan Jay Signe, Elmo Ranolo
Department of Industrial Technology-College of Technology, Cebu Technological University, Carmen Campus, Philippines

***Abstract*—This study presents an initial prototype for a project aimed at assisting visually impaired individuals using deep learning techniques. The proposed system utilizes the You Only Look Once (YOLOv8) algorithm to detect objects tagged as obstacles. Designed for indoor environments, the system employs a CCTV camera and a computer server running the YOLOv8 model. Additionally, the A-star algorithm is used to determine the optimal path to avoid detected obstacles. Video frames are divided into tiles, each considered a node; nodes with detected objects are marked with a value of 0. The YOLOv8 model currently achieves an initial accuracy rate of 70%, with a mean Average Precision (mAP) at an Intersection over Union (IoU) threshold of 0.5 reaching 0.993 across all classes. This high mAP indicates an exceptional balance between precision and recall, signifying the model's effectiveness in object detection. Furthermore, the model yields an impressive F1-score of 0.99 at a confidence threshold of 0.624, demonstrating a robust balance between precision and recall, which is crucial for minimizing both false positives and false negatives. This prototype being developed assumes that a destination can be set by an operator of the system using the server that connects to the CCTV camera. The system was tested in enclosed environments and was able to provide a path that potentially avoids obstacles. The development of audio commands to guide visually impaired users is ongoing. These audio commands depend on identifying the direction an individual is going, requiring an additional deep-learning model to generate accurate instructions.**

Keywords*—Yolov8; A-star algorithm; pathfinding; deep learning

I. Introduction

Equal chances, particularly for the disabled, are vital in today's culture because technology is used by everyone and advancement is happening at a faster pace. For example, one of the groups most at risk of facing various challenges is the visually impaired. Finding our way about inside buildings may be quite difficult and dangerous, particularly if we are unfamiliar with the surroundings. More specifically, this significant topic has been taken into consideration in an effort to improve the independence and safety of visually impaired individuals through the use of modern technology, namely through the development of the in-door pathfinding program with the capability of obstacle detection.

The main goal of this study is to restore the independence and freedom of visually impaired individuals. The visually impaired are genuinely disabled in several ways, including but not limited to the fact that indoor spaces are primarily laden with barriers. They are supported by friends, family, or both, and they need traditional walking aids like canes and guide dogs. They can move around independently with the assistance of an improved navigational aid that can identify impediments, which will boost their efficiency and sense of self. Additionally, safety needs to be improved. In the midst of all of these considerations, improving safety is also crucial. It becomes clear that there are many risks and difficulties in any indoor setting, such as an office building, hospital, airport, or retail center. Occasionally, traditional mobility aids are unable to provide adequate or timely information about these risks. The application will be safe for visually impaired people to use if it can detect impediments in real-time, thereby reducing or eliminating the hazards related to falls and blind guiding.

This research is in phase 1 of creating a total pathfinding mobility application for the visually impaired in indoor scenarios. The result of this research only covers pathfinding and obstacle avoidance. The guiding voice is still in the development phase which will be the next phase of this study. The main focus is on the current obstacle detection and generating the best path to avoid obstacles using yolov8 and graph theory algorithms.

II. Related Work

Despite global efforts in eye care, over 2.2 billion people suffer from vision impairment, with at least 1 billion cases preventable or unaddressed. The burden is higher in low-income countries, older populations, and disadvantaged communities. Initiatives like "Vision 2020" have contributed to improvements, but challenges remain, including limited service coverage, workforce shortages, and health system fragmentation. With aging populations and lifestyle changes, vision impairment is on the rise. The World Report on Vision advocates for Integrated People-Centered Eye Care (IPEC) to provide comprehensive, universal eye care and well-being [10].

Enhancing mobility and independence for visually impaired individuals relies heavily on advanced technologies. These innovations use artificial intelligence, machine learning, and robotics to navigate complex environments safely. By combining sensory data from cameras, LiDAR, and ultrasonic sensors, systems can identify and bypass obstacles instantly. Recent studies emphasize the development of reliable methods that enhance user experience and accessibility. This research is promising for creating assistive technologies that provide greater autonomy and safety for visually impaired individuals. To begin with, planning a path is an important part of any navigation framework. Since the 1970s, the issue of route planning has been a fascinating subject of study, particularly for

robotics enthusiasts [2]. In order to save time, effort, and resources, it seeks to determine the best path between two sites. To guarantee user safety, additional design limitations must be added when incorporating these robotics-derived techniques into navigation systems. Furthermore, given that the user's tastes and wants may differ due to underlying path features and the topography of the environment, an optimal path for the VI is frequently not just the shortest path. In order to discover an optimal route while taking the preferences of the VI user into account and minimizing collisions with obstacles, this work presents a path-planning algorithm that is based on Ant Colony Optimization (ACO). Every stage has taken into account human issues, especially the unique requirements of the VI user [2]. People can manage and process spatial information about where they are physically by using cognitive mapping. In order to successfully complete tasks related to navigation and direction in urban environments, it is essential to develop a conceptual picture of the surrounding space and nearby objects (Fernandes et al., 2017) [1]. Because of obstructions on sidewalks and roadways, blind pedestrians frequently find it difficult or even dangerous to navigate in urban environments, despite their skill at compensating for lost visual information through increased awareness of environmental cues and navigational aids (Yang et al., 2011) [6]. Although blind pedestrians can navigate more safely and effectively thanks to audible traffic signals and landmarks (Weyrer et al., 2013), research on the needs, preferences, and experiences of blind pedestrians during navigation (e.g. Shangguan et al., 2014) and navigation solutions tailored specifically for them are still lacking [4][5]. The system described in the study by Lee and Medioni includes a glass-mounted RGBD camera with inertial measurement unit (IMU) sensors, a vest-type interface with four vibration motors, and a laptop that the user carries in the back and runs the program in real time. It uses visual odometry to estimate the camera location; it uses normal vectors and random sample consensus (RANSAC) to segment the floor; it generates a 2D traversable map and a 3D voxel map; and it directs the user to things of interest in dynamic interior environments. Using the D*Lite technique, an ideal path is calculated in the 2D map. The user receives haptic and audio input while using a smartphone application to select the location [7]. The Google Tango software, which operates on a hybrid phone/tablet Lenovo Phab2 with an integrated 3D depth sensor, gyroscope, and accelerometers, is the foundation of the CCNY smart cane. This device is placed on the user's chest. It can remember important visual elements like doors and rooms (stored in an area description file, or ADF), perform simultaneous localization and mapping (SLAM), and use infrared sensors to locate things. A visually impaired person can be guided to objects of interest in indoor spaces by the system, which uses the A* search algorithm to plot a path and provides both haptic (two vibrating motors installed on the white cane) and audible feedback. The iterative end-point fit algorithm is used to provide easier guidance, giving the user information [3][8][9]. D. Ni et al. [30] developed an assisted walking robot system that relied on computer vision and tactile sense. The system is made up of a rollator framework that offers stable physical support, a Kinect gadget that acts as a blind person's eyesight to record environmental information, and ultrasonic sensors that identify symmetry roads. A wearable vibrotactile belt is intended to provide blind people with information through vibration patterns. For the safe direction, a feature extractor approach based on depth image compression is used [15]. On the other hand, H-C Wang et al. (2021) advanced wearable devices for blind and visually impaired (BVI) individuals by integrating computer vision and haptic feedback to improve navigation and situational awareness. Their study showed how depth cameras and real-time processing help detect objects and obstacles, allowing safer navigation. The design is smaller and less intrusive, addressing previous issues with bulk and audio cues. Ongoing research aims to further develop this intuitive haptic technology for safer, independent mobility for BVI users [17].

According to a recent in-depth analysis of robot path planning in dynamic environments, reactive approaches such as Ant Colony Optimization (ACO) are quickly gaining traction in the field of mobile robot navigation because they perform better than classical approaches in real-time navigation scenarios (Cai et al., 2019) [11].

Kumar et al. (Kumar et al., 2020) recently improved the pheromone evaporation rate in order to address the issue of sluggish convergence of ACO. They suggested a fuzzified ACO (FACO) for mobile robots, in which the path pheromone updating process takes into account both favorable and unfavorable paths. Through simulation, the effectiveness of the FACO method was evaluated, demonstrating that it resolves the problem of delayed convergence [12].

Shiguo et al. (Li et al., 2020) tackled the problem of robot path planning by taking into consideration not only the path length but also the number of turns. They used the A* algorithm to enhance the convergence speed, and turning points were considered in the pheromone concentration. The simulation results assessed the performance of the improved ACO in terms of convergence, the number of iterations, path length, and the number of turns [13].

In order to address the problems of poor convergence speed, trapping into local optimum, and amount of turns, Ma and Mei (Ma and Mei, 2020) suggested an enhanced ACO path planning for mobile robots. To get rid of the less promising nodes, they employed a Jump Point Search (JPS) technique. Both simulation and real-world mobile robot navigation experiments were used to evaluate the algorithm's performance, demonstrating the enhanced algorithm's efficacy and efficiency in resolving path planning for mobile robots [14].

Broersen et al. (2016) developed a basic pathfinding method based on the A* algorithm. The path was calculated via the octree's vacant space. In the pathfinding procedure, two nodes that had the same face were regarded as neighbors. There might be equal-sized, bigger, or smaller neighbors. The neighbors are determined dynamically, and object avoidance is not used [16].

Tapu et al. developed a DEEP-SEE system that detects both dynamic and static items utilizing the You-Only-Look-Once (YOLO) object identification approach. The obstacle category and distance data may be obtained from the system. However, because of its high computing cost, real-time detection on mobile devices is challenging to achieve with this neural network [18]. Some lightweight image-based neural networks, such as ShuffleNet, YOLO-LITE, and MobileNet have

suggested making real-time object identification more accessible to mobile devices. However, while recognizing objects, these lightweight algorithms do not explicitly take distance into account. Therefore, it is possible that the painted item on the ground is misleading to vision-challenged individuals [19][20][21].

Mortari et al. proposed a network creation technique that takes into consideration obstacles in indoor situations. Obstacles were represented as 2D geometry on the floor plane in the prepared models that served as the basis for the approach. Because 2D-floor layouts were abstracted at various height levels, the end product was a 3D network [22]. An approach for 3D indoor path-planning using semantic 3D models encoded in LoD4 CityGML was introduced by Xiong et al. [23]. While the system took barriers into account, tests were conducted on models without obstacles. A true interior navigation approach based on grid models—obtained from 2D-floor layouts with specified obstacles—was developed by Lui et al. [24]. An octree representation of indoor point clouds serves as the foundation for the indoor pathfinding approach that Rodenberg [25] suggested. Because the A* the pathfinding algorithm followed empty nodes and avoided obstructions, it was reliant on the use of heuristics to direct the search. Additionally, Li et al. [26] recently introduced a path-planning technique for drones operating inside that was based on occupancy voxel maps and on which the vacant voxels made up the navigable space.

Tsirpas et al. presented an intriguing Radio Frequency Identification (RFID)--based indoor navigation system for the elderly and visually handicapped. RFID tags have a limited range, which is the primary drawback of systems based on RFID technology (the authors of the research recommend 40 × 40 cm cells). Moreover, considering that the human body may resist radiofrequency signals, its range might be lowered. Another disadvantage is that installing RFID in large spaces may be costly because the tags frequently need to be inserted into the floors, walls, furniture, and other surfaces [27].

Khelifi et al. also studied the utilization of radio and Wi-Fi technologies. The writers went beyond specifics in their discussion of the subject, emphasizing the necessity for consideration of matters like processing costs, implementation costs, and energy efficiency. Grouping relevant works that supported the study according to methodologies and technologies helps the reader grasp the material better and become more objective when reading [28].

Mainetti et al. aimed to cover the key IPS technologies and approaches for locating, mapping, tracking, and navigating people, objects, and animals inside. The primary technologies employed in IPS, according to the authors, are computer vision (mobile and stationary cameras), infrared, ultrasound, Wi-Fi, RFID, and Bluetooth. The primary methods which include those that make use of FM radio and ZigBee technology are thoroughly explained, while the remaining methods are categorized under the heading "other technologies". The authors utilized a model that took accuracy, coverage, cost, complexity, and usual implementation locations into account while presenting the findings of the works that they used as references in the form of graphs and tables [29].

Guerrero proposed a micro-navigation system that tracks the user's position and movements with the use of an infrared camera. Using a tree structure of the room including the relevant data, based on Extensible Markup Language (XML), the system accomplishes static obstacle detection. However, the system's simulation and testing showed that as more people travel through the area, the system performs worse. PERCEPT is an interior navigation system that uses passive RFID tags for location and is based on kiosks. Using the shortest path method, it is a macro-navigation system that lets VI persons move from one room to another on multiple floors. Unfortunately, the technology that was described is unable to identify any environmental constraints that are nearby. The author also suggested that the system be enhanced by offering guidance information depending on user choices and steps [30][31].

Using RFID technology, the authors of Ivanov (2010) created an interior navigation system for the blind. By placing passive RFID tags above and below door handles, the technology makes door-to-door travel easier. This is based on the idea that blind people can identify doors with the use of their white canes. Navigational instructions are stored on tags above the door handles, while adaptive multi-rate (AMR) audio data is stored on tags below the handles. Every door serves as a point of reference for the following location. Each RFID tag has its navigation instructions manually entered into it. The blind user reads these tags with a cell phone that is equipped with RFID. A text-to-speech engine and the mobile phone's keyboard are used for bi-directional communication between the user and the device. The system mostly depends on the user's ability to navigate obstacles with their white canes. The published work did not address the correctness of the system; instead, it just examined the navigation time in its entirety [32].

III. Methodology and Results

Fig. 1 is the system architecture of the system, the CCTV camera streams the video data to a server with the deep learning models installed to detect obstacles within the video frames, and the A-star path-finding algorithm was utilized to generate the path. Each video frame is divided into tiles and treated as 1 node. The details of the algorithm are discussed in the succeeding session.

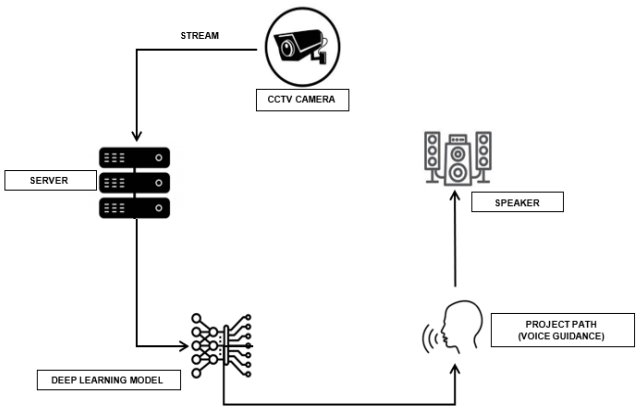

Fig. 1. System architecture.

A. Dataset

The dataset is essential for training YOLOv8 and other object detection models. It consists of images with annotations that include bounding boxes and class labels, providing the model with the information needed to learn object detection. During training, the dataset is divided into subsets: training, validation, and test. The training subset helps the model learn, the validation subset fine-tunes hyperparameters, and the test subset evaluates the final performance. High-quality and diverse datasets are crucial for effective training, ensuring accurate predictions and better model performance. 1200 samples were utilized to train and test the YoloV8 model for obstacle detection.

B. Bounding Boxes

Bounding boxes are essential for object detection models like YOLOv8, as they define the location and size of objects within images. During training, these annotations serve as ground truth, guiding the model to learn object positions and spatial relationships. The model predicts bounding boxes for objects, and its performance is evaluated by comparing these predictions to the ground truth. Discrepancies are used to adjust the model's parameters and reduce errors. After initial training, the model's bounding box predictions are further tested and refined to enhance accuracy. Properly annotated bounding boxes ensure effective object localization and classification, improving overall detection performance.

C. Hyperparameter Tuning for YOLOv8 Model

The training of the YOLOv8 model involves several crucial hyperparameters that significantly influence its performance. As detailed in Table I, the epochs parameter is set to 1000, meaning the model will traverse the entire training dataset 1000 times, which helps in enhancing the model's learning but requires careful monitoring to avoid overfitting. The batch size is configured to 16, indicating that the model will process 16 samples before updating its weights. This setting balances between frequent updates and memory usage, although it may affect the stability of the gradient estimates. The learning rate is set to 0.05, which controls the magnitude of weight adjustments during training. This relatively high learning rate can speed up convergence but may introduce instability if not carefully managed. To mitigate overfitting, weight decay is applied with a value of 0.0005, adding a penalty for large weights and promoting generalization. The image size is fixed at 640 x 640 pixels, determining the dimensions of the input images and impacting both detection accuracy and computational demands. Finally, the momentum is set at 0.937, which helps accelerate the gradient vectors in the correct direction, facilitating faster convergence and reducing oscillations. Together, these hyperparameters are finely tuned to balance learning efficiency and model robustness.

TABLE I. HYPER PARAMETERS

Hyper parameters	Tune value
Epochs	1000
Batch Size	16 (default)
Learning Rate	0.05
Weight Decay	0.0005 (default)
Image Size	640 x 640 (default)
Momentum	0.937 (default)

D. Model Training

Model training for YOLOv8 is essential for optimizing the model's ability to detect objects accurately within a specific dataset. Through training, the model's parameters are adjusted to improve detection precision and recall. This process allows the model to adapt to the unique characteristics of the data, enhancing its performance. For training purposes, 80% of the samples were utilized while 20% were used for evaluating the model. Key aspects of training include tuning hyperparameters like learning rate and batch size, which are critical for achieving the best results while preventing overfitting or underfitting. Regular evaluation using metrics such as mean Average Precision (mAP) helps in assessing the model's performance and making necessary improvements.

E. Performance Metrics for YOLOv8 Model Evaluation

A. Precision is the ratio of true positive detections to the total number of positive detections made by the model. It measures how many of the detected objects are actually correct.

$$\text{Precision} = \frac{number\ of\ True\ positives}{number\ of\ True\ positives + number\ of\ False\ positives} \quad (1)$$

B. Recall is the ratio of true positive detections to the total number of actual objects in the ground truth. It measures how well the model identifies all the relevant objects.

$$\text{Recall} = \frac{number\ of\ True\ positives}{number\ of\ True\ positives + number\ of\ False\ negatives} \quad (2)$$

C. The mean Average Precision (mAP) is the average of Average Precision (AP) scores across all classes. It summarizes the precision-recall curve, offering a single, comprehensive metric to evaluate the model's performance across different thresholds and providing a holistic view of accuracy and object detection capabilities. Higher mAP indicates better object detection performance across various classes and confidence levels.

$$\text{mAP} = \frac{1}{N} \sum_{i=1}^{N} APi \quad (3)$$

D. The F1-score in the context of YOLOv8 refers to a metric used to evaluate the model's performance in terms of both precision and recall. It is a harmonic mean of precision and recall, providing a single score that balances these two metrics.

$$\text{F1} = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad (4)$$

F. Pathfinding Algorithms

*1) A * Algorithm*: The A* (A-star) algorithm is a widely used pathfinding and graph traversal algorithm, especially notable for its efficiency and accuracy in finding the shortest path between two points. It combines the strengths of Dijkstra's algorithm and Greedy Best-First-Search. A* operates by maintaining a tree of paths originating from the start node, extending those paths one edge at a time until the goal node is reached. The algorithm uses a priority queue to explore nodes based on the following cost function:

$f(n) = g(n) + h(n)$

where:

- g(n) is the cost from the start node to the current node (n).
- h(n) is the heuristic estimate of the cost from (n) to the goal.

The heuristic function h(n) is crucial for the A* algorithm's performance. It must be admissible, meaning it never overestimates the true cost to reach the goal, ensuring the optimality of the path found. Common heuristic functions include the Euclidean distance and Manhattan distance, depending on the nature of the problem space.

2) Dijkstra's algorithm: Dijkstra's algorithm finds the shortest path between nodes in a graph. It starts at a chosen node, and then repeatedly selects the unvisited node with the smallest known distance from the start. It updates the distances to neighboring nodes and marks the current node as visited. This process continues until the destination is reached or all nodes are visited. It uses the formula:

$$d[v] = \min(d[v], d[u] + w(u,v))$$

where:

- d[v] is the distance to node v
- d[u] is the distance to node u
- w(u,v) is the weight of the edge from u to v

The algorithm iteratively updates these distances, starting from the source node, until it reaches the destination or visits all nodes. It always selects the unvisited node with the smallest known distance for the next iteration.

While both A* and Dijkstra's algorithms are effective for pathfinding, A* is often preferred in scenarios where efficiency and performance are critical due to the following reasons:

a) Heuristic guidance: A* uses a heuristic to guide its search, which can significantly reduce the number of nodes explored compared to Dijkstra's algorithm. Dijkstra's algorithm explores all possible paths, ensuring the shortest path but often at the cost of increased computation time, especially in large or complex graphs.

b) Efficiency: The heuristic function in A* allows the algorithm to focus on the most promising paths towards the goal. This focused search typically results in faster pathfinding, as fewer nodes are expanded compared to Dijkstra's algorithm, which treats all nodes with equal importance.

c) Optimality and completeness: A* is both optimal and complete, meaning it will always find the shortest path if one exists, provided the heuristic is admissible. Dijkstra's algorithm also guarantees the shortest path, but without the heuristic guidance, it often requires more extensive node exploration.

d) Flexibility: A* can be easily adjusted for different types of pathfinding problems by changing the heuristic function. This adaptability makes it suitable for various applications, from simple grid-based pathfinding to more complex navigation in dynamic environments.

e) Practicality: In many practical applications, such as robotics, video games, and geographical mapping, the A* algorithm's balance of optimality and performance makes it more practical than Dijkstra's algorithm. It provides quicker responses, which is crucial for real-time decision-making processes.

For this study, the A* algorithm would be chosen over Dijkstra's algorithm because of its heuristic-driven approach, which improves efficiency by minimizing the number of nodes investigated while ensuring the shortest path. This makes A* ideal for large-scale and sophisticated pathfinding applications in which performance and speed are important.

G. Pathfinding with YOLO Integration

As shown in Fig. 2, the provided flowchart illustrates how to use YOLO predictions to navigate and find the shortest path between two places. The process starts with an initialization stage, which prepares the system to capture and process frames. To navigate all of the instances, the input frame is divided into grid coordinates with 1s and 0s. Areas filled with 0s signify obstacles, whereas 1s indicate a clear pathway.

As illustrated in Fig. 3, the system processes the YOLO prediction results by tracking the center of each instance's bounding box. The system then identifies two distinct points within the scene, referred to as Point A and Point B, based on their respective (x, y) coordinates. Using these coordinates, the system calculates the shortest path between the two locations, employing pathfinding algorithms such as the A* algorithm. A feedback loop is incorporated, allowing the system to continuously capture new frames and update predictions, as well as pathfinding computations. This dynamic process enables the system to adapt to changes in the environment in real-time, ensuring precise navigation and efficient pathfinding.

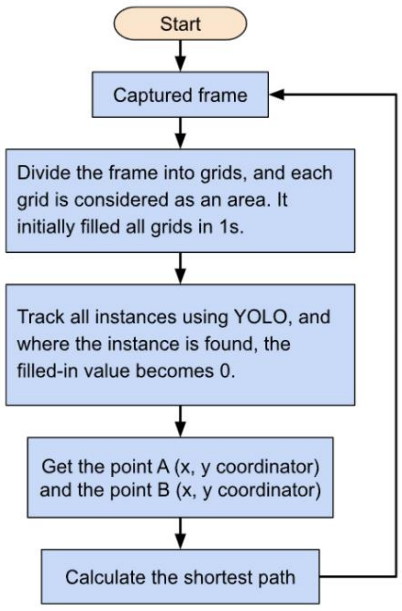

Fig. 2. Path-finding obstacle detection flowchart.

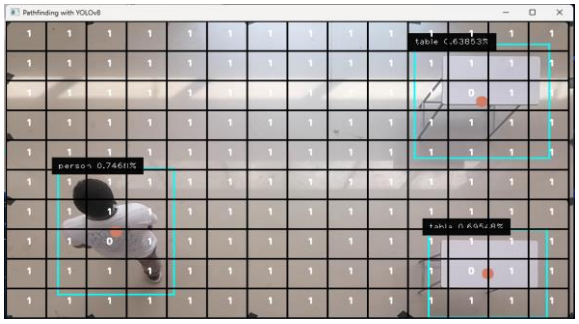

Fig. 3. Applied 1s and 0s on the captured frame.

IV. Experimental Result

A. Model Training Results

1) Confusion matrix: As illustrated in Fig. 4, the confusion matrix provides a comprehensive summary of the performance of the YOLOv8 small model across different classes. It is a powerful tool for understanding how well the model distinguishes between the classes: person, robot, table, and background.

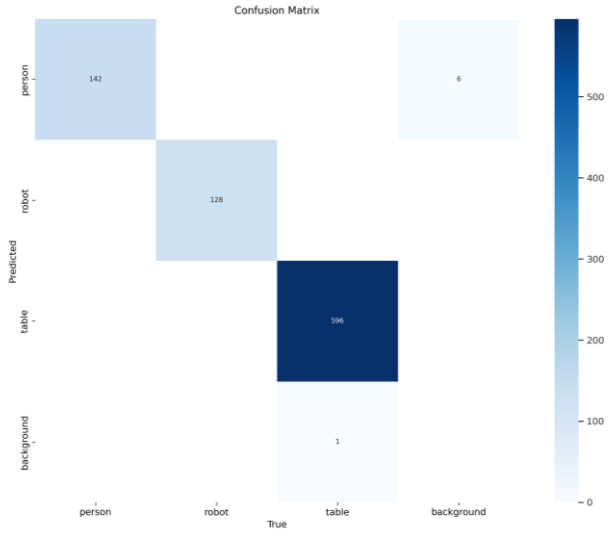

Fig. 4. Confusion matrix result.

In the confusion matrix, the "person" class has 142 true positives, indicating that the model correctly identified 142 instances of "person". However, there are six false positives where background instances were misclassified as "person." There are no false negatives, meaning all actual instances of "person" were correctly identified, showcasing the model's high accuracy for this class. For the "robot" class, the model performs perfectly, with 128 true positives and no false positives or false negatives. This indicates that the model consistently identifies "robot" instances without any errors, demonstrating its robustness in detecting this class. The "table" class also shows high accuracy with 596 true positives. There is only one false positive where a background instance was misclassified as "table," and no false negatives, meaning all actual instances of "table" were correctly identified. This indicates the model's high precision and recall for the "table" class. The confusion matrix also reveals that the model occasionally misclassified background instances. Specifically, six background instances were predicted as "person," and one background instance was predicted as "table." These misclassifications indicate room for improvement in distinguishing background from specific objects, which could enhance the model's overall performance.

In summary, the confusion matrix highlights that the YOLOv8 small model is highly effective at correctly identifying instances of "person," "robot," and "table." While the model demonstrates high precision and recall for these classes, there are occasional misclassifications of background instances, particularly as "person" and "table". These findings suggest that the model is robust and accurate but could benefit from further refinement to improve its handling of background instances. The confusion matrix serves as a valuable tool in validating the model's effectiveness and guiding future enhancements.

2) Training result graphs: Fig. 5 illustrates a visualization of several metrics and loss functions tracked during the YOLOv8 model's training and validation phases. Each subplot gives information about various aspects of the model's performance and learning process.

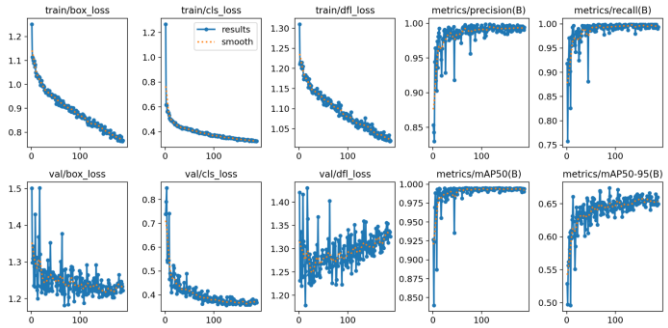

Fig. 5. YOLOv8 training results.

The training box loss (train/box_loss) plot illustrates the model's improvement in predicting bounding boxes. This loss consistently decreases over time, indicating that the model is learning to better match the predicted bounding boxes with the ground truth boxes. A similar trend is observed in the training classification loss (train/cls_loss) plot, where the decreasing loss shows that the model's classification capabilities are improving, allowing it to more accurately categorize objects into the correct classes (person, robot, table).

The training DFL loss (train/dfl_loss) which measures the distribution quality of the predicted bounding boxes, also shows a consistent decline. This suggests that the model is becoming more precise in predicting the exact locations of the bounding boxes. The precision metrics (metrics/precision (B)) and recall metrics (metrics/recall (B)) quickly rise and stabilize close to 1.0. This indicates that the model is highly accurate in its positive predictions, with very few false positives, and effective at detecting almost all relevant objects, with very few false negatives.

For the validation phase, the validation box loss (val/box_loss) decreases significantly, although with more fluctuation compared to the training loss. This suggests that

while the model is learning well, it encounters some variability when applied to new data. The validation classification loss (val/cls_loss) shows a decreasing trend, indicating good generalization, though it also fluctuates, highlighting some variability in performance on the validation set. The validation DFL loss (val/dfl_loss), despite generally decreasing, exhibits higher fluctuation, indicating less stability in predicting bounding box distributions on unseen data compared to the training set.

The mAP50 metrics (metrics/mAP50 (B)), which measure precision and recall at a specific Intersection over Union (IoU) threshold of 50%, rapidly increase and stabilize near 1.0. This demonstrates that the model performs exceptionally well in detecting objects with a high degree of overlap with the ground truth. The mAP50-95 metrics (metrics/mAP50-95(B)), which provide a more comprehensive measure across varying IoU thresholds (from 50% to 95%), show an upward trend and eventual stabilization. This indicates that the model is robust across different IoU thresholds, despite more variability compared to mAP50.

3) Model performance metrics: The YOLOv8 small model, trained to detect three classes (person, robot, and table), exhibits impressive performance metrics. As shown in Table II, the precision for all classes is reported as 1.0 at a confidence threshold of 0.926, which means that every detection made by the model is correct at this high confidence level. This suggests that the model is highly reliable in its detections, making no false positives at this threshold.

TABLE II. PERFORMANCE METRIC RESULT OF YOLOV8 SMALL

YOLOv8 small				
Classes	Precision	Recall	Precision-Recall (mAP@0.5)	F1-score
person robot table	All classes 1.0 at 0.926	All classes 1.0 at 0.000	All classes 0.993	All classes 0.99 at 0.624

The recall for all classes is also 1.0 but at a confidence threshold of 0.000. Recall measures the model's ability to identify all relevant instances in the dataset. Achieving a recall of 1.0 at the lowest confidence threshold means that the model captures all true positive instances, albeit at the expense of potentially including many false positives. This indicates the model's maximum sensitivity, showing it can detect every instance of the specified classes when no confidence filtering is applied.

The mean Average Precision (mAP) at an Intersection over Union (IoU) threshold of 0.5 is 0.993 for all classes. The mAP@0.5 is a comprehensive metric that combines precision and recall across different confidence thresholds. A mAP of 0.993 signifies that the model performs exceptionally well, balancing precision and recall almost perfectly across a range of confidence levels.

The F1-score for all classes is 0.99 at a confidence threshold of 0.624. The F1-score, being the harmonic mean of precision and recall, provides a single metric that accounts for both false positives and false negatives. An F1-score of 0.99 indicates that the model maintains a high balance between precision and recall at this confidence level, making it highly effective for practical applications where both false positives and false negatives need to be minimized.

The perfect precision and recall values demonstrate the model's outstanding capability to correctly identify and detect all instances of the specified classes under certain conditions. However, achieving 1.0 recall at a threshold of 0.000 means the model includes all possible detections, leading to capturing every true positive along with many false positives. The near-perfect mAP@0.5 score confirms the model's effectiveness across different confidence levels, providing a comprehensive view of its capabilities in handling various detection scenarios. The high F1-score further confirms the model's ability to maintain a good balance between precision and recall at a practical confidence threshold.

B. Path Planning Analysis

1) Algorithm implementation: The algorithm below is the exact representation of the source code of the path-finding application with obstacle avoidance.

Algorithm: Path Planning with A * Algorithm
Set top_left, bottom_left, bottom_right, top_right coordinates
Create Pathfinder instance (pf) with video capture, frame name, weights path, coco file path, max frame rows and columns, and entire area dimension
While True:
Read frame from pf.cap
If frame read is unsuccessful:
Reset stream to start
Print "reset stream"
Continue loop
Set target_area to (6, 3)
Get center areas and areas using new frame height and width
Frame, mapping instance, and matrix binary map with tracked objects
Get start area from adjacency matrix using key class index "person"
Create path using matrix binary map, start area, target area, and diagonal movement
Show grid zones on frame
Show obstacle zones on frame using class index "table"

Show A* path on frame
Show 1s and 0s map on frame using center areas
Show frame in window named pf.frame_name

2) Diagonal and non-diagonal pathfinding: Fig. 6 demonstrates a real-time pathfinding and object detection system using YOLOv8 integrated with A* pathfinding. The entire frame is divided into a grid of cells, which helps map out areas for pathfinding and identify object locations. Detected objects include a person and two tables, each indicated by bounding boxes with confidence scores. The grid cell containing the detected person acts as the starting point for the pathfinding algorithm, while a specific grid cell serves as the target area. The system continually checks the person's location to detect any movement errors. The path from the start to the target area is visualized using a sequence of green connected cells, calculated by the A* pathfinding algorithm while considering the grid layout and obstacle positions, with tables acting as obstacles in this sample illustration. Annotations and labels provide the class names and confidence scores of detected objects, with red dots marking their center points. The system utilizes a pre-trained YOLOv8 model for object detection and the A* pathfinding algorithm for computing the shortest path while avoiding obstacles. Visualization of the grid, detected objects, and computed path aids in understanding how the algorithm interprets the environment and plans the path, enhancing the ability of autonomous systems to navigate complex environments.

In Fig. 6, the pathfinding algorithm navigates the grid without diagonal movement. The detected objects, a person and two tables are highlighted with bounding boxes and labeled with confidence scores. The grid cell containing the person acts as the starting point, and a specified grid cell is the target. The A* pathfinding algorithm calculates the path using only horizontal and vertical movements, resulting in a sequence of green connected cells. This path avoids obstacles (tables) by making right-angle turns, which can lead to a longer path with more turns. Fig. 7 below shows diagonal path detection.

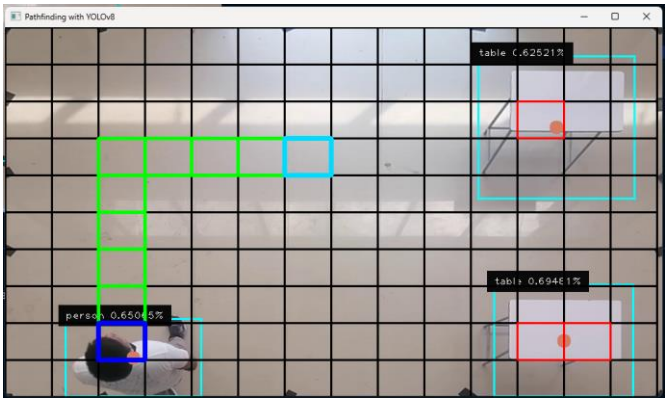

Fig. 6. Without diagonal pathfinding.

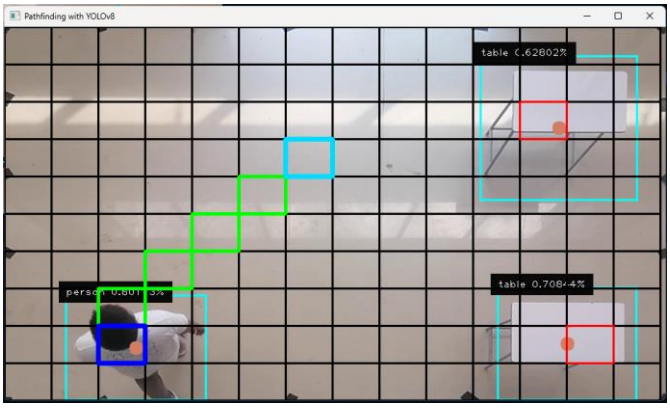

Fig. 7. With diagonal pathfinding.

With diagonal movement in the A* pathfinding algorithm, navigation becomes more efficient. The same grid and detected objects are used, with the person as the starting point and a specified grid cell as the target. Allowing diagonal movement results in a more direct path, visualized with green connected cells, reducing the number of turns and creating a shorter, more efficient route.

Comparing the two figures, the path with diagonal movement is more efficient, with fewer turns and a more direct route to the target area. The system continuously monitors the person's location for movement faults, ensuring that the path remains accurate. By visualizing the grid, detected objects, and computed paths, the system aids in understanding how the algorithm interprets the environment and plans the path, enhancing the navigation capabilities of autonomous systems.

3) Experimental results: In this figure, the A* a pathfinding algorithm is visualized on a grid overlaying a scene with detected objects. The grid cells are used to map out the path from the starting point (a person) to a target location. Detected objects, such as tables, are marked with bounding boxes and confidence scores, which indicate the algorithm's certainty in identifying them. Fig. 8 is a scenario where the system can detect the best path to avoid detected obstacles. In Fig. 8, the person is detected and the target is set to the other side of the person, marked by a blue box.

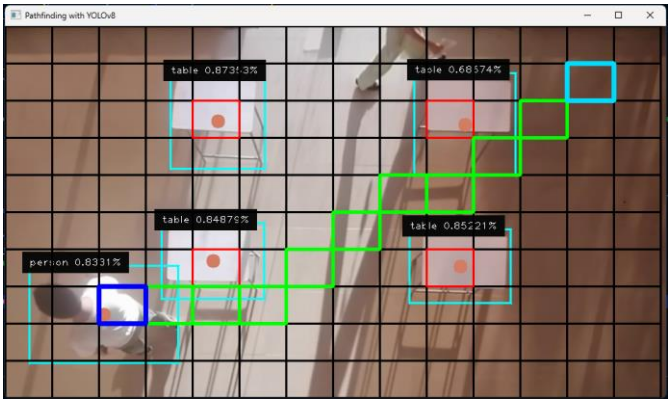

Fig. 8. Test 1: With diagonal path - Obstacle avoidance.

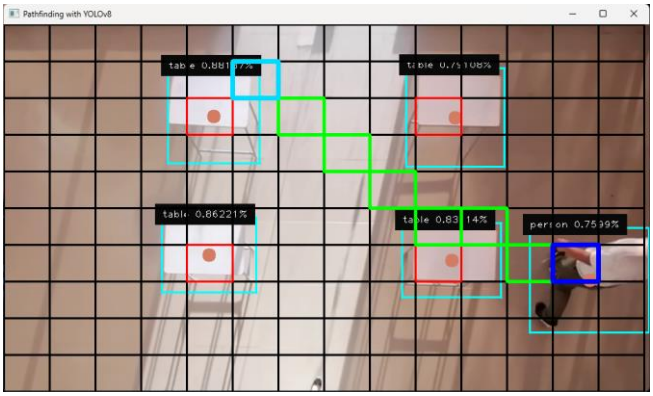

Fig. 9. Test 2: With diagonal path - Obstacle avoidance.

In Fig. 9, the path, highlighted in green, shows the movement direction calculated by the algorithm. However, there's an issue: the path intersects with the bottom right-side table, suggesting a collision error. This means the algorithm did not correctly account for the table as an obstacle, leading to an incorrect path that passes through an object it should avoid.

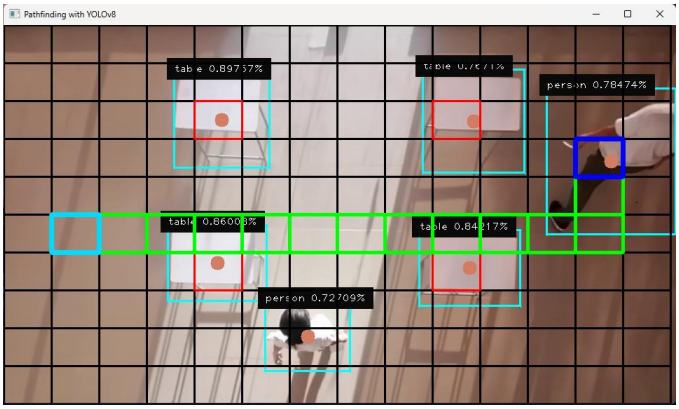

Fig. 10. Test 1: Without diagonal path - Obstacle avoidance.

Fig. 10 shows the green path, representing the calculated route, avoids diagonal movement and instead uses only horizontal and vertical steps. This leads to a less efficient, zigzagging path. However, there's a notable issue: the two tables in the lower part of the grid are incorrectly treated as part of the free path instead of being recognized as obstacles. As a result, the path cuts through areas where it should have avoided obstacles, leading to a flawed and non-optimal route.

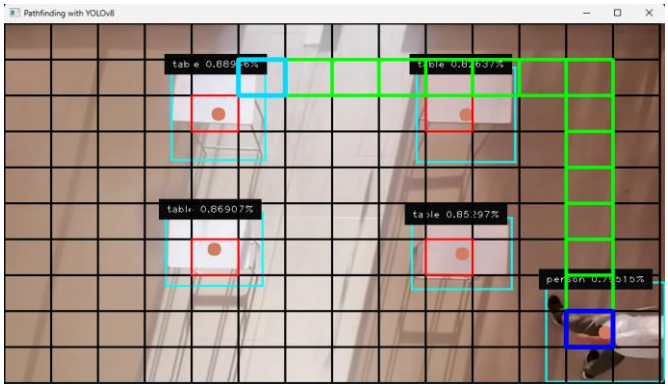

Fig. 11. Test 2: Without diagonal path - Obstacle avoidance.

In Fig. 11, the issue of pathfinding errors becomes clearer. The A* pathfinding algorithm, represented by the green path, incorrectly navigates through areas that should be recognized as obstacles. Specifically, part of the tables in the lower section of the grid is mistakenly treated as free space, allowing the path to pass through what should be blocked areas.

This suggests a problem with how the algorithm maps the obstacle sizes or shapes. The detected tables, outlined with bounding boxes, aren't fully recognized as solid obstacles, leading to an inaccurate and non-optimal path. The path doesn't properly avoid the tables, indicating that the obstacle mapping is not precise enough. This misjudgment by the A* algorithm highlights the need for better obstacle detection and size estimation to ensure that the navigation paths are truly free of obstructions.

4) Strengths and challenges: The A* algorithm demonstrates considerable strengths in pathfinding, particularly in grid-based environments. One of its key advantages is its efficiency in identifying the shortest route between a start point and a target, even when obstacles are present. This makes it highly effective for navigation tasks where quick and accurate pathfinding is crucial. Additionally, the integration of YOLOv8 for object detection enhances the algorithm's ability to identify and account for obstacles in real-time. This capability is essential for dynamic environments where objects may move or change, requiring the algorithm to adapt quickly to maintain a clear path.

However, the algorithm also faces significant challenges. One of the main issues is inaccurate obstacle mapping. In some instances, the algorithm incorrectly treats parts of obstacles, such as tables, as free space, leading to paths that intersect with these objects. This misjudgment can result in collisions or inefficient navigation. Another challenge is the limitation in movement directions. When diagonal movement is not allowed, the algorithm tends to create longer, less efficient paths that zigzag around obstacles. This can cause unnecessary detours and increase travel time. Lastly, the algorithm sometimes overestimates the amount of free space available around obstacles. This can lead to paths that are not truly clear, further increasing the risk of collisions with objects that should have been avoided.

These challenges highlight the need for improvements in obstacle detection, path mapping, and movement direction strategies to optimize the algorithm's performance in complex environments.

V. Conclusion and Recommendation

The current work can present a preliminary deep-learning-based indoor assistance system for the visually impaired. The combination of the YOLOv8 algorithm for the identification of obstacles and the existence of an efficient navigation algorithm like the A* has proved to work effectively. Our system made 60% differentiation of correct paths and mistakes in avoiding the obstacles meaning that our method has the capacity of helping the visually impaired users in improving their navigation. However, the efficacy of the audio commands that guide the users to move forward or turn left or right is still poor.

This shortcoming will be central to the improvement of the applicable theories in the future.

For system improvement, it is suggested that efforts be focused on increasing the reliability of audio directions in the future. This involves improving the processes that define such commands so that they are accurate and can be easily used by the end user. It also outlines the inclusion of diverse test arenas with different obstacles indoors as a way of helping realize copies of different odd experiences. The contribution of visually impaired individuals in testing as well as giving feedback on the product's usability and lapses that need upgrading are also useful. The improvement of the detection algorithm YOLOv8 and the pathfinding algorithm A* needs to be done permanently to enhance the results. Searching for other types of algorithms or integrating with others may also help in obtaining better results. Furthermore, future work should focus on the performance of the real-time system on large environments along with adding more complicated obstacle cases without a large impact on the system. Therefore the above-mentioned points can be used to further develop the system to help visually impaired people maintain independence and avoid hazardous situations indoors. It is recommended as well to have voice recognition so that the user can use a mobile phone to instruct the desired destination within the indoor space.

ACKNOWLEDGMENT

We extend our profound gratitude to the Center for Artificial Intelligence, Big Data, and Cloud Computing of Cebu Technological University Carmen Campus for their unwavering support throughout this research endeavor. Their contributions have been instrumental in the success of our project, providing valuable resources that greatly enhanced our study.

REFERENCES

[1] H. Fernandes, P. Costa, V. Filipe, H. Paredes, and J. Barroso. "A review of assistive spatial orientation and navigation technologies for the visually impaired," Universal Access in the Information Society 18(1): 155–168. Crossref. (2017).

[2] Patle, B.K., Babu, G.L., Pandey, A., Parhi, D.R.K., Jagadeesh, A., 2019, "A review: On path planning strategies for navigation of mobile robots," Defence Technology 15 (4), 582–606. https://doi.org/10.1016/j.dt.2019.04.011.

[3] Q. Chen, M. Khan, C. Tsangouri, C. Yang, B. Li, J. Xiao, and Z Zhu, "CCNY smart cane," in 2017 IEEE 7th Annual International Conference on CYBER Technology in Automation, Control, and Intelligent Systems, CYBER 2017, Jul. 2018, pp. 1246–1251, doi: 10.1109/CYBER.2017.8446303.

[4] Shangguan L, Yang Z, Zhou Z, Zheng X, Wu C, and Liu Y, (2014), "CrossNavi: Enabling real-time crossroad navigation for the blind with commodity phones," In: Proceedings of the 2014 ACM international joint conference on pervasive and ubiquitous computing, Seattle, WA, USA, September 2014, pp.787–798. New York: Association for Computing Machinery.

[5] Weyrer TN, Hochmair HH and Paulus G (2014) "Intermodal door-to-door routing for people with physical impairments in a web-based, open-source platform". Transportation Research Record 2469(1): 108–119.

[6] Yang R, Park S, Mishra SR, Hong Z, Newsom C, Joo H, Hofer E, and Newman MW, (2011) "Supporting spatial awareness and independent wayfinding for pedestrians with visual impairments". In: The proceedings of the 13th international ACM SIGACCESS conference on Computers and accessibility, Dundee, Scotland, October 2011, pp.27–34. New York: Association for Computing Machinery.

[7] Y. H. Lee and G. Medioni, "Wearable RGBD indoor navigation system for the blind," in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 8927, Springer International Publishing, 2015, pp. 493–508.

[8] R. L. Campos, R. Jafri, S. A. Ali, O. Correa, and H. R. Arabnia, "Evaluation of the google tango tablet development kit: A case study of a localization and navigation system," in Proceedings - 2018 International Conference on Computational Science and Computational Intelligence, CSCI 2018, Dec. 2018, pp. 501–506, doi: 10.1109/CSCI46756.2018.00103.

[9] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," IEEE Transactions on Systems Science and Cybernetics, vol. 4, no. 2, pp. 100–107, 1968, doi: 10.1109/TSSC.1968.300136.

[10] World Health Organization, ''World report on vision,'' 2019. https://www.who.int/publications-detail/world-report-on-vision.

[11] Cai, L., Zhu, X., Nov. 2018. ''Intelligent guide cane design based on ant colony algorithm", IOP Conference Series: Materials Science and Engineering, Nanchang. China. https://doi.org/10.1088/1757-899x/423/1/012066.

[12] Kumar, S., Pandey, K.K., Muni, M.K., Parhi, D.R., 2020. "Path planning of the mobile robot using fuzzified advanced ant colony optimization," Lecture Notes in Mechanical Engineering. Springer, Singapore. https://doi.org/10.1007/978-981- 15-2696-1_101.

[13] S. Li, W. Su, R. Huang, and S. Zhang, ''Mobile robot navigation algorithm based on ant colony algorithm with A⁄ heuristic method," 4th International Conference on Robotics and Automation Sciences (ICRAS), Wuhan, China, pp. 28-33, 2020, 10.1109/ICRAS49812.2020.9135055.

[14] X. Ma and H. Mei, ''The global path planning of ant colony system mobile robot based on jump point search strategy," Jiqiren/Robot, vol. 42, no. 4, pp. 494–502, Jul. 2020, 10.13973/j.cnki.robot.190463.

[15] D. Ni, A. Song, L. Tian, X. Xu, and D. Chen, ''A walking assistant robotic system for the visually impaired based on computer vision and tactile perception,'' Int. J. Social Robot., vol. 7, no. 5, pp. 617–628, Nov. 2015.

[16] Broersen, T., Fichtner, F. W., Heeres, E. J., de Liefde, I., Rodenberg, O. B. P. M., Verbree, E., and Voûte, R. (2016). "Project pointless. Identifying, visualizing and pathfinding through empty space in interior point clouds using an octree approach". In AGILE 2016; 19th AGILE Conference on Geographic Information Science, 14-17 June, 2016; Author's version.

[17] H.-C. Wang, R. K. Katzschmann, S. Teng, B. Araki, L. Giarre, and D. Rus, ''Enabling independent navigation for visually impaired people through a wearable vision-based feedback system,'' in Proc. IEEE Int. Conf. Robot. Autom. (ICRA), May 2017, pp. 6533–6540.

[18] R. Tapu, B. Mocanu, and T. Zaharia, ''DEEP-SEE: Joint object detection, tracking and recognition with application to visually impaired navigational assistance,'' Sensors, vol. 17, no. 11, p. 2473, Oct. 2017.

[19] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, ''MobileNets: Efficient convolutional neural networks for mobile vision applications,'' 2017, arXiv:1704.04861. [Online]. Available: http://arxiv.org/abs/1704.04861.

[20] J. Pedoeem and R. Huang, ''YOLO-LITE: A real-time object detection algorithm optimized for non-GPU computers,'' 2018, arXiv:1811.05588. [Online]. Available: http://arxiv.org/abs/1811.05588.

[21] X. Zhang, X. Zhou, M. Lin, and J. Sun, ''ShuffleNet: An extremely efficient convolutional neural network for mobile devices,'' in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., June. 2018, pp. 6848–6856.

[22] Mortari, F.; Zlatanova, S.; Liu, L.; Clementini, E. "Improved geometric network model,"' (IGNM): A novel approach for deriving Connectivity Graphs for Indoor Navigation. ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci. 2014, 2–4, 45–51.

[23] Xiong, Q.; Zhu, Q.; Zlatanova, S.; Du, Z.; Zhang, Y.; Zeng, L.Y. "Multi-Level Indoor Path Planning Method". Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci. 2015, 40, 19–23.

[24] Liu, L.; Xu, W.; Penard, W.; Zlatanova, S. “Leveraging spatial models to improve indoor tracking,” Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.-ISPRS Arch. 2015, 40, 75–80.

[25] Rodenberg, O. “The effect of A* pathfinding characteristics on the path length and performance in an octree representation of an indoor point cloud,” Master’s Thesis, Technical University of Delft, Delft, The Netherlands, 2016.

[26] Li, F.; Zlatanova, S.; Koopman, M.; Bai, X.; Diakité, “A universal path planning for an indoor drone,” Autom. Constr. 2018, 95, 275–283.

[27] Tsirmpas, C.; Rompas, A.; Fokou, O.; Koutsouris, D. “An indoor navigation system for visually impaired and elderly people based on Radio Frequency Identification (RFID)". Inf. Sci. 2015, 320, 288–305.

[28] Khelifi, F.; Bradai, A.; Benslimane, A.; Rawat, P.; Atri, M. “A survey of localization systems in the internet of things,” Mob. Netw. Appl. 2019, 24, 61–785.

[29] Mainetti, L.; Patrono, L.; Sergi, I. “A survey on indoor positioning systems,” In Proceedings of the 22nd International Conference on Software, Telecommunications and Computer Networks (SoftCOM), Split, Croatia, 17–19 September 2014; pp. 1–10.

[30] A. Ganz, J. Schafer, S. Gandhi, E. Puleo, C. Wilson, and M. Robertson, "Percept indoor navigation system for the blind and visually impaired: Architecture and experimentation," Int. J. Telemed. Appl., vol. 2012, no. Article ID 894869, p. 12 page, 2012, doi: 10.1155/2012/894869.

[31] L. a. Guerrero, F. Vasquez, and S. F. Ochoa, "An indoor navigation system for the visually impaired," Sensors, vol. 12, no. 6, pp. 8236 8258, 2012, doi: 10.3390/s120608236.

[32] Ivanov, R. (2010) “Indoor navigation system for visually impaired,” In Proc. 11th Int. Conf. Computer Systems and Technologies and Workshop for PhD Students in Computing on Int. Conf. Computer Systems and Technologies”, pp. 143–149. ACM.