Expert Systems With Applications 212 (2023) 118720

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

DeepNAVI: A deep learning based smartphone navigation assistant for people with visual impairments

Bineeth Kuriakose *, Raju Shrestha, Frode Eika Sandnes

Department of Computer Science, Oslo Metropolitan University, Oslo, Norway

ARTICLE INFO

Keywords:
Navigation assistant
Deep learning
Blind
Visual impairment
Portable
Smartphone

ABSTRACT

Navigation assistance is an active research area, where one aim is to foster independent living for people with vision impairments. Despite the fact that many navigation assistants use advanced technologies and methods, we found that they did not explicitly address two essential requirements in a navigation assistant - portability and convenience. It is equally imperative in designing a navigation assistant for the visually impaired that the device is portable and convenient to use without much training. Some navigation assistants do not provide users with detailed information about the obstacle types that can be detected, which is essential to make informed decisions when navigating in real-time. To address these gaps, we propose DeepNAVI, a smartphone-based navigation assistant that leverages deep learning competence. Besides providing information about the type of obstacles present, our system can also provide information about their position, distance from the user, motion status, and scene information. All this information is offered to users through audio mode without compromising portability and convenience. With a small model size and rapid inference time, our navigation assistant can be deployed on a portable device such as a smartphone and work seamlessly in a real-time environment. We conducted a pilot test with a user to assess the usefulness and practicality of the system. Our testing results indicate that our system has the potential to be a practical and useful navigation assistant for the visually impaired.

1. Introduction

Globally, around 2.2 billion people are diagnosed with vision impairment or blindness (WHO, 2021). Several studies have reported the difficulties faced by people with visual impairments during navigation (Manduchi & Kurniawan, 2011; Riazi, Riazi, Yoosfi, & Bahmeei, 2016). The freedom to move about independently is one component of a dignified life, and tools to support this freedom is the key motivation for this work.

People with visual impairments typically use white canes, guide dogs, and sighted people to assist them (Aspinall, 2012; Hersh & Johnson, 2010) for navigation. White canes have many advantages, including ease of replacement, cost-effectiveness, easy maneuverability, and low maintenance requirements. However, cane users may encounter challenges when navigating crowded areas (Nook, 2020). Furthermore, users cannot use their hands to hold anything else when using canes. Guide dogs can provide companionship, love, comfort, and respect to their handlers while keeping their users (owners) safe. However, training and managing guide dogs are incredibly time-consuming and expensive. Moreover, there are certain places where dogs are not permitted, such as indoor theaters, sticky floors, and some foreign countries that do not have accessibility laws (Austin, 2016; Nook, 2020). People with visual impairments feel safe and comfortable when assisted by someone with sight. Nevertheless, relying on another person to navigate might be a barrier to independence.

To address the issues, such as discomfort in public places, portability issues, expensive and time-consuming training, constant dependability on a second person, etc., various navigation assistant systems have been proposed in the literature (Bhowmick & Hazarika, 2017; Chanana, Paul, Balakrishnan, & Rao, 2017; Real & Araujo, 2019). Some were designed for indoor use, some for outdoor use, and others for both (Real & Araujo, 2019). However, many of these devices are uncomfortable and lead to users' perceived social stigma when using these solutions (Dos Santos, Ferrari, Medola, & Sandnes, 2022).

Researchers have investigated diverse technologies, such as artificial intelligence and machine learning, to find solutions to navigation aids for people with visual impairments. Deep learning models have recently been increasingly explored for obstacle detection in navigation assistance systems. Despite the availability of many object detection models, selecting a suitable model with low inference time and small memory footprint requirement for a real-time navigation environment needs a carefully crafted study and analysis.

* Corresponding author.
E-mail addresses: bineethk@oslomet.no (B. Kuriakose), raju.shrestha@oslomet.no (R. Shrestha), frodes@oslomet.no (F.E. Sandnes).

https://doi.org/10.1016/j.eswa.2022.118720
Received 30 May 2022; Received in revised form 25 August 2022; Accepted 27 August 2022
Available online 31 August 2022

As portability is an important design consideration for navigation support systems, researchers have been exploring the potential of using smartphones (Kuriakose, Shrestha, & Sandnes, 2020). A lot of research attention is being paid to how portable devices can assist people with disabilities in daily activities, such as navigation. Modern smartphones are promising because of their sensors, cameras, increasing processing power, and broader availability (Kuriakose et al., 2020).

We propose DeepNAVI, a portable smartphone-based navigation solution that uses deep learning models for obstacle detection and scene recognition. In addition, DeepNAVI can provide detailed information about the various attributes of obstacles, such as distance to the user, position, and motion status (stationary or moving). Even though there are several smartphone-based navigation assistants or deep learning-based navigation assistance systems proposed in the literature, what differentiates DeepNAVI is its design which gives importance to both technical and usability factors. DeepNAVI has onboard trained deep learning models which do not require internet connectivity to process information. Furthermore, DeepNAVI can be integrated into a smartphone and uses existing sensors for giving navigation directions to people with visual impairments. This work is an extension of our previous work reported in Kuriakose, Shrestha, and Eika Sandnes (2021). However, this work has been extended by making substantial changes by adding more features, training, and testing with better deep learning models with more extensive and extended datasets. The current version of DeepNAVI evolved after design research collaboration with the users incorporated changes from the previous prototype. Some of the components of the current version of DeepNAVI are discussed in detail in the authors' previous works. For example, in distance estimation, scene recognition, and obstacle detection, the authors did elaborate studies and testing and wind up with the ones used in the current version of DeepNAVI. Many similar and recent works in the literature have used existing general datasets or a pre-trained object/obstacle detection model (Rao, et al., 2021; Ashiq, et al., 2022; Joshi, Yadav, Dutta, & Travieso-Gonzalez, 2020; Mukhiddinov & Cho, 2021). But in this work, we have created custom datasets for training models rather than general datasets, which include only relevant objects/obstacles in navigation scenarios. Moreover, this article also provides a detailed evaluation of the performance of different system modules in DeepNAVI. Furthermore, we did extensive pilot testing with a user with visual impairment to assess our proposed system's practicability and usability.

The main contributions of this work are as follows: (1) the proposed design of a smartphone-based navigation assistant after carefully considering various design choices and requirements from target users; (2) the creation of custom datasets for obstacle detection and scene recognition which consist of 20 different types of obstacles and 20 scene categories relevant to navigation domain; (3) lightweight object detection and scene recognition models trained with our datasets, (4) a detailed pilot testing and analysis of our smartphone-based navigation assistant with a smart cane from a visually impaired user perspective, and (5) a consolidated comparative analysis of our system with other similar systems proposed in the domain.

The paper is organized as follows. Section 2 presents related work reported on navigation assistants. Section 3 provides details of our system design and implementation. The development of our smartphone application is described in Section 4. Section 5 provides the experiments and results. It is followed by the discussion in Section 6. The paper concludes in Section 7. We use the terms *people with visual impairments* or *visually impaired users* or *users* throughout this article to refer to people who are legally blind or have reduced vision to perceive visual stimuli.

2. Related works

Many diverse systems are proposed in the literature to assist the navigation of people with visual impairments. Researchers used various criteria to categorize navigation assistants for the visually impaired. Here, we have categorized navigation assistant systems into four major categories based on their processing environments, which refer to the primary device used to process information. They are hardware boards, smart canes, smart glasses, and smartphones. The following subsections describe each, and the example systems fall under each.

2.1. Hardware boards

Systems under this category use hardware boards such as Arduino, Raspberry Pi, or even a laptop to process information acquired from the environment. Such systems use external cameras, BLE beacons, and sensors to acquire information from the navigating environment.

The deep learning-based assistive system proposed in Lin, Wang, Yi, and Lian (2019) consisted of an RGB-D camera, an earphone, a laptop for deep learning processing, and a smartphone for touch-based interaction. The system used segmentation networks that provide semantic information using RGB and depth images. The system claimed to provide reliable feedback to visually impaired people to avoid obstacles. The lack of portability may be one drawback of the system.

Kanwal, Bostanci, Currie, and Clark (2015) introduced a Kinect-based navigation assistant that uses depth values from an infrared sensor. Obstacles are detected by applying the corner detection algorithm to the images, and the depth sensor provides the corresponding distance from the obstacles. The system also suggests a safe path with direction signals and tells the user to stop, move left, or move right. The Kinect camera is intended for stationary home use and is thus not practical from a portability perspective. Moreover, if the system could provide information about obstacles, it could benefit the users. Bhowmick, Prakash, Bhagat, Prasad, and Hazarika (2014) presented an assistive navigation system that also used a Microsoft Kinect depth sensor on board. The Speed-Up Robust Features (SURF) and Bag-of-Visual-Words (BOVW) models extracts features and are used in obstacle detection. The user would receive the audio output through a headset. Although the system could provide information about obstacles, there could be issues related to portability aspects.

Moharkar, Varun, Patil, and Pal (2020) proposed a single-board PC (called an Odroid) based navigation assistant. A USB camera was integrated with the system to capture real-time video. The system detected obstacles, found the distance using a laser, and then provided the results to the user using audio feedback. Obstacle detection and classification were performed using a multimodal fusion-based faster RCNN.

The system introduced by Ashiq, et al. (2022) uses the MobileNet architecture for obstacle detection and uses a Raspberry Pi board to process the information. The user receives audio feedback, and the system can share the location with the user's family/friends. Similarly, in Joshi et al. (2020), the system uses YOLOv3 for obstacle detection that runs on a Raspberry Pi board. The system can recognize different objects, and auditory output is provided to the user in real-time. But a few limitations existing with these systems are lack of feature support (such as distance estimation, scene recognition, or motion detection of obstacles) which could be useful for seamless navigation. Moreover, even though the systems are new in the domain, their architectures for obstacle detection were not the best among the currently available deep learning models that can offer better detection accuracy, inference time, and the capability to get deployed in miniature portable devices like smartphones.

The indoor guidance system proposed in Kahraman and Turhan (2021) uses a hybrid Radio-frequency identification (RFID)/Bluetooth Low Energy (BLE) infrastructure to provide intelligent navigation and guidance to the user in complex indoor environments. The system enables users to input their navigation purpose through a specially designed user interface and provides intelligent guidance through a chain of destination targets, which are determined according to the inherent procedures of the environment. The installation of RFID/BLE

is costly and complex. Furthermore, such a system is not a viable option in outdoor navigation environments.

Barontini, Catalano, Pallottino, Leporini, and Bianchi (2020) introduced a system consisting of an RGB-D camera, a processing unit to compute visual information to avoid obstacles, and a wearable device, which can provide specific force feedback for guidance in an unknown indoor environment. The main limitations of the system are associated with portability, and the operating environment is restricted to indoors.

Navigation assistants that use hardware boards as their central processing unit are often inconvenient and limited in portability. Hardware processing boards and external cameras have power supplies and connecting wires, so users may feel uncomfortable carrying them while navigating.

2.2. Smart canes

Smart canes are electronic devices that fit as a handle on white canes used by people with visual impairments. While white canes can only detect obstacles up to knee height, smart cane detects obstacles from knee to head height (Saksham, 2014). Sonic waves are used to detect obstacles, and intuitive vibrational patterns indicate the presence of obstacles in a smart cane. In recent years, several navigation systems have been proposed that enhance and modify the basic functionality of smart canes. The following section discusses some recent smart cane-based navigation assistants.

Megalingam, Nambissan, Thambi, Gopinath, and Nandakumar (2015) proposed a smart cane with a Bluetooth-enabled obstacle detection module. Obstacles are detected with the help of ultrasonic range finders. The system uses two output modalities: synthetic speech feedback for informing distance through a Bluetooth headset and haptic feedback to warn the user about moving obstacles.

The electronic travel aid developed in Guerrero, Quezada-V, and Chacon-Troya (2018) consists of an ultrasonic sensor to detect possible obstacles, a working range between 0.5 and 5 m, a sound module, and a buzzer to alert the user about possible obstacles. An android application was used to communicate with the smart cane through a GPS and GSM module to help locate the user by sending a text message to a relative to access the user's location and visualizing it through Google Maps.

Saaid, Mohammad, and Ali (2016) proposed a smart cane with range notification. The system used an ultrasonic sensor to measure the distance from the obstacle. Data processing was performed using the National Instruments myRIO-1900 controller. The cane alerted users about obstacles using audio. The authors claim that the system can recognize obstacles both indoors and outdoors. No user-level tests were reported.

Smart cane is easy to learn and can be used easily as a mobility aid with hardly any assistance. However, a person must undergo short training to become an expert user. Moreover, although modern smart canes are expensive and have features such as smartphone integration, they do not provide much information about the surrounding environment or obstacles.

2.3. Smart glasses

Smart glass is a portable device that scans the navigation environment using a camera mounted on the glasses to provide information about obstacles. Various technology firms are developing smart glass solutions to support the various activities of people with vision impairments. This section discusses some research that uses smart glasses as navigational aid for people with visual impairments.

Rao, et al. (2021) proposed a navigation solution using Google Glass. The camera embedded in the smart glasses was used to capture the images of the surroundings, which were analyzed using the Microsoft Custom Vision Application Programming Interface (Vision API) from Azure Cognitive Services. The output of the Vision API about various obstacles was converted into speech and presented to the user. This system required constant network connectivity to process environmental data. This could be a limitation when users travel to basements or areas with limited network connectivity.

The smart glass solution introduced in Mukhiddinov and Cho (2021) includes a transformer-based object detection model and text recognition model that use computer vision and deep learning methods. The proposed system runs these models on an external server connected to a smartphone. The authors claim that the system can detect and recognize obstacles from low-light and dark-scene images to assist users in a nighttime environment. This system also required constant access to a network that could limit the operation, similar to Rao, et al. (2021).

Suresh, Arora, Laha, Gaba, and Bhambri (2017) proposed smart glasses that consist of ultrasonic sensors to detect obstacles during navigation. The central processing part was a Raspberry Pi which analyzes the input data. The system could also provide warning through vibrations in the recognized direction. The software framework was managed in a Robot Operating System (ROS). The embedded external sensors and the Raspberry Pi board might be inconvenient to users during navigation.

Although smart glasses are convenient to use, their main drawback is their high cost. Most users living in middle or low-income countries cannot afford to buy smart glasses.

2.4. Smartphones

The exponential growth of the smartphone industry paved the way to explore them more in navigation assistance research. The navigation assistance system proposed in Bai, Liu, Su, and Fu (2017) used a smartphone to interact with the user through voice input. Stereo cameras were used to capture video from the environment and then send it to the cloud computing platform. Similar to Rao, et al. (2021), Mukhiddinov and Cho (2021), the system also required constant connectivity to the data network to function.

The system introduced in Lin, Lee, and Chiang (2017) consists of an image recognition system integrated with a smartphone application. The system supports two operation modes based on the availability of the network: online and offline. A smartphone was used to capture objects in front of the user and send them to a back-end server. Two algorithms, Faster R-CNN and YOLO, were applied for object recognition. The faster R-CNN algorithm was used in the offline mode of the system to obtain higher accuracy. In contrast, the YOLO algorithm was applied in online mode to get a higher processing speed. After identifying obstacles and their distance, the user would be informed about the results through audio mode.

Bai, et al. (2019) presented a navigation assistance system with an RGB-D camera, an inertial measurement unit (IMU) mounted on a pair of glasses, and a smartphone as its main components. A lightweight CNN was installed on the smartphone to detect obstacles and their position and orientation. Although the authors claim that the system was tested and works indoors and outdoors, the cables connecting various system components might inconvenience the users while navigating.

The system proposed in Fusco and Coughlan (2020) has a real-time smartphone app that combines computer vision, a 2D map, and the IMU of the smartphone to estimate and track the user's location in an indoor environment. At the same time, the app requires the user to hold the smartphone or wear it with the camera facing forward while walking, which might be uncomfortable for the user.

PERCEPT-II is a smartphone-based indoor navigation system proposed in Ganz, Schafer, Tao, Wilson, and Robertson (2014). It is an android application that allows the visually impaired to receive navigation instructions to the target destination when they touch specific landmarks equipped with Near Field Communication tags. The major limitation of the system is associated with NFC tags installation, which involves high maintenance in a large-scale deployment.

ARIANNA (pAth Recognition for Indoor Assisted NavigatioN with Augmented perception) (Croce, et al., 2014) allows the users to find

points of interest in an indoor navigation environment by following a path painted or stuck on the floor. A smartphone camera detects the path, and the phone also generates a vibration signal that provides feedback to the user to correct the direction. Similar to Ganz, et al. (2014), the maintenance and deployment are not practicable in real-life situations.

Tapu, Mocanu, Bursuc, and Zaharia (2013) came with smartphone-based real-time obstacle detection and classification system to help visually impaired people navigate indoor and outdoor environments. The system tried to estimate the camera and background motion using homographic transforms. A HOG descriptor was used with the Bag of Visual Words (BoVW) retrieval framework for obstacle classification in video streams.

Peng, Peursum, Li, and Venkatesh (2010) proposed a smartphone-based navigation assistance system to detect objects on the floor regardless of their height. The proposed system assumes that the user can always keep the smartphone at a tilt angle, such as 45 so that the floor in front of the user is always visible in the image. Because of this assumption, the authors claim that obstacles on the floor in front of the user can be detected in real-time using the proposed system. The system was tested under different floor conditions, and a field trial was conducted with five users. The limitations described in the paper describe most users' difficulty in holding the smartphone at the requested tilt angle (around 45).

In summary, the significant limitations associated with most of the systems reviewed here are associated with portability, which makes them inconvenient during navigation. Although there are portable solutions, many depend on network connectivity and server processing. Users who need to connect to the internet while navigating could raise concerns about privacy and security. Although some have explored general-purpose object detection models with better accuracy, they have not shown how they can be integrated into a portable system with low computational resources. Generally, when used in real-time environments, object/obstacle detection models should have a low inference time without compromising accuracy. However, when general-purpose objection models have been used, they could have adequate accuracy but take more time to deliver the result, which could cause accidents or collisions during navigation due to a delay in response time (Kuriakose, Shrestha, & Sandnes, 2021b).

3. Proposed system and implementation

To identify the limitations of existing navigation systems, we reviewed the findings of our literature survey reported in Kuriakose, Shrestha, and Sandnes (2022) and considered the recommendations we made. We also considered the results of our study on the capability and potential of modern smartphones to be used as a navigation assistant (Kuriakose et al., 2020). Based on all these studies, we focused on several design attributes that could improve the navigation experience of users with visual impairments. Furthermore, these attributes (or *the design choices*) were consolidated with a visually impaired user with experience using technology-based navigation aids. The user was actively involved in our requirement analysis and user testing process. This section discusses the design considerations, our proposed system, and its implementation details.

3.1. Design considerations

Various design considerations that we learned to be significant in a navigation assistant for the visually impaired are as follows:

Accuracy and Speed: The trade-off between the accuracy and speed of obstacle detection is vital in a real-time application. Based on this realization, we choose lightweight deep learning models for object detection and scene recognition in our system. Both models offered a balance between accuracy and speed attributes without compromise. Moreover, it is optimal to deploy on a mobile device to have lightweight models, which all favored selecting appropriate models in our application scenario.

Reduction in Latency: Low-latency systems are believed to have the best user experience. Therefore, instead of deploying the deep learning models on a cloud computing server, we decided to deploy them on a mobile device. The smartphone can act as the core of our navigation system, which computes and provides navigation-related information to the user.

Ensured data privacy: Some of the systems proposed in the literature (Rao, et al., 2021; Bai et al., 2017; Mukhiddinov & Cho, 2021) use an external network or cloud computing services to process information about the environment for navigation. But there are potential privacy issues when sending environment data to an external cloud server. In our case, the information captured from the navigation environment would be processed in the local device. Thus, complete privacy of the data is ensured for the user and the environment entities.

Affordability: The main hardware component of our system is a smartphone. Smartphones are standard and miniature computing devices available to any individual. Since we are not dependent on external cloud computing servers or other hardware devices, such as external cameras, it reduces the cost of implementation.

Low Power Consumption: There is no requirement for WiFi or an external data network to operate our system, hence the power consumption could be lower (Tawalbeh, Eardley, et al., 2016). Furthermore, our system does not use external cameras to capture images, as in Ashiq, et al. (2022), Bhowmick, et al. (2014), Kanwal et al. (2015). All these reasons can contribute to the reduction in power consumption.

Portability: Portability is a crucial design attribute that was missed in many navigation assistants. Our system can be deployed on a portable device such as a smartphone. Hence device could be carried easily by a person (in our case, a person with visual impairment) and does not need additional bulky hardware to capture or process data as in Ashiq, et al. (2022), Kanwal et al. (2015), Lin et al. (2019).

3.2. Proposed system

The main components of our system are a smartphone and a bone conduction headset, and six different software modules, namely obstacle detection, distance estimation, position estimation, motion detection, and scene recognition, as shown in Fig. 1. The smartphone camera captures videos of the navigation environment while the user navigates. The video frames are then sent to the software modules. The obstacle detection module results help estimate the position estimation module results, which is why both modules are connected in dashed lines in Fig. 1. After receiving the results from each module, the output module concatenates them and sends them to the user in audio format. The user receives the navigation information through a bone conduction headset. The reason for using bone conduction headphones is that they enable voice directions from the app without losing situational awareness (Lock, Gilchrist, Cielniak, & Bellotto, 2019). Bone conduction headsets use plates on the cheekbones to send sound vibrations directly through the jaw and skull bone to the cochlea in the inner ear. Using two relatively lightweight hardware components, such as a smartphone for capturing and processing information and the bone conduction headset to output navigation information, we ensured the portability and convenience of our navigation assistant. The six software modules are described below, followed by their implementation.

Obstacle detection: This module gets an image acquired by the smartphone camera as input and detects obstacles. Obstacle detection is critical as it helps users avoid collisions and be vigilant along the navigation path.

Distance estimation: To avoid collision with obstacles while navigating, the knowledge about their distances from the user is crucial. The distance estimation module estimates the distance of the obstacles detected by the obstacle detection module. The distance estimation

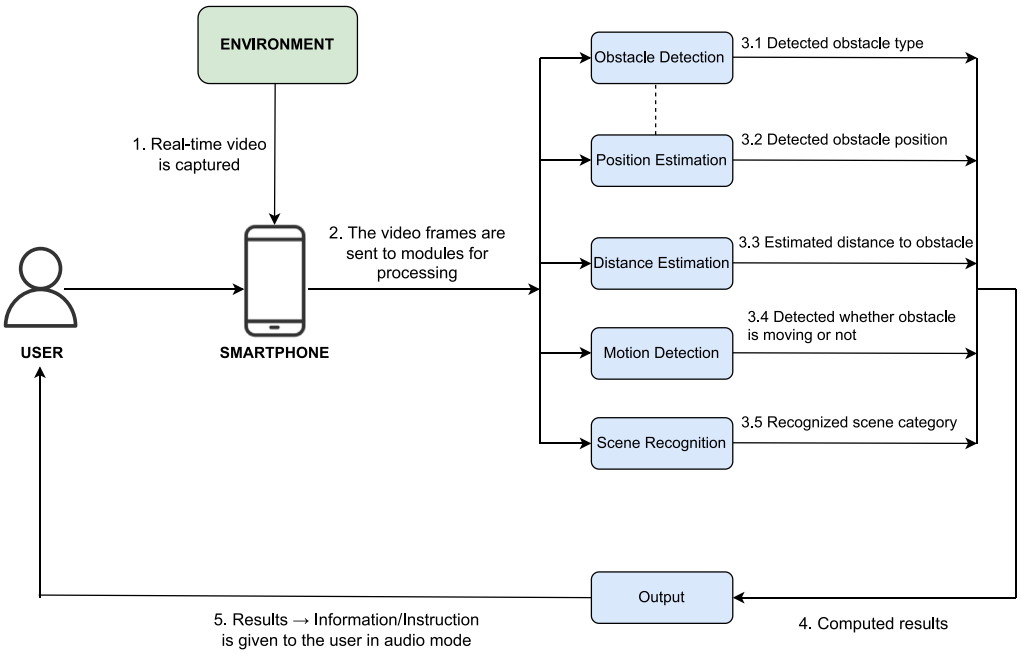


Fig. 1. Modular design of the DeepNAVI navigation assistant.

module shows how far the obstacle is from the user. This can help the user bypass obstacles when they are near.

Position estimation: The position estimation of obstacles gives information about the obstacle's position concerning the user. Position information helps the user locate obstacles and avoid them during navigation.

Motion detection: Motion detection of obstacles helps the user identify whether the obstacle is moving or stationary. In a real-time navigation environment, obtaining information about the movement status of the object is essential as it is valuable information to avoid collisions compared to stationary obstacles.

Scene recognition: Scene recognition can provide a fundamental description of the environment. With a navigation system integrated with a scene recognition module, people with visual impairments could recognize scenes in an emergency, such as *fire escape* or *river*, on the navigation route. Many existing navigation solutions lack this feature (Kuriakose et al., 2021b). From our requirement analysis, we understood the relevance and significance of such a module in the navigation system and therefore included it in our system.

Output: The information from various modules about obstacles and the navigation environment should be given to the user in a suitable output format. The output module converts the results from various modules to an audio format and is given to the user.

3.3. Implementation

3.3.1. Obstacle detection

Several methods from classic machine learning and deep learning are used in the literature for object/obstacle detection. We learned that deep learning-based methods could provide better results by considering the real-time application scenario and the need to deliver the output in minimal time to users. For obstacle detection, we used a lightweight model, EfficientDet-Lite4, from the EfficientDet family. EfficientDet (Tan, Pang, & Le, 2020) model has a scalable framework that expresses the same architecture on different model sizes. EfficientDet uses EfficientNet (Tan & Le, 2019) as the backbone of the network. EfficientNet is a convolutional neural network pre-trained with the ImageNet (Deng, et al., 2009) image database for classification. EfficientDet utilizes several optimizations and backbone tweaks, such as a BiFPN (Bidirectional Feature Pyramid Network) and a compound scaling method that uniformly scales the resolution, depth, and width for all backbones, feature networks, and box/class prediction networks at the same time. Fig. 2 shows the architecture of the EfficientDet obstacle detection model.

The EfficientDet model is evaluated in the COCO (Common Objects in Context) dataset (Lin, et al., 2014), which is considered a general-purpose challenge for object/obstacle detection. According to Tan et al. (2020), the EfficientDet model has been shown to outperform similar-sized models in the benchmark data sets with better mean average precision (mAP) using fewer parameters and less computation. Hence, the model is faster on both the GPU and CPU than other object detectors.

EfficientDet-Lite(versions 0–4) is a family of mobile/IoT-friendly lightweight object detection models derived from the EfficientDet architecture. EfficientDet-Lite models are designed for performance on mobile CPU, GPU, and EdgeTPU and optimized for TensorFlow Lite, an open-source framework for mobile and embedded devices. After comparing the latency and average precision of each version of the EfficientDet-Lite model, we understood that the EfficientDet-Lite4 model is appropriate because of the accuracy and latency offered by the model in a real-time application scenario. The EfficientDet-Lite4 model has the EfficientNet-Lite4 backbone with BiFPN feature extractor, shared box predictor, and focal loss, trained in the COCO 2017 dataset (Lin, et al., 2014).

3.3.2. Distance estimation

Researchers have explored various distance estimation methods using additional sensors and external cameras. As a part of our design choice, we researched various distance estimation methods that can be used with a smartphone alone. Thus, we considered *Rule of 57* for the distance estimation module in our system. We selected this method due to our detailed experiment reported in Kuriakose, Shrestha, and Sandnes (2021a). Moreover, the method could be implemented with a smartphone camera alone, thus offering portability and convenience to the navigation assistant.

The *Rule of 57* indicates that an object (obstacle) with an angular size of 1° is about 57 times farther away than it is significant (see Fig. 3). Therefore, the ratio of an obstacle's angular size (in degrees) to the whole 360-degree circle should equal the ratio of the obstacle's actual size to the circle's circumference at that distance from the

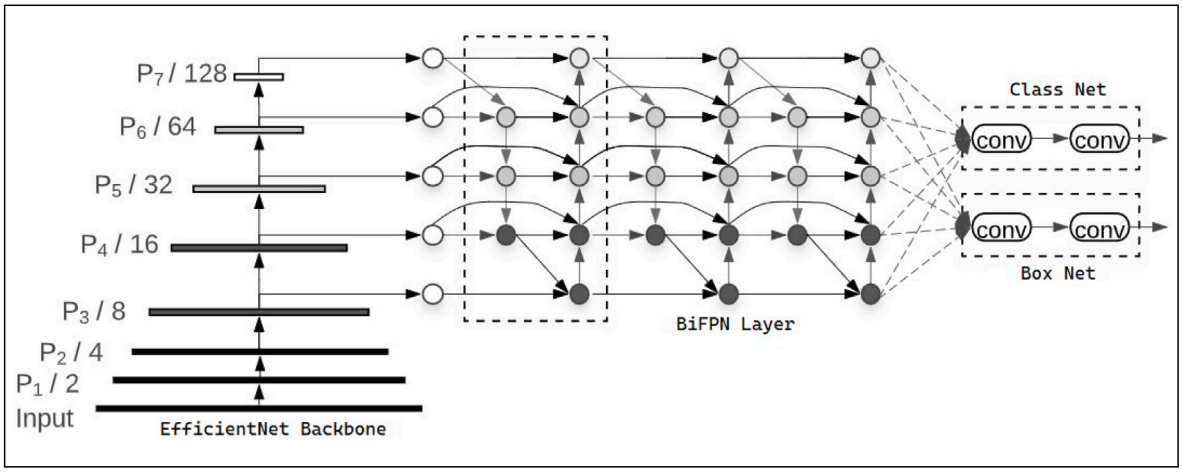


Fig. 2. The EfficientDet model that uses EfficientNet as the backbone network and a BiFPN feature network.
Source: Inspired from Tan et al. (2020).

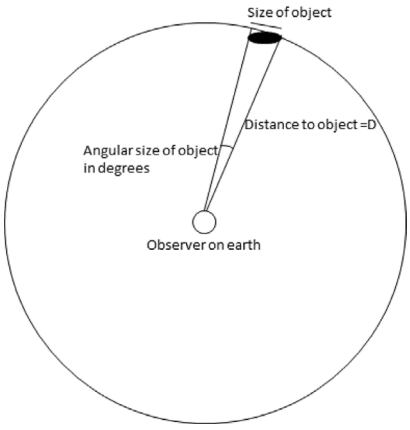


Fig. 3. The Rule of 57.
Source: Adapted from Harvard (2012).

observer. This method has been derived from measuring the distance and angles of the images of telescopes in astronomy (Harvard, 2012). The key to using telescope images to measure distances is to realize that an obstacle's apparent angular size is directly related to its actual size and distance from the observer. This means that the obstacle appears smaller as it is farther away from the observer. However, our experiments found that it can be applied to find the distance to the obstacle even if the obstacle's angular size is more than 1° to the field of view of the smartphone camera sensor.

From Fig. 3, we can write that

$$\frac{angular\_size}{360^\circ} = \frac{actual\_size}{2\pi D} \tag{1}$$

Using the distance of the obstacle for the angular size and the distance of the obstacle for the actual size, we get the following equation to calculate the distance.

$$object\_distance = (object\_size) \times \frac{1}{(angular\ size\ in\ degrees)} \times 57 \tag{2}$$

To use this approach, estimating the obstacle's size is necessary before finding its distance. To calculate the same as that in Eq. (2), we used the method illustrated in Fig. 4. Then the height of the obstacle (H) is calculated as

$$H = \frac{h(\tan(A) + \tan(B))}{\tan(B)} \tag{3}$$

The geomagnetic field sensor, accelerometer, and gyroscope present on the smartphone give the angular size.[1,2,3]

3.3.3. Position estimation

The obstacle detection model returns an array of four numbers representing a bounding rectangle that surrounds its position for each detected obstacle: [top, left, bottom, right]. The image space is divided into three regions *(left, center, and right)*, and then the area where the detected obstacle covers most is found. The position estimation module returns the region covering the central part of the obstacle as its position.

3.3.4. Motion detection

Our motion detection module is inspired by the work described in Moo Yi, Yun, Wan Kim, Jin Chang, and Young Choi (2013) which involves three main steps. Initially, pre-processing on the image is performed with simple spatial Gaussian filtering and median filtering on the image. Then, a dual-mode SGM (single Gaussian model) is performed for background modeling. Finally, a tuned motion compensation using Kanade–Lucas–Tomasi (KLT) is performed for the background movements by mixing models. This method was tested on a smartphone and proved that the time taken to compute the results is less than other similar methods (Moo Yi, et al., 2013).

3.3.5. Scene recognition

We proposed a scene recognition model, SceneRecog, in our previous work (Kuriakose et al., 2021b). We used the same model here but with extended and updated scene classes relevant to navigation. We used the EfficientNet-Lite4[4] model by employing the transfer learning technique with the 20 custom scene classes that can commonly occur in indoor and outdoor navigation environments. The scene classes used are described in detail in Section 3.3.7.

We also incorporated a threshold parameter to minimize false positives with unknown scenes in our implementation. The module reports a scene as *unknown* when the probability is below the threshold value. The threshold was adjusted to 0.7 after trials and errors.

[1] https://developer.android.com/guide/topics/sensors/sensors_position
[2] https://github.com/SensingKit/SensingKit-iOS
[3] https://stackoverflow.com/questions/15949777/how-can-we-measure-distance-between-object-and-android-phone-camera
[4] https://blog.tensorflow.org/2020/03/higher-accuracy-on-vision-models-with-efficientnet-lite.html

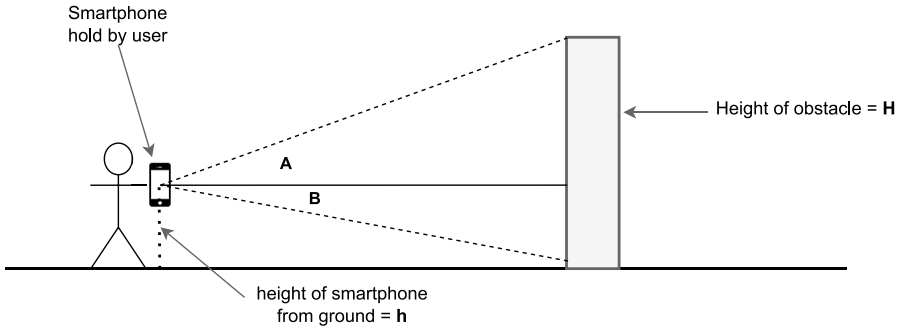


Fig. 4. Estimation of smartphone height from the ground.

3.3.6. Output

The output module converts the textual information about the obstacles and the scene obtained from various modules to audio format and fed to the user through a bone conduction headset. We used Python's Text-to-Speech (TTS) library, *Pyttsx*[5] in the output module to convert textual information into audio output. *Pyttsx* works seamlessly offline on multiple platforms.

3.3.7. Custom datasets

We created a custom dataset comprising 20 different types of obstacles relevant to indoor and outdoor navigation environments to be used in our obstacle detection module. They are *Bench, Bicycle, Billboard, Bookcase, Cabinetry, Car, Chair, Dog, Door, Fire hydrant, Furniture, Kitchen appliance, Person, Plant, Stairs, Stop sign, Table, Traffic light, Tree* and *Waste container*. The dataset was created by collecting images from four different sources, Google Open Images V6 (Kuznetsova, et al., 2020), ImageNet (Deng, et al., 2009), LISA Traffic Sign Dataset (Mogelmose, Trivedi, & Moeslund, 2012), and our own images.

The Google Open Images V6 dataset is used mainly for object detection or segmentation-related research. The ImageNet project is vital in advancing computer vision and deep learning research. The LISA Traffic Sign dataset is a set of images and videos containing annotated frames of US traffic signs. Images for the 20 obstacle classes were extracted from these data sources. After examining the extracted images, we found that many of the images require some preprocessing, such as relabeling. Furthermore, we collected some sets of images from localities and labeled them manually using externally available tools.[6]

For the scene recognition module, we identified 20 common scene categories that are commonly found in indoor and outdoor navigation environments: *Balcony, Basement, Bridge, Bus station, Cafeteria, Classroom, Construction site, Crosswalk, Fire escape, Hospital room, Kitchen, Library indoor, Parking Lot, Playground, Railway track, Reception, River, Shopfront, Street, Supermarket*. The dataset is created by collecting images from three main sources, MIT's Places365 (Zhou, Lapedriza, Khosla, Oliva, & Torralba, 2017), Google Open Images V6 (Kuznetsova, et al., 2020), and Flicker.[7] Moreover, additional images are added to have some real images from the locality to increase the number of images in the dataset and, in turn, improve the model's performance.

4. Android application

An android application is the central part of our DeepNAVI navigation assistant. The trained obstacle detection and scene recognition models were converted to TFLite format and deployed in the android app along with integrating other modules.

The app has an incorporated voice assistant feature. The voice assistant is designed to recognize two different voice instructions and activate the app accordingly. When the user says *Activate Navigation*, the app is activated in navigation mode and scans the environment. The app then provides information about the obstacles in front of the user via audio mode in real-time. The app also displays recognized obstacles and related information as text on the smartphone screen. If the user wants to know the scene of the navigation environment, that can be obtained from the app by giving the *Identify scene* voice command. Then the scene identification module will be activated to recognize the scene and give the output again in audio modality. The working of our android application is illustrated in Fig. 5. The results from the app in two different modules (obstacle detection and scene recognition) are shown in Fig. 6.

5. Experiments and results

This section describes the experiments conducted to evaluate the five core modules: obstacle detection, scene recognition, distance estimation, motion detection, and position estimation, and their results. The section then describes the user testing procedures and the feedback received from the pilot test with a user. Subsequently, our navigation assistant is compared with other similar navigation systems in terms of various features and functionalities.

5.1. Evaluation of object detection and scene recognition models

We used an Intel Xeon processor with 64 GB RAM and an NVIDIA GeForce GTX 1080 Ti GPU to train both the deep learning models for obstacle detection and scene recognition. The experimental platform settings are TensorFlow-GPU 2.4, NVIDIA CUDA toolkit 11.0, and CUDNN 8.1. The models are trained, validated, and tested by randomly shuffling and splitting the data set in the 80:10:10 ratio, respectively. The performance of the obstacle detection model and the scene recognition model is given in terms of the accuracy metric in Tables 1 and 2 respectively. Fig. 7 shows few test results from the obstacle detection module. The results illustrate how the model detects various obstacles.

The accuracy of the obstacle detection model is 87.8%. The accuracy of most trained obstacle dataset images is good (above 80%), although some classes (such as cabinetry and stairs) need improvement (see Table 1). The test results in Fig. 7 show the model's performance in different instances.

The results of the indoor navigation environment (see the first row of Fig. 7) show that the model can recognize most of the obstacles in the environment. In Fig. 7(b), it can be seen that the *stairs* are not detected, but other obstacles are detected. Maybe the model could not detect it because of the image's proximity to *stairs*. Furthermore, in Fig. 7(c), the shelf on top of the *kitchen appliance* is incorrectly detected as a *bookcase*. This could be due to the similarity of both obstacles in terms of color and other features, which caused the model to give an incorrect result.

[5] https://pypi.org/project/pyttsx3/
[6] https://github.com/tzutalin/labelImg
[7] https://www.flickr.com/

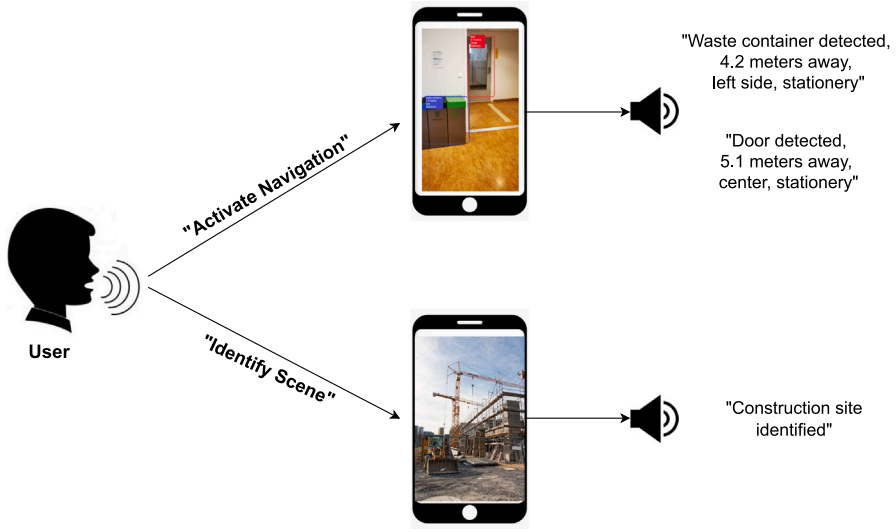
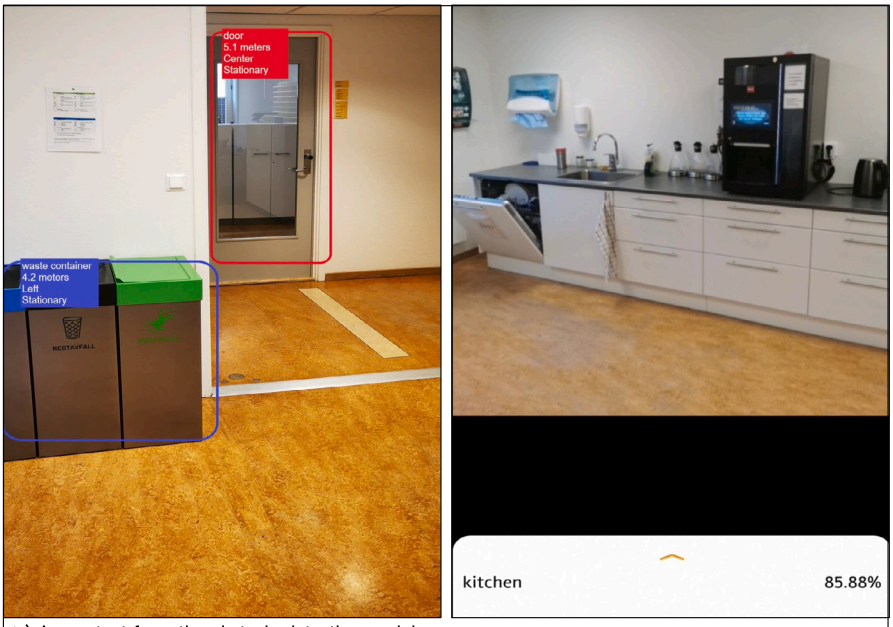


Fig. 5. Working of DeepNAVI android application.



(a) App output from the obstacle detection module. The blue bounding box detects the *waste container* obstacle and says it is 4.2 meters away, left, stationery. The red bounding box detects the *door* obstacle and says it is 5.1 meters away, center, stationary.

(b) App output from the scene recognition module. The module detects the scene as *kitchen* with 85.88% accuracy.

Fig. 6. Results of the smartphone app from obstacle detection and scene recognition modules.

The results from the outdoor navigation environment (see the second row of Fig. 7) show some instances where the model did not recognize obstacles. From Fig. 7(d), it is visible that the model faced some challenges in detecting white *car* and white *door* correctly. In Fig. 7(f), the model incorrectly detects the person in the orange dress as *fire hydrant*. One of the possible reasons for this case might be due to the inability of the model to differentiate the color similarity of the orange dress with *fire hydrant*.

The sample results from the scene recognition model are shown in Fig. 8. The results give an overview of the performance of the scene recognition model with different scene classes predicted from the test dataset. The *green* label indicates the correct prediction of the scene recognition model, and the *red* label indicates the incorrect class predicted by the model. The scene recognition model can give an accuracy of 85% with our dataset. The scene classes that give low accuracy (less than 80%) are *classroom, construction site,* and *reception* (see Table 2).

5.2. Evaluation of distance estimation module

The distance estimation module is evaluated through experiments, where some obstacles of varying sizes were placed at four different distances (1 m, 3 m, 5 m, and 10 m). And the estimated distances from the distance estimation module are compared with the group truth. Table 3 shows the actual and estimated distances of the obstacles. The results

(a) A red bounding box has detected *Door*. Green bounding box has detected *Stairs*. Blue bounding box has detected *Cabinetry*.

(b) An orange bounding box has detected *Dog*. Two maroon bounding boxes have detected *Plants*.

(c) Two green bounding boxes have detected *Cabinetry*. Red bounding box has incorrectly detected *Bookcase*. Yellow bounding box has detected *Kitchen appliance*.

(d) Three maroon bounding boxes have detected *Cars*. Blue bounding box has detected *Fire hydrant*.

(e) Two orange bounding boxes have detected *Persons*. Blue bounding box has detected *Waste container*.

(f) Two green bounding boxes have detected *Cars*. Blue bounding box has detected *Traffic light*. red bounding box has detected *Person*. Grey bounding box has incorrectly detected *Fire hydrant*.

Fig. 7. Example test results from obstacle detection module.

Table 1
Accuracy of obstacle detection.

Class	Accuracy
Bench	93.7
Bicycle	88.5
Billboard	84.6
Bookcase	90.2
Cabinetry	78.8
Car	92.1
Chair	94.7
Dog	87.5
Door	83.2
Fire hydrant	88.9
Furniture	87.5
Kitchen appliance	92.7
Person	84.7
Plant	86.6
Stairs	79.5
Stop sign	91.7
Table	90.9
Traffic light	83.7
Tree	81.5
Waste container	94.6
Average	87.7

Table 2
Accuracy of scene recognition.

Class	Accuracy
Balcony	86.2
Basement	81.4
Bridge	88.2
Bus station	83.6
Cafeteria	88.1
Classroom	79.4
Construction site	78.5
Crosswalk	85.5
Fire escape	84.4
Hospital room	83.3
Kitchen	87.3
Library indoor	84.7
Parking lot	89.6
Playground	84.3
Railway track	88.4
Reception	77.8
River	86.5
Shopfront	88.7
Street	84.7
Supermarket	91.2
Average	85.0

are estimated from the experiments to analyze the performance of the distance estimation module on obstacles of varying sizes at varying distances. During the test case of 1 m, it is visible that small obstacles (such as *chair*, *fire hydrant*, *table*, and *waste container*) have a high deviation from the actual distance compared to other bigger obstacles. The same phenomenon can be observed when the actual distance is 3 m and 5 m. The results in the 5-meter case are estimated after averaging the result from five consecutive outputs since the method could not provide a fixed result. We could not find distances when the obstacle was placed more than 5 m away. That is one reason why no results were reported for 10 m case.

5.3. Evaluation of motion detection and position estimation modules

The motion detection is evaluated with some moving objects. Fig. 9 illustrates objects (class *people*) in motion with two frames in a video. The first figure indicates that almost all moving obstacles in the frame were detected by the module, but the second figure indicates that there were also a few misinterpretations. The movement status of obstacles will be detected, and the users will be indicated about it along with other information related to the obstacles such as its type, distance from the user, etc.

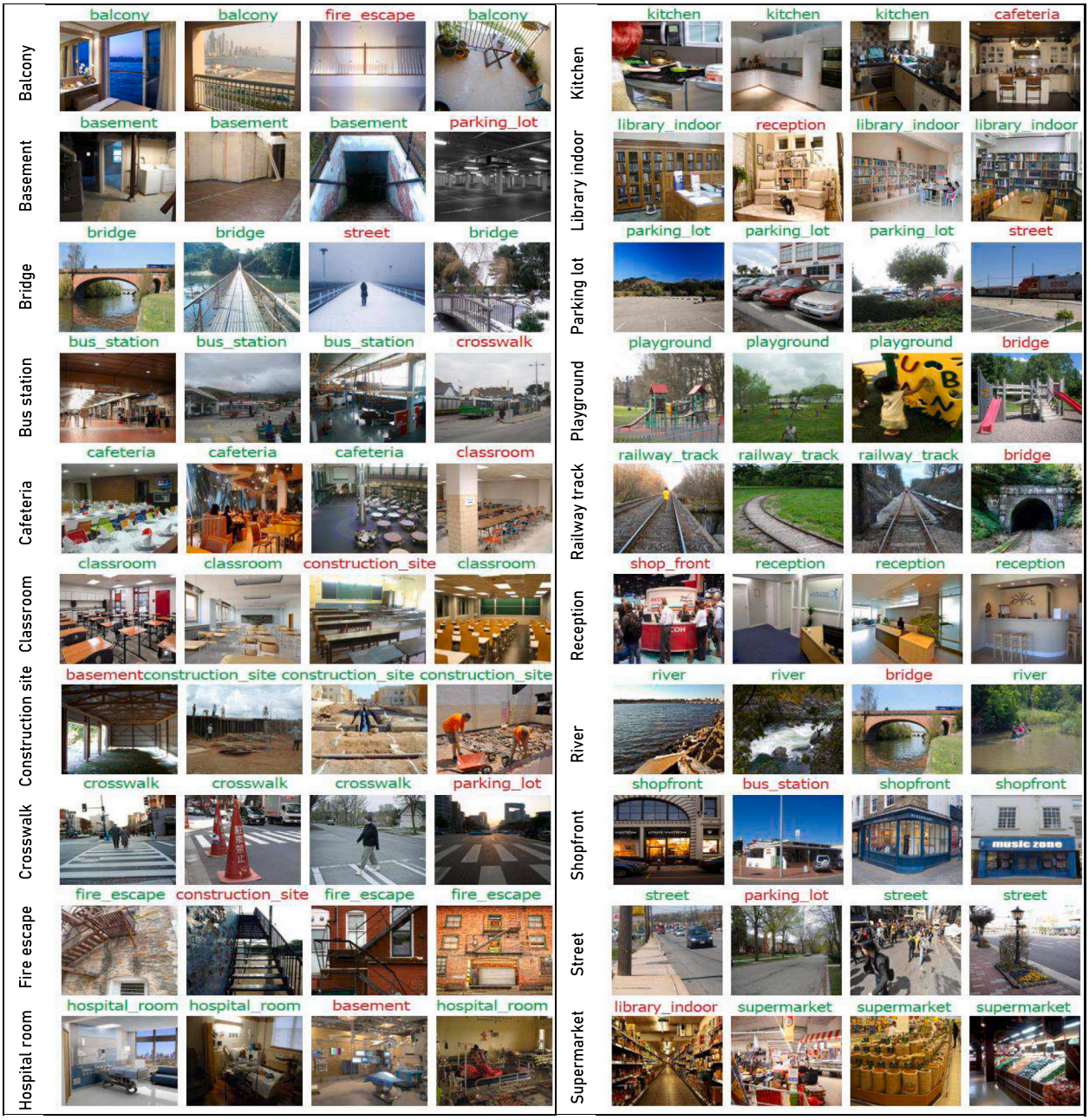
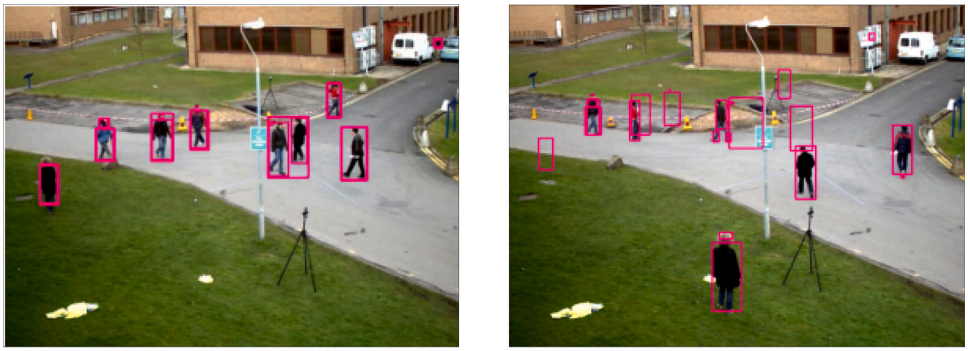


Fig. 8. Example test results from the scene recognition module. The green labels indicate correct detections, and the red labels indicate incorrect detections.



(a) (b)

Fig. 9. Example results showing motion detection from two different frames of the same video.

Table 3
Results from the distance estimation module.

Obstacle	Estimated distance when actual distance was		
	1 m	3 m	5 m
Bench	0.9	2.6	4.5
Billboard	0.9	2.5	4.6
Bookcase	0.9	2.5	4.2
Chair	0.8	2.7	4.1
Door	0.9	2.3	4.5
Fire hydrant	0.8	2.2	3.6
Person	0.9	2.8	4.3
Table	0.8	2.2	4.3
Tree	1.1	2.8	4.6
Waste container	0.8	1.9	3.8

The position estimation module results indicate that it can give the results as intended without any problems (see Fig. 6). The module gives the position of various obstacles, such as *waste container* and *door*.

5.4. User testing

The purpose of user testing is to ensure that the navigation assistant system is working correctly and to assess our navigation assistant's practicality and usability with a user who is legally blind. The user testing experiment was carried out in two test cases: (1) Navigation with the proposed smartphone-based assistant (DeepNAVI) and (2) Navigation with a smart cane with which the user was familiar. These two test cases were designed to compare the effectiveness of both tools from a user perspective. The two test cases were conducted in the same experimental setup, and the results are compared to analyze the effectiveness of the two tools. The experimental setup, process, and the two test cases are described in the following subsection, and the results are finally presented.

Experimental Setup and Process: The test experiment was conducted in an on-campus indoor building. A navigation path of about 100 m was created with various obstacles placed at different points. The environment or test location was decided after considering multiple factors, such as user safety, placement of obstacles that the trained model deployed in the app can detect, and weather concerns. We also consider the security protocols existing in the current pandemic situation due to COVID-19. According to the user, the smart cane does not perform well in conditions such as rain or snow. Considering all these reasons, we decided to select an indoor test location on campus.

The selection of a smart cane to compare the proposed smartphone-based navigation assistant was based on a few reasons. The smart cane is one of the commonly used non-conventional navigation assistants. The participant from the experiment has been using the smart cane for daily navigation for a while and is comfortable with the usage. Consequently, we decided to compare and analyze the comfortability offered by the smart cane against our proposed smartphone-based navigation assistant from a user point of view.

The participant's goal is to travel from a starting point to the destination point. The destination point is defined by an obstacle (a refrigerator (class *kitchen appliance*)) and is already communicated to the participant during the pre-experiment phase. The participant was not familiar with the navigation environment before the test. Various obstacles such as *doors, chairs, tables, billboards, waste bins, cabinetry,* and *kitchen appliances* were present along the navigation path to add more complexity to the test environment. The test location was not blocked from external interferences. Therefore, people can enter the test location at any time. This arrangement is made to simulate much closer to a real-time navigation environment. In such a situation, the mobile app can also detect *persons* and provide information about that particular obstacle to the user. A timer is set to calculate the time it takes for the participant to reach the destination in each of the test cases. Before the experiment, instructions were given to the user about the plan, the tasks, and the goals to be achieved. There was no private data collection during any stage of the experiment. No approval from the ethics committee was required to experiment.

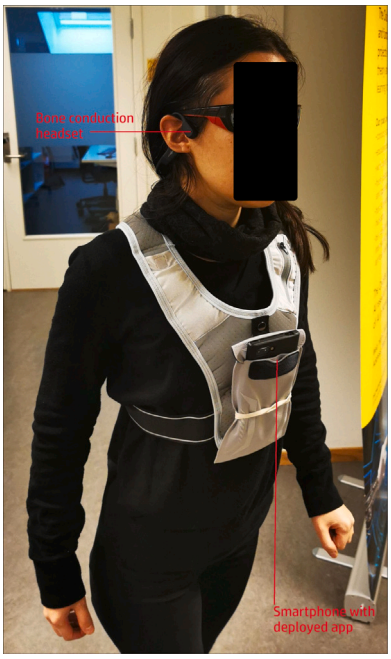


Fig. 10. User testing setup of the navigation assistant. The app is deployed in the smartphone and is placed in the vest where the camera is facing outwards. The user has a bone conduction headset for receiving audio instructions.

After the whole experiment was completed, post-experiment questions were asked to the user related to the experience of using our smartphone-based navigation assistant and the smart cane. Questions about the overall experience of using the navigation aids, difficulties encountered, suggestions/additional features the user considers appearing on the app, etc., were asked.

Test cases: This section describes in detail the two test cases of our pilot study.

(1) Navigation with the proposed smartphone-based assistant (DeepNAVI): A smartphone is placed in a vest with a phone holder with a small hole opening for the smartphone camera, which ensures the camera in a fixed position, reduces the effect of getting it tilted or rotated on image acquisition. The user wears a bone conduction headset connected to the smartphone via Bluetooth. This arrangement ensured that the user did not need to carry anything else and could be free-handed. The proposed DeepNAVI navigation assistant worn by the user is shown in Fig. 10. The user testing setup includes the smartphone with the app installed and the bone conduction headsets. Fig. 11(a) shows a photo taken when the user navigates with the help of the DeepNAVI navigation assistant.

(2) Navigation with the smart cane: The smart cane used in the experiment was WeWalk.[8] The user was familiar with the usage and practicalities of the smart cane, as the device was the default navigation assistant for the user. The cane uses ultrasonic sensors to detect the presence of obstacles. When obstacles are around, the cane handle starts to vibrate, helping the user notice obstacles. Fig. 11(b) shows the user's image navigating with the smart cane's help.

Results: This section describes the results of the two test cases of the experiment.

(a) Using the DeepNAVI navigation assistant: The app provided information about obstacles in the user's path. However, there were a few incorrect outputs from the app for detecting obstacles along the path.

[8] www.wewalk.io/en/

(a) Navigation using DeepNAVI navigation assistant.

(b) Navigation using the smart cane.

Fig. 11. Photos of the participant navigating with DeepNAVI navigation assistant and smart cane.

At one point, the app incorrectly identified a *photocopier machine* on the indoor path as a kitchen appliance (class *refrigerator)*. But when the user touched the obstacle, they realized it was a *photocopier*. On the other hand, the user was happy to know about the presence of an obstacle so that a possible collision was avoided. Also, with the scene recognition feature of the DeepNAVI, the user found it helpful to recognize the *kitchen* environment. Because of that, it was helpful for the user to locate the end goal of the experiment, that is, to locate the *refrigerator* in the kitchen. The time to complete the test case was 3.5 min.

(b) Using the smart cane: Even though the user was repeating the same path for the second test case with the smart cane, the user got lost at one point, trying to identify the objects/obstacles around. The user was assigned a *door* as a point on the spot to determine the destination area from the experience of the previous test case. However, using the smart cane, the user found it challenging to locate *door* on the navigation path and, therefore, the route to the destination point. However, fortunately for the user, a person entered the test area during the experiment, which helped the user spot the *door*'s location and thus the destination. Also, with the smartcane, it was difficult for the user to understand the environment (such as *kitchen*) where the end goal (to identify the *refrigerator*) was located. Hence, the user needs to spend more time than in the previous test case to accomplish the task goal. The time to reach the destination with the smart cane was 5.5 min, more than the time taken with our navigation assistant.

The user was optimistic about the features supported and the convenience offered by the DeepNAVI navigation assistant. Few relevant comments from the user by comparing the experiences with both navigation assistants are mentioned below. Here, the smartphone assistant refers to the proposed DeepNAVI navigation assistant, and the cane refers to the WeWALK smartcane used for the experiment.

> *"I felt comfortable using the smartphone assistant since I have nothing to carry and have freehanded. But while using the cane, I should always need to dedicate one of my hands to the cane".*

> *"I like the feature of knowing obstacles, the distance to them, and other information while using the smartphone assistant. It helped me to keep vigilant about various obstacles on my path".*

> *"I like the feature of telling me about the scene where I am located. When the app said I was in the 'kitchen,' it helped me to finish the end goal easily".*

> *"I felt carrying a cane was difficult for me because of its weight. I might get tired easily if I use it for a long time".*

The user also provided some improvisations that could be considered for the future versions of DeepNAVI, such as (1) including more obstacles to be detected by the navigation assistant, (2) the user should have the possibility to adjust the rate, pitch, or voice used for the app's feedback, (3) in addition to the audio feedback, it would be good to have another feedback modality, such as vibrations/haptic, and a pleasant sound (such as relaxing music) should be played continuously in the background, at a low sound volume while the app is working to ensure that the app is working. It should not be interrupted by another system-level application on the smartphone.

5.5. Comparison of navigation systems

This section gives a comparative analysis of navigation systems similar to our proposed navigation assistant. The comparison is based on qualitative attributes that should be present in a navigation assistant for the visually impaired. These features are described in Table 4. The features were selected and finalized through user discussions through a requirement analysis study. These features could give an idea of the important characteristics that users are looking for in a navigation assistance system.

The comparison analysis between various navigation systems and our proposed DeepNAVI assistant according to the feature table (see

Table 4
Characteristics/features used for comparative analysis and its respective criteria.

Feature	Analysis criteria
Portability	The system should be a single portable device convenient to carry by the user. No external cameras, sensors, or wires should be present.
Lightweight model	If the system uses a deep learning-based model or any computer vision model, it should be lightweight and run on a miniature device without much delay for a real-time operation.
Data network undependability	The system should not depend on an external data network or WiFi for processing or giving outputs.
Coverage (indoor/ outdoor)	The system should work indoors and outdoors.
Obstacle recognition	The system should be able to recognize the type of obstacle or at least detect the presence of an obstacle during the navigation path.
Distance estimation	The system should be able to estimate an approximate distance to the obstacles.
Position estimation	The system should be able to output the obstacle's position.
Scene recognition	The system should be able to recognize the scene (indoor and outdoor) during navigation.
Motion detection	The system should be able to detect if the obstacles are moving.
Multimodal feedback	The system should be able to provide two or more output options.

Table 5
Comparison of features of DeepNAVI with similar systems.

Sl. No	System	Portability	Lightweight model (Real-time)	Data network undependability	Coverage (indoor/ outdoor)	Obstacle recognition	Distance estimation	Position estimation	Scene recognition	Motion detection	Multimodal output
1	Lin et al. (2019)			✓	✓	✓					
2	Kanwal et al. (2015)			✓		✓	✓	✓		✓	
3	Bhowmick, et al. (2014)			✓		✓	✓				
4	Moharkar et al. (2020)			✓	✓	✓	✓				
5	Ashiq, et al. (2022)		✓	✓	✓	✓					
6	Joshi et al. (2020)		✓	✓	✓	✓	✓				
7	Kahraman and Turhan (2021)			✓	✓						
8	Barontini, et al. (2020)			✓	✓	✓					✓
9	Megalingam, et al. (2015)		NA	✓	✓		✓				
10	Guerrero et al. (2018)		NA	✓	✓		✓				
11	Saaid et al. (2016)		NA	✓			✓				
12	Rao, et al. (2021)	✓	✓		✓	✓			✓		
13	Mukhiddinov and Cho (2021)	✓			✓	✓					
14	Suresh, et al. (2017)			✓		✓	✓				
15	Bai, et al. (2019)	✓	✓		✓	✓					
16	Lin et al. (2017)	✓	✓	✓	✓	✓	✓				
17	Bai et al. (2017)				✓	✓					
18	Fusco and Coughlan (2020)	✓	NA	✓							
19	Ganz, et al. (2014)	✓	NA								
20	Croce, et al. (2014)	✓	NA	✓							
21	Tapu et al. (2013)	✓	✓	✓	✓	✓				✓	
22	Peng et al. (2010)	✓	NA	✓							
23	DeepNAVI (Proposed System)	✓	✓	✓	✓	✓	✓	✓	✓	✓	

Table 4) is shown in Table 5. Our DeepNAVI navigation assistant is enriched with more features than other navigation systems. Several systems were developed to work both indoors and outdoors. However, only a few systems used lightweight deep learning models for obstacle detection, which requires working on portable devices in real-time with shorter response times. There is only one system with multimodal output in the comparative study (see Table 5).

6. Discussion

This section provides an elaborated discussion of system performance, experiments conducted with the user, and the results received at various stages.

The obstacle detection model's low accuracy of some obstacle classes may be due to duplicate detection, misclassification, mislocalization, or misclassification and mislocalization. However, the detection accuracy and the inference time to compute the results in a real-time application should be balanced, and we must compromise. It is visible that the model faces a few issues with the white objects (see Fig. 7(d)). One possible reason for this model's performance degradation could be the daylight conditions of the image taken in the outdoor environment. Also, there are some false detections reported by the model at various instances (see Fig. 7(f)). Training the model with a large dataset involving various colored obstacles in the same category could resolve this issue.

A limitation of the distance estimation module is that its results are less accurate than those obtained via stereo vision cameras or other distance estimation devices/sensors. Although, it can provide the user with a sense of distance from obstacles, allowing them to navigate more cautiously. The main advantage of this single-camera-based method is that it can work even on a smartphone without any other external hardware being connected to the system. This ensures portability and hence convenience for the user during navigation. It should also be mentioned that the distance estimation method used for this system cannot provide results when obstacles are more than 5 m from the user/smartphone. This could be a limitation compared to

other methods that use external devices/cameras/sensors to estimate distance.

Our results suggest that the motion detection module needs to improve the response time when working in a real-time environment. The results indicate that the module can accurately detect the state of motion in many cases. But, as mentioned, we had performance issues with speed in computation. Despite this, on the positive side, we realize that as the motion status of an obstacle is introduced, it can improve the navigation experience of users with visual impairment. Thus the module can help make decisions for a user during navigation and act accordingly.

The position of obstacles is detected correctly in our experiments. We defined three different positions based on the relative space of the navigation space to the display space of the smartphone. The method used here could also be improved with more positions such as the top for hanging obstacles and the bottom (or similar) for ground-level obstacles. We must admit that, due to the arrangement of our navigation assistant in the vest, it is challenging to capture ground level and hanging obstacles when they are in proximity to the user.

From the test results (see Fig. 8), it is visible that there are few misclassifications associated with the scene recognition model. And these misclassifications can occur due to labeling ambiguity, the similarity between images, and overlap between two categories in the same scene. A larger dataset with more scenes and accurate labels can mitigate these issues. Our previous work (Kuriakose et al., 2021b) provides a list of possible solutions. Despite this, adding a scene recognition module could be helpful to the user during navigation as it would allow them to learn more about the environment in which they are navigating.

Pilot testing with the user gives an impression of the practicality and usefulness of our system. It was easier for the user to get an idea of the obstacles and avoid them with the additional information provided by the system. Since the system is portable, the user was optimistic about that. Our smartphone-based navigation assistant took the user less time to reach the destination than a smart cane. We received feedback after the evaluation that, in a real-life environment, it would be more useful if the user could know what obstacles to avoid while navigating. Awareness of the obstacles in a user's environment could help the user get to the destination safely. After the user-level testing, the feedback reflects that DeepNAVI performs well and enables the user to reach the destination by providing valuable information such as obstacle type, distance, and position. In contrast with similar navigation systems, DeepNAVI can provide environmental information regarding obstacles and the surrounding environment.

We considered this pilot testing to collect valuable suggestions from the user to further develop and refine our navigation assistant. Since the testing process was done in a controlled environment, we know the experiment's limitations. And at the same time, we got constructive feedback from the user for improving the system. Because of the ethical complications and challenging concerns of recruiting more visually impaired participants, we conducted this testing procedure with a single user. But we plan to extend and elaborate the testing environments and procedures by including more participants in our future work.

The comparative analysis with the quantitative attributes in Table 5 shows that our proposed system is enriched with many features that are not currently present in similar systems. This shows how a portable system such as DeepNAVI is valuable and practical for navigation assistance in indoor and outdoor environments. Based on the comparative analysis, it is evident that we lack multimodal output functionality in our current system version. But we intend to include it in the extended version of our navigation assistant in the future.

Many systems proposed in the literature used miniature hardware boards such as Raspberry Pi as the central processing component. We analyzed the design of such systems and interacted with people with visual impairments regarding the same concept. To our knowledge, those systems are not available in the market, so we could not compare them in our experiments. Such systems need external cameras and sensors to make them a deployable solution. And this involves wires to connect each component to the board. All these factors could question the concept of *portability*. During the interactions with people with visual impairments, we also got responses about the inconvenience caused by the wires entangled all over the body. Therefore, we suspect a system that uses external cameras and connecting wires is not a portable and convenient solution for people with visual impairments to navigate. Moreover, through this research, we aim for a more viable solution that users can use in a real-time environment without much hassle. Hence, the design search concluded with a smartphone-based solution that is already integrated with cameras, sensors, and processing units.

Furthermore, as DeepNAVI does not require any network connectivity, it is more convenient for the users to avoid potential network delays that might arise in a system that utilizes an external server or a cloud service process environment data while navigating. Moreover, depending on an external data network for retrieving results would not always work in areas with no network access (such as underground stations, basements, etc.). But there are few opportunities and scope in having internet connectivity. It can be used for on-the-go model training and collecting relevant images for the dataset by the users. The real challenge in such a scenario is regarding the privacy and security concerns that arise while capturing images from public and private environments or people without concern. Hence, such a data collection and model training plan is another potential research direction to explore.

7. Conclusion

The proposed smartphone-based navigation assistant offers a convenient solution that can work in real-time. Besides providing information on detected obstacles, the assistant also provides necessary information such as distance, position, motion status, and scene information during navigation. The solution also offers convenience, portability, and comfort since users do not need to carry any additional hardware. The results show that our models can perform well in a real-time environment without relying on the external data network. This makes our proposed navigation assistant may be helpful in situations where data networks are not readily available. The current system version can detect 20 different types of obstacles and 20 scenes relevant to indoor and outdoor navigation environments. The system can be extended to detect more obstacles and scenes if necessary after collecting and training additional datasets. A comparison of our navigation assistant with similar systems indicates it provides many features not available in similar systems, including scene recognition, motion detection, position estimation, etc. The pilot testing with a real user validates that our system could be a promising solution for navigation assistance for visually impaired people.

Future enhancements to the proposed solution may include adding features such as multimodal output and a more enriched voice assistant that provides seamless navigation. In addition, adding a reinforcement component to the deep-learning models can be considered, which can be used to re-train models on the go as observed by the users. This might be used to generate a larger dataset and improve the classification and detection accuracy of models. Furthermore, based on the feedback from the user testing, we plan to refine the proposed DeepNAVI system and then conduct elaborated user testing with more users.

CRediT authorship contribution statement

Bineeth Kuriakose: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Resources, Data curation, Writing – original draft, Writing – review & editing, Visualization. **Raju Shrestha:** Conceptualization, Writing – review & editing, Supervision, Project administration, Funding acquisition. **Frode Eika Sandnes:** Writing – review & editing, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The data that has been used is confidential.

Acknowledgments

The authors would like to thank the user for the fruitful discussions that helped in developing and enhancing the DeepNAVI and for participating in the user testing, as reported in the paper.

References

Ali A., H., Rao, S. U., Ranganath, S., Ashwin, T., & Reddy, G. R. M. (2021). A google glass based real-time scene analysis for the visually impaired. *IEEE Access*, *9*, 166351–166369. http://dx.doi.org/10.1109/ACCESS.2021.3135024.

Ashiq, F., Asif, M., Ahmad, M. B., Zafar, S., Masood, K., Mahmood, T., et al. (2022). CNN-based object recognition and tracking system to assist visually impaired people. *IEEE Access*, *10*, 14819–14834. http://dx.doi.org/10.1109/ACCESS.2022.3148036.

Aspinall, A. (2012). Assistive technology: Principles and application for communication disorders and special education. *Journal of Assistive Technologies*, http://dx.doi.org/10.1108/17549451211285799.

Austin, K. (2016). White cane vs. Guide dog. Second Sense. URL https://www.second-sense.org/2016/09/white-cane-vs-guide-dog-why-or-why-not/. (Accessed 01 October 2020).

Bai, J., Liu, Z., Lin, Y., Li, Y., Lian, S., & Liu, D. (2019). Wearable travel aid for environment perception and navigation of visually impaired people. *Electronics*, *8*(6), 697. http://dx.doi.org/10.3390/electronics8060697.

Bai, J., Liu, D., Su, G., & Fu, Z. (2017). A cloud and vision-based navigation system used for blind people. In *International conference on artificial intelligence, automation and control technologies* (pp. 1–6). http://dx.doi.org/10.1145/3080845.3080867.

Barontini, F., Catalano, M. G., Pallottino, L., Leporini, B., & Bianchi, M. (2020). Integrating wearable haptics and obstacle avoidance for the visually impaired in indoor navigation: A user-centered approach. *IEEE Transactions on Haptics*, *14*(1), 109–122. http://dx.doi.org/10.1109/TOH.2020.2996748.

Bhowmick, A., & Hazarika, S. M. (2017). An insight into assistive technology for the visually impaired and blind people: state-of-the-art and future trends. *Journal on Multimodal User Interfaces*, *11*(2), 149–172. http://dx.doi.org/10.1007/s12193-016-0235-6.

Bhowmick, A., Prakash, S., Bhagat, R., Prasad, V., & Hazarika, S. M. (2014). IntelliNavi: Navigation for blind based on kinect and machine learning. In *International workshop on multi-disciplinary trends in artificial intelligence* (pp. 172–183). Springer, http://dx.doi.org/10.1007/978-3-319-13365-2_16.

Chanana, P., Paul, R., Balakrishnan, M., & Rao, P. (2017). Assistive technology solutions for aiding travel of pedestrians with visual impairment. *Journal of Rehabilitation and Assistive Technologies Engineering*, *4*, http://dx.doi.org/10.1177/2055668317725993.

Croce, D., Gallo, P., Garlisi, D., Giarré, L., Mangione, S., & Tinnirello, I. (2014). ARIANNA: A smartphone-based navigation system with human in the loop. In *22nd Mediterranean conference on control and automation* (pp. 8–13). IEEE, http://dx.doi.org/10.1109/MED.2014.6961318.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). ImageNet: A large-scale hierarchical image database. In *IEEE conference on computer vision and pattern recognition* (pp. 248–255). IEEE, http://dx.doi.org/10.1109/CVPR.2009.520684.

Dos Santos, A. D. P., Ferrari, A. L. M., Medola, F. O., & Sandnes, F. E. (2022). Aesthetics and the perceived stigma of assistive technology for visual impairment. *Disability and Rehabilitation: Assistive Technology*, *17*(2), 152–158. http://dx.doi.org/10.1080/17483107.2020.1768308.

Fusco, G., & Coughlan, J. M. (2020). Indoor localization for visually impaired travelers using computer vision on a smartphone. In *Proceedings of the 17th international web for all conference* (pp. 1–11). http://dx.doi.org/10.1145/3371300.3383345.

Ganz, A., Schafer, J. M., Tao, Y., Wilson, C., & Robertson, M. (2014). PERCEPT-II: Smartphone based indoor navigation system for the blind. In *36th Annual international conference of the IEEE engineering in medicine and biology society* (pp. 3662–3665). IEEE, http://dx.doi.org/10.1109/EMBC.2014.6944417.

Guerrero, J. C., Quezada-V, C., & Chacon-Troya, D. (2018). Design and implementation of an intelligent cane, with proximity sensors, GPS localization and GSM feedback. In *IEEE Canadian conference on electrical & computer engineering* (pp. 1–4). IEEE, http://dx.doi.org/10.1109/CCECE.2018.8447741.

Harvard, U. (2012). Measuring size from images: A wrangle with angles and image scale. In *Smithsonian astrophysical observatory*. Harvard University, URL https://lweb.cfa.harvard.edu/webscope/activities/pdfs/measureSize.pdf. (Accessed 01 October 2020).

Hersh, M., & Johnson, M. A. (2010). *Assistive technology for visually impaired and blind people*. Springer Science & Business Media.

Joshi, R. C., Yadav, S., Dutta, M. K., & Travieso-Gonzalez, C. M. (2020). Efficient multi-object detection and smart navigation using artificial intelligence for visually impaired people. *Entropy*, *22*(9), 941. http://dx.doi.org/10.3390/e22090941.

Kahraman, M., & Turhan, C. (2021). An intelligent indoor guidance and navigation system for the visually impaired. *Assistive Technology*, 1–9. http://dx.doi.org/10.1080/10400435.2021.1872738.

Kanwal, N., Bostanci, E., Currie, K., & Clark, A. F. (2015). A navigation system for the visually impaired: a fusion of vision and depth sensor. *Applied Bionics and Biomechanics*, *2015*, Article 479857. http://dx.doi.org/10.1155/2015/479857.

Kuriakose, B., Shrestha, R., & Eika Sandnes, F. (2021). Towards independent navigation with visual impairment: A prototype of a deep learning and smartphone-based assistant. In *The 14th pervasive technologies related to assistive environments conference* (pp. 113–114). http://dx.doi.org/10.1145/3453892.3464946.

Kuriakose, B., Shrestha, R., & Sandnes, F. E. (2020). Smartphone navigation support for blind and visually impaired people-a comprehensive analysis of potentials and opportunities. In *International conference on human-computer interaction* (pp. 568–583). Springer, http://dx.doi.org/10.1007/978-3-030-49108-6_41.

Kuriakose, B., Shrestha, R., & Sandnes, F. E. (2021a). Distance estimation methods for smartphone-based navigation support systems. In *Proceedings of SAI intelligent systems conference* (pp. 658–673). Springer, http://dx.doi.org/10.1007/978-3-030-82196-8_49.

Kuriakose, B., Shrestha, R., & Sandnes, F. E. (2021b). SceneRecog: A deep learning scene recognition model for assisting blind and visually impaired navigate using smartphones. In *IEEE international conference on systems, man, and cybernetics* (pp. 2464–2470). IEEE, http://dx.doi.org/10.1109/SMC52423.2021.9658913.

Kuriakose, B., Shrestha, R., & Sandnes, F. E. (2022). Tools and technologies for blind and visually impaired navigation support: a review. *IETE Technical Review*, *39*(1), 3–18. http://dx.doi.org/10.1080/02564602.2020.1819893.

Kuznetsova, A., Rom, H., Alldrin, N., Uijlings, J., Krasin, I., Pont-Tuset, J., et al. (2020). The open images dataset v4. *International Journal of Computer Vision*, *128*(7), 1956–1981.

Lin, B.-S., Lee, C.-C., & Chiang, P.-Y. (2017). Simple smartphone-based guiding system for visually impaired people. *Sensors*, *17*(6), 1371. http://dx.doi.org/10.3390/s17061371.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., et al. (2014). Microsoft COCO: Common objects in context. In *European conference on computer vision* (pp. 740–755). Springer, http://dx.doi.org/10.1007/978-3-319-10602-1_48.

Lin, Y., Wang, K., Yi, W., & Lian, S. (2019). Deep learning based wearable assistive system for visually impaired people. In *Proceedings of the IEEE international conference on computer vision workshops* (pp. 2549–2557). http://dx.doi.org/10.1109/ICCVW.2019.00312.

Lock, J. C., Gilchrist, I. D., Cielniak, G., & Bellotto, N. (2019). Bone-conduction audio interface to guide people with visual impairments. In *International conference on smart city and informatization* (pp. 542–553). Springer, http://dx.doi.org/10.1007/978-981-15-1301-5_43.

Manduchi, R., & Kurniawan, S. (2011). Mobility-related accidents experienced by people with visual impairment. *AER Journal: Research and Practice in Visual Impairment and Blindness*, *4*(2), 44–54.

Megalingam, R. K., Nambissan, A., Thambi, A., Gopinath, A., & Nandakumar, M. (2015). Sound and touch based smart cane: Better walking experience for visually challenged. In *Intelligent computing, communication and devices. advances in intelligent systems and computing. Vol. 308* (pp. 589–595). India: Springer, http://dx.doi.org/10.1109/IHTC.2014.7147543.

Mogelmose, A., Trivedi, M. M., & Moeslund, T. B. (2012). Vision-based traffic sign detection and analysis for intelligent driver assistance systems: Perspectives and survey. *IEEE Transactions on Intelligent Transportation Systems*, *13*(4), 1484–1497. http://dx.doi.org/10.1109/TITS.2012.2209421.

Moharkar, L., Varun, S., Patil, A., & Pal, A. (2020). A scene perception system for visually impaired based on object detection and classification using CNN. *ITM Web of Conference*, *32*, 03039. http://dx.doi.org/10.1051/itmconf/20203203039.

Moo Yi, K., Yun, K., Wan Kim, S., Jin Chang, H., & Young Choi, J. (2013). Detection of moving objects with non-stationary cameras in 5.8 ms: Bringing motion detection to your mobile device. In *IEEE conference on computer vision and pattern recognition workshops* (pp. 27–34). http://dx.doi.org/10.1109/CVPRW.2013.9.

Mukhiddinov, M., & Cho, J. (2021). Smart glass system using deep learning for the blind and visually impaired. *Electronics*, *10*(22), 2756. http://dx.doi.org/10.3390/electronics10222756.

Nook, C. (2020). Guide dogs vs. White canes: The comprehensive comparison. Clover Nook. URL https://clovernook.org/2020/09/18/guide-dogs-vs-white-canes-the-comprehensive-comparison/. (Accessed 20 April 2022).

Peng, E., Peursum, P., Li, L., & Venkatesh, S. (2010). A smartphone-based obstacle sensor for the visually impaired. In *International conference on ubiquitous intelligence and computing* (pp. 590–604). Springer, http://dx.doi.org/10.1007/978-3-642-16355-5_45.

Real, S., & Araujo, A. (2019). Navigation systems for the blind and visually impaired: Past work, challenges, and open problems. *Sensors*, *19*(15), 3404. http://dx.doi.org/10.3390/s19153404.

Riazi, A., Riazi, F., Yoosfi, R., & Bahmeei, F. (2016). Outdoor difficulties experienced by a group of visually impaired Iranian people. *Journal of Current Ophthalmology*, *28*(2), 85–90. http://dx.doi.org/10.1016/j.joco.2016.04.002.

Saaid, M. F., Mohammad, A., & Ali, M. M. (2016). Smart cane with range notification for blind people. In *2016 IEEE international conference on automatic control and intelligent systems* (pp. 225–229). IEEE, http://dx.doi.org/10.1109/I2CACIS.2016.7885319.

Saksham, S. (2014). Overview of smartcane. Smartcane. Saksham Trust. URL https://smartcane.saksham.org/?page_id=8. (Accessed 25 April 2022).

Suresh, A., Arora, C., Laha, D., Gaba, D., & Bhambri, S. (2017). Intelligent smart glass for visually impaired using deep learning machine vision techniques and robot operating system (ROS). In *International conference on robot intelligence technology and applications* (pp. 99–112). Springer, http://dx.doi.org/10.1007/978-3-319-78452-6_10.

Tan, M., & Le, Q. (2019). Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning* (pp. 6105–6114). PMLR, http://dx.doi.org/10.48550/arXiv.1905.11946.

Tan, M., Pang, R., & Le, Q. V. (2020). Efficientdet: Scalable and efficient object detection. In *IEEE/CVF conference on computer vision and pattern recognition* (pp. 10781–10790). http://dx.doi.org/10.1109/CVPR42600.2020.01079.

Tapu, R., Mocanu, B., Bursuc, A., & Zaharia, T. (2013). A smartphone-based obstacle detection and classification system for assisting visually impaired people. In *IEEE international conference on computer vision workshops* (pp. 444–451). http://dx.doi.org/10.1109/ICCVW.2013.65.

Tawalbeh, M., Eardley, A., et al. (2016). Studying the energy consumption in mobile devices. *Procedia Computer Science*, *94*, 183–189. http://dx.doi.org/10.1016/j.procs.2016.08.028.

WHO (2021). Vision impairment and blindness. In *Blindness and vision impairment*. World Health Organization, URL https://www.who.int/news-room/fact-sheets/detail/blindness-and-visual-impairment. (Accessed 20 April 2022).

Zhou, B., Lapedriza, A., Khosla, A., Oliva, A., & Torralba, A. (2017). Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *40*(6), 1452–1464. http://dx.doi.org/10.1109/TPAMI.2017.2723009.