applied sciences

MDPI

Article

Multi-Modal System for Walking Safety for the Visually Impaired: Multi-Object Detection and Natural Language Generation

Jekyung Lee [1], Kyung-Ae Cha [1,*] and Miran Lee [2]

[1] Department of Artificial Intelligence, Daegu University, Gyeongsan 38453, Republic of Korea; jekyung0204@gmail.com
[2] Department of Computer and Information Engineering, Daegu University, Gyeongsan 38453, Republic of Korea; miran@daegu.ac.kr
* Correspondence: chaka@daegu.ac.kr; Tel.: +82-53-850-6641

Abstract: This study introduces a system for visually impaired individuals in a walking environment. It combines object recognition using YOLOv5 and cautionary sentence generation with KoAlpaca. The system employs image data augmentation for diverse training data and GPT for natural language training. Furthermore, the implementation of the system on a single board was followed by a comprehensive comparative analysis with existing studies. Moreover, a pilot test involving visually impaired and healthy individuals was conducted to validate the system's practical applicability and adaptability in real-world walking environments. Our pilot test results indicated an average usability score of 4.05. Participants expressed some dissatisfaction with the notification conveying time and online implementation, but they highly praised the system's object detection range and accuracy. The experiments demonstrated that using QLoRA enables more efficient training of larger models, which is associated with improved model performance. Our study makes a significant contribution to the literature because the proposed system enables real-time monitoring of various environmental conditions and objects in pedestrian environments using AI.

Keywords: visually impaired; object detection; YOLOv5; natural language generation; KoAlpaca; walking assistance sentence

check for updates

Citation: Lee, J.; Cha, K.-A.; Lee, M. Multi-Modal System for Walking Safety for the Visually Impaired: Multi-Object Detection and Natural Language Generation. *Appl. Sci.* **2024**, *14*, 7643. https://doi.org/10.3390/app14177643

Academic Editor: Nicola Pio Belfiore

Received: 18 July 2024
Revised: 21 August 2024
Accepted: 28 August 2024
Published: 29 August 2024

Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

According to a report by the World Health Organization, approximately 2.2 billion people, or approximately 27% of the global population, will experience near or distance vision impairment by 2023, and nearly half of them (approximately 1 billion) will be unable to prevent vision impairment. Nearly 80% of near vision impairment cases were due to presbyopia. Conditions such as cataracts, refractive errors, age-related macular degeneration, glaucoma, and diabetic retinopathy predominantly contribute to distance vision impairment [1]. Advances in deep learning technology, including computer vision, are being applied across various fields to enhance daily life. This technology is particularly effective in assisting the visually impaired, enhancing their visual functions for tasks, such as mobility and object recognition [2,3]. To advance our study, we investigated the requirements for walking aids for visually impaired individuals through a literature review. Kim et al. [4] surveyed 154 visually impaired adults to identify the essential features for developing an AI-based guide robot and analyzed the results. The most essential feature of a guide robot was "listening and providing guidance", which was rated highest at 48.1%, followed by "exploring the surroundings and providing information", which was rated at 29.9%. Other important features included "providing spontaneous information about places" and "conveying display content through voice guidance". Based on the requirements identified in these references, we designed a system that applies the needs of visually impaired individuals into a walking aid system, utilizing YOLO-based object detection and generating natural language voice outputs to assist them.

One of the primary challenges visually impaired individuals face is the difficulty of navigating independently in complex urban environments characterized by various traffic signal systems, number of pedestrians, intersections, and roundabouts [5]. Walking aids and facilities, such as canes [6,7] and tactile paving [8], have been provided to promote safe travel. However, owing to factors such as technological limitations, road information (such as details about obstacles on the road or the shape of the road), and legal regulations regarding the installation of safety facilities, visually impaired individuals cannot rely solely on these aids and facilities. Therefore, to navigate complex urban roads, walking aids must include functions to detect their surroundings and provide information on traffic signal systems, structures, and road safety facilities that are difficult to recognize visually.

Recently, various studies have introduced methods for providing environmental scanning, traffic, and street information—such as the location and color of traffic lights and the presence of crosswalks—to visually impaired individuals [5–14]. The most used smart cane [6] alerts visually impaired individuals to approaching objects or obstacles through vibrations. However, because the cane must be held manually, its usability is reduced in crowded places.

In [7], a system to detect braille blocks—a walking aid for visually impaired individuals that guides them through various environments by recognizing the geometric features and colors of the blocks—was developed. However, its use is limited to individuals with knowledge of braille blocks. In addition, recognition is limited when the blocks are obscured by obstacles, such as pedestrian traffic and facilities, or when the blocks are discolored and damaged due to aging.

In a subsequent study that enhanced the system's object recognition performance [10], an intelligent navigation support system was developed using deep learning and a neural architecture search (NAS) to provide obstacle information to visually impaired individuals. By designing a high-performance object recognition model based on a NAS, the system offers faster and more accurate recognition compared to previous models. Additionally, it detects obstacles in real time and delivers the results to the user via voice notifications. However, conveying object names through voice notifications, especially when delivering complex information, may pose challenges.

Recently, with advancements in sensor technology, information for visually impaired individuals has increasingly been provided in wearable forms. Chen et al. [11] propose a wearable assistive system for visually impaired individuals that uses object recognition, distance measurement, and tactile feedback. This system utilizes the YOLOv3 object recognition model and two stereo cameras to detect obstacles and measure their distance in real time. The measured information is used to generate various vibration patterns based on the situation, utilizing shape memory alloy (SMA) actuators and small vibration motors. These vibration signals are provided via gloves, offering the advantage of intuitively conveying the direction and distance of obstacles. However, users must understand the different vibration patterns for each situation, and variations in tactile perception among individuals may exist.

Regarding smart devices, some devices combine smart glasses with a depth camera or connect a camera with a smartwatch or smartphone to detect objects and provide auditory warnings to visually impaired individuals [12,13]. However, these devices are limited to individuals wearing smart devices, and their implementation costs are high.

A device for object detection and notification for visually impaired individuals that applies object recognition algorithms to augmented reality devices was developed by [14]. It combines a Microsoft Hololens, a wearable device, with YOLOv2, an object recognition algorithm, to convey the names of the detected objects through voice. However, this device has limitations in providing contextual information for complex situations and has high implementation costs.

Representative walking aid apps for visually impaired individuals include OKO [15], Be My AI [16], and Oorion [17]. These apps utilize smartphone cameras to detect objects, obstacles, and text in real time and convey the recognized information to users via voice.

Particularly, Be My AI combines augmented reality technology to offer a more intuitive user experience by guiding visually impaired users along their walking paths and providing obstacle information. While these apps are similar to this study in that they deliver the names of surrounding objects to users in real time, they have limitations, and they only convey the names of simple objects without expressing the state of complex objects in sentences. By contrast, our system integrates object recognition with natural language generation to provide voice-guided descriptive sentences composed of object information and cautionary phrases rather than just simple object names. This study proposes solving these issues by combining real-time object detection and inference using the object recognition model YOLO [18] with the large-scale natural language model KoAlpaca [19] to generate descriptive sentences that provide complex information about the recognized objects. This sentence is conveyed using text-to-speech (TTS) conversion [20], implementing a walking aid system for visually impaired individuals. The hardware setup reduces the implementation costs using a single board, webcam, and headphones without requiring expensive devices. The main contribution of this study is that the proposed method provides a function that explains the contextual situation of detected objects in natural language for visually impaired individuals. Existing assistive tools have primarily focused on replacing visual functions; however, visually impaired individuals require situational explanations of the detected obstacles. Therefore, the usefulness of assistive tools can be considerably enhanced by combining a natural language generation model to provide situational explanations.

The remainder of this paper is structured as follows. Section 2 discusses related research and the characteristics of the models used in this study. Section 3 outlines multi-object recognition using YOLOv5. Section 4 presents the generation of alert sentences using KoAlpaca, detailing the training data, model training, and evaluation. Section 5 confirms the applicability of the system through the experimental results, explaining the integration and lightweight implementation of the two systems. Finally, Section 6 describes the results and contributions of our research and presents future research directions and limitations.

2. Related Works

2.1. Object Detection

Significant developments in object recognition began with the advancement of deep learning models such as the convolutional neural network (CNN) model [21]. AlexNet [22], a CNN architecture developed in 2012, has achieved groundbreaking results in image classification. The introduction of various object detection models utilizing CNNs, such as R-CNN [23], Faster R-CNN [24], and ResNet [25], has led to significant improvements in classification and object recognition tasks through image feature extraction. Although R-CNN and Faster R-CNN have fast inference speeds, their model structures are complex. AlexNet and ResNet have achieved high image classification accuracy. However, their specialization involves extracting image features for classification tasks rather than object recognition.

The CNN-based YOLO [18], a model that can simultaneously detect multiple objects and perform real-time object inferences, was introduced based on these technological advancements. YOLO, an open-source, real-time multi-object detection model, offers various backbones and versions. It has been applied in various fields, including smart transportation systems, walking aids for the visually impaired, and mobility aids for the disabled. Compared to existing object recognition models, YOLO features a simple model structure and provides fast learning and inference speeds.

Therefore, in this study, we conducted real-time object recognition using YOLO, which has strengths in real-time multiple object recognition in visually impaired walking environments. Additionally, using YOLO's simple model structure, we aim to achieve a lightweight model to enable fast inference on a single board. Among the various YOLO versions, we used YOLOv5 [26] owing to its long time and high stability, reliability, and

accuracy on various datasets. For transfer learning using YOLOv5, we implemented a lightweight system with the smallest weight file, YOLOv5n.pt [27].

2.2. Korean Natural Language Model: KoAlpaca

In this study, we generated visual context sentences for detected objects utilizing KoAlpaca, an open-source Korean language model specialized for Korean language data. KoAlpaca is based on the training methodology of Stanford University's Alpaca [28] model and is trained using the Korean language dataset (instruction set). The Alpaca model is an instruction-following version of the LLaMA [29], making it applicable to various language models. The pretrained KoAlpaca model is based on Polyglot-Ko [30], a large-scale Korean language model that addresses the insufficient understanding of the Korean context in the original Alpaca model. For fine tuning, we use the quantized low-rank adaptation (QLoRA) [31] technique supported by the Hugging Face [32] library. QLoRA is a parameter-efficient fine-tuning (PEFT) [33] method that applies quantization techniques to low-rank adaptation (LoRA), effectively reducing memory and model size while maintaining task performance. Figure 1 shows the memory requirements associated with the application of PEFT. QLoRA allocates the weights of the pretrained model to 4-bit and passes them to the adapter, which is then updated to optimize the model's loss function. During this process, if the GPU memory usage is insufficient, the system pages to the CPU's DRAM. This approach is particularly effective for reducing memory consumption during training.

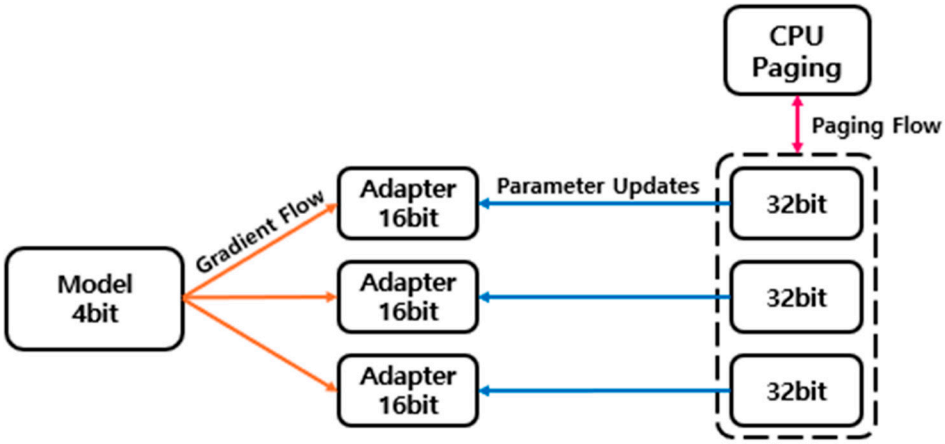


Figure 1. Memory requirements for QLoRA.

The parameters of QLoRA include rank, LoRA alpha, and LoRA dropout. The rank determines the number of trainable parameters in low-rank adapters, optimizing the model's memory requirements. LoRA alpha adjusts the parameter update rate during training to maximize fine tuning efficiency. By appropriately adjusting these parameter values, we effectively trained the KoAlpaca model to generate text efficiently with limited resources. The pretrained model weights can be quantized to a minimum of four bits and are then passed to the adapters to update and optimize the loss function. Because studies on Korean language generation are limited, this study focuses on a Korean text generation system utilizing an optimized Korean natural language model that generates walking information in Korean text, converts it to voice information through a TTS module, and conveys it to the user.

3. Multiple Object Detection with YOLOv5

This study proposes a method to detect situations in the walking environments of visually impaired individuals and automatically generate walking information based on these situations. Real-time video data were processed frame by frame through object detection. The object detection algorithm uses the YOLOv5 model to detect objects in the video input from the camera and obtain labels for the detected objects.

Four classes were defined for the training data: bollards, crosswalks, pedestrian traffic lights (red), and pedestrian traffic lights (green). When an object is detected above a certain threshold, the detected object label is input into the "Instruction" field of the natural language generation model KoAlpaca, which then generates information about the object and outputs it through the "Output" field.

Figure 2 depicts the structure of the full system. The generation of obstacle information automatically compiles the current walking situation and risk factors based on recognized objects. The KoAlpaca model analyzes the input object information and generates walking information, including details regarding objects observed during walking, such as crosswalks and bollards, along with cautionary phrases related to them. The generated walking information is conveyed to the user as a voice through a TTS module. Additionally, for the object detection and natural language generation experiments, the system was implemented on a single-board computer, Jetson Nano [34], enabling rapid and efficient notifications of visually impaired individuals in their walking environment. Figure 2 illustrates the setup, which includes compatible headphones that can be connected via Bluetooth through a single board. This tool is used for the prototype implementation and experiments of the system proposed in this study. To allow visually impaired individuals to perceive environmental sounds, we consider the application of bone-conduction headphones.

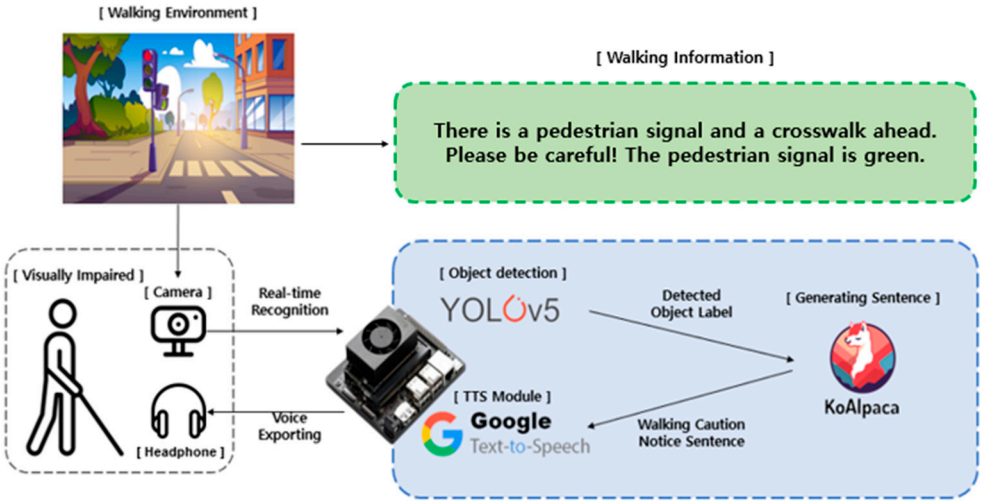


Figure 2. Structure of the proposed system.

3.1. Multiple Object Detection Pipeline

To implement multi-object detection, YOLOv5 is utilized. YOLOv5 can perform real-time multi-object detection and inference simultaneously and is available as open-source software with various versions, depending on the network size. Figure 3 shows the structure of the YOLOv5 model for multi-object detection.

The input image passes through the backbone, neck, and head, ultimately returning the final prediction. The features of the input image are extracted in the backbone field. The multiple convolution and pooling layers alter the image resolution, allowing features to be extracted at various resolutions. The extracted features then pass through the neck field, where they are fused.

At this stage, a path aggregation network [35] is used to fuse the features of low and high resolutions to improve performance. The fused features are then moved to the head field, where convolution layers transform them into the final output. The final output includes the bounding box parameters (x, y, w, and h), confidence score indicating the probability of the object's presence, and class probability, thereby completing the final recognition.

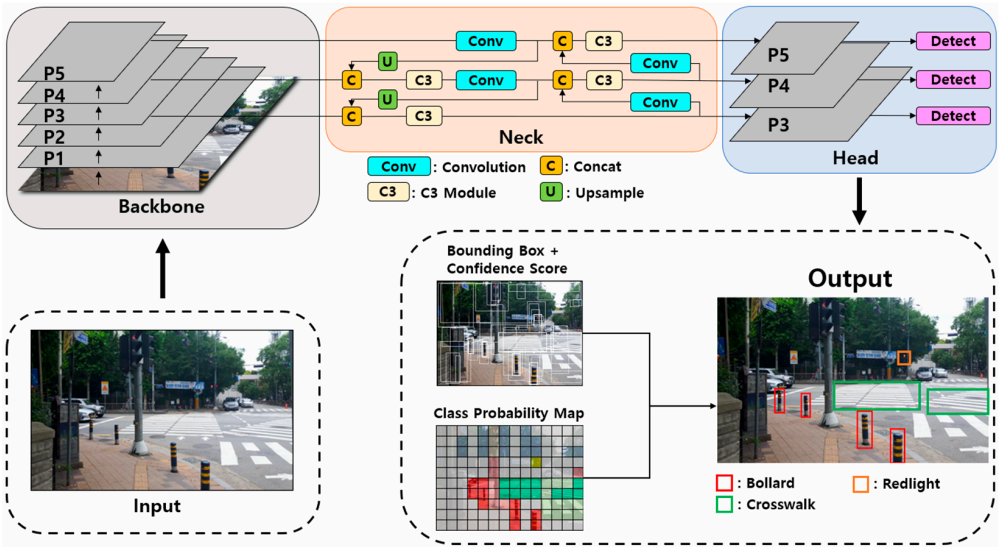


Figure 3. Multiple object detection pipeline.

In this study, YOLOv5 was fine tuned using training data that allowed visually impaired individuals to recognize the appearance and status of obstacles they needed to avoid or be aware of when crossing roads in urban areas.

3.2. YOLOv5 Training Data

To detect situations occurring in the walking environment of visually impaired individuals, the training data combined publicly available image sets obtained from Roboflow [36] with images directly captured by us. By increasing the quantity of road and street images as seen from a pedestrian's eye level, we aim to enable the YOLOv5 model to learn better the objects of the defined four classes (bollards, crosswalks, pedestrian traffic light (red), and pedestrian traffic light (green)).

The collection of public data is explained as follows. The classes were designed to focus on static objects that can be detected while walking. Because the shapes of bollards and signals vary by region, various forms of original images were collected through five Roboflow projects. We collected 3345, 2146, 679, 9497, and 1682 images from the following: the "Traffic-light Computer Vision Project" [37] and the "Capstone for Detection1 Computer Vision Project" [38], which includes walking environments such as signals, crosswalks, and bollards; the "Traffic-sign Project" [39], which includes various forms of bollards and crosswalks, depending on the region; the "Pedestrian Signs and Lanes Computer Vision Project" [40], which includes signals from various regions; and the "July_6 Computer Vision Project" [41], respectively. Images exceeding the permissible range of the object area and sharpness variations were removed, resulting in the construction of 5180 raw images.

Of these, 25% of the raw images contained objects classified into two or more classes. Specifically, 1105 images contained objects classified into only two classes and 193 images contained objects classified into three classes, totaling 1298 images.

To ensure sufficient images from the pedestrian's eye level, fifty-two images corresponding to the four classes were captured between 2 and 4 PM under clear and rainy or cloudy weather conditions. These images were augmented through blurring effects and noise application to triple the number and were refined for use in training. The device used to capture these images was an iPhone 15 Pro with a resolution of 4032 × 3024.

Annotation for creating the training data was conducted using RoboFlow. The four classes were selected considering obstacles encountered in real walking environments. The total number of annotations in the constructed training dataset was 9045. Figure 4 shows examples of collected images. Table 1 lists the specifications of the training data, including the number of images and annotations in each class.

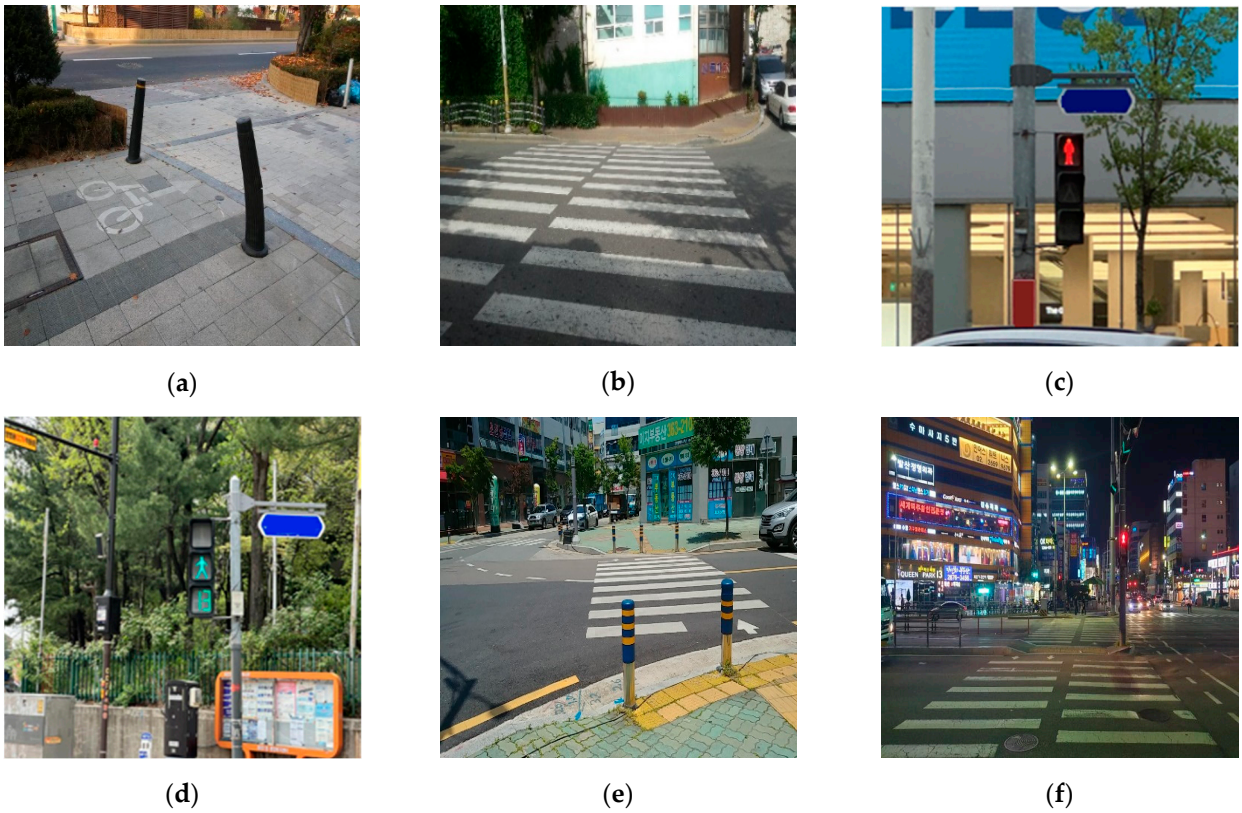


Figure 4. Example raw image: (**a**) bollard, (**b**) crosswalk, (**c**) pedestrian traffic light (red), (**d**) pedestrian traffic light (green), (**e**) bollard and crosswalk, (**f**) bollard, crosswalk, and pedestrian traffic light (red).

Table 1. Specification of training data.

Class	Number of Raw Images	Number of Annotations	Number of Annotation after Augmentations
Bollard	1187	3188	5659
Crosswalk	621	2484	5612
Redlight	1072	2005	5589
Greenlight	1002	1368	5482
Bollard, Crosswalk	363		
Bollard, Redlight	5		
Bollard, Greenlight	1		
Crosswalk, Redlight	583		
Crosswalk, Greenlight	149		
Redlight, Greenlight	4		
Bollard, Crosswalk, Redlight	154		
Bollard, Crosswalk, Greenlight	37		
Bollard, Redlight, Greenlight	1		
Crosswalk, Redlight, Greenlight	1		
Bollard, Crosswalk, Redlight, Greenlight	0		
Total	5180	9045	22,342

YOLOv5 recommends more than 1500 training images per class over 300 epochs [42]. Therefore, we used data augmentation techniques to expand the diversity of the training data from raw images. As presented in column 4 of Table 2, the number of annotations for each class is nearly equal. This demonstrates our ability to expand the training images using data augmentation techniques while simultaneously addressing the imbalance in the number of annotations per class.

Table 2. Training environment for YOLOv5.

Computing Source	Version
CPU	11th Intel(R) i9-11900k
RAM	64.0 GB
GPU	NVIDIA RTX 4090Ti
Anarconda Python v.	3.7.0
CUDA Engine v.	CUDA 11.3
PyTorch v.	11.3

Two augmentation techniques, blurring and noise effects, were selected and applied considering the decreased object detection rate owing to weather conditions during walking. Thus, 12,846 images and 22,342 annotations were constructed, excluding images similar to the original images but with reduced resolution, achieving approximately three times the number of collected images through augmentation. Figure 5 illustrates examples of the augmented data. The constructed dataset was categorized into training, validation, and test datasets with 11,499, 1085, and 262 images, respectively, set at a ratio of 9:0.8:0.2. YOLOv5, owing to its feature pyramid structure [43] and model architecture, downsamples input images in multiples of 32. Therefore, the input images were resized to 640 × 640 pixels to satisfy model requirements.



(**a**) (**b**) (**c**)

Figure 5. Example of image augmentation: (**a**) raw image, (**b**) added blur, and (**c**) added noise.

3.3. Training YOLOv5

In this study, we selected YOLO as the object detection model because of its multi-object detection and rapid inference capability. We used YOLOv5, which combines bottleneck [44] and cross-stage partial network (CSPNet) [45] techniques. The bottleneck technique controls the computational load and information loss through the channels of feature maps extracted from the images, thereby minimizing errors with minimal computation. The CSP technique performs convolution operations only on part of the basic layer before convolution with the next layer, and the remaining part is merged with the results after computation. This reduces the computational load of convolution operations, enhances the learning ability of the model, and reduces computing memory consumption. Therefore, YOLOv5, with its combined BottleneckCSP technology, is a model capable of accurate and fast object detection with low computing memory consumption. YOLOv5 provides five pretrained weight files based on the size of the weight network used.

The development environment for training YOLOv5 is listed in Table 2. We used a single NVIDIA RTX 4090Ti GPU and conducted training in an Anaconda environment. NVIDIA's CUDA was used as the GPU development tool, and training was based on Python and PyTorch.

To implement a lightweight system, we conducted an ablation study by training and comparing the nano and small versions of the YOLOv5 backbone, which are smaller-sized weight files provided by YOLOv5. We used the default learning rate provided by YOLOv5 for parameter training [46]. The epoch and batch size were set to 100 and 16, respectively, and applied equally to both versions to compare the performance and size of the models. Table 3 presents the results of the comparative analysis of the two versions. Although approximately a fourfold difference in model size exists, the performance improvement is only approximately 1%. Therefore, to build an efficient lightweight system, the smaller nano model was used for system implementation.

Table 3. Comparative analysis result for each model of YOLOv5.

Model	Params(M)	Class	Precision	Recall	mAP50	mAP50-95
Nano	1.9	All	0.812	0.808	0.859	0.473
		Bollard	0.886	0.891	0.946	0.61
		Crosswalk	0.633	0.621	0.655	0.309
		Greenlight	0.874	0.865	0.929	0.512
		Redlight	0.857	0.856	0.905	0.463
Small	7.2	All	0.852	0.819	0.873	0.474
		Bollard	0.902	0.902	0.95	0.61
		Crosswalk	0.697	0.591	0.651	0.321
		Greenlight	0.919	0.905	0.955	0.507
		Redlight	0.889	0.879	0.935	0.46

Figure 6 shows an example of the training results. The results show that multiple objects belonging to the four classes are detected. We can generate natural language sentences that describe the situation in advance using the information inferred from these images. For example, in an image where a crosswalk and bollard are inferred together, a natural language sentence can be generated to caution the user, such as "Be careful, there is a crosswalk with bollards ahead".

3.4. Evaluation of YOLOv5

The performance of the trained model was confirmed using the confusion matrix [47] and evaluation metrics observed during model training for model evaluation. The evaluation metrics included step-by-step loss rates, precision, and recall to determine the mean average precision (mAP) to assess model performance [48]. Figure 7 shows the confusion matrix for the test data. The detection rate for each class in the test data was confirmed through the confusion matrix. For bollards, the true detection rate is 92%, the false positive

rate is 8%, and the false negative rate is 17%. No false detections occurred between classes, and the average false positive rate for all classes was 11.8% and the false negative rate was 18%.

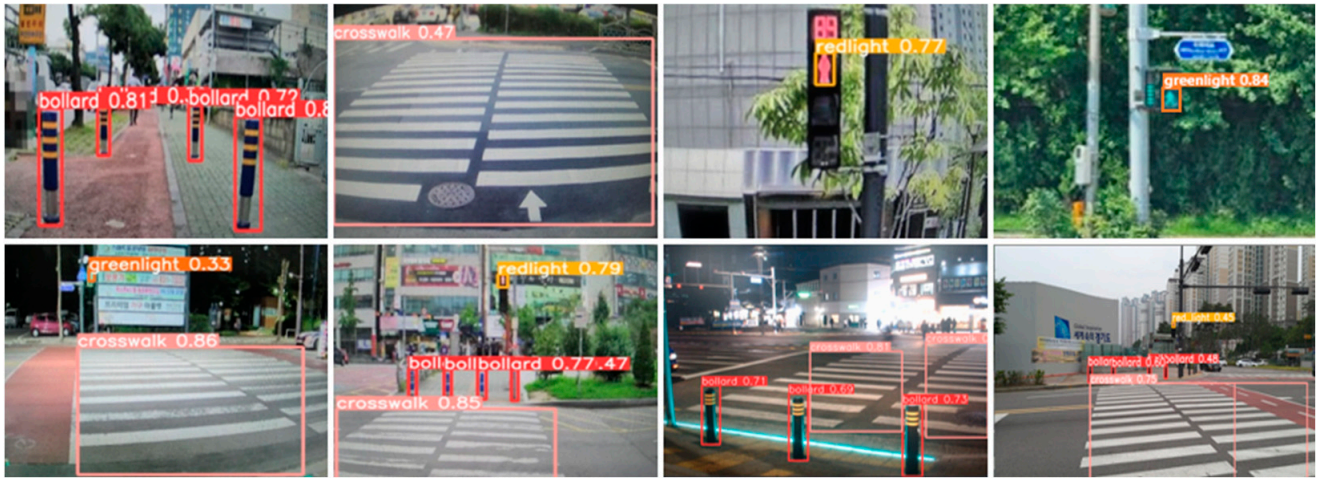


Figure 6. Few results after multiple object detection and recognition.

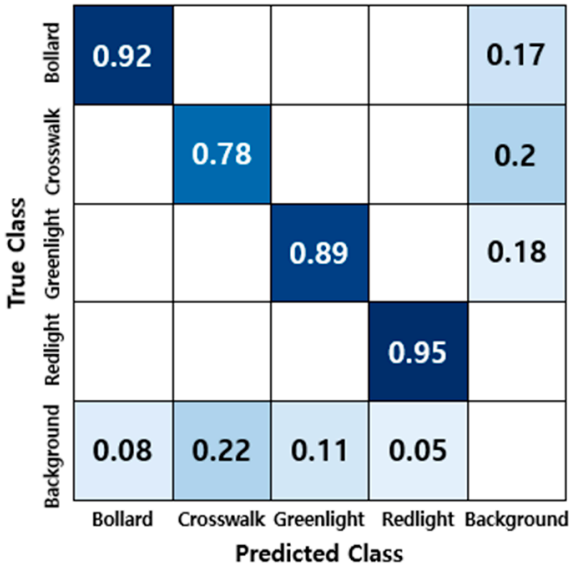


Figure 7. Confusion matrix.

Figure 8 illustrates the precision, recall, and mean average precision (mAP) values of the model for each class. Precision represents the ratio of correctly predicted objects out of all objects predicted by the model; the higher the precision, the greater the model's accuracy. Recall indicates the proportion of actual objects in the input data that the model correctly identified; a higher recall means the model has a better detection rate [48]. The mAP represents the mean precision value and indicates the recognition accuracy. The higher the mAP value, the more accurate the model's predictions are.

Table 4 lists the performance of the classwise model for the test data. The detection and recognition accuracies are 88.84% and 98.68%, respectively. Although the confusion matrix and precision–recall curve show that the recognition accuracy for crosswalks is lower, in the actual experimental results, there are no instances where the bounding box for crosswalk objects is generated or recognized.

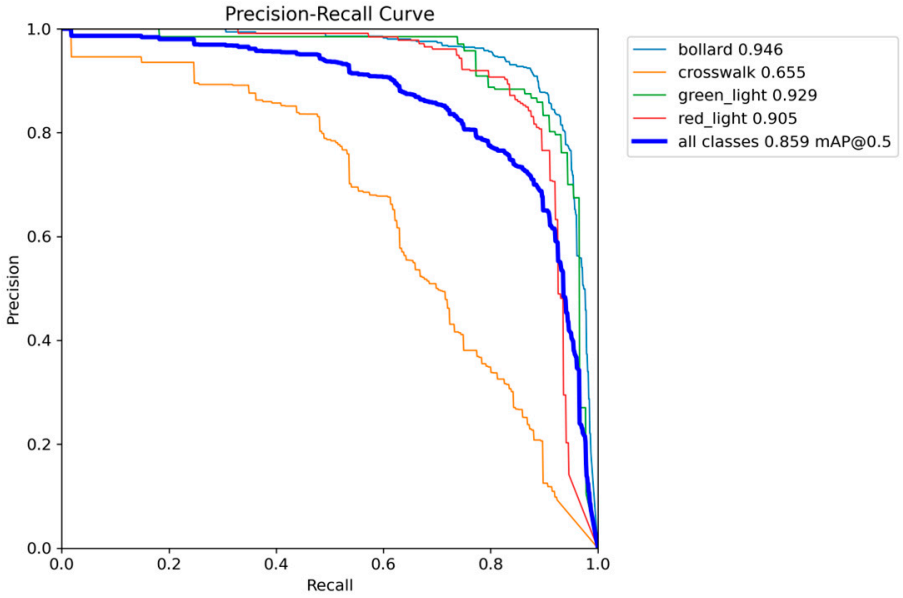


Figure 8. Evaluation result (mean average precision (mAP)) of YOLOv5.

Table 4. Evaluation of model performance on test data.

Class	Total Testing Images	Correctly Detected	Detection Accuracy (%)	Correctly Recognized	Recognition Accuracy (%)	Inference Time (s)
Bollard	65	60	92.30	59	98.33	0.13
Crosswalk	65	51	78.46	50	98.03	0.17
Redlight	65	58	89.23	58	100.00	0.14
Greenlight	65	62	95.38	61	98.39	0.14
Average			88.84		98.68	0.145

4. Generating Caution Notice Sentence Using KoAlpaca

4.1. Walking Assistance Sentence-Generating System

Walking assistance sentences should be generated as situational description sentences for multiple objects encountered during walking. This study utilizes KoAlpaca, an open-source natural language generation model fine tuned with large-scale Korean natural language data using instruction tuning [49], to generate pedestrian assistance sentences.

KoAlpaca is trained using Polyglot-ko, a Korean pretrained version of the multilingual pretrained model Polyglot [50], as its backbone. Training is performed using instruction-following data, enabling it to respond to commands that reflect the user's intent. The instruction-following data comprise three fields: an instruction field for receiving user commands, an input field for additional explanations of the command, and an output field for generating results based on this information. Instruction-following data were constructed using the self-instruction [49] method used to construct the Alpaca training data. Self-instruction is a data construction method that uses self-supply, where data are generated by a large language model (LLM) and used again for another LLM training.

Therefore, we created 160 natural language seed data scenarios for possible situations with detected objects and constructed augmented natural language training data using GPT [51], an open-source generative AI. In addition, during the fine tuning of KoAlpaca, model optimization was performed using QLoRA, which selectively adjusts parameters. The pipeline for the walking assistance sentence generation system is shown in Figure 9.

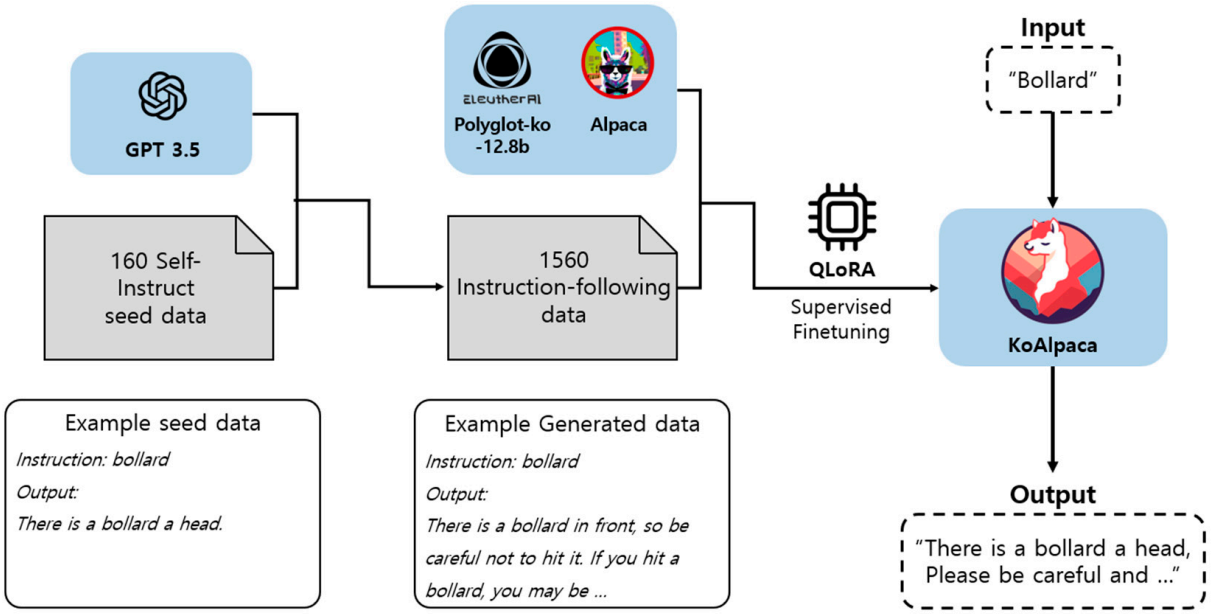


Figure 9. Walking assistance sentence generating system pipeline.

4.2. KoAlpaca Training Data

We constructed natural language training data in the form of sentences indicating the presence of detected objects and included cautionary phrases.

We created 160 natural language data scenarios based on 14 situations involving four objects. These scenarios were used as seed data and input into the GPT, which is an LLM model, to construct the augmented natural language training data. In the initial step, the GPT generates sentences similar to the seed data. In the subsequent steps, the GPT understands the context of the input data and generates new sentences. By repeating this step, many derived contexts and sentences can be obtained. We secured 100 sentences per situation, resulting in 1400 new natural language data points, which were combined with the original seed data, and constructed 1560 augmented training data points. Table 5 presents a portion of the constructed natural language dataset.

The constructed natural language training data in the CSV format are converted into the JSON format using Pandas [52] DataFrame. By adjusting the column names in the Pandas DataFrame to "Instruction", "Input", and "Output", each column is aligned with the corresponding field names to match the instruction-following data structure to train KoAlpaca. Table 6 shows an example of the JSON data composition. Instruction-following data have specific roles in each field, generating results that align with the user's intent. The "Instruction" field is responsible for the command that directs the model's task. The "Input" field provides additional explanations for the "Instruction", but it was not used in this training. The "Output" field contains the model's results, generated based on the "Instruction". As shown in Table 6, when the "Bollard" label detected by the object detection system is assigned to "Instruction", "Output" returns guidance and caution phrases to avoid the "Bollard".

4.3. Training KoAlpaca

KoAlpaca is an open-source natural language model available through Hugging Face. Based on Poly-glot-Ko, it demonstrates excellent performance in generating Korean sentences and offers two model sizes: 5.8 B and 12.8 B. In our research, we conducted four experiments applying two PEFT methods, LoRA and QLoRA, to evaluate memory usage for both model sizes. The experiments were conducted with identical parameters to compare and analyze memory usage and training time for each model size. Table 7 presents the results of the comparative analysis.

Table 5. Example of natural language training data.

Instruction	Input	Output
bollard, crosswalk	null	"There is a crosswalk and bollard. Please be careful."
bollard, crosswalk, greenlight	null	"There is a bollard and a crosswalk ahead, and the traffic light is green for pedestrians."
bollard, crosswalk, greenlight, redlight	null	"There is a crosswalk with a signal ahead. A bollard is also installed, so be careful."
bollard, crosswalk, redlight	null	"There is a bollard and a crosswalk ahead. The light is red."
bollard, greenlight	null	"The light is currently green, but there is a bollard ahead. Please be cautious."
bollard, redlight	null	"There is a bollard ahead and the traffic light is red for pedestrians. It is dangerous."
crosswalk	null	"There is a crosswalk ahead. Please be careful."
crosswalk, greenlight	null	"There is a crosswalk ahead. The traffic light is green for pedestrians."
crosswalk, greenlight, redlight	null	"There is a crosswalk ahead. Please pay attention to the pedestrian traffic light."
crosswalk, redlight	null	"There is a crosswalk and pedestrian traffic light ahead. The light is red."
greenlight	null	"The traffic light ahead is green for pedestrians."
greenlight, redlight	null	"There is a traffic light for pedestrians. Please pay attention to the signal."
redlight	null	"There is a pedestrian traffic light ahead, and the signal is red."

Table 6. Example of JSON data composition.

Filed	Content
Instruction	"bollard"
Input	" "
Output	"There is a bollard in front. Be careful. Please walk while paying attention to the bollards."

Table 7. Comparative analysis results for each model of KoAlpaca.

PEFT Method	Params (B)	GPU Memory (GB)	Training Time (m)
LoRA	5.8	39.8	10
	12.8	70.1	25
QLoRA	5.8	19.1	13
	12.8	28.9	26

The QLoRA four-bit technique was applied to improve memory efficiency during model training. This reduces the memory requirement of the model from 70 to 28.9 GB, a reduction of 41.28%. The increased computation required to reduce memory usage led to a slight increase in training time. However, the saved memory allows for a larger batch size, which can improve training speed. The experiments demonstrated that using QLoRA allows for more efficient training of larger models, which is associated with improved model performance.

Therefore, we used the 12.8 B model out of the two available sizes and fine tuned it using the constructed natural language data to return guidance and cautionary phrases for the detected objects. The training was conducted using an NVIDIA A100 GPU provided by Google Colab. The training parameters for KoAlpaca are listed in Table 8.

Table 8. Parameters of KoAlpaca.

Parameter	Value
Learning rate	1×10^{-4}
Batch size	2
Step	1 k
Warmup	50
Rank	8
LoRA alpha	32
LoRA dropout	0.5

The warm-up step helps the model stabilize during optimization by broadening the learning rate fluctuations during the initial iterations [53]. The rank parameter is the rank with LoRA and affects the training efficiency. A higher rank increases the number of parameters and the memory usage. The LoRA alpha parameter represents the weight update scale factor for the LoRA. The LoRA dropout parameter indicates the dropout rate in the LoRA and is used to prevent model overfitting.

Finally, the constructed walking-assistance sentence generation system generates sentences, averaging 50 tokens in less than 2 s, providing a significant advantage in quickly conveying the information. Some examples of the generated output sentences are given in Table 9. For cases with 50 tokens or fewer, a notification about the object and a simple cautionary statement, such as "There is a bollard ahead, so be careful", are generated. For cases with 100 tokens or more, a more detailed cautionary message is generated, which repeats the caution about the detected object, such as "There is a bollard ahead, so be careful. The structure is sturdy, so be careful not to bump into it".

Table 9. Example of the KoAlpaca results.

Instruction	Tokens	Output
bollard	100	"There is a bollard ahead, so be careful. The structure is sturdy, so be careful not to bump into it."
crosswalk	50	"Nearby, there is a crosswalk, so prioritize safety while walking."
greenlight	50	"The pedestrian traffic light ahead is green."
redlight	50	"A red light has appeared at the signal, so please be careful."
bollard, crosswalk	150	"This is a crosswalk with bollards. Be careful. You can get seriously hurt if you bump into the bollards."
bollard, crosswalk, redlight	150	"There is a crosswalk ahead and a bollard nearby. The traffic light is red, so please be cautious."

4.4. Evaluation of KoAlpaca

The evaluation of a language model is categorized into external and internal evaluations. The external evaluation assesses the model when applied to a specific task, using metrics such as loss and accuracy, while internal evaluation assesses the language model itself, independent of tasks, typically using perplexity [54]. In this study, the language model is applied to a specific task, generating sentences based on situations observed in the walking environment of visually impaired individuals. Therefore, KoAlpaca was evaluated through an external model evaluation, which was confirmed by the loss [55] values during the training step. Through the generation results of KoAlpaca, the generation capability of the model in the actual work was verified, and the difference between the predicted and actual values was examined using the loss value of each training step. The loss value is a metric that measures the difference between a model's predictions and actual values. During training, the model parameters were adjusted to minimize the loss value. A lower loss value indicates a smaller difference, signaling accurate predictions. As shown in

Figure 10, the loss value decreases as training progresses, indicating that the model creation results are becoming similar to the actual data. The final loss value of KoAlpaca is 0.2828.

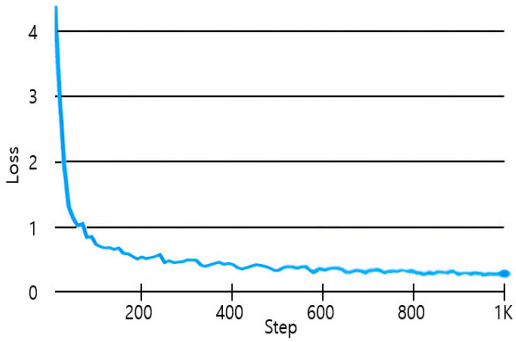


Figure 10. Loss evaluation value of KoAlpaca.

5. Experimental Results

The proposed system was implemented on a single board to detect situations and objects that may occur in the walking environments of visually impaired individuals using a webcam. It automatically generates obstacle notifications and caution sentences based on the names of detected objects.

The proposed system integrates object recognition and natural language generation through the Slack API [56], a cloud-based messaging platform. When the number of frames in which YOLOv5 recognizes an object reaches a certain threshold, the object's name is sent to the Slack user channel. Subsequently, the object name sent to the user channel is retrieved using the "conversations.history" API method, which searches existing messages in the Slack channel and is passed to the KoAlpaca generator. The notification and cautionary sentences generated based on the transmitted object name are then sent back to the Slack channel via the "chat.postMessage" API. Finally, the sent sentences are converted to speech via a TTS module, which is delivered to the user through headphones connected to the single board.

The single board used was a Jetson Orin Nano, an embedded computing board from NVIDIA designed for lightweight, low-power systems. Unlike existing single boards, such as Arduino and Raspberry Pi, it was designed to support AI computations. The implementation on a single board was based on NVIDIA's Ampere GPU architecture [57], using NVIDIA's CUDA as the GPU development tool. The system was developed using Anaconda Python(3.7.0), and PyTorch(11.3) software. Table 10 lists the environment of the computing system in the Jetson Orin Nano.

Table 10. The environment of the computing source in Jetson Orin Nano.

Computing Source	Version
CPU	6-core Arm Cortex-A78EA v8.2
RAM	8 GB
GPU	1024-core NVIDIA Ampere architecture GPU with 32 Tensor Cores
Anaconda Python Ver.	3.7.0
CUDA Engine Ver.	CUDA 11.3
Pytorch Ver.	11.3

A time domain analysis was conducted to verify the applicability of the proposed system. The analysis examined the number of classes and objects within a frame, the average time taken for object detection and sentence generation, the average time of audio prompts, and the total time to process a single frame to verify the performance across various scenarios. Table 11 presents the results of the time domain analysis of the proposed system. We used the Shapiro–Wilk test method to verify the normality of the resulting data

of the four processing indices. As a result, except for the average time of the audio prompt of class 3 and seven objects in the frame (p-value = 0.0270), it was confirmed that all of them followed a normal distribution through normality verification.

Table 11. Processing time of each frame in different conditions in a single board computer. Numbers in parentheses indicate the standard deviations. The Shapiro–Wilk test was used as a statistical method to verify the normality of each result data. The asterisk (*) indicates statistical significance at $p < 0.05$.

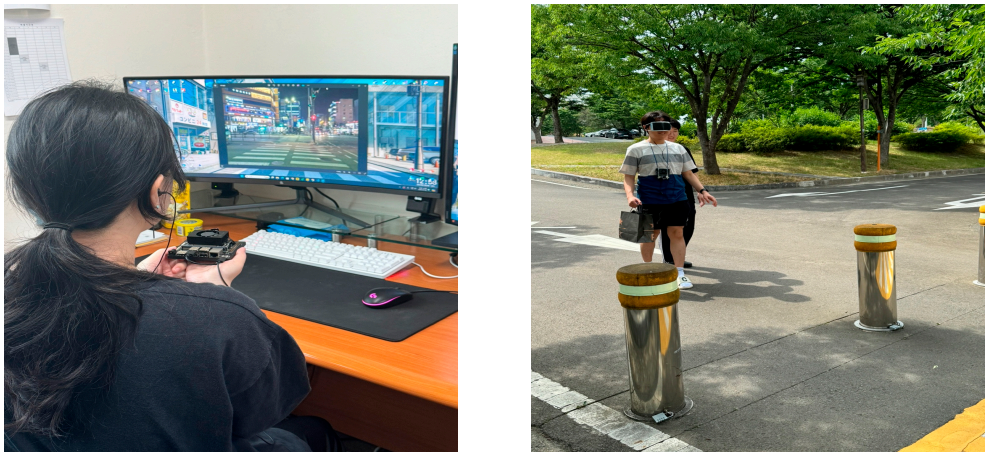
Number of Class	Number of Objects in Frame	Average Time Taken for Object Detection (s)	Average Time Taken for Sentence Generation (s)	Average Time of Audio Prompt (s)	Total Time to Process a Single Frame (s)
1	1	0.30 (0.01)	1.90 (0.01)	0.50 (0.01)	2.70 (0.01)
	p-value	0.0929	0.5344	0.7922	0.2463
1	4	0.30 (0.01)	1.90 (0.01)	0.50 (0.01)	2.70 (0.02)
	p-value	0.9056	0.7922	0.1560	0.8398
2	2	0.31 (0.01)	2.11 (0.02)	0.61 (0.02)	3.02 (0.02)
	p-value	0.8352	0.9827	0.9194	0.8974
2	5	0.30 (0.01)	2.11 (0.01)	0.51 (0.01)	2.91 (0.02)
	p-value	0.8352	0.9116	0.9116	0.8705
3	3	0.30 (0.01)	2.11 (0.02)	0.51 (0.01)	2.91 (0.01)
	p-value	0.9056	0.8733	0.9651	0.9380
3	4	0.30 (0.01)	2.20 (0.01)	0.70 (0.01)	3.20 (0.01)
	p-value	0.0555	0.4082	0.4082	0.1973
3	7	0.30 (0.01)	2.10 (0.01)	0.60 (0.01)	3.00 (0.01)
	p-value	0.0555	0.7922	0.0270 *	0.4082
3	10	0.31 (0.01)	2.20 (0.01)	0.50 (0.01)	3.00 (0.01)
	p-value	0.5224	0.4082	0.7922	0.5573
4	12	0.31 (0.01)	2.10 (0.01)	0.70 (0.01)	3.20 (0.01)
	p-value	0.5224	0.3657	0.4082	0.0672

The proposed system automatically detects objects and situations and generates cautionary sentences based on them. This allows for conveying contextually understood sentences about complex situations rather than object information. Additionally, by delivering sentences via a TTS module, the system can reduce its reliance on sensory-dependent aids, such as canes and clickers, which visually impaired individuals commonly use. Compared with other methods that use various sensors and devices for object detection and information conveyance to support the visually impaired, the proposed system achieves detection with a low-cost webcam and conveys cautionary sentences through headphones. Table 12 presents the results of comparing the proposed system with related works in this field.

Table 12. Comparison with related works.

Method	Using Device	Result	Connection	Cost
Chen et al. [10]	Single board, neural computer stick2, stereo camera, SMA actuator, vibration motors, and signal amplifier circuit	Detect obstacles and vibration pattern output	Offline	High
Yang et al. [11]	Depth camera on smart glass, laptop, and headphone	Detect obstacles and generate warning clarinet sound	Online	High
Bauer et al. [12]	Camera, smartwatch, and smartphone	Detect obstacles and generate audio output	Online	High
Eckert et al. [13]	RGB-D camera and IMU sensors	Detect obstacles and generate audio output	Online	High
Proposed Method	Webcam, single board, and headphone	Detect obstacles and generate walking information audio output	Online	Low (70 USD)

A qualitative evaluation of the proposed system was conducted through a pilot test with one visually impaired person and three blindfolded healthy people in their 20s. The test for the visually impaired participant was conducted in an indoor desktop environment, whereas the normal participants performed the test while walking on actual roads. Figure 11 shows scenes of the experiment for each participant. All participants used our system with the assistance of a helper. Before the experiment, a brief description of the system was provided to the users to understand the experimental steps. The results were confirmed through usability scores and feedback from participants after the experiment. Scores were out of five, with higher scores indicating better usability of the proposed system. Table 13 lists the obtained experimental results.



(**a**) (**b**)

Figure 11. Scenes of the experiment: (**a**) visually impaired and (**b**) general public.

Table 13. Result of usability evaluation.

User Type	Score	Review
Visually Impaired	4.0	It was nice to receive accurate recognition and notification. Compared to previously provided apps, the system could recognize structures from afar and deliver accurate notifications about structures in sentences, enabling understanding of complex walking situations rather than simple word notifications. However, it would be good if it could be implemented even in offline situations and if the voice type could be changed during voice transmission.
General Public A	4.1	I liked that object recognition was smooth even when there was a distance, so even when crossing a crosswalk, you could immediately tell there was a bollard in front of you.
General Public B	3.9	It was disappointing that it took a while for a notification to appear after recognizing an object. However, it was good that obstacle recognition was accurate, reducing the risk of unexpected accidents.
General Public C	4.2	I felt that the fact that it recognizes obstacles well and provides accurate notifications to users and that it provides warning notifications before approaching an object would bring convenience to users and prevent accidents.

The pilot test results indicated an average usability score of 4.05. While the participants expressed some dissatisfaction with the notification conveying time and online implementation, they highly praised the system for its ability to detect objects from a distance and its accuracy.

In particular, the participants praised the system's ability to convey sentences that described the visual context of detected objects rather than simply conveying the presence of an obstacle or a warning sound. The visually impaired participant noted that compared to the visual impairment assistance app they were currently using, the proposed system pro-

vided faster notifications and helped them quickly understand complex walking situations through voice that described the walking scenarios in sentences. These reviews confirm that the proposed system can provide accurate notifications to users and demonstrate its applicability in real walking situations.

6. Discussion

6.1. Results and Contribution

In this study, we developed a multi-modal walking environment information generation system for visually impaired individuals by merging object recognition and cautionary sentence generation systems using YOLOv5 and KoAlpaca, respectively. We applied image data augmentation to ensure the diversity of training data for object recognition in various environments and used GPT, a generative AI, to build our own natural language data and train the pedestrian caution warning generation system. The training of each model was evaluated using the evaluation metrics. The system was implemented on a single board, and a comparative analysis with related studies was conducted. In addition, a pilot test with visually impaired individuals was conducted to verify the applicability and flexibility of the system in a real walking environment.

The constructed system is expected to bring about significant changes when applied to various wearable devices, and it has the following outcomes. First, it enables real-time monitoring of various situations and objects observed in walking environments using AI. Using images collected from the sight of actual pedestrians as training data, we improved the object detection rate in a real walking environment. Second, an automatic information provision system allows for automating the detection and monitoring of the walking environment, converting environmental information into text, and conveying it in real time. This allows for immediate responses based on AI and offers various potential applications, such as data accumulation.

6.2. Limitations and Future Work

We developed a walking environment information generation system for visually impaired individuals, focusing on four static classes. The system was designed to recognize the walking environment and generate environmental information. A pilot test was conducted with one visually impaired participant and three sighted participants. The system's contribution was assessed through comparison with previous research and by evaluating system execution time. However, this study had limitations. The experiments were conducted in a controlled laboratory environment with a small number of participants. To generalize the proposed system, it should undergo validation with a more diverse and larger group of participants, considering factors such as age, gender, and physical abilities, after obtaining IRB approval. This would allow for a more accurate assessment of the system's effectiveness and usability and help improve its performance by reflecting the actual needs of the participants.

Additionally, for the generalization of the system, the classes should be updated to include static and dynamic objects. This means effectively processing a large and diverse amount of data and applying it to the system. Future research should focus on obtaining high-quality data to enhance the system's performance and generating various types of sentences to provide more information to visually impaired individuals.

Author Contributions: Conceptualization, J.L. and K.-A.C.; methodology, J.L. and K.-A.C.; software, J.L.; validation, J.L., K.-A.C. and M.L.; investigation, J.L.; data curation, J.L.; writing—original draft preparation, J.L. and K.-A.C.; writing—review and editing, J.L., K.-A.C. and M.L.; visualization, J.L.; supervision, K.-A.C.; project administration, M.L.; funding acquisition, M.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Ministry of Education of the Republic of Korea and the National Research Foundation of Korea (NRF-2024S1A5A8019541).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. World Health Organization. Available online: https://www.who.int/news-room/fact-sheets/detail/blindness-and-visual-impairment (accessed on 14 June 2024).
2. Manjari, K.; Verma, M.; Singal, G. A survey on assistive technology for visually impaired. *Internet Things* **2020**, *11*, 100188. [CrossRef]
3. Research and Markets. Available online: https://www.researchandmarkets.com/report/visual-impairment-assistive-technology (accessed on 14 June 2024).
4. Kim, H.Y.; Lee, S.Y.; Kim, D.Y.; Jo, S.J. An analysis on O&M ability status and needs of people with visual impairments for the development of AI guide robot. *Korean J. Vis. Impair.* **2023**, *39*, 99–121. (In Korean) [CrossRef]
5. El-taher, F.E.-z.; Taha, A.; Courtney, J.; Mckeever, S. A systematic review of urban navigation systems for visually impaired people. *Sensors* **2021**, *21*, 3103. [CrossRef] [PubMed]
6. Panazan, C.-E.; Dulf, E.-H. Intelligent cane for assisting the visually impaired. *Technologies* **2024**, *12*, 75. [CrossRef]
7. Yang, Z.; Gao, M.; Choi, J. Smart walking cane based on triboelectric nanogenerators for assisting the visually impaired. *Nano Energy* **2024**, *124*, 109485. [CrossRef]
8. Takano, T.; Nakane, T.; Akashi, T.; Zhang, C. Braille block detection via multi-objective optimization from an egocentric viewpoint. *Sensors* **2021**, *21*, 2775. [CrossRef]
9. Walle, H.; De Runz, C.; Serres, B.; Venturini, G. A survey on recent advances in AI and vision-based methods for helping and guiding visually impaired people. *Appl. Sci.* **2022**, *12*, 2308. [CrossRef]
10. Said, Y.; Atri, M.; Albahar, M.A.; Ben Atitallah, A.; Alsariera, Y.A. Obstacle detection system for navigation assistance of visually impaired people based on deep learning techniques. *Sensors* **2023**, *23*, 5262. [CrossRef] [PubMed]
11. Chen, Y.; Shen, J.; Sawada, H. A wearable assistive system for the visually impaired using object detection, distance measurement and tactile presentation. *Intell. Robot.* **2023**, *3*, 420–435. [CrossRef]
12. Yang, K.; Bergasa, L.M.; Romera, E.; Cheng, R.; Chen, T.; Wang, K. Unifying terrain awareness through real-time semantic segmentation. In Proceedings of the 2018 IEEE Intelligent Vehicles Symposium (IV), Changshu, China, 26–30 June 2018; pp. 1033–1038. [CrossRef]
13. Bauer, Z.; Dominguez, A.; Cruz, E.; Gomez-Donoso, F.; Orts-Escolano, S.; Cazorla, M. Enhancing perception for the visually impaired with deep learning techniques and low-cost wearable sensors. *Pattern Recognit. Lett.* **2019**, *137*, 27–36. [CrossRef]
14. Eckert, M.; Blex, M.; Friedrich, C.M. Object Detection featuring 3D audio localization for Microsoft HoloLens—A deep learning based sensor substitution approach for the blind. In Proceedings of the 11th International Joint Conference on Biomedical Engineering Systems and Technologies, Funchal, Portugal, 19–21 January 2018; pp. 555–561. [CrossRef]
15. Ayes. OKO App Leverages AI to Help Blind Pedestrians Recognize Traffic Signals. 2023. Available online: https://www.ayes.ai/post/oko-app-leverages-ai-to-help-blind-pedestrians-recognize-traffic-signals (accessed on 1 August 2024).
16. Be My Eyes. Introducing Be My AI. 2023. Available online: https://www.bemyeyes.com/blog/introducing-be-my-ai (accessed on 1 August 2024).
17. OOrion. OOrion Official Website. 2024. Available online: https://www.oorion.fr/en/ (accessed on 1 August 2024).
18. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once: Unified, real-time object detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
19. Github. Available online: https://github.com/Beomi/KoAlpaca (accessed on 18 June 2024).
20. Sasirekha, D.; Chandra, E. Text to speech: A simple tutorial. *Int. J. Soft Comput. Eng. (IJSCE)* **2012**, *2*, 275–278.
21. O'Shea, K.; Nash, R. An introduction to convolutional neural networks. *arXiv* **2015**, arXiv:1511.08458. [CrossRef]
22. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet classification with deep convolutional neural networks. *Commun. ACM* **2017**, *60*, 84–90. [CrossRef]
23. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Columbus, OH, USA, 23–28 June 2014; pp. 580–587.
24. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2016**, *39*, 1137–1149. [CrossRef]
25. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
26. Terven, J.; Córdova-Esparza, D.-M.; Romero-González, J.-A. A comprehensive review of YOLO architectures in computer vision: From YOLOv1 to YOLOv8 and YOLO-NAS. *Mach. Learn. Knowl. Extr.* **2023**, *5*, 1680–1716. [CrossRef]
27. Github. Available online: https://github.com/ultralytics/yolov5 (accessed on 18 June 2024).
28. Github. Available online: https://github.com/tatsu-lab/stanford_alpaca (accessed on 18 June 2024).

29. Touvron, H.; Lavril, T.; Izacard, G.; Martinet, X.; Lachaux, M.A.; Lacroix, T.; Rozière, B.; Goyal, N.; Hambro, E.; Azhar, F.; et al. LLaMA: Open and efficient foundation language models. *arXiv* **2023**, arXiv:2302.13971. [CrossRef]
30. Ko, H.; Yang, K.; Ryu, M.; Choi, T.; Yang, S.; Hyun, J.; Park, S.; Park, K. A Technical report for Polyglot-Ko: Open-source large-scale Korean language models. *arXiv* **2023**, arXiv:2306.02254. [CrossRef]
31. Dettmers, T.; Pagnoni, A.; Holtzman, A.; Zettlemoyer, L. QLoRA: Efficient finetuning of quantized LLMs. *arXiv* **2023**, arXiv:2305.14314.
32. Huggingface. Available online: https://huggingface.co/ (accessed on 18 June 2024).
33. Houlsby, N.; Giurgiu, A.; Jastrzebski, S.; Morrone, B.; De Laroussilhe, Q.; Gesmundo, A.; Attariyan, M.; Gelly, S. Parameter-efficient transfer learning for NLP. In Proceedings of the International Conference on Machine Learning, Long Beach, CA, USA, 9–15 June 2019; pp. 2790–2799.
34. Nvidia Developer. Available online: https://developer.nvidia.com/ko-kr/blog/develop-ai-powered-robots-smart-vision-systems-and-more-with-nvidia-jetson-orin-nano-developer-kit/ (accessed on 18 June 2024).
35. Liu, S.; Qi, L.; Qin, H.; Shi, J.; Jia, J. Path aggregation network for instance segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 8759–8768.
36. Roboflow. Available online: https://roboflow.com/ (accessed on 18 June 2024).
37. Roboflow. Available online: https://universe.roboflow.com/trafficlight-7r04p/traffic-light-szdih (accessed on 18 June 2024).
38. Roboflow. Available online: https://universe.roboflow.com/dkdkd/capstone-for-detection1 (accessed on 18 June 2024).
39. Roboflow. Available online: https://universe.roboflow.com/sajotuna/traffic-sign-bykpq (accessed on 18 June 2024).
40. Roboflow. Available online: https://universe.roboflow.com/ps7/pedestrian-signs-and-lanes (accessed on 18 June 2024).
41. Roboflow. Available online: https://universe.roboflow.com/dkdkd/july_6 (accessed on 18 June 2024).
42. Ultralytics. Available online: https://docs.ultralytics.com/yolov5/tutorials/tips_for_best_training_results/#model-selection (accessed on 18 June 2024).
43. Lin, T.Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature pyramid networks for object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 2117–2125.
44. Li, J.; Liu, D. Information bottleneck theory on convolutional neural networks. *Neural Process. Lett.* **2021**, *53*, 1385–1400. [CrossRef]
45. Wang, C.Y.; Liao, H.Y.M.; Wu, Y.H.; Chen, P.Y.; Hsieh, J.W.; Yeh, I.H. CSPNet: A new backbone that can enhance learning capability of CNN. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, Seattle, WA, USA, 14–19 June 2020; pp. 390–391.
46. Ultralytics. Available online: https://docs.ultralytics.com/yolov5/tutorials/hyperparameter_evolution/#before-you-start (accessed on 18 June 2024).
47. Liang, J. Confusion matrix: Machine learning. *J. POGIL Act. Clear.* **2022**, *3*, 4.
48. Chen, W.; Luo, J.; Zhang, F.; Tian, Z. A review of object detection: Datasets, performance evaluation, architecture, applications and current trends. *Multimed. Tools Appl.* **2024**, *83*, 65603–65661. [CrossRef]
49. Wang, Y.; Kordi, Y.; Mishra, S.; Liu, A.; Smith, N.A.; Khashabi, D.; Hajishirzi, H. Self-instruct: Aligning language models with self-generated instructions. *arXiv* **2022**, arXiv:2212.10560. [CrossRef]
50. Wei, X.; Wei, H.; Lin, H.; Li, T.; Zhang, P.; Ren, X.; Li, M.; Wan, Y.; Cao, Z.; Xie, B.; et al. Polylm: An open-source polyglot large language model. *arXiv* **2023**, arXiv:2307.06018. [CrossRef]
51. OpenAI. Available online: https://openai.com/blog/chatgpt (accessed on 22 June 2024).
52. Pandas. Available online: https://pandas.pydata.org (accessed on 22 June 2024).
53. Xiong, R.; Yang, Y.; He, D.; Zheng, K.; Zheng, S.; Xing, C.; Liu, T. On layer normalization in the transformer architecture. In Proceedings of the International Conference on Machine Learning, Virtual, 13–18 July 2020; pp. 10524–10533.
54. WhyLabs. Evaluating Large Language Models (LLMs) 2024. Available online: https://whylabs.ai/learning-center/introduction-to-llms/evaluating-large-language-models-llms (accessed on 1 August 2024).
55. Terven, J.; Cordova-Esparza, D.M.; Ramirez-Pedraza, A.; Chavez-Urbiola, E.A. Loss functions and metrics in deep learning. A review. *arXiv* **2023**, arXiv:2307.02694. [CrossRef]
56. Slack. Slack API Documentation. 2024. Available online: https://api.slack.com/ (accessed on 1 August 2024).
57. Nvdia. Available online: https://www.nvidia.com/en-us/data-center/ampere-architecture/ (accessed on 22 June 2024).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.