Smartphone Apps for Visually Impaired that Detects Obstacles and Obtains those Distances

Manabu Shimakawa*, Naho Minami*, Chiharu Okuma*, Kimiyasu Kiyota*

*National Institute of Technology (KOSEN), Kumamoto College
2659-2 Suya, Koshi, Kumamoto 861-1102 Japan

*Corresponding Author: shimakawa@kumamoto-nct.ac.jp

Abstract

The authors have been developing a smartphone app to support for visually impaired walking when they go out. This app has functions to detect obstacles from RGB camera image that hinder walking and to alert dangerous situation to the visually impaired. It uses CNN, a kind of deep learning technique, to detect obstacles, but it can detect only one obstacle from the image. Now we are trying to implement a multiple obstacles detect function into this app. This paper deals with a method to obtain the distance to the obstacles detected from the multiple obstacles detection.

Keywords: visually impaired, obstacle detection, distance obtaining, smartphone apps.

1. Introduction

In Japan, as of 2016, there are approximately 312,000 visually impaired people who have a certificate of the physically disabled. It is predicted that the visually impaired will increase year by year as the population increases and ages (1). Such visually impaired people find it very inconvenient in their daily lives, especially when walking. Investigation results indicate that about half of walking accidents are collision accidents. It will be helpful that some tool to detect properly the obstacles in front of their walking. We developed smartphone apps that detect obstacles using CNN (Convolutional Neural Network), a kind of Deep Learning (2-3). Since this method only determines whether there is an obstacle or not on the image, it is not possible to know where the obstacle is. Therefore, this method the distance to the obstacles cannot be obtained. Currently, we have been trying to implement the function that detects multiple obstacles at the same time and obtains the distance to each obstacle into the smartphone app. This paper shows how to measure distances to the detected obstacles using the tilt angle and the view angle of the smartphone camera, and the usefulness of the method will be shown by experimental results.

2. Obstacle Detection App

2.1 Overview

Fig.1 shows a flow diagram of this system. A visually impaired wears a smartphone equipped with a camera at around chest height to take RGB images of forward. This system works as an application program on the smartphone. This app recognizes forward situations from taken RGB images and classifies them into some categories by using CNN. If it is recognized as an obstacle that disturbs walking, this system notifies it to the user by sound and vibration.

2.2 Obstacle Detection

There are many kinds of obstacles that disturb walking on road. In this study, we selected the following basic
situations that should be detected:

- Sidewalk
- Crosswalk
- Bicycle
- Stairs

CNN learns by using a large amount of supervised training images, and given an input image, it can classify an object included in the images into a category. The final layer of CNN is a Fully Connected Layer, and the ReLU function

$$\varphi(x) = \max(0, x)$$  \hspace{1cm} (1)

is used as activation function for each neuron unit. The most likely candidate neuron unit outputs a highest value.

The output value of each candidate neuron unit to be detected objects is converted to a value of 0 to 1 by the Softmax function

$$p_i = \frac{\exp(y_i)}{\sum_{j=1}^N \exp(y_j)}$$  \hspace{1cm} (2)

where $y_i$ is the output value of $i$-th candidate neuron unit, $N$ is the number of candidates. The value $p_i$ can be used as a probability for $i$-th candidate.

This study makes efficient use Inception-v3 which is already learned CNN model. Inception-v3 was also developed by Google and was learned to classify images into 1,000 classes for the image identification task of ILSVRC\textsuperscript{(4)}. This study uses only the structure of the Inception-v3 model that has abilities of such good identification and re-learned the CNN model to recognize the specific obstacles that are needed to detection for walking support. Fig.2 shows a diagram of the re-learning flow, where TensorFlow is an open-source framework of machine learning developed by Google.

2.3 Experiments

The experiments were conducted during the day in city area. The experimenter kept a smartphone at chest position about 140 cm height from the ground, did not fix the angle of it. This experiment verified whether the app could detect target obstacle or situation while walking.

Fig. 3 shows screenshots of situations in which the classification succeeded. All these situations, (a) sidewalk, (b) crosswalk, (c) bicycle and (d) stairs, were classified successfully with high probability values greater than 0.8.

The specification of smartphone is shown in Table 1. This specification is a middle level model in 2017. The app can perform detection processing 3 to 4 times per second.

We prepared 400 to 2000 images and divided them into two groups, 80% for training and 20% for test. The
training images had undergone image processing for each category and were used to make a CNN model by re-learning. The number of iterations is 4000 times. Table 2 summarizes the experimental results. The correct answer rates are over 80% in any category, which shows that it is a good model.

Table 1. Specification of Smartphone.

| Model         | ASUS ZenFone3 (ZE520KL) |
|---------------|-------------------------|
| OS            | Android 7.0             |
| CPU           | Snapdragon 625 (Octa-core 2.0GHz) |
| Camera        | 16.0 MPixels            |

Table 2. Classification Accuracy

| Scenes     | Correct | Correct Ratio |
|------------|---------|---------------|
| Sidewalk   | 50      | 41            | 82.0%         |
| Crosswork  | 47      | 41            | 87.2%         |
| Bicycle    | 50      | 40            | 80.0%         |
| Stairs     | 45      | 39            | 86.7%         |
| TOTAL      | 192     | 161           | 83.9%         |

3. Multiple Obstacles Detection

3.1 Why Multiple Obstacles Detection

The basic CNN mentioned above only determines if there are obstacles on the image, so you can't know where the obstacles are. If multiple obstacles are in front, this method will detect only one of the most likely obstacles. Even if there is a danger due to other obstacles, it will be ignored. It is necessary to detect multiple obstacles at the same time so as not to overlook the danger.

3.2 YOLO

YOLO\(^5\) is one of the object recognition algorithms using CNN and is named from the acronym of “You Only Look Once”. It can recognize multiple objects and acquire the area on the image at the same time. It has a feature that the processing time is shorter in compared with other similar methods using the window sliding approach or region proposal method. Fig. 4 show an example of multiple obstacles detection using YOLO.

4. Distance Obtaining

4.1 Aims

By using YOLO, it is possible to know the position of the recognized each obstacle on the image. If you have the physical information of the smartphone at the time of shooting, you can calculate the distance to the obstacle. This chapter describes the method and shows its effectiveness by experiments.

4.2 Obtaining Method of Distance to the Obstacles

See Fig. 5, which shows the parameters for calculation. When the height at which the smartphone is held is \( h \) and the angle from the vertical direction to the obstacle is \( \theta \), the distance \( L \) to the obstacle can be calculated by the following equation

\[
L = h \tan \theta
\]

Using the tilt angle of the smartphone, the angle of view of the smartphone camera and obstacle position on the image, the angle \( \theta \) can be calculated by the equation

\[
\theta = \varphi_\alpha + \left(0.5 - \frac{y}{P_y}\right)\varphi_\beta
\]

where

Fig. 4. Multiple Obstacles Detection by YOLO.

Fig. 5. Parameters for Calculation of Distance.
\(\varphi_\alpha\): tilt angle of the smartphone,
\(\varphi_\beta\): angle of view of the smartphone camera,
\(P_y\): size in the y direction of the image [Pixel],
y: position of the recognized obstacle from the top on the image [Pixel].

### 4.3 Experiments

We had experiments to verify that the distance to an obstacle can be obtained correctly by this method using two types of smartphones with different angles of view shown in Table 3. As shown in Fig. 6, the acquired image was divided into four in the y-axis direction, and the distances of the five points from (A) to (E) were calculated using Eq. (3) and Eq. (4). The tilt angle of the smartphone was changed from 0 degree to 90 degree in 10 degree increments, and the calculated and measured distances of each angle were compared. The height \(h\) was fixed at 1.2m assumed as a chest position.

Table 4 shows the results of using the smartphone A. Table (a) are of calculated distances and Table (b) are of measured values. Fig. 7 shows result graph of them. The calculated value expresses a tendency of the measured value, and it is considered that the calculated value is within acceptable range although there are some errors.

The experimental results of using the smartphone B are shown in Table 5 and Fig.8. The angle of view and image size of this smartphone are different from the smartphone A mentioned above, but it also shows good results as well.

| Tilt angle \(\varphi_\alpha\) [deg] | (A) | (B) | (C) | (D) | (E) |
|----------------------------------|-----|-----|-----|-----|-----|
| 0                               | 0.00| 0.35| 0.78|
| 10                              | 0.21| 0.60| 1.11|
| 20                              | 0.08| 0.44| 0.89| 1.59|
| 30                              | 0.30| 0.70| 1.27| 2.36|
| 40                              | 0.15| 0.53| 1.01| 1.81| 3.91|
| 50                              | 0.37| 0.80| 1.43| 2.75|
| 60                              | 0.62| 1.15| 2.10| 5.08|
| 70                              | 0.91| 1.63| 3.30|
| 80                              | 1.30| 2.42| 6.83|
| 90                              | 1.85| 4.05|

**Table 4. Experimental Results using Smartphone A.**

(a) Calculated Distance [m]

(b) Measured Distance [m]

| Tilt angle \(\varphi_\alpha\) [deg] | (A) | (B) | (C) | (D) | (E) |
|----------------------------------|-----|-----|-----|-----|-----|
| 0                               | 0.00| 0.33| 0.68|
| 10                              | 0.18| 0.57| 0.99|
| 20                              | 0.04| 0.40| 0.86| 1.44|
| 30                              | 0.24| 0.64| 1.24| 2.14|
| 40                              | 0.12| 0.44| 0.95| 1.77| 3.58|
| 50                              | 0.32| 0.70| 1.36| 2.75|
| 60                              | 0.55| 1.00| 1.94| 5.18|
| 70                              | 0.82| 1.45| 3.16|
| 80                              | 1.19| 2.16| 6.38|
| 90                              | 1.72| 3.54|

**Fig. 7. Experimental Result Graph using Smartphone A.**

![Fig. 7. Experimental Result Graph using Smartphone A.](image-url)

![Fig. 6. Parameters of Experiments.](image-url)
Table 5. Experimental Results using Smartphone B.

(a) Calculated Distance [m]

| Tilt angle $\phi_a$ [deg] | (A) | (B) | (C) | (D) | (E) |
|---------------------------|----|----|----|----|----|
| 0                         | 0.00 | 0.42 | 0.96 | |
| 10                        | 0.22 | 0.68 | 1.38 | |
| 20                        | 0.02 | 0.45 | 1.00 | 1.97 |
| 30                        | 0.23 | 0.70 | 1.40 | 3.10 |
| 40                        | 0.03 | 0.45 | 1.01 | 2.03 | 6.00 |
| 50                        | 0.24 | 0.71 | 1.43 | 3.27 | |
| 60                        | 0.47 | 1.04 | 2.09 | 6.46 | |
| 70                        | 0.73 | 1.47 | 3.32 | |
| 80                        | 1.06 | 2.16 | 6.97 | |
| 90                        | 1.50 | 3.43 | |

(b) Measured Distance [m]

| Tilt angle $\phi_a$ [deg] | (A) | (B) | (C) | (D) | (E) |
|---------------------------|----|----|----|----|----|
| 0                         | 0.00 | 0.42 | 0.86 | |
| 10                        | 0.19 | 0.68 | 1.26 | |
| 20                        | 0.00 | 0.40 | 1.00 | 1.85 | |
| 30                        | 0.15 | 0.65 | 1.46 | 3.00 | |
| 40                        | 0.00 | 0.35 | 0.95 | 2.15 | 6.03 |
| 50                        | 0.20 | 0.60 | 1.40 | 3.70 | |
| 60                        | 0.41 | 0.89 | 2.05 | 7.00 | |
| 70                        | 0.67 | 1.29 | 3.30 | |
| 80                        | 0.97 | 1.87 | 7.12 | |
| 90                        | 1.40 | 2.90 | |

smartphone and showed that the proposed calculation method is appropriate by the experiments. If the actual distance was within 2m, the average error was 7cm, and the maximum was 18cm. Although there were such slight errors, it was considered to be within an acceptable range when it is assumed that the visually impaired will be notified of the existence of an obstacle.

As a future prospect, we plan to implement a function to recognize multiple obstacles by applying YOLO etc. into the smartphone application and obtain the distance to the obstacle using its position on the image.

Fig. 8. Experimental Result Graph using Smartphone B.

5. Conclusions

This paper explained the method of obtaining the distance to the obstacle using the acquired image by smartphone and showed that the proposed calculation method is appropriate by the experiments. If the actual distance was within 2m, the average error was 7cm, and the maximum was 18cm. Although there were such slight errors, it was considered to be within an acceptable range when it is assumed that the visually impaired will be notified of the existence of an obstacle.

As a future prospect, we plan to implement a function to recognize multiple obstacles by applying YOLO etc. into the smartphone application and obtain the distance to the obstacle using its position on the image.

References

(1) Ministry of Health, Labour and Welfare: "Facts about persons with disabilities or illnesses, etc. and government initiatives", Annual Health, Labour and Welfare Report 2018, Chapter 1 (2018)

(2) Manabu Shimakawa, Kosei Matsushita, Issei Taguchi, Chiharu Okuma, Kimiyasu Kiyota: "Smartphone Apps of Obstacle Detection for Visually Impaired and its Evaluation", 7th ACIT International Conference on Applied Computing & Information Technology (ACIT2019), pp.148-153 (2019)

(3) Mamabu Shimakawa, Issei Taguchi, Chiharu Okuma and Kimiyasu Kiyota: "Smartphone Application Program of Obstacle Detection for Visually Impaired People", ICIC Express, Part B: Applications, Vol.10, No.3, pp.219-226 (2019)

(4) Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, Zbigniew Wojna, "Rethinking the Inception Architecture for Computer Vision", arXiv:1512.00567 (2015)

(5) Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi, "You Only Look Once: Unified, Real-Time Object Detection", https://arxiv.org/abs/1506.02640 (2018)