Distance Estimation Methods for Smartphone-based Navigation Support Systems

Bineeth Kuriakose*, Raju Shrestha, and Frode Eika Sandnes

Department of Computer Science, Oslo Metropolitan University, Oslo, Norway
*bineethk@oslomet.no

Abstract. Distance estimation is a key element of a navigation system. Various methods and instruments are used in distance estimation procedures. The methods and instruments used usually depend on the contexts of the application area. This paper compares the accuracy of five practical distance estimation methods that can be used on portable devices. Some of the methods selected for this study have currently not yet been used in the context of navigation systems. The experimental results show that *Rule 57* and *AR based* distance estimation methods hold great potential for the practical application of navigation support as they provide adequate accuracy and computational efficiency.

Keywords: distance estimation, object detection, computer vision, navigation, smartphones, assistive technology

1 Introduction

Navigation involves monitoring or controlling the movement of a vehicle or person, or any machine from one location to another through an environment with constraints and obstacles. Much research and development have recently been reported in the domain of navigation systems. The advancements in the field of computer vision and machine learning have probably contributed to the accelerated developments in this area. Autonomous cars [1,2], robotic navigation [3,4], navigation of people with or without accessible needs [5,6,7] are some of the areas receiving research attention. In addition to obstacle (object) detection, distance estimation is also a key component in navigation systems [8]. Obviously, information about how far an object is from the viewer can be used to avoid collisions during navigation.

Distance estimation methods can be classified as active methods and passive methods [9]. Active methods send signals to the obstacle, and based on the time the signal takes to reach the obstacle and bounce back, the distance to the obstacle is estimated [10,11]. Such methods may use laser beams [12,13], ultrasound [14,15], or radio signals [16,17]. Passive methods estimate the distance by receiving information about the object's position by applying computer vision techniques on camera images. Passive methods can be based on monovision (single camera) [18,19] or stereo vision systems (double cameras separated by a small distance) [20,21].

This is a post-peer-review, pre-copyedit version of a conference proceeding published in IntelliSys 2021: Intelligent Systems and Applications; Proceedings of the 2021 Intelligent Systems Conference (IntelliSys) Volume 2, which is part of the Lecture Notes in Networks and Systems book series (volume 295). The final authenticated version is available online at DOI: https://doi.org/10.1007/978-3-030-82196-8_49

The use of distance estimation methods in portable navigation support systems for assisting people with visual impairment is the primary concern of this work. According to the World Health Organization (WHO), more than 22 hundred million people worldwide have a vision impairment or can be classified as technically blind[1]. People with visual impairments often experience challenges during navigation. However, tools such as white canes and guide dogs give the users more freedom to navigate independently. WHO uses *assistive technology* as an umbrella term to refer to both systems and services for people with reduced functioning. The primary purpose of assistive devices and technologies is to maintain or improve independence and dignity, to facilitate participation and enhance overall well-being[2].

Portability is one of the vital requirements in navigation systems for people with visual- impairment [22]. Portability is mainly characterized by small physical dimensions and low mass. Portable systems provide comfort and convenience during carriage and usage. When prioritizing portability, smartphones emerge as a feasible and practical technological platform. Several works have reported on how smartphones can be used in navigation systems for people with visual impairments [23,24,25,26]. Moreover, one should use the functionalities available on smartphones and avoid the use of any peripheral hardware such as bulky cameras or add-on sensors that increase the overall system dimensions and mass [25]. Some studies have also reported on the use of various sensors for navigation distance estimation [27].

The main objective of this work was to evaluate five distance estimation methods in the context of a portable navigation system for people with visual impairments. The methods are chosen based on their potential to be used with smartphones. Eventhough many works are reported on object distance estimation in general, few studies have addressed distance estimation in context of smartphone assisted navigation for blind users specifically. To the best of our knowledge this is one of the first attempts at experimenting and analyzing established object distance methods in the problem domain of smartphone-based navigation support.

The paper is organized as follows. Section 2 discusses the distance estimation in general and the major components involved. Section 3 describes the five distance estimation methods that we considered for our experiments. Section 4 describes the experimental procedures. The results are presented and discussed in Section 5. The paper ends with the conclusion in Section 6.

2 Distance Estimation

Many methods have been proposed for the estimation of the distance between the object (or obstacle) to the viewer. The distance estimation process in navigation usually consists of object detection and computation of distance from the objects. They are described in the following subsections.

[1] https://www.who.int/news-room/fact-sheets/detail/blindness-and-visual-impairment
[2] https://www.who.int/news-room/fact-sheets/detail/assistive-technology

2.1 Object detection

Object detection is the task of detecting instances of objects of interest in an image. Different challenges such as partial/full occlusion, varying illumination conditions, poses, and scale are needed to be handled while performing object detection [28,29]. As an essential navigation component, object detection is sometimes considered a pre-phase in the distance computation procedure.

Object detection methods, in general, use machine learning and deep learning algorithms. A typical object detection pipeline consists of a Convolutional Neural Network (CNN). CNN is a type of feed-forward neural network and works on the principle of weight sharing. Some benchmarked datasets, such as MS COCO [30] and ImageNet [31], make object detection using deep learning a preferable choice among developers [29].

In addition to different object detection models that are computationally expensive, various lightweight object detection models intended to be used in mobile devices are available. You Only Look Once (YOLO) [32,33,34], Single Shot Detector (SSD)-MobileNetV1 [35] are some examples. YOLO divides each image into grids, and each grid predicts bounding boxes of detected objects with confidence values. The SSD architecture is a single convolution network that learns to predict bounding box locations and classify these in a single pass [36]. SSD combined with MobileNet as its base network gives better object detection results in terms of accuracy compared to similar other deep learning models [37].

This study uses the SSD-MobileNetV2 [36,38] object detection model as a prephase to the distance estimation methods. The main reasons for this choice are portability and usability. Moreover, SSD-MobileNet has the highest mean average precision (mAP) among the models that facilitate real-time processing. Even though the latest version of the series, MobileNetV3 offers higher accuracy and speed in general classification tasks than MobileNetV2, while MobileNetV2 provides higher performance for object detection tasks than MobileNetV3 [39].

In the SSD-MobileNetV2 model, input image features are extracted by the CNN layers in the MobileNetV2, and SSD predicts the obstacles based on the feature maps. The MobileNetV2 architecture is shown in Figure 1. MobileNetV2 is based on an inverted residual structure with residual connections between the bottleneck layers. The intermediate expansion layer uses lightweight depthwise convolutions to filter features as a source of nonlinearity. The MobileNetV2 architecture contains an initial fully convolution layer with 32 filters, followed by 19 residual bottleneck layers. ReLU6 is used as the nonlinearity because of its robustness when used in devices with low computational power [38].

After the extraction of basic features, several layers of depth-wise separable convolution are operated to generate several feature maps with decreasing scales. SSD performs on multiscale feature maps to predict multiscale objects [36]. Each feature map is evenly divided into cells; every cell predicts k bounding boxes and c category confidences. For each category, the top *'n'* bounding boxes are retained. Then the non-maximum suppression is performed to filter out the bounding boxes with considerable overlap, and finally it outputs the

detection results. The model output comprises the detected objects along with their bounding box coordinates.

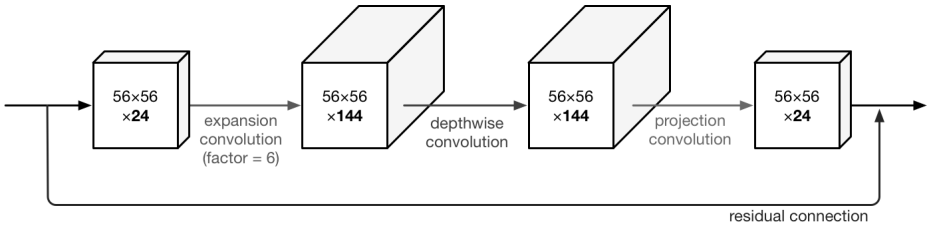

Fig. 1. The architecture of MobileNetV2 (adapted from [38]).

2.2 Distance computation

Several distance computation methods have been proposed in the literature. Some require object detection as a prerequisite to compute distance, while others compute distance based on information from specific sensors. Examples of such sensors include stereo cameras [40,41], monocular cameras [42,43], ultrasonic sensors [44,45], Light Detection and Ranging (LiDaR) [46], and Time of Flight (ToF) sensors [47].

With the stereo vision method, two cameras separated by certain distance capture two images of a scene from two slightly different vantage points, which are used to calculate their disparity. This disparity helps to estimate the depth, enabling projection of the scene to a 3D world that can be used for navigation [40,41]. With the monocular vision-based distance estimation, the image captured by a single camera is used to compute distance. The distance is estimated either using a traditional distance estimation model, such as the pinhole imaging model or deep learning-based methods [42,43]. Ultrasonic distance sensors are also widely used to measure the distance between the source and target using ultrasonic waves [44,45]. In robotics applications and autonomous vehicles, LiDaR is commonly used to measure the distance by illuminating the target with laser light and measuring the reflection with a sensor. LiDaR uses the Simultaneous Localization and Mapping (SLAM) technique, which builds a map representing its spatial environment while keeping track of any robot or vehicle within the map of the physical world [46]. Time-of-flight (ToF) sensors are also widely used in range imaging camera systems. A ToF sensor computes the distance between the camera and the object for each point of the image by measuring the round-trip time of an artificial light signal from a laser or an LED [47].

Besides sensors, different computational methods can be used to compute the distance. Optical methods [48] and Rule of 57 [49] are examples of methods that do not require any hardware besides a smartphone. Therefore, the purpose of

this work was to compare these methods. These methods are described in detail in the next section.

3 Distance Estimation Methods

Among the distance estimation methods described in Section 2 we selected five methods that can be used on a smartphone without any additional hardware and computational complexity barrier. Some of the distance estimation methods explored in this work are being used in different scenarios other than in the context of navigation support. But through this work we explored how to map those methods to use with a smartphone device. To ensure an unbiased assessment, the SSD-MobileNetV2 was used for object detection with all distance estimation methods.

3.1 Optical method

The relationship between the object distance and the image distance is defined by the *lens makers* equation is defined as [48],

$$\frac{1}{f} = \frac{1}{d_o} + \frac{1}{d_i} \tag{1}$$

where, f is the focal length of a camera lens, d_o is the distance from the lens to the target object, and d_i is the distance between the lens and the projected image. The following expression is used to compute the distance (d_o) from the object's bounding boxes [48],

$$distance(in\ inches) = \frac{(2 \times \pi \times 180)}{(w + h \times 360) \times 1000 + 3} \tag{2}$$

where, w and h are the width and height of the bounding box of the object detected by the object detection model. We refer to [50] for details on the derivation of the expression.

3.2 Smartphone position sensors based

Most of today's smartphones come with two types of position sensors that can help determine a device's position: the geomagnetic field sensor and the accelerometer. These position sensors can be used to determine the physical position in the world's frame of reference. The combination of the geomagnetic field sensor with the accelerometer can be used to determine a device's position relative to the magnetic north. These sensors can also be used to determine the device orientation with respect to the frame of reference defined in the app. Both the Android and the iOS platforms support specific functions to access these sensor data [51,52,53].

If the angle between the camera and the object is a, then by the right-angled triangle property, the tangent of the angle a gives the expression to find the distance d from the camera and the object.

$$d = h \times tan\ a \tag{3}$$

where h is the height from the base (camera height), and d is the distance from the object to the camera.
The estimation of the angle can be computed using the sensors present in the smartphone. In an Android platform, it is possible to access this sensor data using its sensor framework [54]. Using the different sensors supported in a smartphone, such as accelerometer and magnetometer, it is possible to find the angle between the object and the phone camera [51,53,52].

The implementation details for Android are as follows: From the accelerometer and magnetometer sensor values, it is possible to compute a rotation matrix and an orientation matrix. The rotation matrix involves mapping a point in the phone coordinate system to the real-world coordinate system. And the orientation matrix is derived from the rotation matrix. From the orientation matrix, we can compute the *pitch* and *roll*. Using *pitch* or *roll* depending on whether the phone is in portrait or landscape mode, the distance can be estimated using the following equation.

$$distance, d = h \times tan(pitch \mid roll \times \pi/180) \tag{4}$$

where, h, denotes the height of the camera from the base in meters, and in our case, it was set to 1.4.

3.3 Augmented Reality based methods

Augmented reality (AR) can be described as an enhanced version of the real physical world that is executed through the use of digital visual elements, sound, or other sensory stimuli delivered using technology. Several commercial stakeholders such as Google and Apple are incorporating AR technology into smartphones for multiple applications. ARCore is Google's Augmented Reality (AR) developer platform, which provides simple but powerful tools to developers for creating AR experiences [55]. The *com.google.ar.core* package helps to design applications that make it possible to determine the distance from a smartphone's camera to an object. The *anchor* class in the same package describes various methods to find a fixed location and orientation in the real world. Besides, to stay at a fixed location in physical space, the numerical description of that position will update as ARCore's understanding of the space improves. The limitation on the usage of the ARCore is that it only supports to work in ARCore compatible devices[3].

ARCore can create depth maps containing data about the distance between surfaces from a given point, using the primary RGB camera of a supported

[3] https://medium.com/@shibuiyusuke/measuring-distance-with-arcore-6eb15bf38a8f

device. ARCore uses the Simultaneous Localization and Mapping (SLAM) technique, to understand where the phone is relative to the world around it. It detects visually distinct features in the captured camera image called feature points and uses them to compute its location change. The visual information is combined with inertial estimation from the device's IMU to estimate the camera's pose (position and orientation) relative to the world over time.

Using ARCore, it is possible to place an anchor, a fixed location in the real world, and find the camera's distance to the anchor. Both the anchor position and the camera position can be acquired as x, y, and z values (width, height, and depth) corresponding to the world position of objects in the ARCore package [56]. Once the two positions are known, it is straightforward to calculate the Euclidean distance between them.

3.4 Method based on *Rule of 57*

The *Rule of 57* states that an object with an angular size of 1° is about 57 times further away than it is big (see Figure 2). Therefore, the ratio of an object's angular size (in degrees) to a whole 360-degree circle should equal the ratio of the object's actual size to the circumference of a circle at that distance from the observer. This method has been derived for measuring distance and angles from telescope images in astronomy [49]. The key to using telescope images to measure distances is to realize that an object's apparent angular size is directly related to its actual size and distance from the observer. It means that if the object appears to be smaller as it is farther away from the observer. However, in our experiments, we found it can be applied to find the distance to the object even if the angular size of the object is more than 1° to the field of view of the smartphone camera sensor.

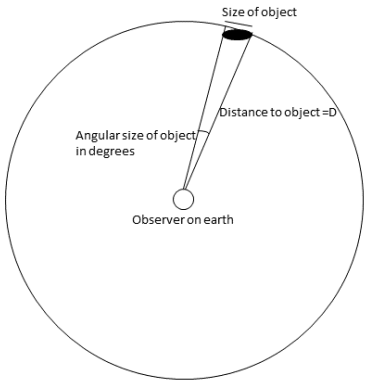

Fig. 2. The Rule of 57 (adapted from [49]).

From Figure 2, we can write that,

$$\frac{angular\_size}{360^\circ} = \frac{actual\_size}{2\pi D} \tag{5}$$

Using object distance for angular size and object size for actual size, we get the following equation to calculate the distance,

$$object\_distance = (object\_size) \times \frac{1}{(angular\ size\ in\ degrees)} \times 57 \tag{6}$$

In order to use this approach, it is necessary to get an estimate of the size of objects before finding the distance from them. We measured the size of the object (height) for our experiment. The geomagnetic field sensor and the accelerometer sensor measure the angular size [51,53,52]. Both sensors are used in a similar manner as described in Section 3.2.

3.5 DisNet method

DisNet uses multi-layer neural network to estimate the distance [57]. The method can be used to learn and predict the distance between the object and the camera sensor. A six-dimensional feature vector (v) can be obtained from the bounding box of the object detected by the object detection model is used as input to the DisNet as,

$$v = [1/B_h \quad 1/B_w \quad 1/B_d \quad C_h \quad C_w \quad C_b] \tag{7}$$

where, B_h, B_w and B_d denotes the height, width, and diagonal of the object bounding box in pixels/image, respectively. And C_h, C_w and C_b represents the values of average height, width, and breadth of the particular class's object. For example, for the class *person*, C_h, C_w and C_b are, respectively, 175 cm, 55 cm and 30 cm. These values were chosen based on an average case assumption. The features C_h, C_w, and C_b are assigned to objects labeled by the SSD+MobileNetV2 detector as belonging to the particular class to complement more information to distinguish different objects.

Finally, the DisNet model outputs the estimated distance of the object to the camera sensor. In the original work of DisNet [57], YOLO was used as the object detector. However, in our work, we used the SSD+ MobileNetv2 object detection model, typical for all methods already described. An illustration of how the model works is given in Figure 3.

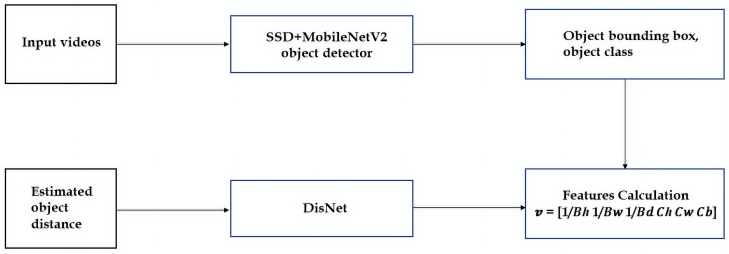

Fig. 3. The DisNet based distance estimation (adapted from [57]).

The SSD+MobileNet model was pretrained with the COCO dataset [30]. We applied transfer learning to train more classes such as *person*, *bag*, and *chair*. The images were collected using a smartphone camera. For the DisNet model, it is necessary to collect the distance to the objects (classes) in addition to the images. For the class *person*, the dataset was already available from the reference paper [57]. For the other two (*chair* and *bag*), along with the images, its ground truth distance to the camera was measured and recorded. To train the network, the input dataset was randomly split into a training set (80% of the data), validation set (10% of the data), and a test set (10% of the data). After calculating the input vector, the DisNet model was trained using the custom dataset. The output of the model gives the distance of the object from the camera sensor.

4 Experiment

An Android app was developed and deployed in Huawei P30 Pro smartphone to assess the performance of each of the methods. One of the reasons for the selection of smartphone device is the requirement of AR-enabled device for AR-based distance estimation.

The independent variables included the ground truth distance and object size. Observed distance and computational overload were the dependent variables. Other evaluation parameters include how a moving camera could affect the distance estimation, and how the object's size, and distance and accuracy in each method are related.

Four types of objects were used to estimate the distance from them in our experiments. The selection of objects was made with varying physical sizes, namely, a *bottle*, *bag*, *chair*, and *person*. This particular selection was made to understand the effect of varying sizes on each distance estimation method. We use the term *distance marker* to refer to ground truth values. The selection of different distance markers was made to analyze the estimation method's effect at various distances (near, medium, and far).

The distance markers were placed at four different spots (very near-1m, near-2m, medium-5m, far-10m) away from the observation point. In the first round of the experiment, the *bag* object was placed one meter from the marker. Then we

measured the distance using the five different methods. However, we were unable to estimate distance when the object is placed in a 1 meter distance marker using all methods. Next, the object was kept 2 meters away. Again, five methods were used to obtain the measurements and these measurements were recorded. The next step involved measuring the distance at 5 meters. Finally for the 10 meter case, we were unable to estimate the distance from the object. We repeated the same procedure using other objects (such as *chair* and *person*). However, we observed the same situation as in the previous case. We were unable to measure the distance at 1 and 10 meters but were able estimate the distance when the object was placed at a distance of 2 and 5 meters. A detailed explanation of the possible causes for this is given in the discussion section.

To identify whether the size was a factor in distance estimation, we placed a smaller object *bottle* at the 1m marker. Surprisingly, we were able to measure the distance this time with four methods. We were unable to estimate the distance using the DisNet method in the 1m case. However, with other distance markers (2m, 5m, 10m), we were unable to find the distance to the *bottle* object.

To identify the maximum range of the distance methods, we moved the object from 10m to closer to the camera point. However, we observed that distance estimation was not possible beyond 5m. We therefore concluded that the maximum distance possible with the methods studied is 5m. Furthermore, beyond the distance marker of 5m, it is impossible to compute the distance using any of the methods described here. All the experiments were conducted in a controlled indoor environment (room) during midday.

5 Results and Discussions

5.1 Results

We took five samples using each method and then calculated the mean and standard deviation of the value corresponding to each object. The results of the experiments with different objects using different distance computation methods are given in Table 1. Results are also graphically shown in Figure 4.

Table 1. Mean and standard deviation of the estimated distances of the different objects for three different distance markers using the five methods.

Method	**1m***	**2m**		**5m**	
	Mean	Mean	SD	Mean	SD
Optical method	1.3	2.36	0.05	6.17	0.32
Position sensors	1.5	2.53	0.05	4.39	0.37
AR method	0.95	2.34	0.11	5.11	0.34
Rule 57 method	0.76	2.01	0.35	4.94	0.57
DisNet	-	2.86	0.21	6.03	0.32

** for 'bottle' object only*

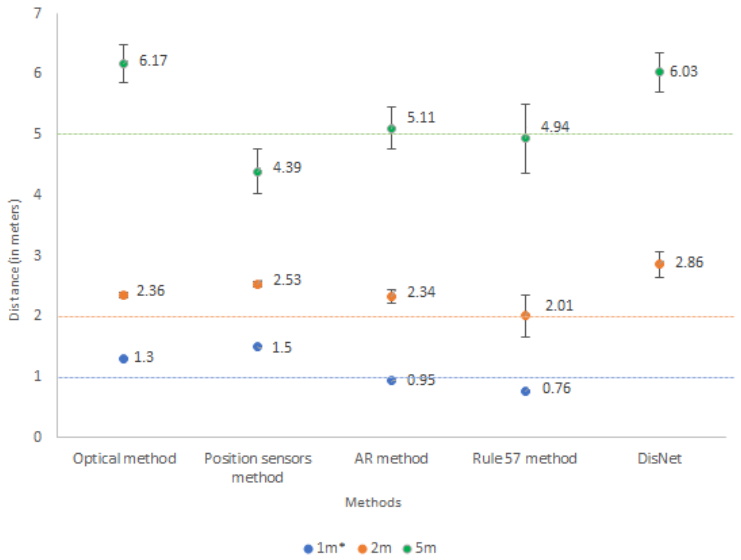

Fig. 4. Mean distances (error bars shows the standard errors), estimated with the five distance estimation methods.

The 1 meter case is only applicable to the smallest object *bottle*. The 10 meters case is out-of-range since we were unable to obtain any estimates. There were variations in the 5 meters distance marker estimation for all objects for each distance estimation method.

5.2 Discussion

From Figure 4, it is clear that the deviation from the ground truth is the smallest when using the ARCore method when the object is placed 1m away. However, when the object is placed either 2m or 5m away, the *Rule 57* method gives the most accurate results. The largest variation is observed when the object is placed at a 1 meter distance marker with the smartphone sensors. We also considered how various factors can affect distance estimation. They are discussed below.

Effect of distance: When we tried to compare different objects (*bag, chair, person*), the distance estimation methods showed fair results up to 4 meters. However, there were varying results when the object is placed more than 4 meters away. Hence, we decided to consider the values of 5 samples and find the mean and standard deviation when the object marker is placed at a distance of 5 meters. When the object marker is placed more than 5 meters away, no distance estimate could be obtained.

Size of object: The size of the object is also an important parameter that affects distance estimation. It was understood from the experiment that the distance could not be estimated through the above-selected methods when the object size is small or if it is placed far away. However, we did a small experiment on how well the distance estimation method could perform when the object size is small and placed in one meter in our experiment. The testing with the *bottle* object shows that the distance estimation is possible within 1 meter. When we tried to place the object far away from that point, none of the distance estimations showed any results.

Moving camera: We tried to estimate the distance to the moving camera. For this, we placed the object at a 5 meter distance. And marked path with the distance markers - 1 meter and 2 meters away from the starting point (initial position of the camera). We moved with the camera in each of those distance markers in parallel (left to right direction) to the object. We observed that there were fluctuations in the distance estimation when the camera were moving. However, still, it was able to do distance estimation. However, in some methods, such as (AR-based or smartphone sensor-based), camera focusing is required to estimate the object's distance.

We assume there are some reasons for these observations. The camera sensor's size is one reason that can affect the estimation of distant objects and objects with small sizes. Since we tested all methods with a smartphone camera, the limitation of the same may involve detecting distant objects. This results in the fact that all methods considered for the study cannot be used for long range applications. Probably by using a long range camera can elude the limitation. However, since we are focusing only on portable navigation solutions, the idea of using long range cameras which can increase the system weight does not hold well in our case. When the size of the object is small, and the object is kept far, it is not easy to get detected using the methods described in the experiment. Another factor we think of is the lighting effect of the environment. Since the experiment was done in an indoor setting, we should consider the effect of lighting as well. Furthermore, the smartphone used for the experiment was held in bare hands without any anti-motion devices. Therefore, when the camera experiences fluctuations, this could have affected the object detection. This might be one reason which affects the varying observations at the same distance marker. However, it also to be noted that the smartphone we used for the experiment has a good video image stabilization[4].

The AR-enabled smartphones available today are already used to estimate various metrics such as calculating an object's length. But the smartphone AR features can be further enhanced and used in the applications such as navigation to assist people. Moreover, the distance estimation methods explored in this work are tested in smartphone devices can be used in other portable devices or miniature computing devices such as Raspberry Pi to develop applications other than navigation.

[4] https://www.dxomark.com/huawei-p30-pro-camera-review/

6 Conclusion

This study does an analysis of different distance estimation methods and their performance. We did a controlled and structured experiment on how a smartphone can be used in distance estimation tasks without any additional hardware. Our findings reveal which distance estimation method that is appropriate for short-range navigation applications. The *Rule 57* distance estimation method holds great potential. The *AR-based* method could also be considered a viable alternative, though it requires an AR-compatible device. Moreover, the result also shows that none of the methods are suitable for long-range applications beyond 5 meters. We believe that this study could help developers and researchers in making informed choices of technologies when designing systems involving distance estimation. Future work involves in using the results from this research in the development of a smartphone-based navigation system for people with visual impairments.

References

1. Sarma Emani, KP Soman, VV Sajith Variyar, and S Adarsh. Obstacle detection and distance estimation for autonomous electric vehicle using stereo vision and DNN. In *Soft Computing and Signal Processing*, pages 639–648. Springer, 2019.
2. Dragan Obradovic, Henning Lenz, and Markus Schupfner. Fusion of map and sensor data in a modern car navigation system. *Journal of VLSI signal processing systems for signal, image and video technology*, 45(1-2):111–122, 2006.
3. Haiyang Chao, Yu Gu, and Marcello Napolitano. A survey of optical flow techniques for robotics navigation applications. *Journal of Intelligent & Robotic Systems*, 73(1-4):361–372, 2014.
4. Hiram Ponce, Jorge Brieva, and Ernesto Moya-Albor. Distance estimation using a bio-inspired optical flow strategy applied to neuro-robotics. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7. IEEE, 2018.
5. Ramanamurthy Dantu. Methods and systems for indoor navigation. https://patents.google.com/patent/US20120143495A1/en, June 2012. Accessed: 01-10-2020.
6. Navid Fallah, Ilias Apostolopoulos, Kostas Bekris, and Eelke Folmer. Indoor human navigation systems: A survey. *Interacting with Computers*, 25(1):21–33, 2013.
7. Vítor Filipe, Filipe Fernandes, Hugo Fernandes, António Sousa, Hugo Paredes, and João Barroso. Blind navigation support system based on Microsoft Kinect. *Procedia Computer Science*, 14:94–101, 2012.
8. Jing Zhu and Yi Fang. Learning object-specific distance from a monocular image. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3839–3848, 2019.
9. Abdelmoghit Zaarane, Ibtissam Slimani, Wahban Al Okaishi, Issam Atouf, and Abdellatif Hamdoun. Distance measurement system for autonomous vehicles using stereo camera. *Array*, 5:100016, 2020.
10. Faisal Mufti, Robert Mahony, and Jochen Heinzmann. Robust estimation of planar surfaces using spatio-temporal RANSAC for applications in autonomous vehicle navigation. *Robotics and Autonomous Systems*, 60(1):16–28, 2012.

11. A Prusak, O Melnychuk, H Roth, Ingo Schiller, and Reinhard Koch. Pose estimation and map building with a time-of-flight-camera for robot navigation. *International Journal of Intelligent Systems Technologies and Applications*, 5(3-4):355–364, 2008.
12. Farhad Aghili and Kourosh Parsa. Motion and parameter estimation of space objects using laser-vision data. *Journal of guidance, control, and dynamics*, 32(2):538–550, 2009.
13. Fitri Utaminingrum, Tri Astoto Kurniawan, M Ali Fauzi, Rizal Maulana, Dahnial Syauqy, Randy Cahya Wihandika, Yuita Arum Sari, and Putra Pandu Adikara. A laser-vision based obstacle detection and distance estimation for smart wheelchair navigation. In *2016 IEEE International Conference on Signal and Image Processing (ICSIP)*, pages 123–127. IEEE, 2016.
14. Seong Jin Kim and Byung Kook Kim. Dynamic ultrasonic hybrid localization system for indoor mobile robots. *IEEE Transactions on Industrial Electronics*, 60(10):4562–4573, 2012.
15. Jaroslaw Majchrzak, Mateusz Michalski, and Grzegorz Wiczynski. Distance estimation with a long-range ultrasonic sensor system. *IEEE Sensors Journal*, 9(7):767–773, 2009.
16. Pedro Coronel, Simeon Furrer, Wolfgang Schott, and Beat Weiss. Indoor location tracking using inertial navigation sensors and radio beacons. In *The Internet of Things*, pages 325–340. Springer, 2008.
17. Helena Leppäkoski, Jussi Collin, and Jarmo Takala. Pedestrian navigation based on inertial sensors, indoor map, and WLAN signals. *Journal of Signal Processing Systems*, 71(3):287–296, 2013.
18. Koray Celik, Soon-Jo Chung, and Arun Somani. Mono-vision corner SLAM for indoor navigation. In *2008 IEEE International Conference on Electro/Information Technology*, pages 343–348. IEEE, 2008.
19. Kamilla Run Johannsdottir, Lew B Stelmach, et al. Monovision: a review of the scientific literature. *Optometry and vision science*, 78(9):646–651, 2001.
20. David J Kriegman, Ernst Triendl, and Thomas O Binford. Stereo vision and navigation in buildings for mobile robots. *IEEE Transactions on Robotics and Automation*, 5(6):792–803, 1989.
21. Annett Stelzer, Heiko Hirschmüller, and Martin Görner. Stereo-vision-based navigation of a six-legged walking robot in unknown rough terrain. *The International Journal of Robotics Research*, 31(4):381–402, 2012.
22. Bineeth Kuriakose, Raju Shrestha, and Frode Eika Sandnes. Tools and Technologies for Blind and Visually Impaired Navigation Support: A Review. *IETE Technical Review*, pages 1–16, 2020.
23. Dianyuan Han and Chengduan Wang. Tree height measurement based on image processing embedded in smart mobile phone. In *2011 International Conference on Multimedia Technology*, pages 3293–3296. IEEE, 2011.
24. Clemens Holzmann and Matthias Hochgatterer. Measuring distance with mobile phones using single-camera stereo vision. In *2012 32nd International Conference on Distributed Computing Systems Workshops*, pages 88–93. IEEE, 2012.
25. Bineeth Kuriakose, Raju Shrestha, and Frode Eika Sandnes. Smartphone Navigation Support for Blind and Visually Impaired People-A Comprehensive Analysis of Potentials and Opportunities. In *International Conference on Human-Computer Interaction*, pages 568–583. Springer, 2020.
26. Juan Zhang and Xin-yuan Huang. Measuring method of tree height based on digital image processing technology. In *2009 First International Conference on Information Science and Engineering*, pages 1327–1331. IEEE, 2009.

27. Shangwen Chen, Xianyong Fang, Jianbing Shen, Linbo Wang, and Ling Shao. Single-image distance measurement by a smart mobile device. *IEEE transactions on cybernetics*, 47(12):4451–4462, 2016.
28. Wang Hechun and Zheng Xiaohong. Survey of deep learning based object detection. In *Proceedings of the 2nd International Conference on Big Data Technologies*, pages 149–153, 2019.
29. Licheng Jiao, Fan Zhang, Fang Liu, Shuyuan Yang, Lingling Li, Zhixi Feng, and Rong Qu. A survey of deep learning-based object detection. *IEEE Access*, 7:128837–128868, 2019.
30. Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.
31. Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.
32. Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.
33. Joseph Redmon and Ali Farhadi. YOLO9000: Better, Faster, Stronger. *arXiv preprint arXiv:1612.08242*, 2016.
34. Joseph Redmon and Ali Farhadi. YOLOv3: An Incremental Improvement. *arXiv*, 2018.
35. Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.
36. Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.
37. Wattanapong Kurdthongmee. A comparative study of the effectiveness of using popular DNN object detection algorithms for pith detection in cross-sectional images of parawood. *Heliyon*, 6(2):e03480, 2020.
38. Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.
39. Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1314–1324, 2019.
40. Emre Dandil and Kerim Kürşat ÇeviİIk. Computer vision based distance measurement system using stereo camera view. In *2019 3rd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT)*, pages 1–4. IEEE, 2019.
41. Rahul Kala. *On-road intelligent vehicles: Motion planning for intelligent transportation systems*. Butterworth-Heinemann, 2016.
42. Lin Chenchen, Su Fulin, Wang Haitao, and Gao Jianjun. A camera calibration method for obstacle distance measurement based on monocular vision. In *2014 Fourth International Conference on Communication Systems and Network Technologies*, pages 1148–1151. IEEE, 2014.

43. Xuanyin Wang, Bin Zhou, Jiayu Ji, and Bin Pu. Recognition and distance estimation of an irregular object in package sorting line based on monocular vision. *International Journal of Advanced Robotic Systems*, 16(1):1729881419827215, 2019.
44. Gabriel Găşpăresc and Aurel Gontean. Performance evaluation of ultrasonic sensors accuracy in distance measurement. In *2014 11th International Symposium on Electronics and Telecommunications (ISETC)*, pages 1–4. IEEE, 2014.
45. Lianjun Zhang and Lifang Zhao. Research of ultrasonic distance measurement system based on DSP. In *2011 International Conference on Computer Science and Service System (CSSS)*, pages 2455–2458. IEEE, 2011.
46. Guolai Jiang, Lei Yin, Shaokun Jin, Chaoran Tian, Xinbo Ma, and Yongsheng Ou. A simultaneous localization and mapping (SLAM) framework for 2.5 D map building based on low-cost LiDAR and vision fusion. *Applied Sciences*, 9(10):2105, 2019.
47. S Burak Gokturk, Hakan Yalcin, and Cyrus Bamji. A time-of-flight depth sensor-system description, issues and solutions. In *2004 conference on computer vision and pattern recognition workshop*, pages 35–35. IEEE, 2004.
48. M. A. Khan, P. Paul, M. Rashid, M. Hossain, and M. A. R. Ahad. An AI-Based Visual Aid With Integrated Reading Assistant for the Completely Blind. *IEEE Transactions on Human-Machine Systems*, pages 1–11, 2020.
49. Measuring size from images: A wrangle with angles and image scale. https://www.cfa.harvard.edu/webscope/activities/pdfs/measureSize.pdf, November 2012. Accessed: 01-10-2020.
50. L. Xiaoming, Qin Tian, Chen Wanchun, and Y. Xingliang. Real-time distance measurement using a modified camera. In *2010 IEEE Sensors Applications Symposium (SAS)*, pages 54–58, 2010.
51. Google. Position Sensors. https://developer.android.com/guide/topics/sensors/sensors_position, April 2018. Accessed: 01-10-2020.
52. StackOverFlow. How can we measure distance between object and android phone camera. https://stackoverflow.com/questions/15949777/how-can-we-measure-distance-between-object-and-android-phone-camera, May 2013. Accessed: 05-11-2020.
53. Kleomenis Katevas. SensingKit/SensingKit-iOS. https://github.com/SensingKit/SensingKit-iOS, Oct 2019. Accessed: 2020-10-01.
54. Google. Sensors Overview: Android Developers. https://developer.android.com/guide/topics/sensors/sensors_overview, 2020. Accessed: 05-11-2020.
55. Google ARVR. Build new augmented reality experiences that seamlessly blend the digital and physical worlds. https://developers.google.com/ar, Jun 2016. Accessed: 05-11-2020.
56. Ian M. How to measure distance using ARCore? https://stackoverflow.com/questions/45982196/how-to-measure-distance-using-arcore, Aug 2017. Accessed: 05-11-2020.
57. Muhammad Abdul Haseeb, Jianyu Guan, Danijela Ristic-Durrant, and Axel Gräser. Disnet: A novel method for distance estimation from monocular camera. *10th Planning, Perception and Navigation for Intelligent Vehicles (PPNIV18), IROS*, 2018.