Indoor Query System For The Visually Impaired

Lizhi Yang[0000−0002−6101−7610], Ilian Herzi, Avideh Zakhor*, Anup Hiremath, Sahm Bazargan, and Robert Tames-Gadam

University of California, Berkeley
{lzyang,ilianherzi,avz,anup.h,sahm.bazargan,robert_gadams}@berkeley.edu

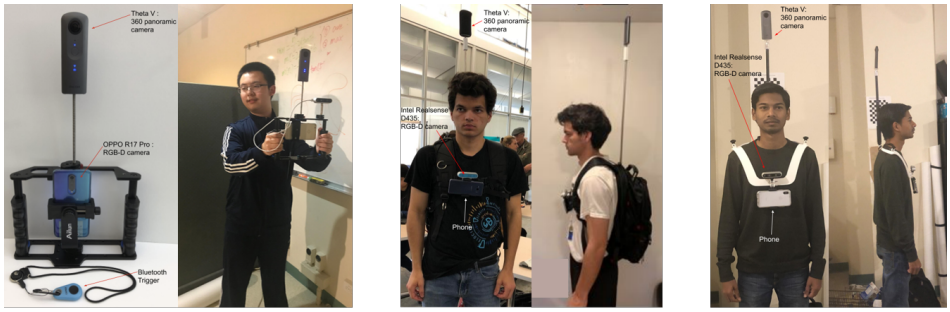
Abstract. Scene query is an important problem for the visually impaired population. While existing systems are able to recognize objects surrounding a person, one of their significant shortcomings is that they typically rely on the phone camera with a finite field of view. Therefore, if the object is situated behind the user, it will go undetected unless the user spins around and takes a series of pictures. The recent introduction of affordable, panoramic cameras solves this problem. In addition, most existing systems report all "significant" objects in a given scene to the user, rather than respond to a specific user-generated query as to where an object located. The recent introduction of text-to-speech and speech recognition capabilities on mobile phones paves the way for such user-generated queries, and for audio response generation to the user. In this paper, we exploit the above advancements to develop a query system for the visually impaired utilizing a panoramic camera and a smartphone. We propose three designs for such a system: the first is a handheld device, and the second and third are wearable backpack and ring. In all three cases, the user interacts with our systems verbally regarding whereabouts of objects of interest. We exploit deep learning methods to train our system to recognize objects of interest. Accuracy of our system for the disjoint test data from the same buildings in the training set is 99%, and for test data from new buildings not present in the training data set is 53%.

Keywords: non-visual query system · object detection · RGB-D · panoramic camera system

1 Introduction

According to the 2019 WHO World Report on Vision, at least 2.2 billion people have a vision impairment globally[17]. Thus, it is no surprise that there has been consistent interest in providing non-visual descriptions of the environment to this population. Existing methods use GPS and RFID[23], or multiple cameras around the user[24] requiring pre-configured environments. Others draw

* This project was funded by Microsoft AI for Accessibility program.



(a) (b) (c)

Fig. 1. Three designs: (a) Handheld design consisting of a panoramic and a mobile phone with a time-of-flight camera. (b) Wearable design, backpack with depth sensor, panoramic camera and mobile phone. (c) Wearable custom 3-D printed ring with panoramic camera, depth sensor and mobile phone.

inspiration from biomimetics and use a dual-camera setup[11] to detect the object and its distance from the user. Recently, methods using RGB-D cameras have gained traction due to a simpler hardware setup. Some combine the aforementioned methods with state-of-the-art computer vision techniques[5], using a RGB-D camera for object detection and distance sensing[1]. A few commercial apps include the Be My Eyes phone app[6], a platform connecting the visually impaired person with sighted volunteers for assistance, and the TapTapSee[4] app utilizing the CloudSight Image Recognition API to identify objects. A recent advancement in commercial apps is the Microsoft Seeing AI[15], which carries out numerous tasks such as describing the scene around the user and identifying currency bills when paying.

While these existing systems are able to recognize objects surrounding a person, one of their significant shortcomings is that they typically rely on the phone camera with a finite field of view (FoV) to take a picture of the scene before objects are detected and reported to the user. As such, if the object is situated behind the user, it will go undetected unless the user spins around and takes a series of pictures or a video of the scene. Recent introduction of affordable, easy to use panoramic cameras, can easily solve the "spin around" problem by taking one image which can be used to detect objects of interest surrounding a person, thus obviating the need for the user to look or spin around. In addition, most existing systems merely report all "significant" objects in a given scene to the user, rather than respond to a specific user-generated query. In practice, the user might want to inquire whether or not a specific object is in his/her environment, and if so, approximately where it is located. Along these lines, our informal survey of the visually impaired population shows particular interest in navigation-related objects such as doors, exit signs, staircases, elevators, restrooms, as well as utilitarian objects such as electric plugs on the walls. The recent introduction of text-to-speech and speech recognition capabilities on mo-

bile phones paves the way for such user generated queries, and for audio response generation to the user.

In this paper, we propose a wearable query system for the visually impaired utilizing a panoramic camera and a mobile phone. The concept of operation of our system is as follows: the user pushes a button on a Bluetooth device to connect to the phone to activate the app and asks for a specific object. The phone then triggers the panoramic camera to take a picture of the surroundings and runs the recorded audio signal on a speech recognition engine. Once the phone receives the captured 360° picture from the panoramic camera, it runs a pre-trained object recognition model with the requested object as input, in order to identify it in the panoramic image; it then reports the direction of the object back to the user via the text-to-speech engine. An optional commercially available inexpensive depth-sensing camera can also identify the distance of the object from the user, if the object is within the FoV of the depth camera.

2 System Overview

2.1 Hardware Design

Fig.1 shows the evolution of three designs of our proposed system, all consisting of a mobile phone as well as a panoramic camera. Fig.1(a) shows our first design which is a handheld device with two handles. Upon informal survey of the visually impaired community, we migrated to our second design shown in Fig.1(b) which is a wearable backpack and hands-free. Specifically, we found the user community to have a strong preference for a hands-free versus a handheld system since their hands are already occupied by a cane. In the backpack system of Fig.1(b) the panoramic camera is connected to a rod which is then secured inside the backpack with additional hardware; in addition, a mobile phone and a depth-sensing camera is attached to the backpack in front of the user. Fig.1(c) shows our third design inspired by Project BLAID at Toyota[21], consisting of a ring that the user wears on his/her shoulder. Here, a depth camera and a mobile phone is attached to the front and the panoramic camera on a rod is attached to the back of the ring. In all three designs the panoramic camera is high above the user's head, enabling it to capture 360° images surrounding the user without much occlusion. While from a technical point of view, this approach results in non-occluded panoramic imagery and improves object detection, we readily acknowledge that from an aesthetic point of view, this might not be the best choice. In particular, we have received feedback from the visually impaired community regarding the stigma attached to wearable devices that draw attention to them. As such, in our latest design in Fig.1(c), we have included two mounting screws on the two shoulders to accommodate future designs with two panoramic cameras pointing sideways. This obviates the need for having a camera high above the users head, since the two panoramic cameras can capture the entire surroundings of the user, to the right, left, up and down.

As seen, the designs in Fig.1(b) and 1(c) include a front facing depth camera, which can relay the distance of the object to the user as long as the object is in

front of the object, i.e. within the FoV of the depth camera. Since there are no panoramic depth cameras currently on the market, to find distances to objects in all directions, our future design might need to have multiple depth cameras pointing at different directions. For the design shown in Fig.1(a) we have used a depth-sensing mobile phone, namely Oppo R17 Pro, rather than using a separate depth-sensing device. Clearly, in all three designs, it is possible to either use a depth-sensor enabled mobile phone or a regular mobile phone with an additional depth sensor. The advantage of the former over the latter is fewer components and therefore increased robustness to failure. For the experiments in this paper, we use the RGB-D depth-sensing camera Realsense D415 from Intel[9] , Android mobile phones such as the Google Pixel 2 or the Samsung Galaxy S8, and the panoramic camera Theta V from Ricoh[19]. The Bluetooth trigger is hung around the neck of the user for safekeeping and easy access purposes.

2.2 System Operation

We now describe the details of the operations of our proposed system: Upon a single click of the Bluetooth trigger, the user talks to the app to ask for a specific object of interest. We use the Kaldi speech recognition toolkit[18] for transcribing the user speech. At the same time the phone signals the panoramic camera via WiFi connection to capture a picture with resolution $5,376 \times 2,688$. The mobile phone then downloads, splits and recti-linearizes the image into 4 pictures of size $1,344 \times 2,688$ using OpenMVG[16], which corresponds to the front, right, left and back quadrants around the user respectively. The 4 images are then passed into our TensorFlow Lite model trained on our custom dataset to localize the object of interest. It then reports the results through a text-to-speech engine, where the objects are described in a clock coordinate system with the user's front corresponding to 12:00, right to 3:00, left to 9:00 etc. Fig.2 showcases the visual output of object detection in one of the recti-linear pictures created from a panoramic capture.

By clicking the Bluetooth trigger twice, the user again talks to the app to ask for a specific object of interest, but this time the depth detection pipeline will be invoked, capturing a recti-linear RGB image and its associated depth map. For the handheld device in Fig.1(a) we use the Oppo phone depth sensor, while for the wearable backpack and ring systems in Fig.1(b) and 1(c) we use the Intel Realsense D415 depth device. In both cases, there is a need to align the RGB image and depth map by mapping each point in the depth map to a point in the RGB image. This is achieved by using the intrinsic camera matrices for both the RGB camera and the depth camera, taking into account focal length and optical center in x and y. Fig.2(a) shows the ground truth and 2(b) the result of detected objects. The TensorFlow Lite model trained on RGB images detects the queried object in front of the user, calculates and reports back its distance through the text-to-speech engine. If the confidence level of the recognized object is too low, the panoramic camera is invoked to locate the desired object in the 360° surrounding of the user. In this case, if the detected object via the panoramic camera is found to be within the FoV of the depth camera with high confidence,

then the distance to the user will be reported via the text-to-speech engine on the phone.

Fig. 2. Detection example. (a)Ground-truth (b)Results from recognition model

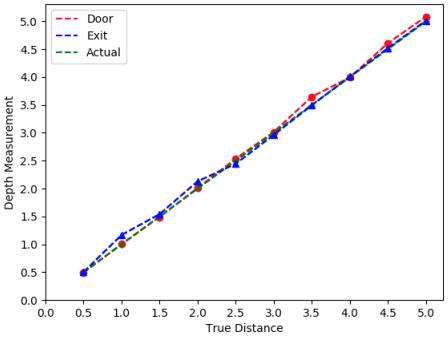


Fig. 3. Measurements of object depth vs. actual distance.

In order to calculate the distance of the object to the user, we average over the depth values near the center of the detected bounding box for the object. The size of the region is taken proportional to the size of the bounding box. The depth to RGB point mapping is stored in a two dimensional K-d tree[3] to provide fast look up of the points near the center of the detected object.

The accuracy of the depth detection is tested by taking measurements of 2 representative classes, namely doors and exits, at 0.5 meter steps from 0.5 meter to 5 meters with objects being fronto-parallel to the camera. The result is shown in Fig.3, indicating a close match between measured and ground truth distance.

3 Object Detection Model

3.1 Dataset Collection

We manually collect panoramic pictures using the setup in Fig.1(a), with the panoramic camera held above the head as to emulate the height of the final design. We then recti-linearize the panoramic pictures into 4 pictures of equal resolution for the front, right, back and left of the person respectively using OpenMVG[16]. We use LabelImg[22] to create the ground truth bounding boxes. Fast inference time is a crucial requirement of a real-time query system; thus we choose SSD with MobileNetV2[20] as the object detection model architecture in our system[8]. Specifically MobileNetV2 is used as the feature extractor and 6 additional SSD layers are used for bounding box regression and object classification. To compare the performance of the model against better feature

extractors, we also trained the model with ResNet-50 FPN[7, 12] as the backbone. The models are pretrained on the Microsoft COCO dataset[14] and then finetuned on our collected dataset.

3.2 Model Implementation

The training pipelines are adapted from [8], an implementation of SSD w/ MobileNetV2 and SSD w/ResNet-50 FPN in TensorFlow, and a machine setup of an Intel Core i7-6850K CPU @ 3.60GHz with one GeForce GTX 1080Ti is used in the training process. All images are resized to 400×800 pixels per the memory limit of the GPU. Twelve augmentations including contrast, brightness, hue changes and various crops are performed on the dataset in order to improve the robustness of the model. To overcome the imbalance of objects in the dataset, we used the Adam optimizer[10] together with focal loss[13] to train the model. After training, the model is exported to TensorFlow Lite, which allows us to store the trained model directly on the phone for faster detection.

Table 1. The confusion matrix with "same" and "different" tests of SSD w/MobileNetV2 with 12 augmentations. Nothing means that either an object of a class of interest is not detected, or an object detected is not in the ground-truth.

SSD w/MobileNetV2 @0.5 IOU ("same" test)		Prediction					
		exit	elevator	door	bathroom	plug	nothing
Ground Truth	exit	204	0	0	0	0	8
	elevator	1	95	3	5	0	5
	door	0	1	341	5	0	24
	bathroom	0	0	2	46	0	3
	plug	0	0	2	0	109	20
	nothing	18	29	57	13	19	0

SSD w/MobileNetV2 @0.5 IOU ("different" test)		Prediction					
		exit	elevator	door	bathroom	plug	nothing
Ground Truth	exit	64	0	2	0	0	2
	elevator	0	19	9	0	0	7
	door	0	10	193	13	0	8
	bathroom	0	4	8	12	0	12
	plug	0	0	1	0	2	0
	nothing	24	22	124	21	7	0

Table 2. Precision and recall for each class.

SSD w/MobileNetV2 @0.5 IOU ("same" test)		
Class	Precision	Recall
Exit	0.92	0.96
Elevator	0.76	0.87
Door	0.84	0.92
Bathroom	0.67	0.90
Plug	0.85	0.83

SSD w/MobileNetV2 @0.5 IOU ("different" test)		
Class	Precision	Recall
Exit	0.73	0.94
Elevator	0.35	0.542
Door	0.572	0.861
Bathroom	0.261	0.333
Plug	0.222	0.667

3.3 Experimental Results

We train the model with batch size of 8. The initial learning rate was chosen in the range of [0.001, 0.0008, 0.0004, 0.0002, 0.0001], with 0.0002 performing the best in the first 3000 steps. We use logarithmic decreasing learning rate and train for a total step number of 250K steps chosen empirically to perform best. We train the model on 518 pictures from Cory, Soda and Stanly Halls on U.C. Berkeley campus with a train/test split of 0.8/0.2. We refer to the results from this test as "same", indicating that training and test data were disjoint but from the same buildings. In addition, we test the model on 228 pictures taken from new buildings not in the training set, namely Dwinelle and Evans Halls. We refer to this test as "different" since the test data was from different buildings than the training data. The confusion matrix of the SSD w/MobileNetV2 model, along with precision and recall of each of the classes are also shown in Tables 1 and 2. As seen, for both tests the precision and recall values are best for exits, followed by doors. This is not surprising as they have the largest number of training examples. As expected, there is a drop in precision and recall from "same" to "different" test. We also trained the models with fewer augmentations, without random cropping as a comparison. The accuracy of all the models can be seen in Table 3. The accuracy drop from same to nfew test buildings is in agreement with [2], as it is difficult for the model to generalize to the objects of interest not in the training set. As seen, increased augmentation from 5 to 12 improves accuracy and generalization for new test buildings from 44.8% to 53.6%.

Table 3. Accuracy of different models measured by mAP @0.5 IOU

Models	Augmentations	"Same" Test	"Different" Test
SSD w/MobilNetV2	12	90.9	52.9
SSD w/MobilNetV2	5	99.6	51.7
SSD w/ResNet-50 FPN	12	86.2	53.6
SSD w/ResNet-50 FPN	5	99.8	44.8

From the results in Tabel 3 we observe that SSD w/ResNet-50 FPN performs marginally better than MobileNetV2 with 12 augmentations. However the latency of ResNet-50 FPN as the backbone to be higher than MobileNetV2[8]. The ResNet-50 FPN feature extractor is also worse at detecting small objects such as plugs. Thus we choose SSD w/MobileNetV2 as our model, allowing a very slight accuracy drop for faster and more usable detection.

Future work involves improving accuracy for new buildings, new hardware designs with less conspicuous cameras, and extensive user studies.

References

1. Bai, J., Liu, Z., Lin, Y., Li, Y., Lian, S., Liu, D.: Wearable travel aid for environment perception and navigation of visually impaired people. CoRR abs/1904.13037 (2019), http://arxiv.org/abs/1904.13037

2. Balamurugan, A., Zakhor, A.: Online learning for indoor asset detection. 2019 IEEE 29th International Workshop on Machine Learning for Signal Processing (MLSP) pp. 1–6 (2019)
3. Bentley, J.L.: Multidimensional binary search trees used for associative searching. Commun. ACM 18(9), 509–517 (Sep 1975). https://doi.org/10.1145/361002.361007, https://doi.org/10.1145/361002.361007
4. CloudSight: Taptapsee (2012)
5. Dai, J., Li, Y., He, K., Sun, J.: R-FCN: object detection via region-based fully convolutional networks. CoRR abs/1605.06409 (2016), http://arxiv.org/abs/1605.06409
6. Eyes, B.M.: Be my eyes (2015)
7. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. CoRR abs/1512.03385 (2015), http://arxiv.org/abs/1512.03385
8. Huang, J., Rathod, V., Sun, C., Zhu, M., Balan, A.K., Fathi, A., Fischer, I., Wojna, Z., Song, Y., Guadarrama, S., Murphy, K.: Speed/accuracy trade-offs for modern convolutional object detectors. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) pp. 3296–3297 (2016)
9. Intel: Realsense d415 (2018)
10. Kingma, D.P., Ba, J.: Adam: A Method for Stochastic Optimization. arXiv e-prints arXiv:1412.6980 (Dec 2014)
11. Kotyan, S., Kumar, N., Sahu, P.K., Udutalapally, V.: Drishtikon: An advanced navigational aid system for visually impaired people. CoRR abs/1904.10351 (2019), http://arxiv.org/abs/1904.10351
12. Lin, T., Dollár, P., Girshick, R.B., He, K., Hariharan, B., Belongie, S.J.: Feature pyramid networks for object detection. CoRR abs/1612.03144 (2016), http://arxiv.org/abs/1612.03144
13. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal Loss for Dense Object Detection. arXiv e-prints arXiv:1708.02002 (Aug 2017)
14. Lin, T., Maire, M., Belongie, S.J., Bourdev, L.D., Girshick, R.B., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft COCO: common objects in context. CoRR abs/1405.0312 (2014), http://arxiv.org/abs/1405.0312
15. Microsoft: Seeing ai (2017)
16. Moulon, P., Monasse, P., Perrot, R., Marlet, R.: Openmvg: Open multiple view geometry pp. 60–74 (2016)
17. Organization, W.H.: World report on vision. World Health Organization (2019)
18. Povey, D., Ghoshal, A., Boulianne, G., Burget, L., Glembek, O., Goel, N.K., Hannemann, M., Motlícek, P., Qian, Y., Schwarz, P., Silovský, J., Stemmer, G., Veselý, K.: The kaldi speech recognition toolkit (2011)
19. Ricoh: Theta v (2017)
20. Sandler, M., Howard, A.G., Zhu, M., Zhmoginov, A., Chen, L.: Inverted residuals and linear bottlenecks: Mobile networks for classification, detection and segmentation. CoRR abs/1801.04381 (2018), http://arxiv.org/abs/1801.04381
21. Toyota: Project blaid (2016)
22. Tzutalin: Labelimg (2015)
23. Yelamarthi, K., Haas, D., Nielsen, D., Mothersell, S.: Rfid and gps integrated navigation system for the visually impaired pp. 1149–1152 (Aug 2010). https://doi.org/10.1109/MWSCAS.2010.5548863
24. Yi, C., Flores, R., Chincha, R., Tian, Y.: Finding objects for assisting blind people. Network modeling and analysis in health informatics and bioinformatics 2, 71–79 (07 2013). https://doi.org/10.1007/s13721-013-0026-x