Demo Abstract: A Monocular Vision-based Obstacle Avoidance Android/Linux Middleware for the Visually Impaired

Xiaoyang Qiu*
qiux@kean.com
Dept of CS, Wenzhou-Kean University
Wenzhou, Zhejiang, China

Arjun Keerthi, Teppei Kotake
Aniruddha Gokhale
{arjun.b.keerthi,teppei.kotake,a.gokhale}@vanderbilt.edu
Dept of EECS, Vanderbilt University
Nashville, Tennessee, USA

Abstract

Increasing maturity in Fog/Edge computing has enabled many latency-sensitive Internet of Things (IoT) applications to achieve better performance and shorter response times. Our work on the URMILA middleware [1] proposed a performance and mobility-aware Fog/Edge resource management solution to support cognitive assistance to the visually impaired. However, URMILA was evaluated only in lab-based emulated scenarios. We overcome this limitation by presenting an affordable, unobtrusive and simple-to-use solution for the visually impaired. Alongside the long cane or the guide dog, our application aims to provide the visually impaired with a more detailed description of their environment. Using a single-camera on the Sony SmartEyeglass SED-E1 as the only sensor and an Android/Linux application, we were able to perform both per-pixel depth prediction and object detection on each image frame. By combining the information from these two sources, we provide users with a descriptive audio feedback assisting them in avoiding obstacles and thus better situational awareness. URMILA is used as before to manage the fog and edge resources in the system. We show the effectiveness of obstacle detection and recognition by creating both an outdoor and indoor scenario.

CCS Concepts • **Human-centered computing** → **Displays and imagers**; • **Computing methodologies** → *Object detection*; *Distributed algorithms*;

Keywords visually impaired, obstacle detection, monocular vision, real-time, assistive technology

*Work performed by first three authors as a summer undergraduate internship research team at Vanderbilt University during summer 2019

Middleware '19, December 9–13, 2019, Davis, CA, USA

ACM ISBN 978-1-4503-7042-4/19/12.
https://doi.org/10.1145/3366627.3368113

ACM Reference Format:
Xiaoyang Qiu, Arjun Keerthi, Teppei Kotake, and Aniruddha Gokhale. 2019. Demo Abstract: A Monocular Vision-based Obstacle Avoidance Android/Linux Middleware for the Visually Impaired. In *Middleware '19: International Middleware Conference Demos and Posters, December 9–13, 2019, Davis, CA, USA.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3366627.3368113

1 Problem Motivation

Mobility assistance, particularly obstacle detection and avoidance, offers the possibility for increased navigational confidence and awareness to the visually impaired. However, the complexity and accessibility of technical assistance remains one of the major hurdles to broader adoption. Although many assistive solutions for navigation or obstacle detection for the visually impaired exist [2, 3], these approaches do not address both the accessibility and affordability for those with special needs. Therefore, this research presents a solution to create an affordable, unobtrusive and simple-to-use obstacle avoidance solution for the visually impaired.

To that end we leverage the URMILA middleware [1] which is our performance and mobility-aware Fog/Edge resource management solution to support cognitive assistance to the visually impaired. The URMILA solution was, however, evaluated in lab-based emulated scenarios only. In this work, we have designed a real application to exploit URMILA. The proposed system is not a replacement for the long cane or guide dog; rather it provides situational awareness to the visually impaired. In the rest of this paper, we describe our solution and results from experimentation.

2 Technical Approach

We have used the Sony SmartEyeglass SED-E1 with a built-in camera as our sensor, which streams real-time images to a connected smartphone carried by the user. Our obstacle detection solution is more flexible and directional because of the orientation of the built-in camera in accordance with the head movement but it can also work with the built-in camera on the phone or any image source as input.

2.1 System Architecture

The architecture of our proposed system consists of five major components as shown in Figure 1. The Sony SmartEyeglass is connected to an edge device (an Android smartphone) using Bluetooth, where images captured by the eye glass are streamed to the edge device. URMILA manages dynamic adaptation of image processing between fog and edge nodes to reduce the latency and conserve edge resources. Both edge (Android) and fog (Linux) nodes can perform object detection by processing the image stream and identify obstacles.

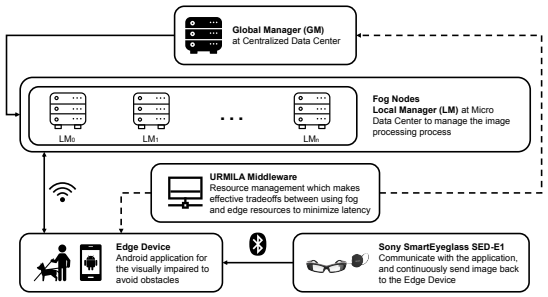

Figure 1. Proposed System Architecture.

The overall pipeline for this navigational system involves the phone sending images streamed from the Sony SmartEyeglass to the fog node, where depth prediction and object detection are run on the image. This information is sent back to the phone, where the beeps and messages are played to the user. When a server is not available, the application switches to only executing object detection on the phone.

2.2 Data Analysis and User Feedback

We used the image stream from a single camera. On this stream, we performed per-pixel ground-truth depth data prediction using Monodepth2 [4] and object detection using TensorFlow and PyTorch. To "translate" per-pixel depth data and object detection results into descriptive acoustical feedback, we needed to first find a suitable way to convert the scaled depth values (between 0 and 1) outputted from the depth prediction system into information that reasonably describes the environment. Because the depth prediction model was trained on a camera with a different focal length and image ratio from our setup, calculating ground-truth depth directly is unreliable. To address this issue, we divided the scaled depth image into a 3x4 grid. This was achieved by averaging the scaled depth values for the pixels in the corresponding section of the image. Subsequently, we can decide whether objects are close and should be notified to the user by finding the sum of values in different areas ("ahead", "left", "right") of the grid and classifying that area with a danger level. Finally, we inform the user of danger areas by reading the area direction once and sounding repeated beeps that increase in frequency as the danger level changes. Detected objects are also read to the user when the environment changes sufficiently. Details on the algorithm are not provided due to space limitations.

3 Implementation and Experimental Results

Our edge device was a Samsung Galaxy S7 (SM-G930U) with the Snapdragon 820 chipset. We modified the Monodepth2 software to fit the depth data prediction requirements of our application. An SSD Mobilenet v2 object detection implementation optimized for the NVIDIA Jetson Nano was used for the server-side computation [5]. The phone-based computation utilized TensorFlow Lite's Object Detection API. We tested using different devices for implementing the fog nodes including the NVIDIA Jetson Nano and NVIDIA GeForce GTX 960M. The average processing speed for each frame on different devices is listed below.

Table 1. Test Result under Different Scenarios.

Device		Average Frames Per Second (fps)	Average Processing Time Per Frame (s)
Edge Node	Samsung Galaxy S7	3.590	0.278
Fog Node	NVIDIA Jetson Nano	6.706	0.149
	NVIDIA GeForce GTX 960M	18.516	0.054

4 Conclusions

This paper describes an edge computing system comprising computer vision application that exploits a fog/edge resource management middleware to enable cognitive navigational assistance to the visually impaired.

Acknowledgments

This research was supported in part by NSF REU funds on US Ignite CNS 1531079, AFOSR DDDAS FA9550-13-1-0227 & VUSE SUGRE Program. All views presented are those of the authors and do not reflect the views of sponsors.

References

[1] Shashank Shekhar, Ajay Chhokra, Hongyang Sun, Aniruddha Gokhale, Abhishek Dubey, and Xenofon Koutsoukos. 2019. URMILA: A Performance and Mobility-Aware Fog/Edge Resource Management Middleware. In *2019 IEEE 22nd International Symposium on Real-Time Distributed Computing (ISORC)*. IEEE, 118–125.

[2] Ayat A Nada, Mahmoud A Fakhr, and Ahmed F Seddik. 2015. Assistive infrared sensor based smart stick for blind people. In *2015 Science and Information Conference (SAI)*. IEEE, 1149–1154.

[3] Florbela Pereira, João C Ponte-e Sousa, Rui PS Fartaria, Vasco DB Bonifácio, Paulina Mata, Joao Aires-de Sousa, and Ana M Lobo. 2013. Sonified infrared spectra and their interpretation by blind and visually impaired students. *Journal of Chemical Education* 90, 8 (2013), 1028–1031.

[4] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel Brostow. 2018. Digging into self-supervised monocular depth estimation. *arXiv preprint arXiv:1806.01260* (2018).

[5] AastaNV. 2019. TensorRT Python Sample for Object Detection. https://github.com/AastaNV/TRT_object_detection