2019

Ambient awareness on a sidewalk for visually impaired

Faruk Ahmed

AMBIENT AWARENESS ON A SIDEWALK FOR VISUALLY IMPAIRED

by

Mohammad Faruk Ahmed

A Dissertation

Submitted in Partial Fulfillment of the

Requirements for the Degree of

Doctor of Philosophy

Major: Engineering

The University of Memphis

December 2019

Abstract

Safe navigation by avoiding obstacles is vital for visually impaired while walking on a sidewalk. There are both static and dynamic obstacles to avoid. Detection, monitoring, and estimating the threat posed by obstacles remain challenging. Also, it is imperative that the design of the system must be energy efficient and low cost. An additional challenge in designing an interactive system capable of providing useful feedback is to minimize users' cognitive load. We started the development of the prototype system through classifying obstacles and providing feedback. To overcome the limitations of the classification-based system, we adopted the image annotation framework in describing the scene, which may or may not include the obstacles. Both solutions partially solved the safe navigation but were found to be ineffective in providing meaningful feedback and issues with the diurnal cycle. To address such limitations, we introduce the notion of free-path and threat level imposed by the static or dynamic obstacles. This solution reduced the overhead of obstacle detection and helped in designing meaningful feedback. Affording users a natural conversation through an interactive dialog enabled interface was found to promote safer navigation. In this dissertation, we modeled the free-path and threat level using a reinforcement learning (RL) framework. We built the RL model in the Gazebo robot simulation environment and implanted that in a handheld device. A natural conversation model was created using data collected through a Wizard of OZ approach. The RL model and conversational agent model together resulted in the handheld assistive device called Augmented Guiding Torch (AGT). The AGT provides improved mobility over white cane by providing ambient awareness through natural conversation. It can inform the visually impaired about the obstacles which are helpful to be warned about ahead of time, e.g., construction site, scooter, crowd, car, bike, or big hole. Using the RL framework, the robot avoided over 95% obstacles. The visually impaired avoided over 85% obstacles with the help of AGT on a 500 feet U-shape sidewalk. Findings of this dissertation support the effectiveness of augmented guiding through RL for navigation and obstacle avoidance of visually impaired users.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

According to the estimates from the World Health Organization (WHO), about 285 million people are visually impaired worldwide: 39 million are blind, and 246 million have low vision (Bourne et al., 2017; Fricke et al., 2018). 2.3% people have a visual disability in the USA according to the Disability Status Report (Erickson, Lee, & von Schrader, 2008). Also, according to traffic safety facts data, there were $4,735$ pedestrians killed in 2013 in USA (Administration et al., 2017). Hence, ambient awareness on the sidewalk is critical for safe navigation for the people who are blind or visually impaired. Ambient awareness may include (but are not limited to) static and dynamic obstacles including potholes, debris, ongoing construction, a person riding a bike, or person with a pet. Fig. 1.1 depicts a typical scenario of ambient awareness on a sidewalk. Despite the needs and advances in assistive technologies, designing a functional and easy to use system remains a challenge.

Avoiding transient (e.g., puddle, a person with a bike) and static (e.g., electric pole, tree) obstacles on a sidewalk is challenging for visually impaired. There are numerous electronic travel aids (ETAs) to help blind with obstacle detection, obstacle avoidance, and navigation. Blind people often use a cane, smartphone apps, or other ETA devices while walking. These technologies provide some information for safe navigation but not successful in many cases. For example, devices with global positioning system (GPS) provide direction only but unable to notify about a transient obstacle, e.g., potholes, puddles, or traffic cone. The visually impaired may use guide cane to become aware of the path and obstacle by continuously striking

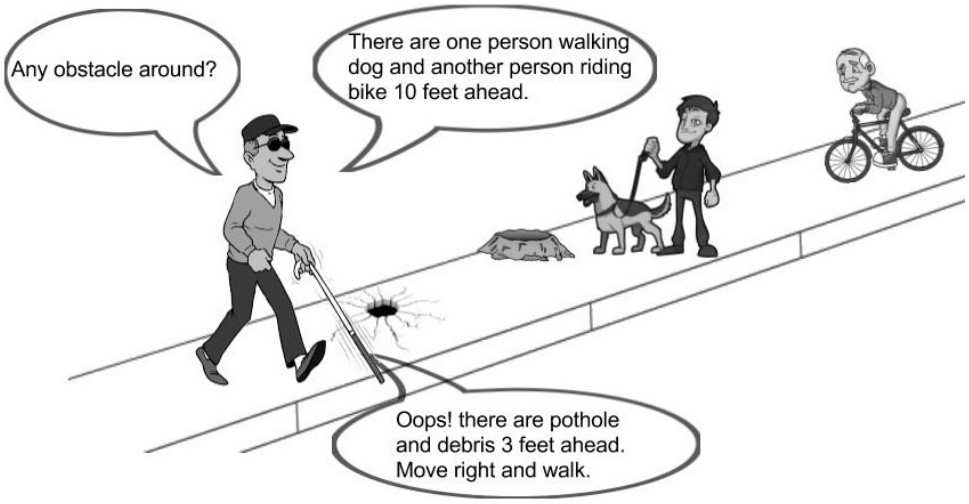


Figure 1.1: The blind person walking on the sidewalk with AGT.

on the surface. Slippery surfaces, as well as slope, are not detected by the guide cane. Some mobile apps provide turn-by-turn instructions but do not serve for ambient awareness. A few applications detect obstacles in the indoor and outdoor environment but do not identify sidewalk obstacles, e.g., a person with bike or crowd (Aladren, López-Nicolás, Puig, & Guerrero, 2016). Most of the reported applications are not interactive. Those applications are limited to single-mode operations (e.g., audio feedback or haptics). Besides, the design of feedback mechanism suffers from insufficient personalization and reconfigurability. Poor design of feedback mechanism induces cognitive load. (Ahmed, Mahmud, Al-Fahad, Alam, & Yeasin, 2018) proposed image captioning to reduce the cognitive load for the application of sidewalk ambient awareness.

In the continuum of developing assistive solutions since 2010 (e.g., Computer Vision and Perception Analysis (CVPIA) lab), we have built a number of systems, and some are prototype system and some are practically working; some are in the phase of going to production (see Figure 1.2). Some are for printed text recognition, some are for non-verbal communication, but all these systems are built over the last 10 years, we have learned number of strategies to design and build a system. For example, when we design the assistive system –

- We work **with** the people not only **for** the people.
- To work with the people, we start with a participatory design with the users, to get the

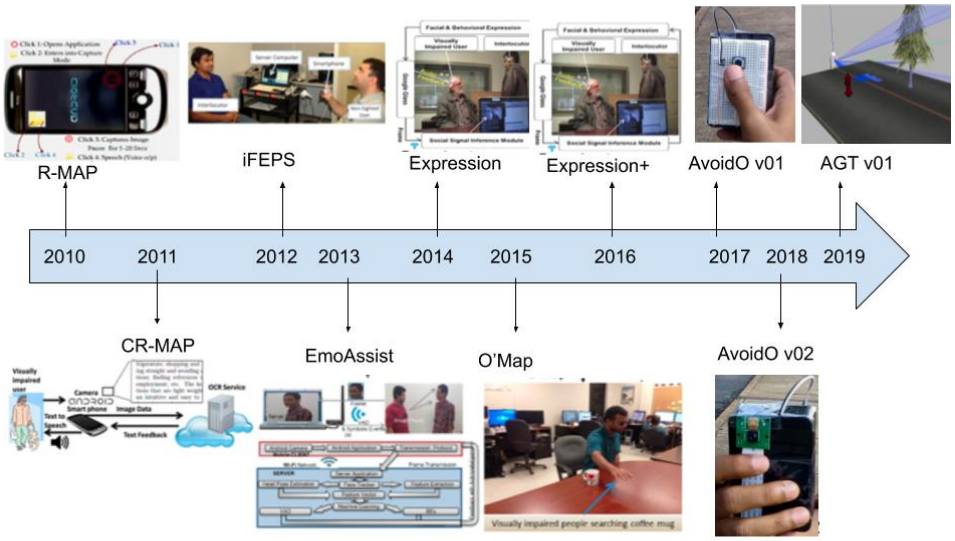


Figure 1.2: The trend of CVPIA lab's assistive technology solution development.

functional requirements of the system.

- We take into account the "design thinking aspect" of it. That is how to design the feedback, what would be the shape of the device, what is the energy consumption, what is the look-and-feel. Because people want to project a positive image to others. So, the design thinking aspect matters, where to position the system, is it going to be hand-held or body mounted.

- We consider the "system thinking aspect" of the device to make it energy efficient, portable, and low cost. When we started developing assistive solutions, the cost was over $2000 (e.g., using Google Glass). In spite of the existence of the expensive technology solutions, visually impaired may not afford those. Our goal is to bring the cost down to the affordable limit. That is why we focus on using the low form factor machine (e.g., $35 computer Raspberry Pi 3).

- We design interface with minimal cognitive load. If the interface imposes too much cognitive load, or the user does not have any clue of how to use it, the system will be

useless.

- We design a system that can respond in real time under dynamic conditions. User requires responses from the system within the time limit of mean walking speed (e.g., 1.4 meters per second (m/s)). Response time has to be reduced to meet safety requirements. There is a trade-off between the response time, safety, as well as the affordability.

While walking on a sidewalk, a visually impaired person needs adequate information to create a mental map of the environment. A description of a visual scene could provide that required information. The description should be meaningful (e.g., "a person with a bike is coming towards you") of a visual scene that is useful for assistive solutions. An incomplete description may lead to poor perception and mis-representation of dangers that may lie ahead. Incomplete or partial characterization of a scene such as "person with bike" can be irrelevant or meaningless feedback to a person who is blind. The first sentence is an example of a caption or description of an image, whereas the second sentence is the example of a class label. However it is accomplished, informing a visually impaired with a vivid description of an obstacle can save them from imminent danger and increase their confidence and independent mobility. Recently Deep Neural Networks (DNNs) solved many challenging problems efficiently. Researchers from both the academia and industries have been using the power of DNNs for speech recognition (Abdel-Hamid et al., 2014), text categorization (T. Wang, Wu, Coates, & Ng, 2012) and image recognition (Deng et al., 2009a), just to name a few. Convolutional Neural Network (CNN) based deep learning architectures are state-of-the-art for visual recognition tasks. Reinforcement Learning (RL) is able to beat human playing game (Bellemare, Naddaf, Veness, & Bowling, 2013; Mnih et al., 2015). Recurrent Neural Network (RNN) is generating sentences and recognizing spoken language (Mikolov, Karafiát, Burget, Černockỳ, & Khudanpur, 2010; Graves, Mohamed, & Hinton, 2013). In this dissertation, we attempt to bridge the gap between DNN and assistive technology solutions. DNN based assistive technology solutions can be useful to augment the perceptual capability of the visually impaired, and this can be an aid for them. We name it "Ambient Awareness on a Sidewalk".
There are numerous challenges to design the ambient awareness on a sidewalk application:

- Building robust model for the environment which is time efficient (image classification,

image annotation, reinforcement learning) and deploying this model on a low-cost computing platform (RPi).

- Building an energy efficient system.
- Efficient feedback design which is very important for the visually impaired for safe navigation.
- System evaluation challenge.

This research work started with participatory design to find the functional requirements of ambient awareness for a sidewalk application. From participatory design, we have identified five different obstacle classes and constructed a unique dataset. We were able to study the performance of CNN on this dataset. Though the classification performance was above 80%, it did not meet the requirements of the visually impaired. Moreover, the classifier failed for multiple obstacles and diurnal effect[1]. Then we evolved to image annotation expecting that the machine would generate a vivid description of the image that met the need of the visually impaired to avoid obstacles. Even though we obtained a partial solution for multiple obstacle avoidance from image annotation, still it did not meet user expectations. That is why, to provide a better feedback, and to avoid all the complexities of object recognition we had reanalyzed the problem and redefined it. Instead of modeling the obstacles, we modeled the free path in combination with the conversational agent. This novel approach provides better intuitive solution of the ambient awareness on a sidewalk problem. We present the details of this evolution in next few chapters.

[1] A diurnal cycle is any pattern that recurs every 24 hours as a result of one full rotation of the Earth around its own axis (source: Wikipedia). Changes of shadow is an effect of diurnal cycle.

Chapter 2

Literature Review

Assistive technology solutions for the visually impaired drew the attention of researchers as a prominent research area in the mid-90s. Researchers have conducted studies and developed applications to improve the mobility of visually impaired. Generally, two types of applications are available for visually impaired, *a) standalone device* and *b) mobile apps*. Classical computer vision and DNN are primary technologies for image-based assistive solutions.

2.1 Standalone devices

Many research projects have focused on indoor path navigation and avoiding obstacles for the visually impaired. Among the standalone devices Drishti (Helal, Moore, & Ramachandran, 2001) and GuideCane (Ulrich & Borenstein, 2001) used GIS information hosted on a central server. They continuously queried the server for GPS information to facilitate navigation. GuideCane used an ultrasonic sensor and embedded computer to detect obstacles, but the field of view of the sensor was very narrow. To circumvent the problem, Shoval, Ulrich, and Borenstein (2003) proposed an array of ultrasonic sensors mounted on a belt (Shoval, Ulrich, & Borenstein, 2003). However, the belt became too bulky, along with being power and resource hungry. GuideCane therefore, along with other smart cane project, focused on obstacles that are of head-height to make it lighter (Wu et al., 2008; Singh et al., 2010; Wahab et al., 2011). There is a talking navigation cane that allows voice command and provide navigation information via audible messages and haptic feedback (Jesie, 2015). They used the GPS

to accomplish the localization of the user. With the revolution of the smartphone, research focus shifted towards developing the vision-based systems as well as assistive apps.

Bradley and Dunlop (2005) investigated the difference between sighted and visually impaired peoples' mental and physical demand of following verbal instructions of direction (Bradley & Dunlop, 2005). They used the "Wizard of OZ" techniques, in which participants were given pre-recorded verbal directions via a Minidisk to navigate to landmarks and the researcher controlled the timing of verbal messages. We adopted the same method in this dissertation to construct conversation data for agent training. Probabilistic inertial-visual odometry (PIVO) was developed for an occlusion-robust navigation system (Solin, Cortes, Rahtu, & Kannala, 2017). In this work, the Inertial Measurement Unit (IMU) sensors and the monocular camera information are fused to construct odometry. The application is robust even if the camera is covered for an extended amount of time. However, this is not usable by the visually impaired people because the camera and the IMU sensors have to be at a specific orientation. Travi-Navi is another navigation system based on vision-guidance. It records high-quality images and sensor readings during a guider's walk. The reading is compressed into a navigation trace. This trace is used by another navigator to navigate safely (Zheng et al., 2017). This navigation system works only indoors because outdoors are dynamic and image-based navigation trace is not suitable. A beacon-based navigation system is more accurate and provides far better navigation help. But the deployment of several beacons is an expensive task and needs an expert for the implementation (Ahmetovic et al., 2017).

Recently WiFi-based positioning has drawn attention (C. Yang & Shao, 2015; H.-H. Liu & Yang, 2011). This type of positioning and navigation system determines the approximate position of cellular devices by using radio frequency (RF) signals and a triangulation mechanism. It depends on the signal strength and phase, signal transmission time and angle of arrival along with channel state information. Indoor environments are complicated because of multiple access point transmission. Signals are affected by the adjacent and co-channel interference (Mahmud & Uddin, 2018). That is why this method is less reliable both in indoors and outdoors. One system using this approach is ppNav, a mobile app which helps navigate based on previous navigator's trace. It constructs trace from ubiquitous WiFi signal along

with visual features (Yin, Wu, Yang, & Liu, 2017).
Chen reported a mobile robot navigation algorithm which fuses the odometry and compass data. They used an extended Kalman filter algorithm for the fusion (Chen & Zhang, 2017).

2.2 Mobile apps

Liu, Wu, Tseng, and Tsai (2015) developed an app to facilitate daily activities like reading the text, voice activated dialing, and walking using distance and directions feedback (K.-C. Liu, Wu, Tseng, & Tsai, 2015). Zhong, Garrigues, and Bigham (2013) presented an app that performs a real-time scanning of objects by using key frame extraction from the video (Zhong, Garrigues, & Bigham, 2013). It sends those frames to cloud-based recognition engine for identification. The purpose of this application is to help visually impaired take a good picture. There are some apps for specific assistive task. For example, a visual search can be performed by using Zensors (Laput et al., 2015) and VizWiz (Bigham et al., 2010). SeeClickFix is an app to report non-emergency issues to the city government (*SeeClickFix*, n.d.).
Li, Shu, Karlsson, Lin, and Moscibroda (2017) at Microsoft Research started the scalable indoor navigation system "FollowUs", an easily deployable application (Li, Shu, Karlsson, Lin, & Moscibroda, 2017). This was further developed into "Path Guide"(Shu & Karlsson, 2017). This app works on peer-to-peer or leader/follower model. A person goes from one point to another point and records the trace. This trace is shareable with others.

2.3 Computer vision applications

Rao, Prasad, Shetty, Hegde, and Bhakthavathsalam (2012) used video and frame by frame processing to detect a pre-modeled obstacle and provided three different pitches of sound to avoid obstacles (Rao et al., 2012). On the other hand, Aladren et al. (2016) used RGB-D sensor and computer vision technique to segment the floor in an indoor environment to find any barriers (Aladren et al., 2016). However, this system is cumbersome and does not address the issues in a dynamic outdoor environment (e.g., sidewalk). Leung and Medioni (2014) developed an application which determines the egomotion in the highly dynamic outdoor environ-

ment (Leung & Medioni, 2014). The purpose of this application is to predict visual odometry to help blind people navigate, based on a predefined map; it does not provide ambient awareness. However, none of these applications is capable of catering to ambient awareness.

2.4 Deep learning applicatios

Google Goggles has been used to "search" based on pictures taken by handheld devices. The Orcam MyEye (Shashua, 2016) is efficient in reading texts, road-signs, and traffic signals. "Clarifai" developed the image recognition engine whose underlying technique is Deep Convolutional Neural Networks (DCNN). Beside those commercial apps Szegedy, Toshev, and Erhan (2013) used DNN to detect and localize objects of various classes including bicycle, dog, person, car, and bus (Szegedy, Toshev, & Erhan, 2013).
SqueezeNet achieves AlexNet-level accuracy on ImageNet with fifty times fewer parameters (Iandola et al., 2016). Another important network announced by Google is the MobileNet (Howard et al., 2017). This network is built to achieve a balanced trade-off between accuracy and resources available in a mobile hand-held device. DNN require huge computing resource to train and test as well as in production. That is why optimizing DNN became an important branch of the research. Research work related to optimizing the performance of DNN using the deep compression network (Denton, Zaremba, Bruna, LeCun, & Fergus, 2014) is mentionable. Various methods based on vector quantization (Gong, Liu, Yang, & Bourdev, 2014), hashing techniques (Chen, Wilson, Tyree, Weinberger, & Chen, 2015), circulant projection (Cheng et al., 2015), and tensor train decomposition (Novikov, Podoprikhin, Osokin, & Vetrov, 2015) were reported with better compression capability.

2.5 Other related studies

Shinohara, Bennett, and Wobbrock (2016) performed a study to investigate the designers regard disability and accessible design thinking for the disabled and non-disabled population (Shinohara, Bennett, & Wobbrock, 2016). According to them, designing for both surface challenges and tensions lead to better accessible design. Kawas, Karalis, Wen, and Ladner

(2016) performed a qualitative study to understand real-time captioning experiences of deaf and hard of hearing (DHH) students in a classroom setup (Kawas, Karalis, Wen, & Ladner, 2016). They discovered that the accuracy and reliability of the technology are still the most important issue of current captioning solutions. Wilson and Brewster (2015) presented a study suggesting the accuracy of peripersonal reaching can be improved by the use of dynamic sound from both the objects to reach for and the reaching hand itself (via a word speaker) that changes based on the proximity of the hand to the object (Wilson & Brewster, 2015). Part of this research is useful for the ambient awareness application on a sidewalk because if an assistive technology solution produces dynamic sound, it will help the blind person to draw a mental map of the ambient environment. Kane, Jayant, Wobbrock, and Ladner (2009) interviewed 20 participants with visual and motor disabilities and asked about their current use of mobile devices, including how they select them, how they use them while away from home, and how they adapt to accessibility challenges when on the go (Kane, Jayant, Wobbrock, & Ladner, 2009). They showed that people with visual and motor disabilities use a variety of strategies to adapt inaccessible mobile devices and use them to perform everyday tasks and navigate. The assistive solution with an accessible design will help visually impaired improve their daily life, and they will be able to carry out everyday tasks with ease.

Chapter 3

Methods and Materials

In the process of assistive technology development, we used convolutional neural network, recurrent neural network, reinforcement learning, and conversational agent. In this chapter, we describe these components briefly.

3.1 Machine Learning

Machine learning is improving rapidly due to the availability of massive data. In this dissertation, we used a convolutional neural network, recurrent neural network, and reinforcement learning from machine learning.

3.1.1 Convolutional Neural Network

The Convolutional Neural Network (CNN) is one of the prominent categories of neural network for image classification. It takes an image as input and classifies to a particular label (e.g., human, cat). The network sees the image as an array of pixels. The pixels are arranged in Height, Width, and Depth. For example, an RGB image has a certain height, width, and depth of 3 (e.g., Red, Green, and Blue) and the gray image has a certain height, width with a depth of 1.

The CNN model takes input images and passes through a series of convolution layers with filters (kernels), pooling, fully connected layers, and a softmax layer. The output of the softmax

layer is a probabilistic value which maps to a label of the image (see Figure 3.1).

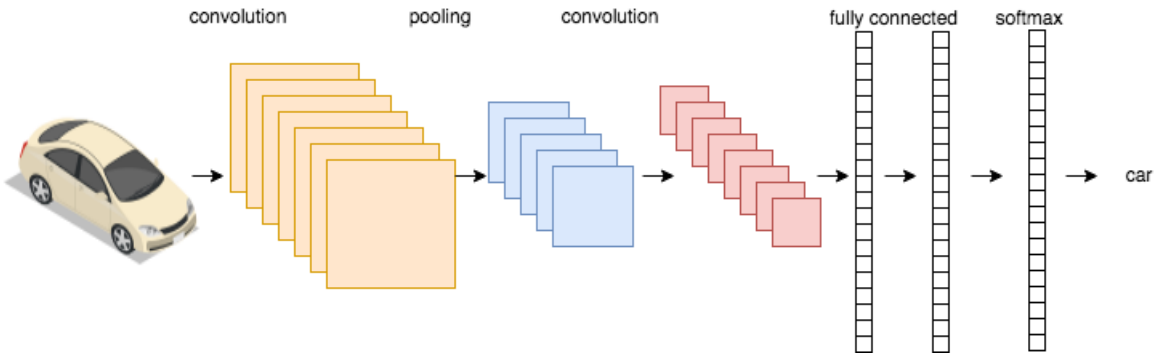


Figure 3.1: Convolutional neural network.

Convolution Layer

A convolutional neural network consists of convolutional layers. In convolution layers, the convolution is performed between the input and the filters. The convolution is essentially a dot product between the entries of a filter and the input. The filters are a small matrix in height and width but increases in depth after the convolution with input volume. As an example, a typical filter on the first layer of a network might have size 11x11x3, which is 11 pixels wide, 11 pixels high, and depth 3. If the number of filters in the layer is 96, then the output of this layer will have 55x55x96 for an input image of size 224x224x3 with a 4-pixel stride. ”Forward pass” means passing an image as input to the layers of convolution, and ”backpropagation” means propagating the gradients towards the input layer, which updates the weights of the filters along the way. The convolution produces a 2-dimensional activation map that gives the responses of that filter. In other words, the network learns filters that activate at particular visual features.

Stride

The Stride is the number of pixels each time the filter shifts over the input. Stride moves 1 pixel at a time when Stride is 1, and it moves 2 pixels at a time when Stride is 2. Sometimes the filter does not fit appropriately with the input image. To forcefully fit the filters sometimes the image is padded with zero (zero-padding) or a portion of the image is dropped.

Pooling

Pooling layers reduces the number of parameters. Pooling layer plays a vital role to make the neural network deep and to learn features with a reasonable number of parameters. Sub-sampling or down-sampling is another name for spatial pooling. There are max-pooling, average-pooling, and sum-pooling, which takes the maximum, average, or summation of the values from filter output (see Figure 3.2).

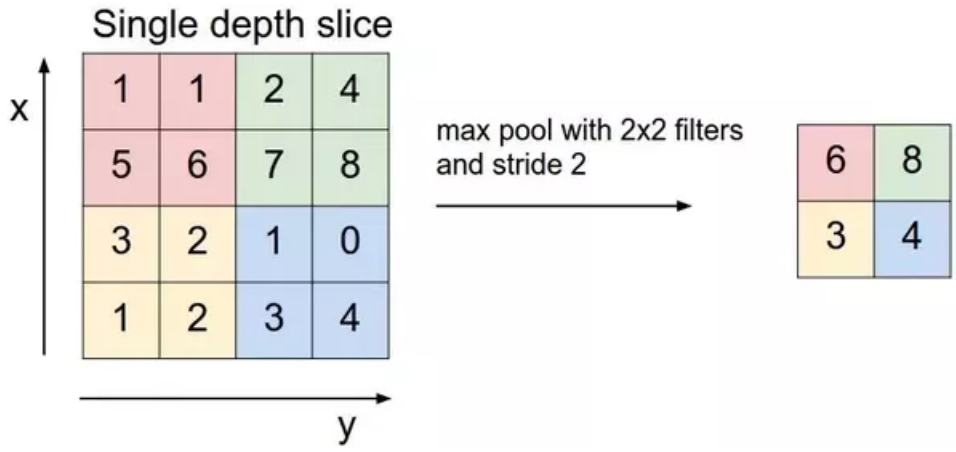


Figure 3.2: Max pooling, Source: Stanford's CS231n GitHub.

Nonlinearity

Neural networks add non-linearity through a non-linear operation. The Rectified Linear Unit (ReLU) is very popular operation among researchers. It is defined as $f(x) = max(0,x)$. There are other non-linear functions such as $tanh$ or $sigmoid$.

Activation

Usually, the final layer of a deep network is a the fully connected layer. In this layer, the feature matrix is flattened into a vector and fed into the activation function. $softmax$ or $sigmoid$ is the example of activation function.

3.1.2 Recurrent Neural Networks (RNNs)

Other than RNNs the assumption for neural network is that the inputs and outputs are independent of each other. However, many tasks especially sequential fail for this assumption because the output depends on the previous computation. That is why RNNs are discovered to utilize the sequential information efficiently and effectively. For example, if we want to predict a word in a sentence, we must know the words already appeared and the sequence of appearance.

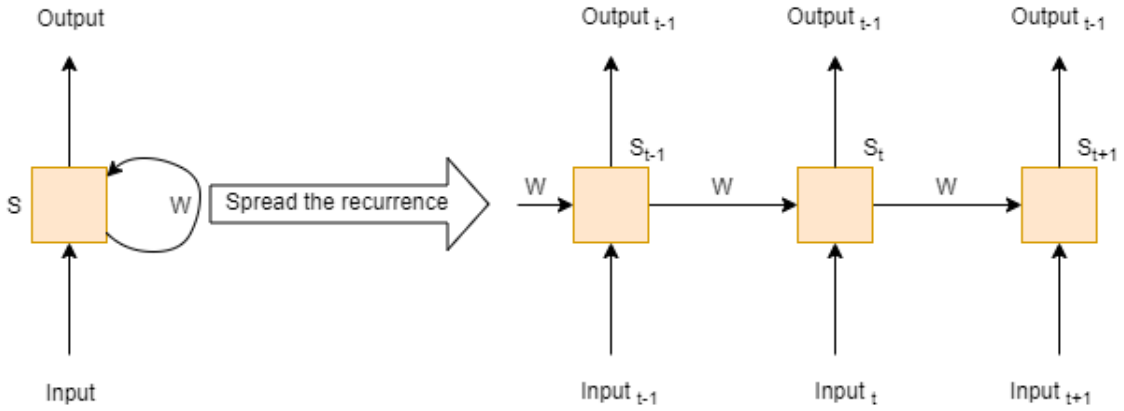


Figure 3.3: An RNN and the spread out recurrent steps in time.

A typical RNN looks like the Figure 3.3. The right hand side is a picture of stretched out of steps which RNNs take. It does the same operation for every input of a sequence and that is why it is called recurrent (Britz, 2015). The spreading out means that the steps are written out network for the complete sequence. For example, if we have a sequence of 10 words, the network would be spread out to a 10-layer neural network, one layer for each word.

- $Input_t$ is the input at time step t. It could be a one-hot vector of a word of a sentence.
- s_t is the hidden state at time step t. This is called "memory" of the network. s_t is calculated based on the previous hidden state and the input at the current step.
- $Output_t$ is the output at step t. For example, if we wanted to predict the next word in a sentence it would be a vector of probabilities across the list of words.

The memory of the network s_t contains information about all the previous time steps. The output at step $Output_t$ is calculated solely based on the memory at time t. In practice, s_t can

not capture information from too many time steps back. The RNN shares the same parameters across all steps with different inputs that is the reflection of performing the same task at each step. Sharing the same parameters reduces the total number of parameters to learn. The Figure 3.3 has output at each time step, but taking output depends on the nature of the task. For example, the output at the final step is sufficient to predict the sentiment of a sentence. RNNs have shown success in language modeling, generating text, machine translation, speech recognition, and generating image descriptions. The main problem of RNNs is the vanishing gradient (Bengio, Simard, Frasconi, et al., 1994) which is solved by LSTM (Hochreiter & Schmidhuber, 1997) and LSTM is the most widely used RNN. It is efficient at capturing long-term dependencies than vanilla RNN. LSTM has the same concept as the RNN with a different way of computing the hidden state.

3.1.3 Reinforcement Learning (RL)

RL is more or less about an agent that interacting with the environment and learning to take actions. There are three aspects a problem should have, to become a reinforcement learning problem. Those are

- Different actions yield different rewards
- Reward for an action is conditional on the state of the environment
- Rewards are delayed over time

Here, we present a simple tutorial of exploring Q-Learning algorithms (Juliani, 2016). For that, we will take the FrozenLake environment of OpenAI gym (Brockman et al., 2016). The OpenAI gym provides many different ready environments to explore RL algorithms.

Table 3.1: The configuration of a frozen lake defined by openai gym.

SFFF	S: starting point
FHFH	F: frozen and safe
FFFH	H: hole, dangerous
HFFG	G: goal, target

The FrozenLake environment is made up of a 4x4 grid of blocks. S is the start block, F is frozen block, H is the hole which is a dangerous block, and G is the goal block. This tutorial should train an agent to navigate from the start to the target block without falling into a hole. At any given time, the agent chooses to move either up, down, left, or right. There is an uncertainty that the agent ends up in a block which it did not choose, because of slippery surface or the wind speed. Taking the correct step every time is not possible, but learning to avoid the holes and reach the goal is possible - every step the agent acquires 0 but reaching destination incur 1. Thus, the Q-learning algorithm learns expected long-term rewards.
The most straightforward implementation of Q-Learning is a table of possible values for every state (row) and action (column). For our custom FrozenLake, we have sixteen possible states and four possible actions (left, up, right, down). The states and actions require $16x4$ table of Q-values. The table initially filled with uniform zero, and then the agent observes the reward for the actions taken. Each time the agent updates the Q-value in the table by following the Bellman equation 3.1.

$$Q(s,a) = r + \gamma(max(Q(s',a'))) \tag{3.1}$$

In the equation 3.1 s is state, a is action, r is reward, and γ is discounted future reward. By updating over time, the table starts to guess the correct measures of the expected future reward for a given action in a given state. The Q-learning and the SARSA algorithms are shown in algorithm 1 and algorithm 2 (Sutton & Barto, 1998).

Algorithm 1: Q-learning algorithm

```
1 Initialize Q(s,a) arbitrarily
2 foreach episode do
3     Initialize s
4     foreach step of episode until s is terminal do
5         Choose a from s using policy derived from Q (e.g., ε-greedy)
6         Take action a, observe r, s'
7         Q(s,a) = Q(s,a) + α[r + γ max_α(Q(s',a')) − Q(s,a)]
8         s = s'
```

In the algorithm 1 α is the learning rate, set between 0 and 1. Setting it to 0 means that there is no learning (e.g., Q-values are never updated). Setting a higher value close to 1 means the

learning will occur quickly. γ is a discount factor, and it ranges from 0 to 1. It interprets that future rewards are less worth than immediate rewards. max_α is the maximum reward that is obtainable in the state following the current state.

Algorithm 2: SARSA learning algorithm

```
Initialize Q(s,a) arbitrarily
foreach episode do
    Initialize s
    Choose a from s using policy derived from Q (e.g., ε-greedy)
    foreach step of episode until s is terminal do
        Take action a, observe r, s'
        Choose a' from s using policy derived from Q (e.g., ε-greedy)
        Q(s,a) = Q(s,a) + α[r + γmax_α(Q(s',a')) − Q(s,a)]
        s = s' and a = a'
```

A sample learned Q-table using lookup-table approach is shown in table 3.2. Note that the rows $6, 8, 12, 13$ contains 0. In the sample program the C style row numbers are used i.e., row 6 is actually state 5, row 8 is state 7 and so on. Because those rows represent state of holes and the agent gets punished by being into those states, thus it learned that those holes are dangerous to move in. The final state 16 also contains 0 because the agent reaches the goal.

Table 3.2: Sample Q values learned by the agent to travel frozen-lake.

	left move	top move	right move	bottom move
1	5.95873773e-03	3.24337047e-03	2.41046373e-02	3.61873573e-03
2	5.62644971e-04	8.11715890e-04	5.92301797e-04	1.68971343e-01
3	8.78774934e-04	4.76184814e-02	1.17030650e-03	3.08992350e-03
4	6.74146328e-04	5.04143648e-04	7.33285360e-04	3.28750708e-02
5	6.54950007e-02	1.54778278e-03	4.29140128e-04	8.50202331e-04
6	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
7	9.34385891e-06	5.96117359e-04	9.89125224e-03	1.35366473e-05
8	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
9	3.19627637e-04	2.09251364e-04	1.39997535e-04	2.20680987e-01
10	3.58018940e-04	5.92961422e-01	1.31089143e-04	6.45863380e-04
11	8.22868609e-01	9.18343158e-05	4.09722043e-04	0.00000000e+00
12	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
13	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
14	4.97094456e-07	1.20964024e-03	2.78848966e-01	5.83403326e-05
15	0.00000000e+00	0.00000000e+00	9.70693218e-01	0.00000000e+00
16	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00

3.2 Conversational Agent

A conversational agent is a software system that enables a user to communicate with it using natural language. At the beginning of the conversational agent, it used speech to text conversion to perceive natural conversation. However, nowadays, the agent uses body movements, facial expressions along with language features to understand the conversation. Naturally spoken sentences (e.g., ”What is .....”, ”Where is ....”, ”Book a ticket”) are used to communicate with the agent, and the agent replies mostly with relevant answers or relevant tasks.

3.2.1 Components

Conversational agents are made for special purposes. But, most of the conversational agents share some common components such as:

- Input
- Language parser
- Classifier (e.g., intent)
  - String matching and template-based answer
  - Case Based or Rule-based reasoning
  - Batch Learning from the set of conversation
- Output (The agent’s response)

The above are the minimum components of a conversational agent (see fig 3.4). There are commercial and open-source conversational agent frameworks available, namely Microsoft Bot Framework (Washington, 2016), Dialogflow (Di Fabbrizio et al., 2011), IBM Watson (High, 2012), and RASA Stack (Bocklisch, Faulkner, Pawlowski, & Nichol, 2017). We used the RASA stack because it is open source and easy to customize.

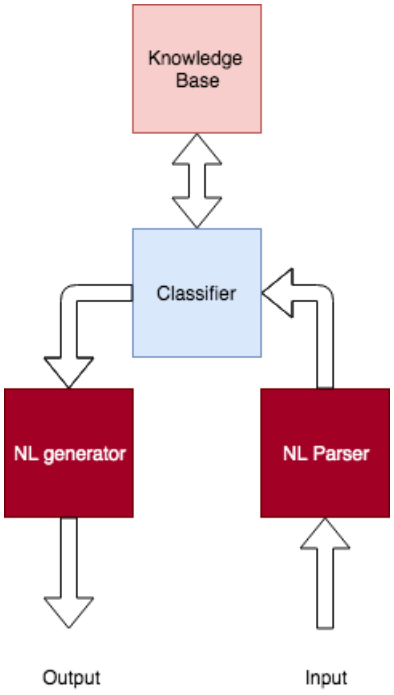


Figure 3.4: Building block of conversational agent.

3.3 Robot OS and Gazebo

The Robot Operating System (ROS) is a framework developed by the Stanford AI Laboratory in 2007 for developing robots and maintained by Open Source Robotics Foundation. The concept of ROS is more than just a framework. It provides hardware abstraction, low-level device control, implementation of commonly-used functionality, message-passing between processes, and package management. A single-board computer such as Raspberry Pi is sufficient to install and test ROS.

In general, ROS consists of code and tools that help running the robot code. It is a loosely coupled system where a process is called a node, and every node should be responsible for one task. Nodes communicate with each other using messages passing via logical channels called topics. Each node can send or receive data from the other nodes using the publish/subscribe model. Looking deeper, ROS is not an OS but OS like framework. Three main components are there in ROS ecosystem namely *Communications infrastructure, Robot-specific features*, and *Tools*.

The communications infrastructure consists of message passing, recording, and playback of

messages, remote procedure calls (RPC), and distributed parameters system. Robot-specific features are standard message definitions for robots, robot geometry library, robot description language, preemptable remote procedure calls, diagnostics, pose estimation, localization, mapping, and navigation: the command-line tools, rviz, and rqt provided by ROS help to debug a robot.

The best part of ROS is that it integrates with GAZEBO, OpenCV, Pointcloud library (PCL), MoveIt, and ROS industrial seamlessly. We used Gazebo, OpenCV, and the PCL library.

Gazebo (Koenig & Howard, 2004) is a 3D indoor and outdoor multi-robot simulator, complete with dynamic and kinematic physics, and a pluggable physics engine. Integration between ROS and Gazebo is provided by a set of Gazebo plugins that support many existing robots and sensors. Because the plugins present the same message interface as the rest of the ROS ecosystem, we can write ROS nodes that are compatible with simulation, logged data, and hardware. We can develop our application in simulation and then deploy to the physical robot with little or no changes in your code.

OpenCV (Bradski, 2000) is the premier computer vision library, used in academia and in products around the world. OpenCV provides many common computer vision algorithms and utilities that you can use and build upon. ROS provides tight integration with OpenCV, allowing users to easily feed data published by cameras of various types into OpenCV algorithms, such as segmentation and tracking. ROS builds on OpenCV to provide libraries such as image_pipeline, which can be used for camera calibration, monocular image processing, stereo image processing, and depth image processing. If any robot has cameras connected through USB, Firewire, or Ethernet, ROS and OpenCV will make life easier.

PCL (Rusu & Cousins, 2011), the Point Cloud Library, is a perception library focused on the manipulation and processing of three-dimensional data and depth images. PCL provides many point cloud algorithms, including filtering, feature detection, registration, kd-trees, octrees, sample consensus, and more. Any work with a three-dimensional sensor like the Microsoft Kinect or a scanning laser, then PCL and ROS will help collect, transform, process, visualize, and act upon that rich 3D data.

Chapter 4

Assistive technology with image classification

This chapter is about applying image classification for the assistive solution. In this stage of work, we optimize and compare the performance of different deep learning architectures for awareness on a sidewalk using small form factor devices such as Raspberry Pi 3. The main objective is to find a deep-learning architecture that is complex enough to classify a set of obstacles on the sidewalk accurately. Our selection criteria of efficient deep architecture are a minimum number of parameters, lower power consumption, and robustness against the effect of the diurnal cycle. In particular, we compare the performance of GoogleNet, ResNet, InceptionV3, MobileNet, and VGG-16 on a database constructed for sidewalk applications. Empirical evaluation on that dataset suggests that the performance of ResNet is superior compared to other architectures' classification accuracies. To further our objective, we optimize the hyperparameters of ResNet to find architecture with a lower number of parameters without losing accuracy. Furthermore, we investigate the efficacy of different color spaces to address problems related to the diurnal cycle and power usage without sacrificing accuracy and generalizability.

4.1 Sidewalk obstacle image dataset construction

Training CNN for image classification requires databases. There are widely used image datasets available, e.g., MNIST, ImageNet, CIFAR, Caltech, STL-10 (Coates, Ng, & Lee, 2011), SVHN, NIST. These databases cover many classes of objects from handwritten digits to house numbers and vehicles to animals. Flores and Manduchi (2018) did a thorough study of mobile vision (Manduchi, 2012). They also created an inertial sensor time-series dataset which can be used to model turn-taking, step counting of blind people (Flores & Manduchi, 2018). However, there is no custom database related to the obstacles on a sidewalk. Most existing databases do not have objects affected by diurnal cycles and shadows. Therefore, we decided to create an image dataset that incorporates obstacles identified by representative users. To the best of our knowledge, no publicly available database collects explicitly for sidewalk obstacles.
To select relevant obstacles on the sidewalk, we interviewed 50 visually impaired people [1]. Among the participants, 20 people have complete vision loss (10 of them are congenitally blind), and 30 have partial sight of different degrees. We conducted the interviews by asking them open-ended questions. The questions were: (1) how often do you walk on a sidewalk? (2) what are the difficulties you face on a sidewalk? (3) how do you resolve those difficulties? From their answers the top 10 difficulties (obstacles) are listed in Table 4.1. The distribution of the age group and gender group of the participants is presented (see Figure 4.1). From the list of obstacles, we observe that some obstacles seemed less important (e.g., crowd) but showed up on the list with 18%. More crowd is visible in urban areas compared to rural areas. This result implies that the list of obstacles is generalized, not area-specific.
In this pilot dataset, we consider five classes of obstacles: construction, crowd, pothole, a person with a bike, and a person with a pet. Each class has $10,000$ images, and the total number of images is $50,000$. Each image is 96×96 pixels with RGB channels. The STL-10 natural image dataset (Coates et al., 2011) inspired the size of the images. Figure 4.2 shows the sample data from the database (Ahmed & Yeasin, 2017). The images were collected by taking photographs from the sidewalk and also using Google image search. We also used virtual example creation to increase the size of the database for training and testing the models.

[1] IRB approvals at the University of Memphis 16322937, 22627312, 29904158

Table 4.1: List of top 10 obstacles identified by representative users.

Name	Percentage (%)
Potholes / Damaged Sidewalk	29
Crowd	18
Construction	14
Person with pet	8
Person with bike / bike	6
Curbs	4
Slope	4
Poles	3
No sidewalk (sidewalk ending)	3
Narrow sidewalk	2

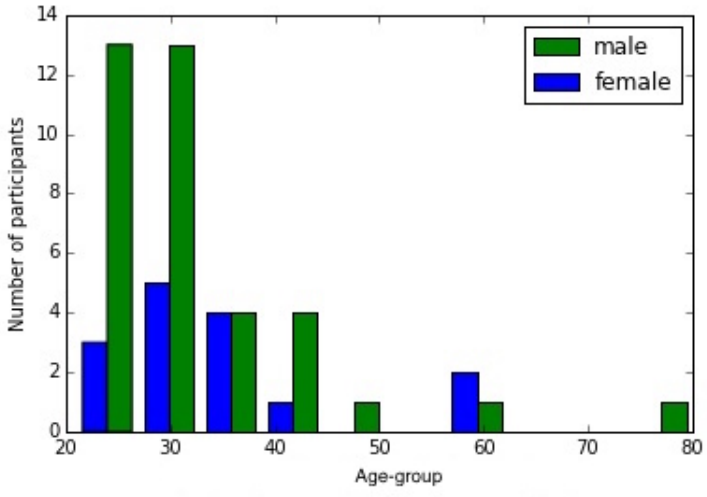


Figure 4.1: Distribution of participants.

There are noticeable variabilities in the image database. 25% of the images have occlusion. Some of them have multiple obstacles, for example, construction and person-pet, construction and person-bike, pothole, and construction. The database contains blurred images, images captured under diurnal cycle from little light to bright light, affine transformed images that may occur due to the position of the sensor on the obstacles, different types noises to account for ambient conditions. These variabilities make image recognition difficult. Some of the images contain shadow. Seasonal variations such as snow or rainfall do not belong to this version of the database. Additionally, the database does not have images of different terrains.

Figure 4.2: Samples from the sidewalk obstacle dataset.

4.2 Optimization and evaluation of DNNs

After building the image dataset, we plan to deploy an optimized DNN to RPi3. Finding the best DNN for the small form factor machines (SFF) (e.g., RPi3) is a challenge. That is why we studied the performance of deep architectures. We considered matrices, e.g., accuracy, network size and parameters, and power consumption. Our goal was to experiment with different architectures and color spaces to build a model that is robust against the diurnal cycle. We also required the network to be small enough to run on SFF machines with minimum power consumption.

We trained ResNet (He, Zhang, Ren, & Sun, 2016), GoogleNet (Szegedy et al., 2015), VGG-16 (Simonyan & Zisserman, 2014), InceptionV3 (Szegedy, Vanhoucke, Ioffe, Shlens, & Wojna, 2016), and MobileNet (Howard et al., 2017) from the scratch to see the performance. Results tabulated in Table 4.2.

There was one issue common with all of these networks, which is the diurnal effect (Sims & Dunigan, 1984; Wirz-Justice, 2008). An obstacle appears different from the variation of the diurnal state. For example, in the morning and the afternoon, the shadow of obstacle falls in the opposite direction. This formation of shadow makes the obstacle tough to recognize. It is

Table 4.2: Top-5 accuracy of model training and testing. The sidewalk image data is used to evaluate in all cases.

DNN	From scratch	Transfer learning	Fine-tune
ResNet	69%	55%	86%
GoogleNet	58%	50%	83%
VGG-16	54%	46%	77%
InceptionV3	63%	51%	87%
MobileNet	71%	56%	87%

easy to note that the effect of diurnal cycle manifest as a variation in the luminance channel while chrominance channel information remains relatively unchanged (Figure 4.3). We also observed that if the image represented in CIELAB color space (i.e., $La*b*$ color space), the true neutral gray value which is $a*$ does not contain the shadow at all (Figure 4.4). We, therefore, converted the images from RGB color space to $La*b*$ color space and evaluated the DNNs.

Then we performed transfer learning (Torrey & Shavlik, 2010) of pre-trained models available. The output of a network trained on ImageNet is 1000 class scores. In *transfer learning,* researchers take a pre-trained model and remove the last fully connected layer. Then they consider the remaining of the network as a fixed feature extractor for a new dataset. The new dataset consists of a fewer number (e.g., less than 1000) of classes that are fed into the network to train a linear classifier (e.g., softmax). Hence the learning of the original model is adjusted to the new dataset through the linear classifier. Backpropagation does not change any weights of the fixed feature extractor. On the other hand, in *fine tuning* (Yosinski, Clune, Bengio, & Lipson, 2014) all or some of the layers of fixed feature extractor are retrained with the new dataset (e.g., the weights are backpropagated). This way, the higher-level layers become progressively more specific to the details of the classes (Karpathy, 2017).

The five deep architectures trained themselves from scratch with same obstacle image dataset used before. The obtained results are presented in Table 4.2. The MobileNet achieved approximately 71% accuracy to classify obstacles correctly. This result is the highest among the three networks. The Inceptionv3, VGG-16, GoogleNet, and ResNet accuracies were 63%,54%,58%, and 69%, respectively. At this stage, we investigated the transfer learning of the pre-trained standard models. We froze all the trainable layers but the topmost layer and retrained it with

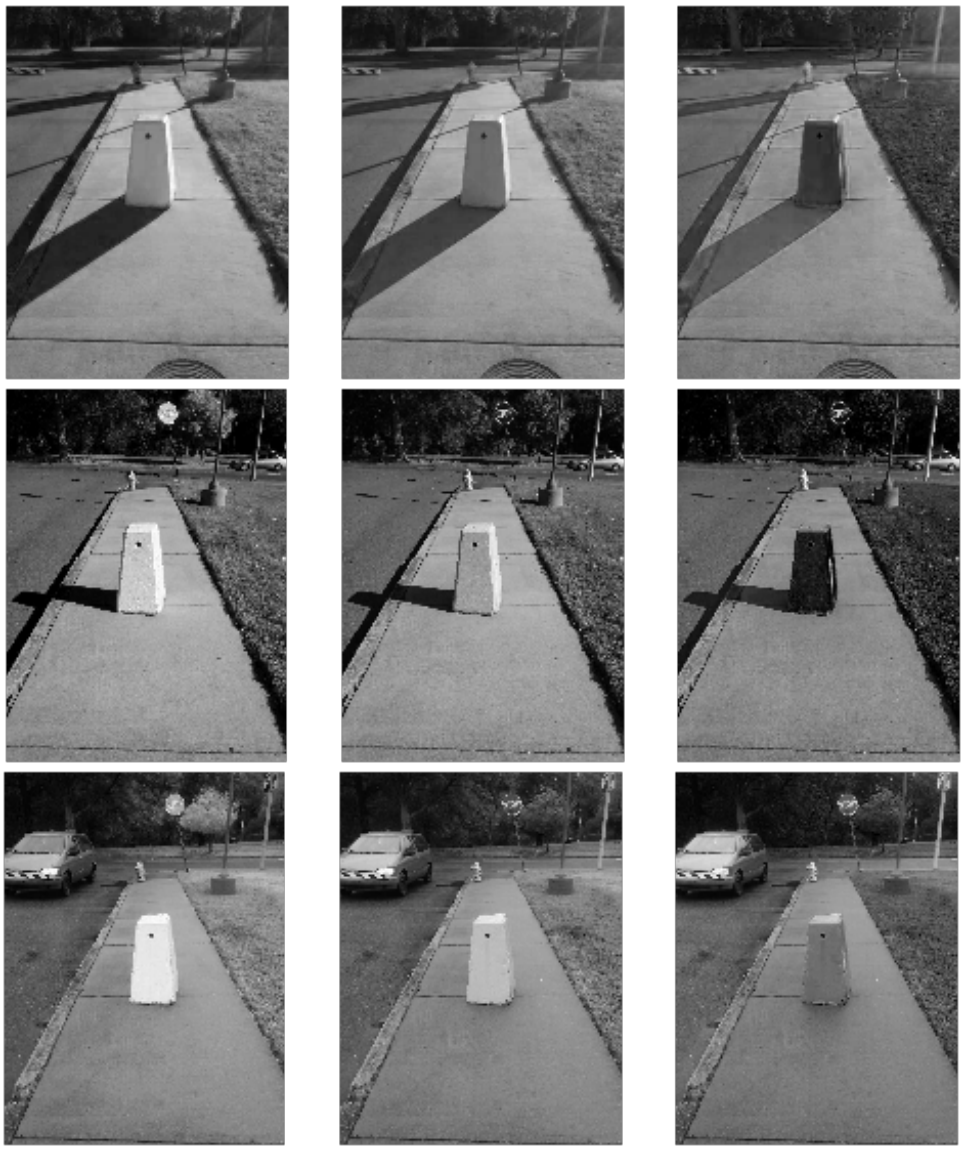


Figure 4.3: RGB components of the same obstacle at 8am, 1pm, and on a cloudy weather in gray scale. Note that the shadow is visible in all components.

AS dataset of five classes. Surprisingly the performances dropped below 56%. This drop is due to the different nature of the dataset. The transfer learning curve and the fine-tuning curve of MobileNet is shown in Figure 4.5 and 4.6, respectively. From the transfer learning curve in Figure 4.5, we observed that the model overfitted after four steps. The standard model was trained on ImageNet (Deng et al., 2009b). However, ImageNet does not have sidewalk obstacle images, but sidewalk obstacle image dataset contains obstacle images.

Furthermore, the fine-tuning curve of MobileNet showed significant improvement in the performance according to Figure 4.6. At the stage of fine-tuning, we un-freeze the lower convolutional layers and retrained. The batch size was 10, the learning rate was 0.001, and there

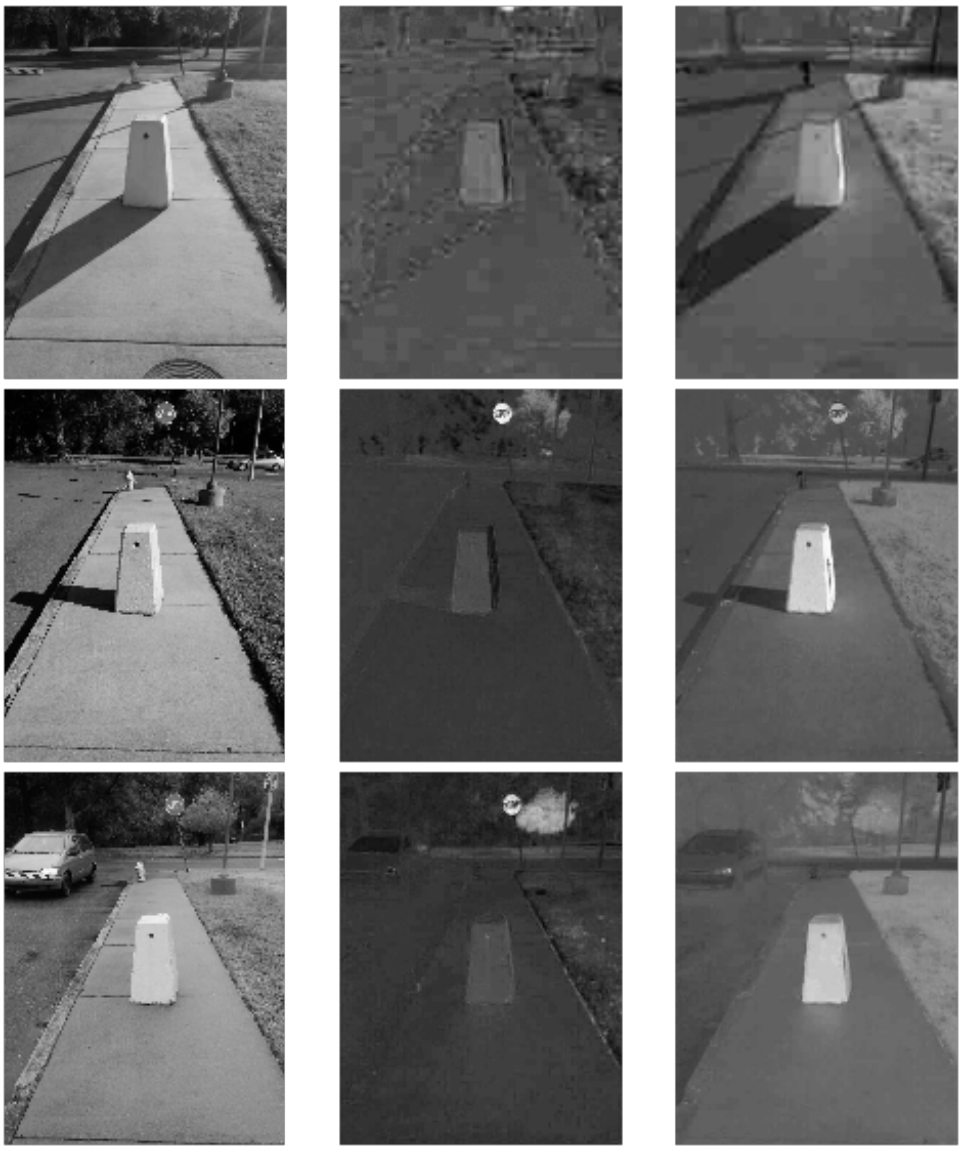


Figure 4.4: $La*b*$ representation of the same obstacle shown in 4.3. Note that the a* component does not have the shadow.

were 20 epochs. The training set contains 45k, and validation set 5k out of 50k. All the three networks performed with similar (over 86%) accuracy after fine-tuning. The learning curve of InceptionV3 and ResNet exhibited a similar pattern which is not presented to avoid redundancy.

The number of parameters of the three deep architectures are given in table 4.3 (Chollet et al., 2015). We choose MobileNet to build the deep model. This architecture has fewer number of parameters compared to InceptionV3 and ResNet. Moreover, the architecture is deep enough to capture the complexities in the sidewalk obstacle dataset and fast enough to detect obstacles on RPi3 or cloud in soft real-time. In addition, the size of the trained model is smallest

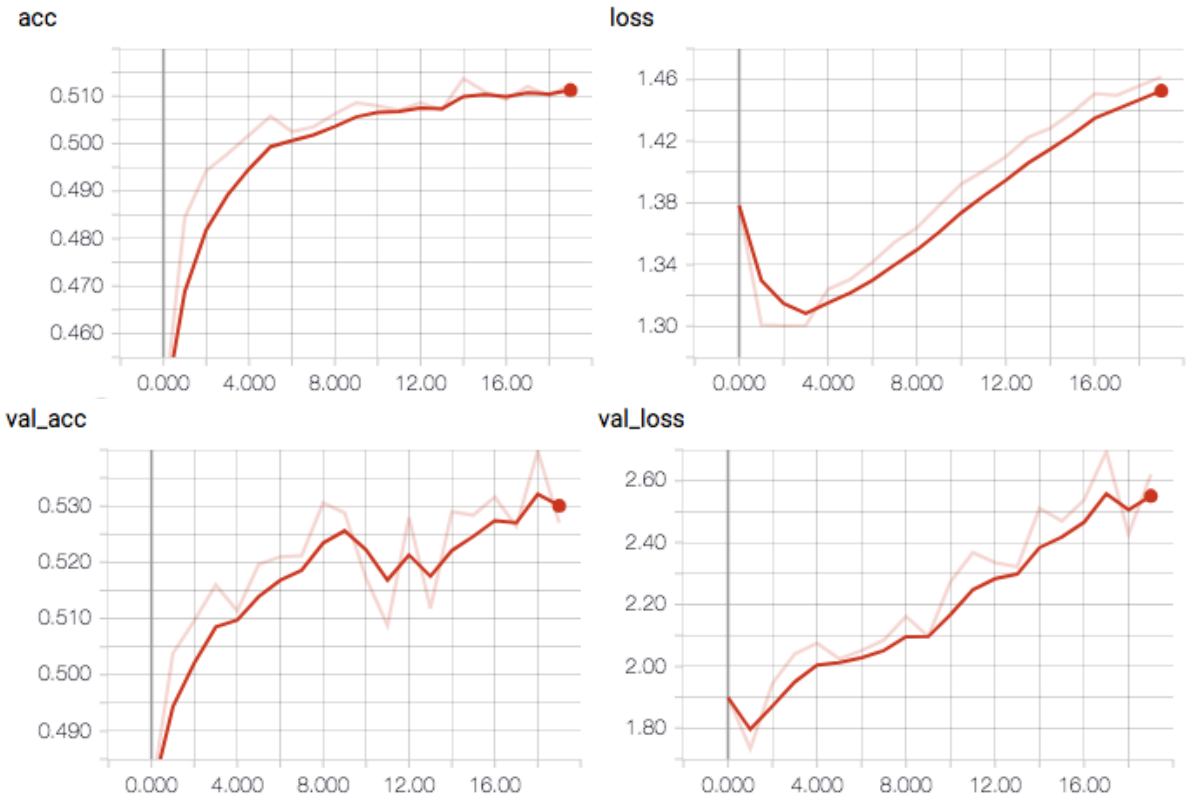


Figure 4.5: Transfer learning curve of MobileNet on AS dataset.

(16 MB) among these three architectures. The smaller size of the pre-trained model is significant because of the limited memory storage and I/O capability of RPi3.

Table 4.3: DNN architectures with number of layers, parameters, depth, and model size.

DNN	Layers	Size	Parameters	Depth
InceptionV3	48	83 MB	23.8 million	159
MobileNet	28	16 MB	4.2 million	88
ResNet	50	98 MB	25.6 million	168

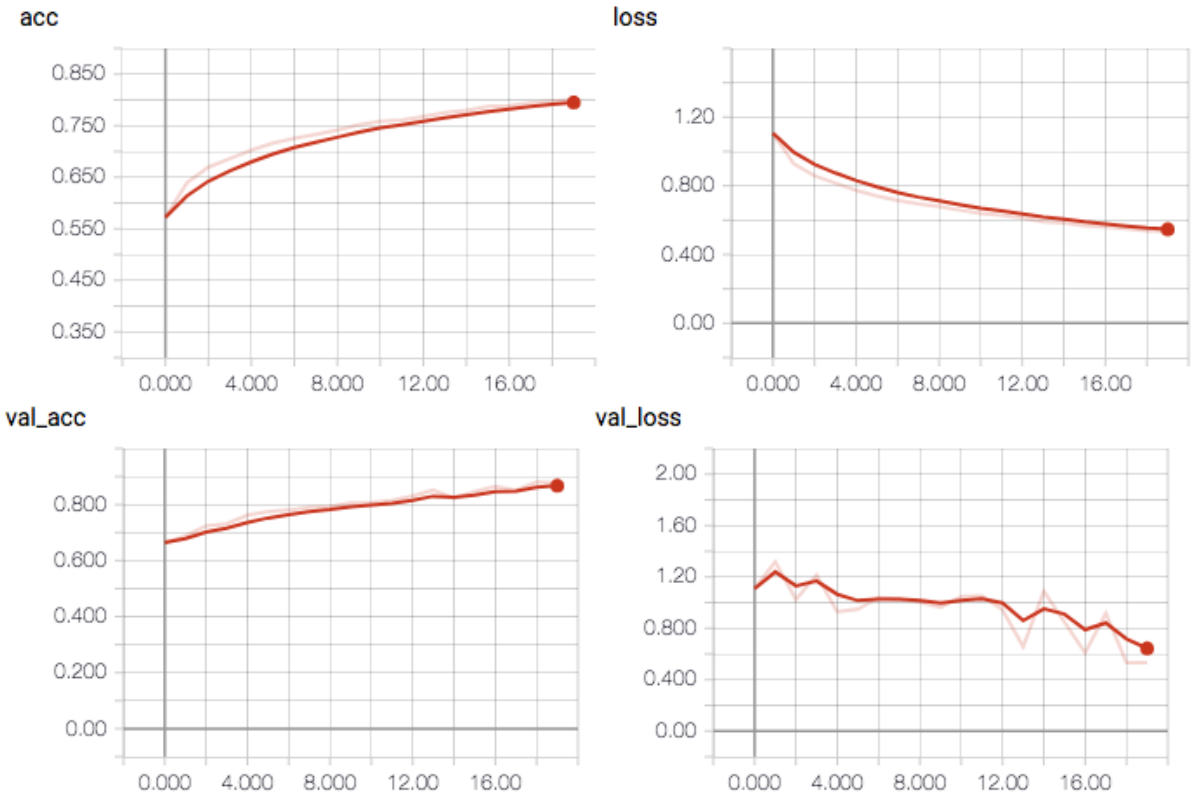


Figure 4.6: Fine-tune learning curve of MobileNet on AS dataset.

Chapter 5

Improved assistive technology with image captioning

In this chapter, we present the outcome of experiments with off-the-shelf image captioning systems. Design and implementation of a system embedded in an RPi3 is part of the experiment. The main components of the system includes *(a) generation of the meaningful caption from images, (b) implementation of personalized feedback mechanism for efficient communication with a minimal cognitive load*, and *(c) an interactive user interface and energy-efficient integration that account for multiple configurations to be more inclusive*. In particular, the performance of off-the-shelf image captioning systems was compared, e.g., Microsoft Cognitive Service, Clarifai, Google Vision API, and IBM BlueMix to determine the best platform for meaningful caption generation. We implemented three different schemes, namely text-to-speech synthesis, haptics, and ring tone, to provide personalized feedback. The implemented system interface is energy efficient and interactive to provide ambient awareness. We tested the fully integrated system on the sidewalk to get an objective evaluation. In particular, we focus on the accuracy of the captioning system and the usage analytics to provide helpful tips on the spot and to understand long term system behavior.

Table 5.1: Captioning time for cloud API.

API Host	Time (ms)
Clarifai	381
Google	617
IBM BlueMix	838
Microsoft	1380

5.1 Image captioning

Obstacle image classification has limitations. For example, if there are multiple obstacles, a classifier recognizes only the most apparent one. A classifier provides probability measurements of the multiple obstacles but this information is challenging to incorporate into an application. Because to provide feedback to a visually impaired spelling out the obstacles with probability is not suitable for the time constraint. Instead of classifying the obstacle image, it is more optimal to generate a caption of the image because the caption usually contains the description of the image including obstacles in the image. Image captioning performs very well for describing a general-purpose image. The sidewalk obstacle image requires an accurate description so that the visually impaired is well aware of the ambient scene. Image captioning techniques are still immature to perform this task.

For image captioning purposes, we investigated off-the-shelf APIs available in the market. We found that only Microsoft Cognitive service has image captioning capability among those computer vision APIs. Google, IBM BlueMix, and Clarifai do image tagging and concept generation but not image captioning. Here we report the details of the experiment.

In the first step, we initiate the API call from a desktop computer. The calls were made using a single image to Google Vision, IBM BlueMix, Microsoft Cognitive service, and Clarifai. On an average Clarifai took a shorter time and Cognitive service took longer time to tag the image. Table 5.1 shows quantitative evaluation result. The variation of the time is happening due to usage of free version of the APIs, the distance of the geo-location, or the data communication network delay.

In the next step, we deployed the application in RPi3. Five sighted volunteers walked on the sidewalk with the device where there is WiFi network available. They collected pictures of the sidewalk obstacle using the device in real-time. The image collection was performed to

capture different variability that include, different diurnal circle, lighting condition shadow, low and bright sunlight, forward-backward motion of obstacle, moving obstacle, rotation, and occlusion.

We observed that the captioning performs better when the image is a frontal view, and the obstacle occupies most space in the image. Performance reduces where there are multiple objects and obstacles. Some of the correct captioning (based on human judgment) cases are shown in Figure 5.1. Most of the caption contains the word "sitting" and start with "a". The frequent presence of these words occurred because of the original dataset (the paired images with captions) on which the model was trained contained these words frequently.

(i) a car parked on the side of a road

(ii) a traffic light sitting on the side of a road

(iii) a construction site

(iv) a fire hydrant on the side of the street

(v) a pole sitting in the middle of a sidewalk

Figure 5.1: Example of correct captions.

There are a few situations where the visual description was not correct. Examples are in Figure 5.2. In the picture (i), the bollards are thought to be a bench, and there is no obstacle in the picture (ii), but the caption talks about fire hydrant. Moreover, in the picture (iv), the pothole looks like a dog laying; in the picture (v), the pole is described as fire-hydrant. The image (v) is not bad because even the pole is described as fire-hydrant. At least from this description, the visually impaired receives a clue.

Rotation. To observe the rotational case, we manually rotated the camera to get 90 degrees rotation which and 45 degrees rotation we obtained through an image rotation function. Figure 5.3 shows some captions of rotated image. The captions generated for the rotated images does not describe the image properly. The 45 degrees rotated image captions is not correct due to the dark portion due to the rotation. It is promising that at least the model was able to identify the sidewalk in all four rotated cases, though the captions are not relevant.

(i) a bench on a sidewalk

(ii) a fire hydrant on the sidewalk

(iii) the side of a road

(iv) a dog laying on a sidewalk

(v) a fire hydrant on the sidewalk

Figure 5.2: Example of incorrect captions.

(i) person walking down a sidewalk

(ii) a sandwich on a sidewalk

(iii) a fire hydrant on the side of a fence

(iv) a man laying on a sidewalk

(v) a cat is standing on a sidewalk

Figure 5.3: Incorrect captions due to rotation.

Diurnal effect. The captioning task for image also was not correct where there is shadow due to diurnal effect. The images in Figure 5.4 are taken from a certain place of the same obstacle at a different time of the day. We did not use $La*b*$ because the image captioning presumably performs better with color images than only with the $a*$ component of images. However, the captions are different and sometimes very irrelevant. The first image talks about a "red train", but there is no train in the image. Probably the long shadow looked like a train. Other three images are somewhat relevant to the construction site, which is apparent.

(i) a red train traveling down tracks next to a fire hydrant

(ii) a construction site

(iii) a fire hydrant on the side of a fence

(iv) a construction sign on the side of a road

Figure 5.4: Example of diurnal effect (pictures taken at 8AM, 11AM, 1PM, and 4PM of the same obstacle).

Chapter 6

Modeling ambient awareness on a sidewalk

Researchers are continuously improving models for their interests using innovative ideas and techniques. This dissertation is about building a sidewalk model with obstacles. Avoiding those obstacles is an essential part of safe navigation for visually impaired. Most of the models of obstacles and obstacle avoidance address the problem of the static and intransient natural obstacle. However, avoiding the transient obstacle, e.g., puddle, scooter, and pothole, remains a challenge. In this chapter, we demonstrated a novel approach of finding free of obstacles navigable path using reinforcement learning. Instead of modeling only the obstacles, the RL learns the path to navigate avoiding obstacles in this approach. As long as there is a path to navigate, the user is safe to walk. The RL model, along with the conversational agent, improved the obstacle avoidance experience for the visually impaired.

One of the essential solutions in the scientific community to model obstacles is image-based. This solution depends on the visible light because to capture an image light is necessary. However, for a visually impaired, it is difficult to capture a perfect image where the obstacle is clearly in the picture. Besides, the classifiers miss-classify the image of the same obstacle in the morning and evening because of the diurnal effect. For this reason, there should be an alternate way to sense the obstacles.

The possible alternative of the image-based solution is the use of point cloud (PC). The PC

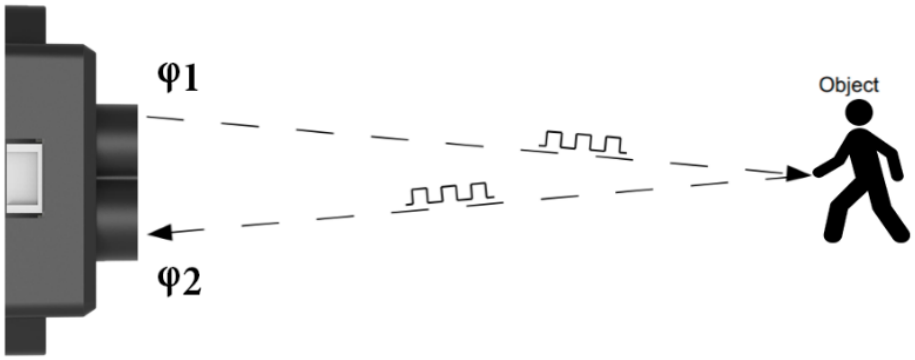


Figure 6.1: The schematics of TOF working principle (image from TFmini datasheet).

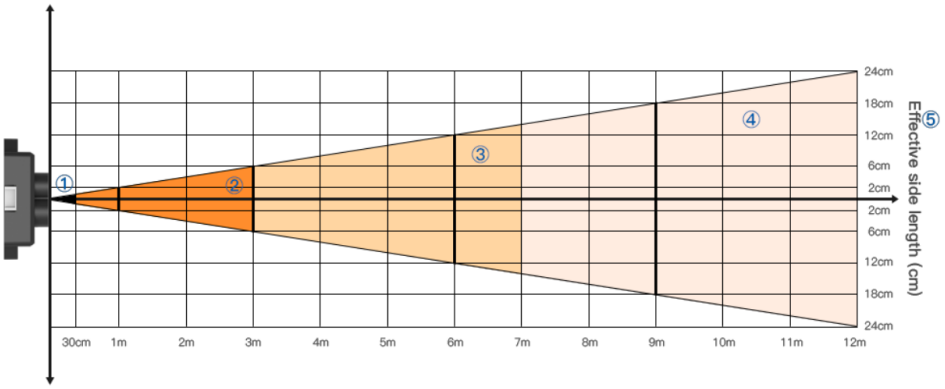


Figure 6.2: The schematics of range of distance measurement and effectiveness (image from TFmini datasheet).

is a set of data points in space (Lee, Moon, Ko, Choi, & Lee, 2017a) and usually constructed from laser technology-based cameras (e.g., Intel RealSense, Microsoft Kinect). The PC contains both RGB and depth information (RGB-D). An additional advantage is that the PC is not highly dependent on visible light. There are various depth cameras available in the market (`https://rosindustrial.org/3d-camera-survey`) for building PC effectively. Some of those are bulky, less energy efficient and some are smaller as well as less energy consuming. Initial construction of PC is performed using the TFmini-IC form `www.benewake.com`. This works on the time of flight (TOF) principle. It transmits modulation wave of the near infrared ray on a periodic basis, which reflects after contacting objects. The device obtains TOF by measuring round-trip phase difference and then calculates the distance from the equation $D = \frac{C}{2} \times \frac{1}{2\pi f}\Delta\phi$ Fig 6.1.

- ① is the blind area $0-30cm$
- ② operating range under extreme condition, which is $0.3-3m$. Extreme condition refers to the outdoor glare (of which illuminatin intensity is around 100klux e.g., at noon in summer) and detection of black target (with reflectivity of 10%).
- ③ operating range under normal sunshine condition (with illumination intensity of around $70klux$) which covers the range $0.3-7m$.
- ④ operating range at the indoor environment or considerably weak ambient light environment $0.3-12m$. This is more useful at night.
- ⑤ is the minimum side length of effective detection at different distances. To get reliable data, the side length should be equal to or more than the minimum side length. The minimum side length of effective detection depends on the field of view (FOV). The FOV of tfmini is the smaller value between the receiving angle and the transmitting angle, which is calculated by $d = 2 \times D \times \tan\beta$. d is the minimum side length of effective detection; D is detecting range; β is the half of the value of the receiving angle 1.15°. Table 6.1 represents the list of minimum effective side length and detection range.

Table 6.1: Minimum side length of effective detection corresponding to detection range

Detection range (m)	1	2	3	4	5	6	7	8	9	10	11	12
Minimum side length (cm)	4	8	12	16	20	24	28	32	36	40	44	48

The geometric arrangement of the LiDAR sensor is given in Figure 6.3. The sensors require to place very close to each other to reduce the size. The coverage has to be as broad as a typical width of a sidewalk. There are 5 sensors in each row and 4 sensors in each column. According to the U.S. Department of Transportation Federal Highway Administration, minimum sidewalk width is 5 feet or $1.5m$ (*Walkways, Sidewalks, and Public Spaces*, n.d.; Kim, Choi, & Kim, 2011). Covering $150cm$ of width of sidewalk at a distance of $7m$ with the 5 sensors, the angular placement should be at least at a $\theta = \frac{s}{r}$ radian or $\frac{28cm}{7m} = 0.04r$ or 2.29° Fig. 6.4. Each row of the sensors is used to cover the obstacles at ground level, knee-height, and head height. The average height of men and women in the United States range from $161cm$ to

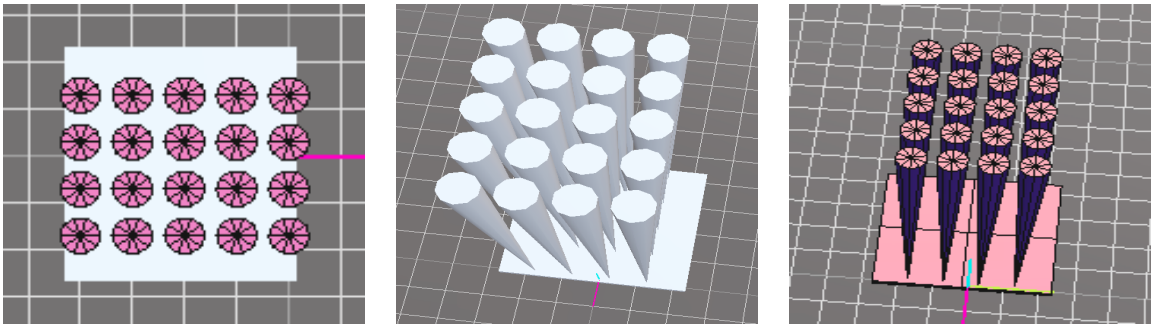


Figure 6.3: Geometric arrangement of the LiDAR sensors.

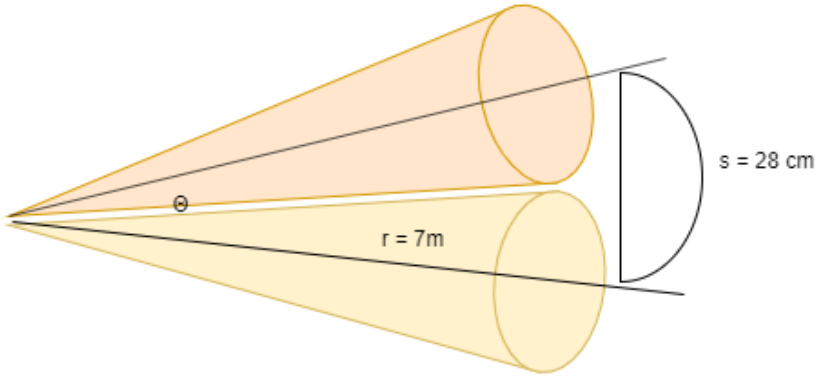


Figure 6.4: The sensors placement angle.

176cm. Considering the average height of men the rows of sensors placed at $\frac{44cm}{700cm} = 0.06r$ or 3.43° for a safe distance of 7m,

Dimension of the 20 assembled TFmini sensors was 10.6×3 inch. For building assistive device, the size of the device must be smaller as much as possible. Because of the over-width of the hand-made TOF sensor, we choose the Intel RealSense D435 as an alternate. Its dimension is 2×0.7 inch which is 75% reduction of the width. Moreover, it has the RGB sensors builtin along with the laser sensors.

6.1 Definition of free-path

Generally, the sidewalk consists of static and dynamic obstacles. The dynamic obstacles have motion. The visually impaired person walking on the sidewalk has motion as well. The mean comfortable walking speed of adult (aged between 20 to 70 years) ranges approximately from 100 cm/s to 150 cm/s. (Bohannon, 1997).

Suppose $\chi = (M, d)$ is a discrete metric space from euclidean space $\mathbb{R}^n$, where $M \subset \mathbb{R}^n$ is the set of points and d is the distance metric. The density of M in the ambient euclidean space may not be uniform due to perspective distortion. There exist a set of functions f that take χ as input and produce clusters satisfying a set of constraints (e.g., points at a given neighborhood distance or color) (Charles, Su, Kaichun, & Guibas, 2017). In this dissertation $n = 3$ meaning the spaced is three dimensional.

In the given χ the *free path* is defined as $f(\chi) = \phi$ which indicates there is no obstacle along the direction of interest. If $f(\chi) = C$, where C is a set of clusters in χ. The *threat level* t is inversely proportional to the distance of the cluster c_i ($C \in \{c_1, c_2, ..., c_i\}$), that is $t \propto \frac{1}{d_i}$ (Morales, Toledo, Acosta, & Sánchez-Medina, 2017; Lee, Moon, Ko, Choi, & Lee, 2017b).

6.2 Reinforcement learning for modeling free-path

Researchers are spurred to improve the mobility of visually impaired by devising obstacle avoidance mechanism. There are vision-based solutions to model the obstacles (Escobar-Alvarez et al., 2018; Barry, Florence, & Tedrake, 2018). The obstacles are modeled using traditional computer vision algorithm or modern DNN (Zhou, Li, Cao, Wang, & Wu, 2018). Both traditional and DNN algorithms have a limited capacity of modeling dynamic nature and huge number of obstacles. Dynamic nature refers to stationary and moving obstacles along with their sizes, shapes, motion speed, and colors. The dynamic number refers to the unknown number of obstacles. Any object blocks the mobility of the people is an obstacle. There is research to combine camera and Inertial Measurement Unit (IMU) sensors [1] with improving the model of obstacles. In this approach, the system becomes too much complex. Simple sensor-based algorithms are prevalent nowadays to reduce complexity. Yang, Wang, Lin, Bai, Bergasa, and Arroyo (2018) proposed pairs of sensors for this purpose (K. Yang et al., 2018). RealSense R200 and IMU are mounted on smart glass at eye-level and RealSense RS410 at waist level. This system is efficient to detect low-lying obstacles. Wang, Yang, Hu, and Wang described stixel representations of 3D world combined with pixel-wise semantic

[1] This is an electronic device that measures and reports orientation, velocity, and gravitational forces through the use of accelerometers and gyroscopes and often magnetometers

segmentation for the navigation aid (J. Wang, Yang, Hu, & Wang, 2018; Cordts et al., 2017). All the above-mentioned technologies are limited to certain class of obstacles. For example, the CNN based models are capable of recognizing only the classes of obstacles belong to the training classes. Moreover, the model has to see the obstacle beforehand. To overcome the shortcomings and to simplify the navigation on a sidewalk, the proposal in this dissertation is "free-path." The idea of free-path is to find a safe area on a sidewalk instead of trying to model the dynamic environment of obstacles. We utilized a RealSense D435 depth camera as well as the custom LiDAR to collect PC of the sidewalk. The PC is then used to model the free-path using reinforcement learning.

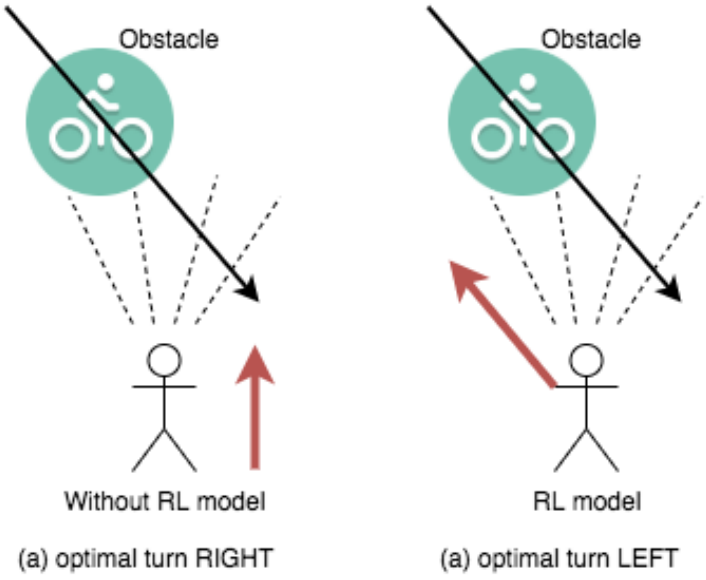


Figure 6.5: Optimal turn decided by RL model.

Another aspect is that the position of the dynamic obstacles has to be communicated to visually impaired. The visually impaired would take necessary action based on that. In a situation with a dynamic obstacle, the outcome of actions performed by visually impaired people is delayed. For example, to avoid a bike rider, the visually impaired may stop and stand on a side or keep walking towards a direction. She does not know if the bike rider is avoided until passed. In this case, her beginning actions (e.g., stopping, standing aside, or walking) are delay rewarded. To model this behavior, the RL is a perfect fit for both static and dynamic obstacles. The reason is explained with an example. Let us say a biker is approaching a user in a crossing pattern from left to right figure 6.5 (a). A model without RL will see an immedi-

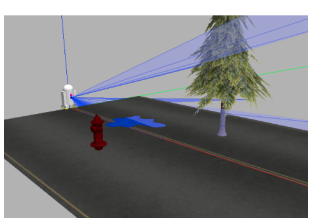
ate empty space in front and will decide that as a free path, whereas the biker will reach that space after some time. On the other hand, the model with RL takes the movement of the biker into account and decides to move left instead of going forward, figure 6.5 (b). In this dissertation, we choose RL to teach the robot the dynamic and static nature of the obstacles.

In order to build an RL model, there has to be an agent and environment. The agent placed in this environment can learn from the interaction with the environment. Building a real environment to train an agent is expensive, especially a sidewalk. Moreover, there must exist a practical way of implementing the punishment mechanism every time the agent makes mistakes. To understand the complexity of real sidewalk and to study the feasibility of the system the simulated environment is extremely suitable. It is easy to program, modify, and various types of agent can be placed in the environment. Implementing algorithm, training, testing much easier than real environment. That is why, we selected the simulation to train RL model and real environment to test it.

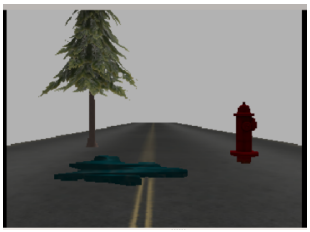
The RL model trained in Gazebo (Koenig & Howard, 2004) simulation environment. We place a robot with a virtual AGT on a virtual sidewalk, where there are obstacles (e.g., curb and grass beside the sidewalk, pothole, cone, fire hydrant, electric scooter, electric pole, dumpster, and tree). The RL algorithm stays in Robot OS (ROS). In this setup, we let the robot walk in the sidewalk with $10,000$ episodes and 1000 steps in each episode. The physics engine of Gazebo environment makes it easy to detect collision, fall, displacement, and other physical measurements. It also provides a way to set the base speed of the robot and we set the base speed equal to the mean walking speed of men. Whenever the robot collides an obstacle or fells down by going out of the sidewalk, it gets penalized by -1, and the simulation resets and robot starts from initial position. There are rewards of $+1$ for actions which do not cause collisions or falls. We present a depiction of a metaphor between the simulated sidewalk and real sidewalk in figure 6.6. Once the RL model is built, then it was transferred to the device for the testing and evaluation.

The following aspects makes the free-path problem to be solved by RL

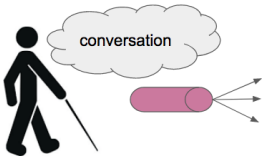
- Different actions yield different rewards. For example, when trying to avoid obstacle in a sidewalk, going left may lead to an avoidance, whereas going right may occur colli-



(a) Gazebo simulation



(b) What robot sees



(c) Real sidewalk

(d) What AGT sees

Figure 6.6: Analogy with AGT and Gazebo simulation.

sion.

- Reward for an action is conditional on the state of the environment. In figure 6.5, going left may be ideal at a certain position in the path, but not at others.
- Rewards are delayed over time. This just means that even if going left (Fig 6.5) is the right thing to do, we may not know it till the obstacle is completely out of sight.

We have defined the environment, state, action, and reward in terms of sidewalk in the following manner.

Environment: The sidewalk environment consists of static and dynamic obstacles. The static obstacle does not move whereas the dynamic obstacle moves. The sidewalk has curb and it has brick pavement. There are grass beside the sidewalk which is different in color than the sidewalk itself.

State: The state space is a set of all possible relative position of agent and the obstacles on the sidewalk. That is why the number of states are infinite. The agent finds useful information from the states to make right action.

Action: There are five actions namely stop, left, forward, right, and backward movement. The agent encounters infinite number of states and takes one of these actions in the action space set. There are four more actions (e.g., movements) under consideration, those are movement towards 45°, $-45°$,135°,$-135°$.

Reward: If an action performed by the agent causes collision then the reward is -1. Agent keeps getting +1 as reward until there is no collision.

With the above environment the Gazebo simulation is created. We implemented three algorithms, Q-learning, SARSA, and deep Q-learning network (DQN). The optimal parameters found for those algorithm are listed in table 6.2.

Table 6.2: Parameters for the learning algorithms

Parameters	Q-learning	SARSA	DQN
learning rate	0.5	0.5	0.001
discount factor	0.9	0.9	0.95
exploration probability	0.1	0.1	0.1
exploration decay	0.99	0.99	0.99

6.3 Active Interface: conversational agent

In daily life, any matter not apparent to the user becomes more transparent through the conversation. That is why the teachers request students to ask questions, and the managers ask the employee to ask questions. Through the conversation, the real scenario becomes evident. In this dissertation, we are adopting this concept. The user communicates with the agent, and the agent talks about what it sees ahead. Through the conversation, the ambiance become more apparent to the user. The agent mentions any obstacle on the walkway to the user. How to avoid that obstacle depends on the user. The AGT device will not command to do a particular action. Instead, the user decides the next action based on the conversation. This conversational agent is an active interface.

For the basic understanding of the conversational agent we should understand few keywords.

Intent: The intent is the end meaning of what the user is trying to say. For example, if the user says, “Find the fire hydrant” the intent can be classified as to find obstacle.

Entity: An entity is to extract useful information from the user input. From the example above, “Find the fire hydrant” the entities extracted should be the *name* of the obstacle. The name, for example, is a fire hydrant.

Stories: Stories define the sample interaction between the user and the conversational agent

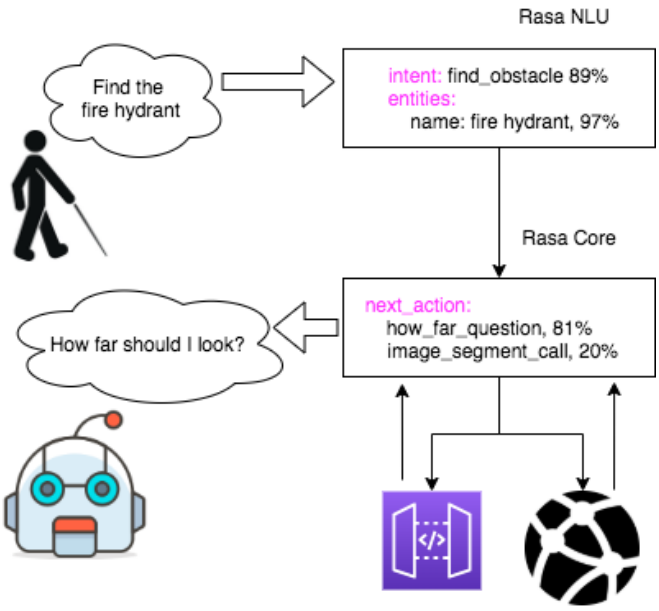


Figure 6.7: Block diagram of Rasa NLU and Rasa Core.

connecting intent and action performed by the agent. In the example above agent got the intent of finding the obstacle and entities like the name of the obstacle, but still, there is an entity missing - how far should it look. That would make the next action from the agent.

Actions: Actions are the operations performed by the agent. It could be either asking for some more details to get all the entities or integrating with some APIs or querying the RL model to get any information.

Templates: The templates are the sample replies from the agent which can be used as actions.

The conversational agent, a software system, enables a user to talk with it in natural language. RASA, an open-source machine learning framework, serves as the engine of the conversational agent. It is easy to customize. We can build, deploy, or host RASA internally in our server or environment with complete control. Confidential conversation data cannot be shared with third party. The majority of the conversational agent tools available are cloud-based and provide software as a service. We cannot run them internally in our environment, and we need to send data to the third party. With RASA, there is no such issue.

The RASA comprises of two main components *Rasa NLU* and *Rasa Core*. Rasa NLU is a library for natural language understanding (NLU), which does the classification of intent and extract the entity from the user input and helps the agent to understand what the user

is saying. Rasa Core, on the other hand, is a conversational agent framework with machine learning-based dialogue management capabilities. It takes the structured input from the NLU and predicts the next possible best action using a probabilistic model like long short-term memory (LSTM) recurrent neural network. Rasa NLU and Rasa Core are independent, and we can use NLU without Core, and vice versa. But using both NLU and Core enhance performance. A block diagram of RASA is shown in figure 6.7.

Three types of files are necessary to train Rasa NLU. NLU training file, Stories file, and Domain file. The training file contains some training data with user inputs along with the mapping of intents and entities present in each of them. The more varying examples we provide, better the agent's NLU capabilities become. Stories file contains sample future interactions between the user and the agent. Rasa Core creates a probable model of interaction from each story. The Domain file lists all the intents, entities, actions, templates, and some more information. The conversational data obtained from the WoZ experiment is converted to text and processed to create the above-mentioned training files. The training files are stored in markdown format. Samples form an NLU file is presented in listing 6.1.

```
## intent:greet
- hey
- hello
- are you there?
- are you ready?
- ready?

## intent:greet_ask
- Yes ready, are you ready?
- Ready, want to start?.
- I am here, start walking?

## intent:greet_normal
```

```
- yes
- yap
- let's go

## intent:find_obstacle
- Find [obstacle](obstacle)?
- What is [there](obstacle)?
- What is [that] (obstacle)?
- Do you see [anything](obstacle)?
- [There] (obstacle)?
- [Here] (obstacle)?
- This [way](obstacle)?
- That [way](obstacle)?

## intent:find_distance
- [Where](distance)?
- How [far](distance)?
- How long to [reach](distance)?
- Is it [close](distance)?
- Is it very [close](distance)?

## intent:bye
- bye, let me know
- bye now
- i am here, bye
```

Listing 6.1: Samples from an NLU file.

Once the RL model is trained, AGT integrates that model. Through this RL model the device sees obstacles and recommends an action. The AGT does not dictate the turn or move; it gives the ambient information about the obstacle, and the person decides which direction to move.

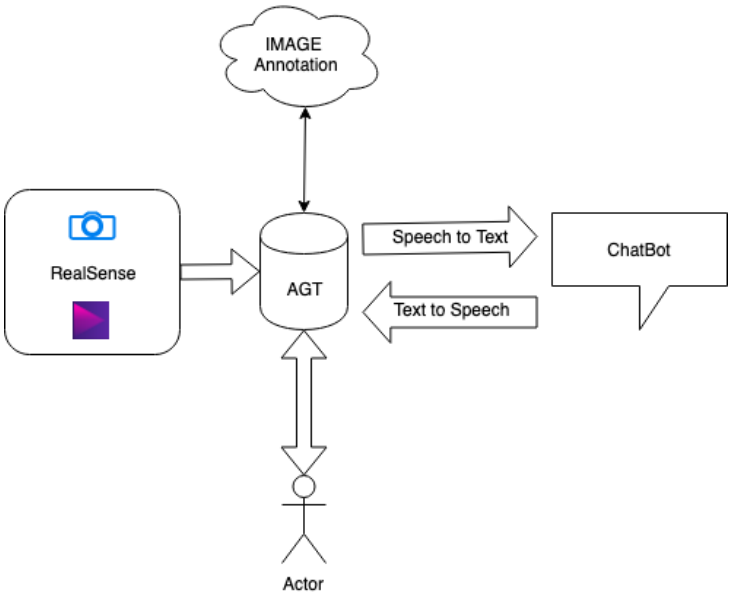


Figure 6.8: Block diagram of AGT.

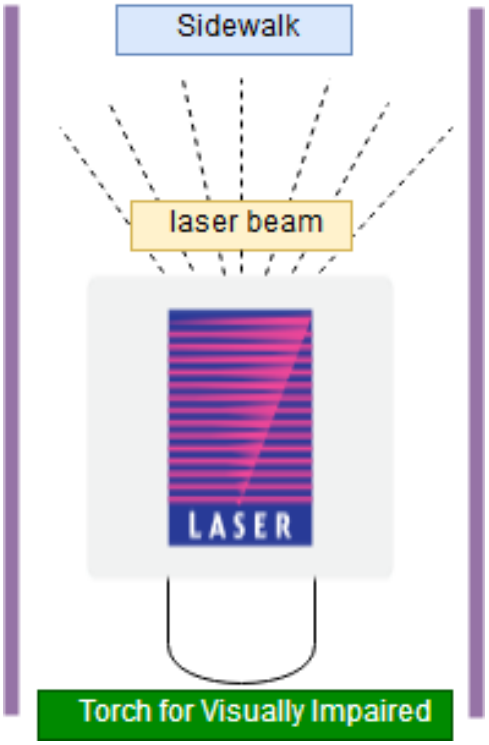


Figure 6.9: The Torch for Visually Impaired.

6.4 Augmented guiding torch (AGT)

AGT contains two essential modules free-path finder and conversational agent. The free-path finder module uses RL to find the obstacle-free path, and the CA helps visually impaired in-

formed about the ambiance through active conversation.

There are camera and LiDAR sensor connected to computing engine (e.g., Raspberry Pi, Jetson Nano) in the AGT. The upgraded version of AGT uses only the RealSense depth camera, which provides both RGB and LiDAR data, to make it lighter and smaller in size. The block diagram of AGT is shown in figure 6.8 and the working principal is shown in figure 6.9.

6.5 Evaluation

In the Gazebo simulated training environment, the robot is equipped with the depth camera. From this depth camera, the robot can sense the depth, and it senses color, texture from the RGB sensor. Figure 6.10 shows sample pictures from depth camera. The blobs in the pictures are laser beams which forms PC. Picture (a) is depth image taken during daytime, (b) is an infrared image also taken during daytime, (c) is taken at nighttime and it is an infrared image. The PC is the input to RL both in real sidewalk as well as in simulation. Base moving speed of the robot is set to the mean walking speed of men to make the simulation close to the real sidewalk. The lighting condition is set to ambient light, which gives an approximation of daylight. We were able to set the wind speed of the ambient environment. However, there were ways to create a sidewalk with a slippery surface, with ice, snow, and slope. Nevertheless, to avoid the extreme complexity of the implementation, we skipped these aspects within this dissertation scope.

(a) (b) (c)

Figure 6.10: Sample pictures obtained from depth camera.

In the training environment, we examined the learning of the three algorithms. The same sidewalk was used for training all of these. Q-learning, SARSA, and DQN include in the list of training algorithms. Within 200 episodes, the DQN learned best among the three, and SARSA

learned better than the Q-learning. Figure 6.11 shows these findings. The derivative of the learning curve of the reward increased over the number of episodes. In another way, we can say that the more interaction the robot makes with the obstacles, it learns to avoid it. That is why the reward increases after a couple hundred episodes.

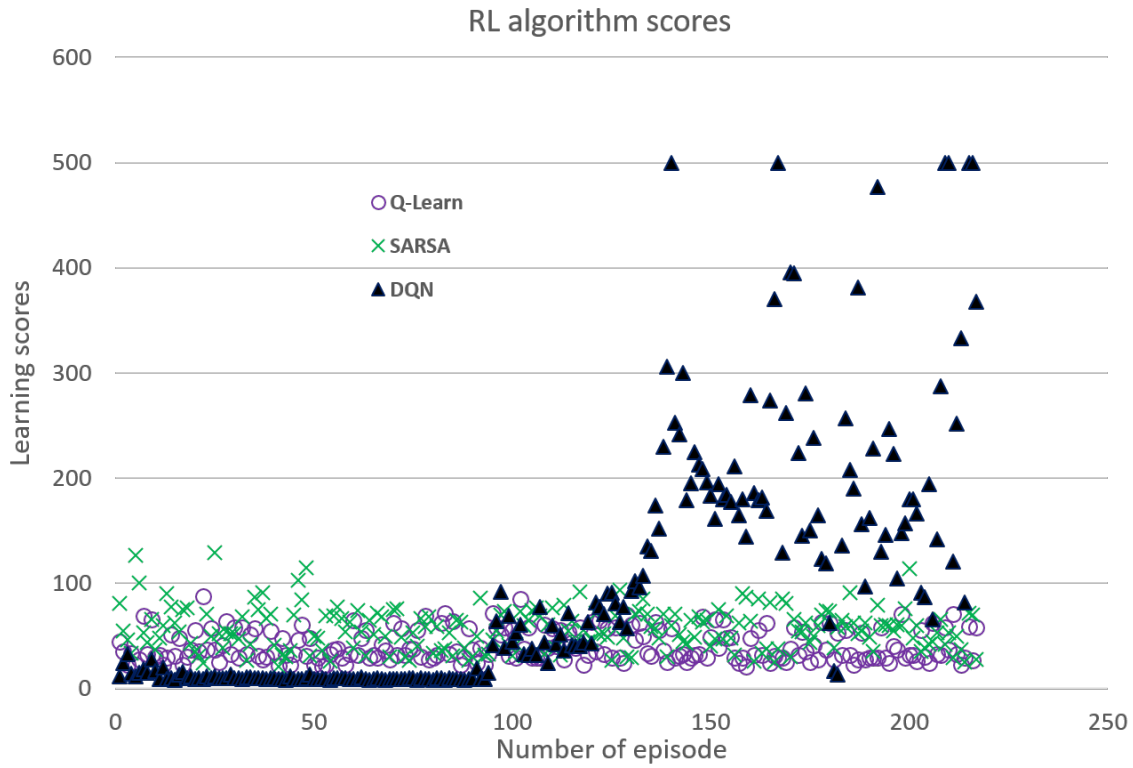


Figure 6.11: Learning score comparison of Q-learning, SARSA, and DQN.

The testing environment of the RL model is the real sidewalk. A visually impaired person volunteered to test the prototype. The IRB approval of the blind-ambition umbrella project is used for this testing as it involves human subjects. Five hundred feet of the u-shape sidewalk was selected for the evaluation of the prototype. There were trees, electric pole, pothole, dumpster, iron fence, visible curb, bollard, and a fire hydrant on this sidewalk. We manually placed a couple of electric scooters, yellow construction cones, water to form a puddle. The user was mostly happy about knowing about upcoming objects ahead of time. He could quickly point to any direction and ask “what is there?” and get names of the segmented objects. The obstacles which stand above the ground were found easily, but the ground level obstacle such as pothole and puddle was hardly found. On a narrow sidewalk, the RL got confused with the sidewalk fence (not the construction fence) as an obstacle. Though there are

limitations, according to the volunteer, the overall performance of the assistive device was found satisfactory.

RASA (Bocklisch et al., 2017) framework is the base engine for building a conversational agent. To train it, it requires conversational data, which we have obtained from the WOZ experiment on the sidewalk. We carefully annotated the spoken sentences of the visually impaired into proper intent, and we identified the entities and actions from those. Executing actions requires developing a service engine. An entity is passed as a parameter to the action. The RASA stack provides a light-weight SDK for this purpose. We used this SDK to develop the action end-point.

The input and output of the conversational agent is text. From an audio input device, the speech is converted to text and fed into the agent. The reply from the agent again converted back to speech and sent to the audio output device. We have used the speech-to-text engine for the speech to text conversion, and it generates words with correct spelling words. Because the RASA stack always receives words with correct spelling, we did not have to train it with incorrectly spelled words. For example, we avoided training the conversational agent with the variation of “hi”, “hey”, or “hai”.

The Bluetooth headset acts as an audio input and out interface. This device connects to the prototype of the assistive device and provides a partial scope of the private conversation. That is, people may hear what the visually impaired person is asking for, but they cannot hear what the device is replying.

We show the basic block diagram (see Figure 6.8) of the prototype. The user has the option to ask the AGT to take a picture and segment it. Amazon Rekognition does the segmentation of images in AGT. The text-to-speech and speech-to-text service is used from Google. Of course, to use the Google and AWS services, there is a need for internet connectivity.

Table 6.3 contains the results obtained from a test simulation. In the testing phase, we let the robot walk from one side to the other side of the sidewalk $10,000$ times which is the number of episodes. The robot found the construction cone most of the time but failed to see the pothole. It is reasonable, because the pothole is on the ground whereas the construction cone, fire hydrant, stopper, electric scooter stand above the ground. Among the above ground level ob-

stacles, the AGT is less able to detect electric scooter than other obstacles. This less detection is due to the size and shape of the scooter. Few other obstacles are shown in Figure 6.12.

Obstacle	Image	% signaled to avoid
Pothole		40
Construction Cone		90
Fire hydrant		89
Electric Scooter		70
Electric Pole		73
Dumpster		93
Tree		86

Table 6.3: Obstacles and simulated avoidance results

The AGT got confused with the obstacle in Figure 6.13. The real obstacle is the electric scooter but it was talking about the fence as well as obstacle.

Figure 6.12: Other example obstacles detected by the assistive device

Figure 6.13: The confused obstacle is the sidewalk fence, though that is not obstacle.

Chapter 7

Conclusion

In this dissertation, I built an assistive technology prototype device. The purpose of this prototype is to augment the means of avoiding obstacles for visually impaired. The obstacles include both static and dynamic nature, and the device is useful during a walk on the sidewalk. The journey of this prototype began with the image classification technique. The image classification technique requires an image dataset. As there was no image dataset specially built for sidewalk obstacles, we built one with 50K images, which contains five classes (pothole, a person with a bike, a person with pet, construction, and a crowd).

The outcome from the image classification technique is that the classifier does incorrect classification where the image is affected by the diurnal cycle (i.e., shadow is different in the morning and afternoon) and where the image has multiple obstacles (i.e., it picks up the one which is dominant). We partially solved the problem of diurnal effect during image classification by using $La*b*$ representation of the image. The assistive device was improved and re-devised based on image caption generation, and that helped to the problems of classification. The caption based system was amazing to the visually impaired, but it did not significantly contribute to avoiding the obstacle because the caption generated from an image does not always mention the obstacle. Besides, there is no notion of distance of the obstacle in the caption.

To overcome the shortcomings of both image classification and image captioning, we developed the free path approach instead of modeling the various obstacles. Reinforcement learn-

ing served as an essential tool for free path modeling. Also, to communicate the free path to the user, we incorporated the conversational agent trained on the RASA stack.

For modeling the free path, we created the simulated sidewalk and the 3D models of obstacles in Gazebo. We placed a robot in the environment, which learns to avoid obstacles through RL. When the RL gets stable, we incorporated it into the AGT.

The conversational agent is trained with the Wizard of OZ conversation data. This conversation is the starting point of the agent to learn to talk. The user asks the agent about the ambient environment. The agent talks back to the user with the necessary information. RASA collects that information from AWS API and the RL model. From this information about the ambient environment, the user decides to take necessary actions.

We observed some limitations of the assistive prototype system during the training and testing. One of those is that the Gazebo obstacles are a purely mathematical model. It means that the physics engine sees a tree as a box for collision though the tree has a particular shape. During testing, we found that this limitation did not matter much because the input to the RL was PC. Another limitation is that sometimes, the conversation takes a longer time. It could be dangerous in a situation where time is crucial, e.g., an oncoming car while crossing the road. Besides, the use of the WiFi network is another limitation. It could be solved by keeping the models and services all in the computing device, but that requires higher computing, storage, and battery capacity. As a trade-off, the WiFi is used. Also, the most critical obstacle, according to the participating volunteer, is the “slope”. Our assistive device can not detect slope.

References

Abdel-Hamid, O., Mohamed, A.-R., Jiang, H., Deng, L., Penn, G., & Yu, D. (2014). Convolutional neural networks for speech recognition. *IEEE/ACM Transactions on audio, speech, and language processing*, *22*(10), 1533–1545.

Administration, N. H. T. S., et al. (2017). *Traffic safety facts 2015 data–pedestrians. washington, dc: Us department of transportation, national highway traffic safety administration; 20175. publication no* (Tech. Rep.). DOT-HS-812-375. Available at https://crashstats.nhtsa.dot.gov/.

Ahmed, F., Mahmud, M. S., Al-Fahad, R., Alam, S., & Yeasin, M. (2018). Image captioning for ambient awareness on a sidewalk. In *2018 1st international conference on data intelligence and security (icdis)* (pp. 85–91).

Ahmed, F., & Yeasin, M. (2017). Optimization and evaluation of deep architectures for ambient awareness on a sidewalk. In *2017 international joint conference on neural networks (ijcnn)* (pp. 2692–2697).

Ahmetovic, D., Murata, M., Gleason, C., Brady, E., Takagi, H., Kitani, K., & Asakawa, C. (2017). Achieving practical and accurate indoor navigation for people with visual impairments. In *Proceedings of the 14th web for all conference on the future of accessible work* (pp. 31:1–31:10). New York, NY, USA: ACM. Retrieved from `http://doi.acm.org/10.1145/3058555.3058560` doi: 10.1145/3058555.3058560

Aladren, A., López-Nicolás, G., Puig, L., & Guerrero, J. J. (2016). Navigation assistance for the visually impaired using rgb-d sensor with range expansion. *IEEE Systems Journal*, *10*(3), 922–932.

Barry, A. J., Florence, P. R., & Tedrake, R. (2018). High-speed autonomous obstacle avoidance with pushbroom stereo. *Journal of Field Robotics*, *35*(1), 52–68.

Bellemare, M. G., Naddaf, Y., Veness, J., & Bowling, M. (2013). The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, *47*, 253–279.

Bengio, Y., Simard, P., Frasconi, P., et al. (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, *5*(2), 157–166.

Bigham, J. P., Jayant, C., Ji, H., Little, G., Miller, A., Miller, R. C., ... Yeh, T. (2010). Vizwiz: Nearly real-time answers to visual questions. In *Proceedings of the 23nd annual acm symposium on user interface software and technology* (pp. 333–342). New York, NY, USA: ACM. Retrieved from `http://doi.acm.org/10.1145/1866029.1866080` doi: 10.1145/1866029.1866080

Bocklisch, T., Faulkner, J., Pawlowski, N., & Nichol, A. (2017). Rasa: Open source language understanding and dialogue management. *arXiv preprint arXiv:1712.05181*.

Bohannon, R. W. (1997). Comfortable and maximum walking speed of adults aged 20—79 years: reference values and determinants. *Age and Ageing*, *26*(1), 15-19. Retrieved from `http://ageing.oxfordjournals.org/content/26/1/15.abstract` doi: 10.1093/ageing/26.1.15

Bourne, R. R., Flaxman, S. R., Braithwaite, T., Cicinelli, M. V., Das, A., Jonas, J. B., ... others (2017). Magnitude, temporal trends, and projections of the global prevalence of blindness and distance and near vision impairment: a systematic review and meta-analysis. *The Lancet Global Health*, *5*(9), e888–e897.

Bradley, N. A., & Dunlop, M. D. (2005). An experimental investigation into wayfinding directions for visually impaired people. *Personal and Ubiquitous Computing*, *9*(6), 395–403.

Bradski, G. (2000). The OpenCV Library. *Dr. Dobb's Journal of Software Tools*.

Britz, D. (2015, September). *Recurrent neural networks tutorial, part 1 – introduction to rnns.* Retrieved from http://www.wildml.com/2015/09/recurrent-neural-networks-tutorial-part-1-introduction-to-rnns/

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., & Zaremba, W. (2016). *Openai gym.*

Charles, R. Q., Su, H., Kaichun, M., & Guibas, L. J. (2017). Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Computer vision and pattern recognition (cvpr), 2017 ieee conference on* (pp. 77–85).

Chen, W., Wilson, J., Tyree, S., Weinberger, K., & Chen, Y. (2015). Compressing neural networks with the hashing trick. In *International conference on machine learning* (pp. 2285–2294).

Chen, W., & Zhang, T. (2017). An indoor mobile robot navigation technique using odometry and electronic compass. *International Journal of Advanced Robotic Systems*, *14*(3), 1729881417711643. Retrieved from https://doi.org/10.1177/1729881417711643 doi: 10.1177/1729881417711643

Cheng, Y., Felix, X. Y., Feris, R. S., Kumar, S., Choudhary, A., & Chang, S.-F. (2015). Fast neural networks with circulant projections. *arXiv preprint arXiv:1502.03436*, *2*.

Chollet, F., et al. (2015). *Keras.* https://keras.io.

Coates, A., Ng, A., & Lee, H. (2011). An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics* (pp. 215–223).

Cordts, M., Rehfeld, T., Schneider, L., Pfeiffer, D., Enzweiler, M., Roth, S., ... Franke, U. (2017, December). The stixel world. *Image Vision Comput.*, *68*(C), 40–52. Retrieved from https://doi.org/10.1016/j.imavis.2017.01.009 doi: 10.1016/j.imavis.2017.01.009

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009a). Imagenet: A large-scale hierarchical image database. In *Computer vision and pattern recognition, 2009. cvpr 2009. ieee conference on* (pp. 248–255).

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009b). ImageNet: A Large-Scale Hierarchical Image Database. In *Cvpr09*.

Denton, E. L., Zaremba, W., Bruna, J., LeCun, Y., & Fergus, R. (2014). Exploiting linear structure within convolutional networks for efficient evaluation. In *Advances in neural information processing systems* (pp. 1269–1277).

Di Fabbrizio, G., Dutton, D. L., Gupta, N. K., Hollister, B. B., Rahim, M. G., Riccardi, G., ... Schroeter, J. (2011, January 11). *Voice-enabled dialog system.* Google Patents. (US Patent 7,869,998)

Erickson, W., Lee, C., & von Schrader, S. (2008). disability status report: United states. *Ithaca, NY: Cornell University Rehabilitation Research and Training Center on Disability Demographics and Statistics*.

Escobar-Alvarez, H. D., Johnson, N., Hebble, T., Klingebiel, K., Quintero, S. A., Regenstein, J., & Browning, N. A. (2018). R-advance: Rapid adaptive prediction for vision-based autonomous navigation, control, and evasion. *Journal of Field Robotics*, *35*(1), 91–100.

Flores, G. H., & Manduchi, R. (2018). Weallwalk: An annotated dataset of inertial sensor time series from blind walkers. *ACM Transactions on Accessible Computing (TACCESS)*, *11*(1), 4.

Fricke, T. R., Tahhan, N., Resnikoff, S., Papas, E., Burnett, A., Ho, S. M., ... Naidoo, K. S. (2018). Global prevalence of presbyopia and vision impairment from uncorrected presbyopia: systematic review, meta-analysis, and modelling. *Ophthalmology*, *125*(10), 1492–1499.

Gong, Y., Liu, L., Yang, M., & Bourdev, L. (2014). Compressing deep convolutional networks using vector quantization. *arXiv preprint arXiv:1412.6115*.

Graves, A., Mohamed, A.-r., & Hinton, G. (2013). Speech recognition with deep recurrent neural networks. In *2013 ieee international conference on acoustics, speech and signal processing* (pp. 6645–6649).

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 770–778).

Helal, A., Moore, S. E., & Ramachandran, B. (2001). Drishti: An integrated navigation system for visually impaired and disabled. In *Proceedings fifth international symposium on wearable computers* (pp. 149–156).

High, R. (2012). The era of cognitive systems: An inside look at ibm watson and how it works. *IBM Corporation, Redbooks*.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9*(8), 1735–1780.

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., ... Adam, H. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*.

Iandola, F. N., Han, S., Moskewicz, M. W., Ashraf, K., Dally, W. J., & Keutzer, K. (2016). Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size. *arXiv preprint arXiv:1602.07360*.

Jesie, R. S. (2015). Advanced talking navigation cane for visually impaired using capacitive touch keypad. In *2015 international conference on circuits, power and computing technologies [iccpct-2015]* (pp. 1–5).

Juliani, A. (2016, August). *Simple reinforcement learning with tensorflow part 0: Q-learning with tables and neural networks.* Retrieved from `https://medium.com/emergent-future/simple-reinforcement-learning-with-tensorflow-part-0-q-learning-with-tables-and-neural-networks-d195264329d0`

Kane, S. K., Jayant, C., Wobbrock, J. O., & Ladner, R. E. (2009). Freedom to roam: a study of mobile device adoption and accessibility for people with visual and motor disabilities. In *Proceedings of the 11th international acm sigaccess conference on computers and accessibility* (pp. 115–122).

Karpathy, A. (2017). *Transfer learning.* Retrieved from `http://cs231n.github.io/transfer-learning/`

Kawas, S., Karalis, G., Wen, T., & Ladner, R. E. (2016). Improving real-time captioning experiences for deaf and hard of hearing students. In *Proceedings of the 18th international acm sigaccess conference on computers and accessibility* (pp. 15–23).

Kim, S., Choi, J., & Kim, Y. (2011). Determining the sidewalk pavement width by using pedestrian discomfort levels and movement characteristics. *KSCE Journal of Civil Engineering*, *15*(5), 883.

Koenig, N., & Howard, A. (2004). Design and use paradigms for gazebo, an open-source multi-robot simulator. In *2004 ieee/rsj international conference on intelligent robots and systems (iros)(ieee cat. no. 04ch37566)* (Vol. 3, pp. 2149–2154).

Laput, G., Lasecki, W. S., Wiese, J., Xiao, R., Bigham, J. P., & Harrison, C. (2015). Zensors: Adaptive, rapidly deployable, human-intelligent sensor feeds. In *Proceedings of the 33rd annual acm conference on human factors in computing systems* (pp. 1935–1944).

Lee, S. J., Moon, Y. S., Ko, N. Y., Choi, H.-T., & Lee, J.-M. (2017a). A method for object detection using point cloud measurement in the sea environment. In *2017 ieee underwater technology (ut)* (pp. 1–4).

Lee, S. J., Moon, Y. S., Ko, N. Y., Choi, H. T., & Lee, J. M. (2017b, February). A method for object detection using point cloud measurement in the sea environment. In *2017 IEEE Underwater Technology (UT)* (pp. 1–4). doi: 10.1109/UT.2017.7890290

Leung, T.-S., & Medioni, G. (2014). Visual navigation aid for the blind in dynamic environments. In *Proceedings of the ieee conference on computer vision and pattern recognition workshops* (pp. 565–572).

Li, Z., Shu, Y., Karlsson, B. F., Lin, Y., & Moscibroda, T. (2017). Demo: Towards flexible and scalable indoor navigation. In *Proceedings of the 23rd annual international conference on mobile computing and networking* (pp. 495–497). New York, NY, USA: ACM. Retrieved from `http://doi.acm.org/10.1145/3117811.3119854` doi: 10.1145/3117811.3119854

Liu, H.-H., & Yang, Y.-N. (2011). Wifi-based indoor positioning for multi-floor environment. In *Tencon 2011-2011 ieee region 10 conference* (pp. 597–601).

Liu, K.-C., Wu, C.-H., Tseng, S.-Y., & Tsai, Y.-T. (2015). Voice helper: A mobile assistive system for visually impaired persons. In *Computer and information technology; ubiquitous computing and communications; dependable, autonomic and secure computing; pervasive intelligence and computing (cit/iucc/dasc/picom), 2015 ieee international conference on* (p. 1400-1405). IEEE.

Mahmud, M. S., & Uddin, M. F. (2018). Mitigating unfairness problem in wlans caused by asymmetric co-channel interference. *International Journal of Mobile Communications*, *16*(3), 307–327.

Manduchi, R. (2012). Mobile vision as assistive technology for the blind: An experimental study. In *International conference on computers for handicapped persons* (pp. 9–16).

Mikolov, T., Karafiát, M., Burget, L., Černockỳ, J., & Khudanpur, S. (2010). Recurrent neural network based language model. In *Eleventh annual conference of the international speech communication association.*

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... others (2015). Human-level control through deep reinforcement learning. *Nature*, *518*(7540), 529.

Morales, N., Toledo, J., Acosta, L., & Sánchez-Medina, J. (2017, July). A Combined Voxel and Particle Filter-Based Approach for Fast Obstacle Detection and Tracking in Automotive Applications. *IEEE Transactions on Intelligent Transportation Systems*, *18*(7), 1824–1834. doi: 10.1109/TITS.2016.2616718

Novikov, A., Podoprikhin, D., Osokin, A., & Vetrov, D. P. (2015). Tensorizing neural networks. In *Advances in neural information processing systems* (pp. 442–450).

Rao, S., Prasad, A., Shetty, A., Hegde, R., Bhakthavathsalam, R., et al. (2012). Acoustic vision-acoustic perception based on real time video acquisition for navigation assistance. *International Journal of Computational Intelligence Techniques*, *3*(2), 102.

Rusu, R. B., & Cousins, S. (2011, May 9-13). 3D is here: Point Cloud Library (PCL). In *IEEE International Conference on Robotics and Automation (ICRA)*. Shanghai, China.

Seeclickfix. (n.d.). Retrieved from `http://seeclickfix.com/`

Shashua, A. (2016). *Orcam - see for yourself*. Retrieved from `http://www.orcam.com/`

Shinohara, K., Bennett, C. L., & Wobbrock, J. O. (2016). How designing for people with and without disabilities shapes student design thinking. In *Proceedings of the 18th international acm sigaccess conference on computers and accessibility* (pp. 229–237).

Shoval, S., Ulrich, I., & Borenstein, J. (2003). Navbelt and the guide-cane [obstacle-avoidance systems for the blind and visually impaired]. *IEEE robotics & automation magazine*, *10*(1), 9–20.

Shu, Y., & Karlsson, B. (2017). *Path guide: A new approach to indoor navigation*. Retrieved from `https://www.microsoft.com/en-us/research/blog/path-guide-new-approach-indoor-navigation/`

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Sims, G., & Dunigan, E. (1984). Diurnal and seasonal variations in nitrogenase activity (c2h2 reduction) of rice roots. *Soil biology and biochemistry*, *16*(1), 15–18.

Singh, V., Paul, R., Mehra, D., Gupta, A., Sharma, V. D., Jain, S., ... others (2010). 'smart'cane for the visually impaired: Design and controlled field testing of an affordable obstacle detection system. In *Transed 2010: 12th international conference on mobility and*

transport for elderly and disabled personshong kong society for rehabilitations k yee medical foundationtransportation research board.

Solin, A., Cortes, S., Rahtu, E., & Kannala, J. (2017). Pivo: Probabilistic inertial-visual odometry for occlusion-robust navigation. *arXiv preprint arXiv:1708.00894*.

Sutton, R. S., & Barto, A. G. (1998). *Introduction to reinforcement learning* (1st ed.). Cambridge, MA, USA: MIT Press.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 1–9).

Szegedy, C., Toshev, A., & Erhan, D. (2013). Deep neural networks for object detection. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, & K. Q. Weinberger (Eds.), *Advances in neural information processing systems 26* (pp. 2553–2561). Curran Associates, Inc. Retrieved from `http://papers.nips.cc/paper/5207-deep-neural-networks-for-object-detection.pdf`

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 2818–2826).

Torrey, L., & Shavlik, J. (2010). Transfer learning. In *Handbook of research on machine learning applications and trends: Algorithms, methods, and techniques* (pp. 242–264). IGI Global.

Ulrich, I., & Borenstein, J. (2001). The guidecane-applying mobile robot technologies to assist the visually impaired. *IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans*, *31*(2), 131–136.

Wahab, M. H. A., Talib, A. A., Kadir, H. A., Johari, A., Noraziah, A., Sidek, R. M., & Mutalib, A. A. (2011). Smart cane: Assistive cane for visually-impaired people. *arXiv preprint arXiv:1110.5156*.

Walkways, sidewalks, and public spaces. (n.d.). Retrieved from `https://safety.fhwa.dot.gov/ped_bike/univcourse/pdf/swless13.pdf`

Wang, J., Yang, K., Hu, W., & Wang, K. (2018). An environmental perception and navigational assistance system for visually impaired persons based on semantic stixels and sound interaction. In *2018 ieee international conference on systems, man, and cybernetics (smc)* (pp. 1921–1926).

Wang, T., Wu, D. J., Coates, A., & Ng, A. Y. (2012). End-to-end text recognition with convolutional neural networks. In *Pattern recognition (icpr), 2012 21st international conference on* (p. 3304-3308). IEEE.

Washington, M. (2016). *An introduction to the microsoft bot framework: Create facebook and skype chatbots using microsoft visual studio and c#* (1st ed.). USA: CreateSpace Independent Publishing Platform.

Wilson, G., & Brewster, S. A. (2015). Using dynamic audio feedback to support peripersonal reaching in visually impaired people. In *Proceedings of the 17th international acm sigaccess conference on computers & accessibility* (pp. 433–434).

Wirz-Justice, A. (2008). Diurnal variation of depressive symptoms. *Dialogues in clinical neuroscience*, *10*(3), 337.

Wu, W., Au, L., Jordan, B., Stathopoulos, T., Batalin, M., Kaiser, W., ... Chodosh, J. (2008). The smartcane system: an assistive device for geriatrics. In *Proceedings of the icst 3rd international conference on body area networks* (p. 2).

Yang, C., & Shao, H.-R. (2015). Wifi-based indoor positioning. *IEEE Communications Magazine*, *53*(3), 150–157.

Yang, K., Wang, K., Lin, S., Bai, J., Bergasa, L. M., & Arroyo, R. (2018). Long-range traversability awareness and low-lying obstacle negotiation with realsense for the visually impaired. In *Proceedings of the 2018 international conference on information science and system* (pp. 137–141).

Yin, Z., Wu, C., Yang, Z., & Liu, Y. (2017). Peer-to-peer indoor navigation using smartphones. *IEEE Journal on Selected Areas in Communications*, *35*(5), 1141–1153.

Yosinski, J., Clune, J., Bengio, Y., & Lipson, H. (2014). How transferable are features in deep neural networks? In *Advances in neural information processing systems* (pp. 3320–3328).

Zheng, Y., Shen, G., Li, L., Zhao, C., Li, M., & Zhao, F. (2017). Travi-navi: Self-deployable indoor navigation system. *IEEE/ACM Transactions on Networking*, *25*(5), 2655–2669.

Zhong, Y., Garrigues, P. J., & Bigham, J. P. (2013). Real time object scanning using a mobile phone and cloud-based visual search engine. In *Proceedings of the 15th international acm sigaccess conference on computers and accessibility* (p. 20).

Zhou, C., Li, F., Cao, W., Wang, C., & Wu, Y. (2018). Design and implementation of a novel obstacle avoidance scheme based on combination of cnn-based deep learning method and lidar-based image processing approach. *Journal of Intelligent & Fuzzy Systems*(Preprint), 1–11.