To Veer or Not to Veer: Learning from Experts How to Stay Within the Crosswalk

Manfred Diaz *
Concordia University †
Montreal, Canada
mdiaz@cim.mcgill.ca

Roger Girgis *
McGill University
Montreal, Canada
rogerg@cim.mcgill.ca

Thomas Fevens
Concordia University
Montreal, Canada
fevens@cse.concordia.ca

Jeremy Cooperstock
McGill University
Montreal, Canada
jer@cim.mcgill.ca

Abstract

One of the many challenges faced by visually impaired (VI) individuals is the crossing of intersections while remaining within the crosswalk. We present a Learning from Demonstration (LfD) approach to tackle this problem and provide VI users with an assistive agent. Contrary to previous methods, our solution does not presume the existence of particular features in crosswalks. The application of the LfD framework helped us transfer sighted individuals' abilities to the intelligent assistive agent. Our proposed approach started from a collection of 215 demonstrative videos of intersection crossings executed by sighted individuals ("the experts"). We labeled the video frames to gather the experts' recommended actions, and then applied a policy derivation technique to extract the optimal behavior using state-of-the-art Convolutional Neural Networks. Finally, to assess the feasibility of such a solution, we evaluated the performance of the trained agent in predicting expert actions.

1. Introduction

Independent navigation of a city is a significantly challenging task for individuals with visual impairment. This challenge is further exacerbated when the environment they are navigating is unknown. For this reason, they tend to remain in known environments [25], as they learn the intersection characteristics of those routes. Some of the problems they face include determining whether the intersection is one- or two-way, orienting to the correct direction for crossing, obtaining the status of the pedestrian signal, and detecting veering during the crossing phase. With regard to the last of these, mobility training focuses on techniques to keep the individual walking as straight as possible while maintaining a safe distance from parallel traffic, i.e., remaining within the marked lines designating pedestrian crossings. Unfortunately, even after training, detection of veering remains difficult [17]. It has been noted that regardless of visual impairment, in the absence of environmental cues, humans tend to walk in circles [43], with diameters as small as 20 meters. This has obvious implications to crossing at intersections, which can be of similar length.

Various assistive devices are available to help the visually impaired explore a city, including talking GPS systems, and those providing information about points of interest around the user [30]. However, these do not solve the problem of safe crossing at intersections, which is generally agreed to be the most difficult and risky aspects of independent travel for visually impaired individuals [42].

Accessible pedestrian signal (APS) systems provide indications of *when* it is safe to cross [40], and in certain cases, offer auditory cues that help the user determine orientation. Unfortunately, these auditory cues are often masked by background noise. More problematically, due to their high cost, estimated at over $25k per new installation, and approximately $8k at intersections with existing poles [28], APS deployment remains limited. For example, according to the Montreal Association for the Blind, the city of Montreal, Canada, with 1875 intersections [14], reportedly has only 133 installed APS systems [1].

A potential alternative, explored by several research efforts, considers the use of embedded sensors, such as ac-

[1]Equally contributed to this work.

[2]Research conducted while visiting Shared Reality Lab at the Centre for Intelligent Machines, McGill University.

celerometers or gyroscopes found in typical smartphones [31, 37, 17] to provide the feedback necessary to prevent veering. However, sensor instability and the potential need for frequent re-calibration pose obstacles to such efforts. Furthermore, while these solutions may reduce veering behavior, they do not help with the initial alignment of the user in the correct direction at the start of crossing.

Relying instead on visual information provided by the smartphone camera represents an attractive alternative. This is especially the case considering that non-visual understanding of the environment is not only less effective and efficient, but also potentially dangerous, compared to scanning the surrounding using vision [6]. However, processing of the wide variety of street scenes to extract the appropriate features, if present, needed for such guidance has long been a daunting challenge. Fortunately, the recent explosion of capability of deep learning systems offers a potential solution. In particular, the convolutional neural network (CNN) architecture has been shown to outperform all other methods for image recognition and classification tasks [18, 12, 19, 21, 26]. Previous work [46] has shown that the features these CNN models learn can be transfered to tackle a different problem. In this paper, we combine such pre-trained models with Learning from Demonstrations [20] techniques to provide real-time feedback to visually impaired individuals before and during the crossing of an intersection, helping both initial alignment and maintenance of a straight path.

2. Related Work

Many systems have been developed in an attempt to tackle the veering and intersection crossing problem encountered by the visually impaired community. While some of these systems are commercially available, many systems remain limited to an academic setting, and are still either in the experimental phase or would be too expensive for widespread commercial deployment. These systems typically lie within two main categories of systems: sensor-based and vision-based systems. In this Section, we present some of these systems and how they motivated the approach presented in this work.

2.1. Sensor-Based Systems

The first category of systems focuses on employing sensors typically mounted on the user. One such orientation and way-finding interface system was proposed in [37] where they explore three different interfaces. The system is comprised of a computer placed in a backpack, to be worn by the user, with an array of speakers placed against the back used to vibrate the direction to follow, a digital compass mounted either on the shoulder or in a hat, and a pair of ear buds mounted on the hat providing stereo audio beeping. The authors found a 31% significant improvement in veering performance when compared to the baseline veer. It is important to note that the authors initially attempted to augment the orientation signal from the digital compass by installing a pedestrian signal system at test intersections which would communicate with the backpack computer. However, this could not be accomplished due to state laws and difficulties with the installation and maintenance of such a system. This further demonstrates the difficulty that one would face in deploying such a system at intersections in a given city. Another drawback with this system is the need to recalibrate the digital compass after every intersection. This makes such an application highly impractical for the intended user group. Finally, the system also assumes that the user is properly oriented at the onset of crossing.

Guth [17] proposes the *Anti-Veering Training Device (AVTD)* which employs a solid state gyroscope to measure the user's cumulative rotation as they walk along a path. The gyroscope also provides tilt and temperature compensation adding robustness to the system. The user is presented with veering correction speech cues and feedback about performance. However, it is not apparent how the system's effectiveness and accuracy were evaluated. Paneels et al. [31] build on this work with their *Walking Straight* application which also uses the gyroscope to measure body sway and orientation. This work also focuses on the feedback modality based on typical mobility training for the blind. The experiment consisted of walking in a straight line towards a 15 m target after initially being positioned in the correct orientation. It was conducted in a controlled outdoor environment, and not at an actual intersection. They find that the system reduced veering to half that encountered during the control condition. Another important result from their experiment was that the most effective method for providing veering feedback was a continuous beep rendered in the ear opposite to the veering direction. As such, the current proposed application uses a continuous beep stimulus.

While these systems can be effective in an ideal setting, sensor stability can prove problematic when the system is continuously used. This is due to the need of recalibrating the sensors, making such systems risky in the intersection crossing task. Additionally, these systems assume that the user was initially properly oriented. However, this can be a difficult task when taking into account the complex sound environment at typical intersections, as discussed in [6]. Moreover, neither of these systems has the capability of providing information regarding the pedestrian signalization status.

2.2. Vision-Based Systems

In recent years, many systems have been proposed that instead employ computer vision techniques. Shen et al. [42] developed a prototype on a Nokia 6681 mobile phone,

utilizing the camera for detection. The system detected zebra-crossings using segmentation of the edges of strips in the pattern. However, the authors explain that the algorithm performed worse than an earlier version that was implemented using a desktop computer and higher resolution camera. With the advances in mobile devices, Ahmetovic et al. build on the work in [42] by building a two-part system comprising of a *Zebralocalizer* [4] and *ZebraRecognizer* [3]. The former is used to interface between the user and an iOS application while the latter uses a 5-step process, computing the position of the zebra-crossing by using a combination of the camera and the device accelerometer as inputs. Furthermore, they transform the problem into three stages: an approaching stage, an aligning stage and a crossing stage. The user detects and crosses an intersection by holding the mobile phone (an iPhone 4) parallel to the ground, with the camera "looking" for the zebra-crossing. The results were positive, with all the subjects successfully capable of crossing a 6-meter road in an average three to five seconds.

One of the major limitations of these systems is the need for a zebra-crossing pattern, as its presence is infrequent in many cities. As a result, users would still lack the ability to fully and autonomously navigate through an unknown area. Later work by Ahmetovic et al. [5] has been done to address the *zebracrossing* scarcity. In this work, the researchers designed a system that mines existing image databases (e.g. Google Street View images) to plan a route that ensures all intersections have *zebracrossings*. While this method offers an elegant solution, it still doesn't provide users with an independent experience. Another limitation of such a system is its inability to deal with occlusions. As a result, these can cause users to move in wrong directions leading to potentially dangerous situations.

Ivanchenko et al. [22] proposed a system that detects the more common two-stripe crosswalk instead of zebra-crossings, removing the reliance on this pattern. The authors develop the *Crosswatch* application running on a Nokia N95 mobile phone. Additionally, they use accelerometer readings to estimate the direction of gravity, making it easier to position the camera in the correct orientation. In addition, they utilize a 3D analysis technique as an attempt to ensure the subject remains within the two-stripes. To perform this analysis, they use the focal length of the camera lens and estimate an average height of 1.5-m for adults. Finally, the system used high-pitched tones to inform the user if their feet were inside the two-lane corridor. The preliminary experiments required that the blind user correctly identify the location of a crosswalk. However, the experiments did not include task of crossing the intersection and, therefore, does not allow for evaluation in effectiveness. Additionally, it not clear how such a system would handle partially or fully occluded stripes.

Moreover, Poggi et al. [34, 33] proposed the use of a pocket-sized device with and embedded CPU, coupled with a custom RGBD camera attached to wearable glasses. This device uses the dense disparity map from the RGBD camera to determine the ground plane, which serves as a reliable way to discriminate between the ground and the rest. Furthermore, they train a CNN model, similar to [26], which takes as input a wrapped image of the ground and, if a crosswalk is present, determines its orientation. The authors report a near-perfect accuracy on their test set, testimonial to the power of these models. However, it is important to note that this system uses custom hardware (e.g. custom RGBD camera) designed by the authors. It is not clear how it would be deployed for the general public in an efficient and cost-effective manner. In addition, the authors only report testing results on a computer in an off-line setting. Thus, it is not clear how this system would perform in a real world experiment with blind participants. Finally, this system was designed for the initial orientation part of the intersection crossing task. We are not aware if it can easily be adapted to provide users with real-time veering feedback.

Both of these types of systems rely on the presence of a zebra-crossing or a two-stripe crosswalk. However, as we have experienced through our investigation of the problem, zebra-crossings are not always present at intersections and the lines in the two-lane corridors are, in many cases, faded or obscured. In such intersections, these systems would be incapable of assisting users in their everyday travels. The system proposed in the present work does not depend on any particular structure at intersection crosswalks as it utilizes recent advances in machine learning to detect veering problems.

3. Street Crossing from Demonstrations

Despite the recent surge of work in intelligent robotics, to our knowledge, the results from this research have scarcely been applied to alleviate sensorial, motor and cognitive impairments in humans [7]. We believe that such research, in particular, the technique of Learning from Demonstration, is well suited to addressing the problem of veering during street crossing.

3.1. Motivation

Learning from Demonstration (LfD) is based on the idea of transferring human behavior to intelligent agents [20] [8]. An agent's policy is a function $\pi : \mathcal{S} \rightarrow \mathcal{A}$ that maps every state $s \in \mathcal{S}$ to an action $a \in \mathcal{A}$. Conceptually, algorithms in the LfD domain aim to acquire the optimal policy π^* for a task from a series of demonstrations $\mathcal{D} = \{d_1, d_2, ..., d_m\}$ that can guide an agent while autonomously performing the task. Following this methodology, we can gather the knowledge of sighted "experts" on the intersection-crossing task,

and transfer these to an intelligent assistive agent for the visually impaired.

To completely formulate an LfD solution, one must establish the structure of the world states $s \in \mathcal{S}$ that the agent may reach, the actions $a \in \mathcal{A}$ that the agent is capable of performing, and a transition function $\mathcal{T}(s'|s,a)$ that expresses the probability of landing in $s' \in \mathcal{S}$ given that the agent executes action a from state s. In most real-life scenarios, the state is not fully observable. Most LfD models handle this uncertainty by relying on the agent's observations of the world, $z \in \mathcal{Z}$, instead of the complete representation of its internal structure [8]. Therefore, our LfD method must determine the optimal policy, $\pi^* : \mathcal{Z} \rightarrow \mathcal{A}$ using demonstrations $d_i = (z_i, a_i) \in \mathcal{D} : \mathcal{Z} \times \mathcal{A}$. We now describe the application of this approach to the street crossing domain.

3.2. Action Space Design

In LfD, a transition $t \in \mathcal{T}$ between states occurs when an agent executes the actions specified by its policy. We choose to discretize the space of possible actions by dividing the agent's field of view into 12 evenly spaced vertical bins as presented in Figure 1, following a similar approach taken in previous research [38, 10, 23]. Each bin, $v \in \mathcal{V}, \mathcal{V} = \{v_1, v_2, ..., v_{12}\}$, is an action in $\mathcal{A}$ an expert would recommend to execute given an observed state in the street crossing task. The bins are intended to capture the heading of the goal relative to the expert's field of view, with bin v_1 corresponding to the agent having to veer maximally to the left, and bin v_{12} representing having to veer maximally to the right.

For situations where an expert could not identify the bin including the goal, for example, in the scenario shown in Figure 2a, our problem model also included an action $unknown \in \mathcal{A}$. As we will discuss in Section 4, this representation allowed us to experiment with different levels of granularity for the action space.

3.3. Task Demonstrations

Once the problem design space was defined, the structure of the demonstration set had to be specified. We divided our collection of demonstrations into two steps: *(i)* demonstrations acquisition and *(ii)* expert's knowledge extraction.

Each demonstrator was asked to stand at the corner of an intersection, holding a smartphone at chest level, and capture, from a first-person perspective, the sequence of actions required to cross the intersection. The motivation for this particular position of the smartphone is the outcome of previous experiments carried out with visually impaired users [30, 31]. As our interpretation of the street crossing task also included an initial orientation phase to the correct direction towards the goal, demonstrators were asked to record the procedure of rotating within a range of $\pm 45^o$ about the appropriate heading from the starting corner to the goal corner.

Furthermore, as suggested by previous work [35, 45, 38], the high sensitivity of LfD techniques to the quality of demonstrations greatly impacts their generalization ability. A comprehensive set of samples $(z, a) \in D$ should capture not only the optimal behavior of the task, but also states that could only be reachable by some suboptimal action sequence. To ensure that this was the case, the demonstrators were asked to include suboptimal behaviors in their crossings, along with the corresponding corrective actions.

Our demonstrators recorded 215 videos of approximately 25 s each from street intersections in downtown Montreal, Canada, registering the sequence of states transitioned by sighted individuals performing the task. As a compromise between data quantity and a desire to minimize redundancy of frames at a high framerate of 30 frames per seconds (fps), we extracted frames from the collected videos at a rate of 2 fps, which resulted in a total of 8125 observations.

Figure 1: Action space discretization into vertical bins $\mathcal{V} = \{v_1, .., v_{12}\}$ from left to right.

3.4. Experts' Knowledge Extraction

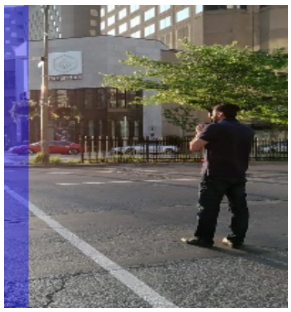
As our method did not incorporate a technique to capture the demonstrators' actions on-site, we relied on three experts' knowledge to extract optimal behavior from those observations, in a post-demonstrations procedure. For this, each expert was presented with frames randomly sampled from the observations, in a structure similar to the one depicted on Figure 1. They were then asked to select the bin $v \in \mathcal{V}$ that contained the position of the goal.

To ensure some resiliency to occlusions in the derived policy, we instructed the experts to choose the bin closest to the presumed goal position in scenarios in which the goal was occluded or otherwise not visible, provided that its location could be assumed (e.g., Figure 2b). We expected that under most conditions, a sighted individual could quickly estimate the relative orientation towards the goal from a single observation. For those exceptional cases where it was

not possible to infer the target position, the experts were asked to assign $unknown$ as the recommended action (e.g., Figure 2a).

By virtue of symmetry, we were able to mirror each image around its central vertical axis and associate the flipped image with the corresponding inverse action (i.e., swapping left-to-right with right-to-left). This allowed us to create a set of synthetic observations which, combined with the demonstration examples gathered, doubled the size of $\mathcal{D}$ and ensured a balance between the states explored and the optimal behavior observed.

(a) total occlusion of the environment



(b) optimal actions inferred even if the goal is not visible.

Figure 2: Examples of demonstrations' frames including corner cases in our dataset.

3.5. Policy Derivation Technique

The literature on LfD suggests the existence of three categories of policy derivation methods: direct learning, indirect learning, and execution plans, only differentiated by how much understanding of the environment each algorithm requires while inferring a policy [20, 8]. The algorithms contained in the Direct Learning category are mostly independent of beliefs about the internal state of the environment, thus easier to implement. Then, the family of direct policy learning algorithms was our preference to solve the street crossing veering problem.

Based on the discretization of our actions space and the reduction of the observations to features, we choose to implement our policy extraction strategy as an image classification problem. A classification problem is one where a classifier $c(x) : X \rightarrow Y$ is used to predict the class y of an instance x, having $y \in Y$, $Y = \{y_1, y_2, ..., y_m\}$ a discrete set of classes. Usually, $x \in X$ is a vector $\vec{f} = \{f_1, f_2, ..f_n\}$ of features that reduce the dimensionality of the samples in X. In a supervised learning setting, the classifier is trained using a dataset $\mathcal{N}$ of samples in the form $(\vec{f_i}, y_i)$. Thus, we established the equivalence: $\mathcal{D} \equiv \mathcal{N}, \mathcal{Z} \equiv X, Y \equiv \mathcal{A}$ where the classifier $c(x) : \mathcal{Z} \rightarrow \mathcal{A}$, a CNN model, was trained to infer our policy π^* directly from samples on $\mathcal{D}$.

3.5.1 CNN for Classification Tasks

As an image usually contains irrelevant and redundant information for the resolution of visual tasks, it is better to deal with a condensed representation of such knowledge. Computer Vision techniques often rely on the extraction of salient attributes as a way to minimize the dimensionality of the information contained in an image. Manual extraction of those features requires a comprehensive understanding of the environment and the task at hand. The appearance of Convolutional Neural Networks (CNN) has come to alleviate this need while achieving human-level performance on computer-assisted visual tasks.

Notably, CNN architectures have eliminated the prerequisite of hand-crafted feature extraction algorithms by learning the required features and the task at hand, simultaneously [9]. Since ImageNet Large Scale Visual Recognition Challenge 2012 [39], CNN have obtained state of the art results [24] on benchmark datasets in image classification, segmentation or object detection like ImageNet or PASCAL Visual Object Classes Challenge (VOC) [13].

Yosinksi et al. [47] analyzed why CNN has performed remarkably well on visual tasks and concluded that the way convolutional filters are organized explains this success in part. In a CNN, each convolutional filter learns to search for specific patterns in an image. Filters on first layers of these models learn to detect low-level characteristics (e.g., edges), while filters in deeper layers are fine-tuned to compose the low-level patterns into high-level features (e.g., the shape of a flower), according to a hierarchical structure. Therefore, we used CNN architectures to convert our z component of the demonstrations $d_i = (z_i, a_i)$ to a vector $z : \vec{f} = \{f_1, f_2, ..., f_n\}$ of features and to map these features into our discrete action space $\mathcal{A}$, thus generating an optimal policy π^*.

3.5.2 Transfer Learning

Training a CNN for classification using randomly initialized filters, or even with traditional heuristics [16], is usually a challenging and time-consuming task as the space of the models' hyper-parameters has to be explored. Moreover, our dataset had significantly fewer instances that the ImageNet dataset (8725 vs. 1.2 million instances) and the dimensionality of the classification task is significantly lower (13 vs. 1000 classes). Consequently, the direct application of models designed for ImageNet could lead to overfitting our dataset and to the loss of generality on the predicted actions.

In this regard, the notion of transfer learning helped us to overcome those obstacles. The theory of transfer learning establishes that the knowledge on a source problem space

$\mathcal{P}_s$ of a learned task $\mathcal{T}_s$ could help improve the learning of a target task $\mathcal{T}_t$ on a target problem space $\mathcal{P}_t$. How much knowledge is transferable from one domain/task to the another is directly associated with the amount of overlap between the problem areas in both [29].

Therefore, there exists a proven transferability property between features of a CNN trained on different visual tasks [46]. Although the overlapping between our demonstrations and the training samples on the ImageNet dataset is not clear, we still relied on models pre-trained on the latter as a starting point for fine-tuning different classifiers. Consequently, the high-level features of our problem were built upon the low-level features in the pre-trained models by retraining the appropriate deeper layers in each model.

Interestingly, the derivation of policies with supervised learning has presented some weakness in the past when the independence and identical distribution of the samples collected on $\mathcal{D}$ cannot be guaranteed (i.i.d principle)[38]. To guarantee such independence, each frame and the corresponding expert's action was considered a self-contained demonstration. Recent applications of LfD and CNN to navigation problems in robotics [10, 15, 23] have disregarded the sequential interpretation of a go-to-goal process thus inferring a stationary (time independent) policy. Moreover, the observations presented to the experts for labeling were randomized, ensuring their action (class) recommendation was independent of a sequential analysis of the frames.

4. Results and Discussion

4.1. Training the Agent

The accuracy of CNN models has significantly improved in recent years relative to their computational complexity [11]. However, state of the art results remain dependent on models relying on high-performance hardware, especially Graphics Processing Units (GPUs) to carry out their inference within adequate time constraints for real-life or real-time applications. Recent work [21, 19, 36, 32] has explored CNN architectures that aim to achieve a balance between the human-level accuracy results of their predecessors and the prediction time, thus making the application of deep learning techniques to a real-time problem, such as street crossing, feasible.

In this work, we experimented with four state of the art CNN architectures. Firstly, Resnet50 [18] and Xception [12] have a reported top-5 accuracy over 90% on the ImageNet dataset. This motivated our exploration of their potential as policy extractors. Moreover, we were curious to investigate the performance of network models that have been designed specifically to achieve a balance between classification accuracy and training/inference time. Thus, we selected Squeezenet [21] and Mobilenet [19] as our testbed for a mobile deployable solution.

Our transfer learning approach was based on the fine-tuning of each model by removing the latest layers, containing high-level features, and training our custom structure from scratch. In the cases of Xception, Mobilenet and Squeezenet, after removing those high-level-feature layers from each model, we added a $3 \times 3 \times 32$ convolutional layer, followed by a $1 \times 1 \times |\mathcal{A}|$ convolutional layer, both activated with ReLUs [27]. Finally, we added a softmax activation layer with a size of $|\mathcal{A}|$. Because of the particular structure of residual networks [18], we could only add to Resnet50 an extra fully connected layer converging to the number of actions and, similarly to the models above, this layer was followed by a softmax activation layer.

After introducing these modifications, we fine-tuned the models, while holding the pre-trained layers constant, and only trained the final layers we added. Each model was trained with a small learning rate (0.0002), using the RMSprop optimizer [44] ($\rho = 0.9, \epsilon = 1 \times 10^{-8}, \delta = 0.0$) and a categorical cross-entropy loss. The values of these hyper-parameters were selected empirically. With this configuration, we aimed to ensure the stability of the pre-trained values of each model.

We then experimented with reducing the dimensionality of the action space. Starting from the arrangement of 12 bins, we generated the following three configurations:

- 4-actions space: $\mathcal{V}_1$, by combining $\{v_1, ..., v_4\}$, $\{v_5, ..., v_8\}$ and $\{v_9, ..., v_{12}\}$ into $\{v_{\text{left}}, v_{\text{straight}}, v_{\text{right}}\}$ respectively, plus the $unknown$ action, as shown in 3a.

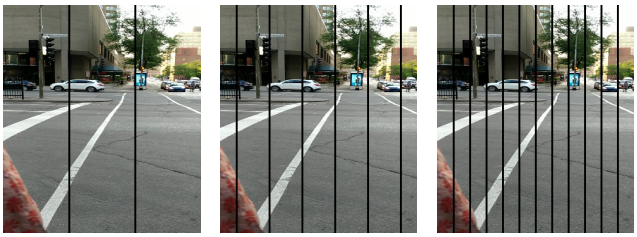
- 8-actions space: $\mathcal{V}_2$, by combining $\{v_2, v_3\}$, $\{v_4, v_5\}$,..., $\{v_{10}, v_{11}\}$, reserving bins $\{v_1\}$ and $\{v_{12}\}$ for those situations when the goal is not visible but its position can be inferred, as shown in Figure 3b

- 13-actions space: $\mathcal{V}_3$, retaining the full configuration of as shown in Figure 3c

For each of these configurations, we modified the associated Softmax layer to accord with the sizes of $\mathcal{A}_1 = \mathcal{V}_1, \mathcal{A}_2 = \mathcal{V}_2, \mathcal{A}_3 = \mathcal{V}_3$, and added $v_0 = unknown$. We then trained the CNN classifiers and evaluated their performance.

4.2. Testing the Agent

To evaluate the generalization of the learned policy, we created a second demonstration dataset from different intersections that were not included in the training set. Following the procedures described in Section 3, a supplementary collection of 51 videos was acquired, resulting in a new set $\mathcal{O} : \mathcal{Z}_o \times \mathcal{A}_o$ of 1170 observations. The optimal action for those samples was crowd-sourced to another ten experts



(a) 4-actions configuration (b) 8-actions configuration (c) 13-actions configuration

Figure 3: All actions-space configurations experimented while training and testing the agent.

who labeled each sample at least five times. The conditions described for labeling the initial set $\mathcal{D}$ were also followed here.

Table 1 presents the accuracy of the derived policy, applied over the observations on $\mathcal{O}$. These results are computed based on the best-predicted action of the classifier compared to the action that received the most votes from our experts. However, we note that best-action accuracy metrics are not meaningfully indicative of the model's actual performance on a practical task. Instead, Table 2 presents the mean absolute error in the agent's predicted action, a measurement computed by taking the absolute difference between the index of the action inferred by the policy from an *observation* and the index of the winning vote from the experts. Considering that our distribution of the action space is dependent on the spatial arrangement of the bins, we excluded the results of the action *unknown* in this calculation.

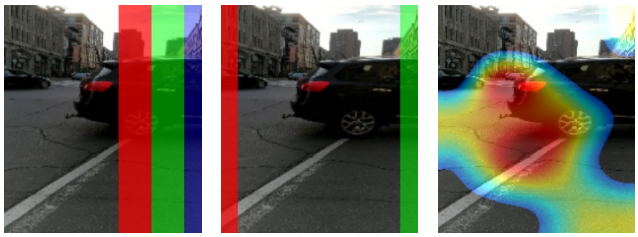
Model	4-Action	8-Action	13-Action
ResNet-50	0.746	0.635	0.503
Xception	0.822	0.615	0.526
Squeezenet	0.775	0.483	0.393
Mobilenet	0.822	0.599	0.467

Table 1: Accuracy of each model in predicting the correct action, compared to the experts' optimal action.

Model	4-Action	8-Action	13-Action
ResNet-50	0.27 ± 0.03	0.61 ± 0.07	1.14 ± 0.12
Xception	0.20 ± 0.03	0.59 ± 0.07	1.05 ± 0.12
Squeezenet	0.26 ± 0.03	0.83 ± 0.07	1.37 ± 0.12
Mobilenet	0.20 ± 0.03	0.71 ± 0.08	1.24 ± 0.12

Table 2: Each model's mean absolute difference between predicted action and the experts' optimal action, presented with the corresponding 95% confidence margin.

As can be seen, relying solely on the accuracy metric would suggest that the agent exhibits poor performance. However, given the mean absolute error reported—typically within a difference of a single bin—the average performance of the system is actually satisfactory across all model types and action space configurations. This can be verified by analysis of the confusion matrix for each action-space configuration. One can observe in Figure 5 a strong tendency around the diagonal in all four models, indicating that errors in the agent's prediction are most often the result of confusion with an adjacent, i.e., very similar, action. Thus, a mean absolute error metric is more appropriate than a simple correctness percentage score to characterize the performance of the model. Although we only present here the 8-actions configuration, similar behavior was exhibited for the other action-spaces tested.



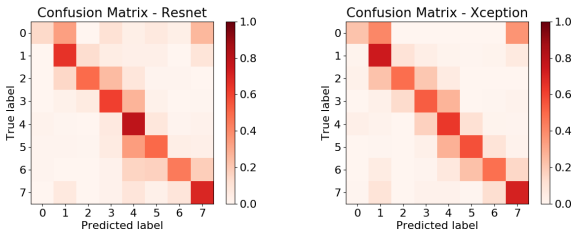
(a) experts' prediction (b) agent's prediction (c) activation map

Figure 4: Mobilenet top-3 predictions ($blue, green, red$) vs. experts' predictions on the 8-action-space. A missing bin corresponds to $unknown$. (c) shows the CNN activation maps [41].

It is also interesting to note that for situations where the experts' optimal action was $unknown$ (i.e., the correct label is 0), the agent would most often confuse it with the extreme veering conditions (i.e., actions 1 and 7). This suggests that when the expert is unsure of the required action, the agent's predictions recommend rotation. We suspect that this behavior is related to the way experts chose the optimal action in the training demonstrations; when the goal was not seen, the expert would choose the edge column that they guessed was the best direction to which one should rotate. Figures 4 and 5 make it evident that the agent has also learned this behavior. Although some perfect agreements between the policy and the expert's judgment are represented in the first row of Figure 6, there are still scenarios in which the goal is occluded and the policy is not capable of inferring the correct behavior, as shown in the third row of Figure 6.

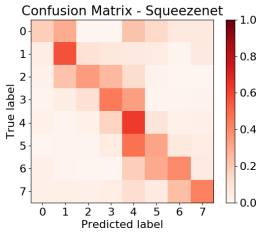
4.3. Mobile Prototype

To evaluate the potential of our solution, we developed a prototype application, initially for the Android platform, presuming our users would possess nothing more than a smartphone and bone-conduction headphones as an aid to

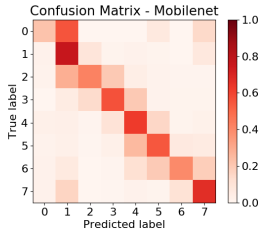


(a) Resnet50

(b) Xception



(c) Squeezenet



(d) Mobilenet

Figure 5: Confusion Matrix for each model trained on the 8-action-space configuration.

complete the task. While designing this prototype, we decided not to pursue an evaluation of factors such as energy efficiency, traffic data, or inference times. Instead, we implemented a mobile-only approach to gauge the feasibility of such an implementation empirically.

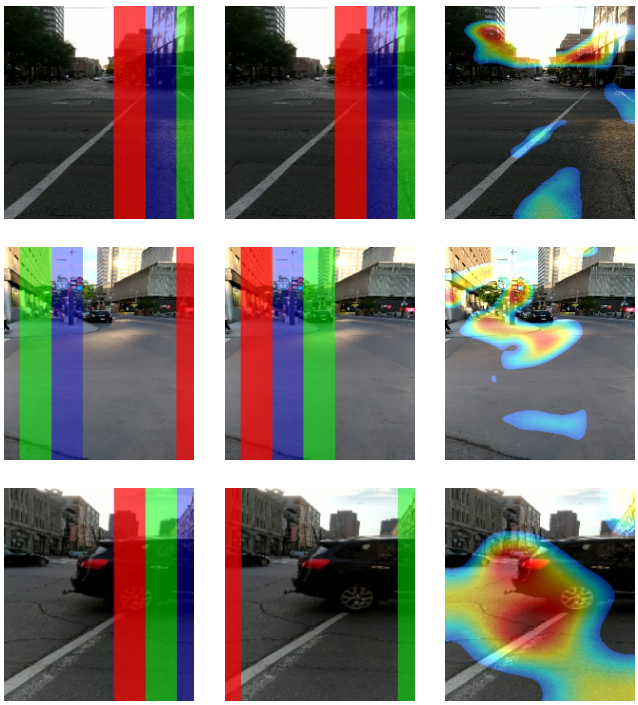
Our prototype performs three high-level tasks. First, using the front-facing camera of the smartphone, the system captures and pre-processes video frames. Next, a batch of these frames is fed into the pre-trained network for inference, based on Google's Tensorflow API for Android [2], and the results are collected. The relative position of the goal is extracted from the policy contained in the model, and converted to an angle from the subdivision of the 90^o region in front of the user.

Based on those results, a stereo panning audio signal is rendered, representing a beacon the user should follow to reach the goal. This approach of rendering is intended to mimic the general assistive behavior of the Accessible Pedestrian Signal. Testing of this system with human subjects is expected to begin shortly.

5. Conclusion and Future Work

This paper presented a basis for the construction of an intelligent assistive agent with the aim of helping the visually impaired with safely crossing road intersections. As discussed in Section 4, the LfD method implemented has effectively eliminated the need of extracting specific features, such as zebra stripes, from the environment (e.g. second row of Figure 6). Instead, it uses the advances in deep learning to extract the necessary features to derive an optimal policy. In all model configurations, we observed that the agent was capable of predicting close to optimal actions (when compared to experts), as seen by the confusion matrices in Figure 5.

However, as the analysis of the results showed, a lot of work still remains. Firstly, our method would greatly benefit from the collection of a larger number of demonstrations for both the training and testing processes. This increase in amount of data would undoubtedly help the generation of a robust agent, more capable of handling usual roads configurations. Another interesting problem would be to explore human-computer interaction aspect.



(a) experts' predictions (b) agent's predictions (c) activation maps

Figure 6: Mobilenet top-3 predictions $(blue, green, red)$ vs. experts' predictions on the 8-action-space. A missing bin corresponds to $unknown$. (c) shows the CNN activation maps [41].

That is, how should one render the agent's outputs with each of the action-space configurations we presented. To answer this question, future work should aim at developing a smartphone application, built on these trained models. A user study with visually impaired individuals should then be conducted to evaluate the effectiveness of the various rendering possibilities. Moreover, it would be interesting to benchmark each model's battery consumption statistics and inference times.

References

[1] List of audible pedestrian signals installed and to come. http://www.mabmackay.ca/pages/158/INFO-Accessibility?langue=en. Accessed: 2017-06-30.

[2] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[3] D. Ahmetovic, C. Bernareggi, A. Gerino, and S. Mascetti. Zebrarecognizer: Efficient and precise localization of pedestrian crossings. In *ICPR*, 2014.

[4] D. Ahmetovic, C. Bernareggi, and S. Mascetti. Zebralocalizer: identification and localization of pedestrian crossings. In *Mobile HCI*, 2011.

[5] D. Ahmetovic, R. Manduchi, J. M. Coughlan, and S. Mascetti. Mind your crossings: Mining gis imagery for crosswalk localization. *ACM Transactions on Accessible Computing (TACCESS)*, 9(4):11, 2017.

[6] A. Arditi, J. D. Holtzman, and S. M. Kosslyn. Mental imagery and sensory experience in congenital blindness. *Neuropsychologia*, 26 1:1–12, 1988.

[7] B. D. Argall. Human autonomy through robotics autonomy. https://youtu.be/od6V1tt0ctc, 2016.

[8] B. D. Argall, S. Chernova, M. Veloso, and B. Browning. A survey of robot learning from demonstration. *Robotics and autonomous systems*, 57(5):469–483, 2009.

[9] Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(8):1798–1828, Aug. 2013.

[10] M. Bojarski, D. D. Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, X. Zhang, J. Zhao, and K. Zieba. End to end learning for self-driving cars. *CoRR*, abs/1604.07316, 2016.

[11] A. Canziani, A. Paszke, and E. Culurciello. An analysis of deep neural network models for practical applications. *CoRR*, abs/1605.07678, 2016.

[12] F. Chollet. Xception: Deep learning with depthwise separable convolutions. *CoRR*, abs/1610.02357, 2016.

[13] M. Everingham, L. Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision (IJCV)*, 88(2):303–338, June 2010.

[14] D. Fernandes. *Vehicle-pedestrian Accidents at Signalized Intersections in Montreal*. PhD thesis, McGill University, 2013.

[15] A. Giusti, J. Guzzi, D. C. Cireşan, F.-L. He, J. P. Rodríguez, F. Fontana, M. Faessler, C. Forster, J. Schmidhuber, G. Di Caro, et al. A machine learning approach to visual perception of forest trails for mobile robots. *IEEE Robotics and Automation Letters*, 1(2):661–667, 2016.

[16] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 249–256, 2010.

[17] D. Guth. Why does training reduce blind pedestrians veering. *Blindness and brain plasticity in navigation and object perception*, pages 353–365, 2007.

[18] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[19] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *CoRR*, abs/1704.04861, 2017.

[20] A. Hussein, M. M. Gaber, E. Elyan, and C. Jayne. Imitation learning: A survey of learning methods. *ACM Computing Surveys (CSUR)*, 50(2):21, 2017.

[21] F. N. Iandola, M. W. Moskewicz, K. Ashraf, S. Han, W. J. Dally, and K. Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and <1mb model size. *CoRR*, abs/1602.07360, 2016.

[22] V. Ivanchenko, J. Coughlan, and H. Shen. Staying in the crosswalk: A system for guiding visually impaired pedestrians at traffic intersections. *Assistive technology research series*, 25(2009):69, 2009.

[23] D. K. Kim and T. Chen. Deep neural network for real-time autonomous indoor navigation. *arXiv preprint arXiv:1511.04668*, 2015.

[24] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems*, NIPS'12, pages 1097–1105, USA, 2012. Curran Associates Inc.

[25] O. Lahav, D. Schloerb, S. Kumar, and M. Srinivasan. A virtual environment for people who are blind a usability study. *Journal of Assistive Technologies*, 6(1):38–52, 2012.

[26] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[27] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21-24, 2010, Haifa, Israel*, pages 807–814, 2010.

[28] N. Y. C. D. of Transportation. Accessible pedestrian signals program status report, 2015.

[29] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, Oct 2010.

[30] S. Panëels, A. Olmos, J. Blum, and J. R. Cooperstock. Listen to it yourself! evaluating usability of "what's around me?" for the blind. In *Human Factors in Computing Systems (CHI)*, 2013.

[31] S. A. Panëels, D. Varenne, J. R. Blum, and J. R. Cooperstock. The walking straight mobile application: Helping the visually impaired avoid veering. In *Proceedings of ICAD13*, pages 25–32, 2013.

[32] A. Paszke, A. Chaurasia, S. Kim, and E. Culurciello. Enet: A deep neural network architecture for real-time semantic segmentation. *CoRR*, abs/1606.02147, 2016.

[33] M. Poggi and S. Mattoccia. A wearable mobility aid for the visually impaired based on embedded 3d vision and deep learning. In *Computers and Communication (ISCC), 2016 IEEE Symposium on*, pages 208–213. IEEE, 2016.

[34] M. Poggi, L. Nanni, and S. Mattoccia. Crosswalk recognition through point-cloud processing and deep-learning suited to a wearable mobility aid for the visually impaired. In *International Conference on Image Analysis and Processing*, pages 282–289. Springer, 2015.

[35] D. A. Pomerleau. Advances in neural information processing systems 1. chapter ALVINN: An Autonomous Land Vehicle in a Neural Network, pages 305–313. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1989.

[36] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, June 2016.

[37] D. A. Ross and B. B. Blasch. Wearable interfaces for orientation and wayfinding. In *Proceedings of the fourth international ACM conference on Assistive technologies*, pages 193–200. ACM, 2000.

[38] S. Ross and J. A. Bagnell. Efficient reductions for imitation learning. In *In Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS*, 2010.

[39] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[40] A. Scott, J. Barlow, B. Bentzen, T. Bond, and D. Gubbe. Accessible pedestrian signals at complex intersections: Effects on blind pedestrians. *Transportation Research Record: Journal of the Transportation Research Board*, (2073):94–103, 2008.

[41] R. R. Selvaraju, A. Das, R. Vedantam, M. Cogswell, D. Parikh, and D. Batra. Grad-cam: Why did you say that? visual explanations from deep networks via gradient-based localization. *arXiv preprint arXiv:1610.02391*, 2016.

[42] H. Shen, K.-Y. Chan, J. Coughlan, and J. Brabyn. A mobile phone system to find crosswalks for visually impaired pedestrians. *Technology and disability*, 20(3):217–224, 2008.

[43] J. L. Souman, I. Frissen, M. N. Sreenivasa, and M. O. Ernst. Walking straight into circles. *Current Biology*, 19(18):1538–1542, 2009.

[44] T. Tieleman and G. Hinton. Lecture 6.5—RmsProp: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural Networks for Machine Learning, 2012.

[45] J. Togelius, R. D. Nardi, and S. M. Lucas. Towards automatic personalised content creation for racing games. In *2007 IEEE Symposium on Computational Intelligence and Games*, pages 252–259, April 2007.

[46] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328, 2014.

[47] J. Yosinski, J. Clune, T. Fuchs, and H. Lipson. Understanding neural networks through deep visualization. In *In ICML Workshop on Deep Learning*, 2015.