FINAL REPORT: OBJECT DETECTION AND DEPTH PERCEPTION SYSTEM FOR VISUALLY IMPAIRED PEOPLE(AWAREAI)

Rick Lin
Student# 1007977076
rick.lin@mail.utoronto.ca

Ruichen (Spencer) Li
Student# 1007623831
spencer.li@mail.utoronto.ca

Jason Tai
Student# 1008122363
jason.tai@mail.utoronto.ca

Danjie Tang
Student# 1008008941
danjie.tang@mail.utoronto.ca

ABSTRACT

Since the inception of the idea of Artificial Intelligence (AI), there has been extensive discussions about the potential of combining AI and healthcare. AI is considered an increasingly potent tool across various healthcare domains, such as medical image interpretation, diagnosis, and risk prediction. In this context, we present AwareAI, a deep-learning-based system designed to assist visually impaired individuals in becoming more aware of their surroundings. Our system seamlessly integrates deep learning frameworks with voice output capabilities to aid users. Through the analysis of input images, our system generates audio outputs that offer information about nearby objects and obstacles, including their distances. This task is divided into two components: depth estimation and object detection, both of which leverage deep learning techniques. The system excels in providing precise depth estimation and accurate object segmentation. Finally, we have successfully deployed our system on a Raspberry Pi and conducted several tests to validate its effectiveness.

—-Total Pages: 9

1 INTRODUCTION

Millions of people worldwide currently suffer from vision impairment, with an estimated 253 million individuals affected according to the World Health Organization. Our project's primary objective is the creation of a system enabling visually impaired individuals to utilize their smartphone camera for enhanced environmental awareness. This is achieved by reporting through voice nearby obstacles and their positions relative to the user. For instance, the system may say, "A chair is approximately 2 meters ahead on your left." By presenting this information audibly, users can heighten their awareness, thus mitigating the risk of collisions, stumbles, and falls.

Deep learning emerges as the optimal approach for our project's development due to its exceptional effectiveness in classifying objects within images. To ensure maximum accessibility, we have opted for a single camera rather than multiple cameras or LiDAR technology. Despite the constraints of a solitary camera, deep learning remains the most potent technique for accurate depth estimation. As a result, our system processes individual images using deep learning frameworks to precisely calculate distances.

2 BACKGROUND AND RELATED WORK

We conducted extensive research to explore approaches aiding individuals with visual impairments in understanding their surroundings. As we deconstruct this challenge into depth estimation and object detection, we delve into relevant studies encompassing both facets. Bauera et al. (2020)

examines the feasibility of leveraging cost-effective sensors to assist visually impaired individuals in comprehending their environment. He et al. (2018b) tackles depth perception by integrating the camera's focal length into an end-to-end trained deep neural network (DNN). They incorporate the focal length as an input to the model, with the aim of augmenting depth estimation accuracy.

Furthermore, Ito et al. (2019) introduces a modular network architecture for estimating indoor scenes, adapted from the GAN (Goodfellow et al. (2020)) and AlexNet (Krizhevsky et al. (2012)) architectures. In the realm of object detection, Fast R-CNN, proposed by Girshick (2015), adeptly classifies object proposals using deep convolutional networks. Similarly, Redmon et al. (2016) introduces Yolo as a novel approach to object detection. They reframe object detection as a regression problem for spatially separated bounding boxes and associated class probabilities. All these works bear close relevance to the quandary we are addressing, offering invaluable insights during the design of our unique system.

3 Data Preprocessing

Our project involves the training of 2 distinct models, one for depth estimation and another for object detection. As a result, two separate datasets are needed, with each requiring its own unique data preprocessing.

3.1 Depth Map

We selected the NYU Depth V2 (Nathan Silberman & Fergus (2012)) for depth estimation due to its substantial volume of training samples and its user-friendly interface. In order to generate valid input image, depth map pairs, identical transformations must be applied simultaneously. We implemented our own data augmentation classes and specifically, random cropping and horizontal flipping were chosen as our transformation methods. To enhance training and inference speed, we downscale the input images from 480×640 to 240×320 and limit the output depth maps' dimensions to 60×80. Figure 1 and figure 2 show the input image and the ground truth depth map for the depth model

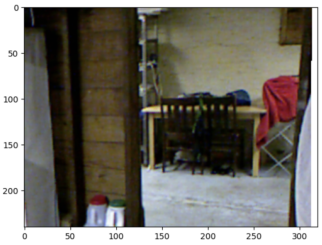

Figure 1: Input image

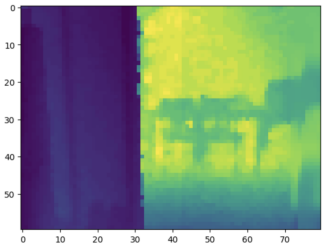

Figure 2: Ground truth depth map

3.2 Object Detection

We used MIT indoor scenes dataset Torralba & Sinha (2009) for our object recognition and segmentation model because it includes images of indoor settings, our use case, with labeled objects and polygon masks.

To process this dataset to be used for training, we started by copying all images that had an associated annotation file into a folder named; "cleanImages", and their annotation files into a folder named; "cleanAnnotations". We then parsed the copied XML annotation files, and for the files that could not be parsed, we removed them and their respective images. We then created a dictionary of all the different object classes and how many times they were labeled. We determined that there are 2970 different classes in this dataset. We then chose 50 classes that were common in the dataset and would likely be an obstacle for someone navigating indoors and added them to a desired classes list. We then went through all the annotation files in the "cleanAnnotations" folder and deleted annotation files along with their images that did not have objects that were in our desired classes list. This resulted in a final dataset size of 2517 images.

We then converted each polygon mask (list of points) in the annotation file of each image into a new image of the same dimension as the original image, but having 0 represent not the object, and

1 representing the object as shown in figure 3. (shows up as black for the background and white for the object in visualization) Then we created a bounding box and encoded the object class into its index in our list of desired classes. We then resized all images and masks to 500x500, and by doing this, the images would be deformed by stretch or compression figure 4, which aids in training our model to be more robust.

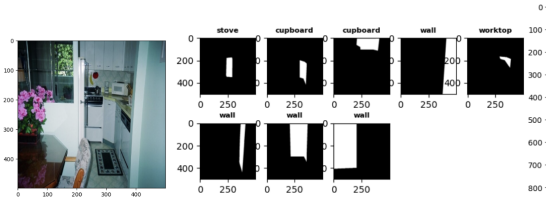

Figure 3: Example of image and its ground truth. Torralba & Sinha (2009)

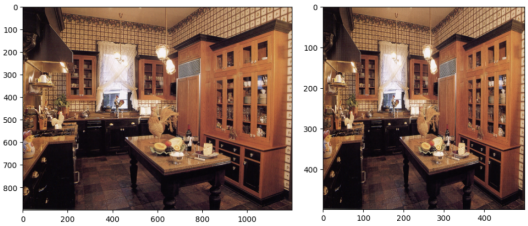

Figure 4: Example of image that is compressed to fit 500x500. Torralba & Sinha (2009)

4 Model Architecture

Our overall project architecture comprises an object detector, a depth map generator, and a text-to-speech converter. The object detector is responsible for labeling detected objects and creating segmentation masks for them. The depth map generator takes the original images as input and generates corresponding depth maps. The system then combines the image with the masked objects and its depth map. This allows us to establish a one-to-one relationship between detected objects and their depths. To compute the distance of an object, the system simply averages the depths of each pixels of the mask of the object. The object's relative distance is determined based on its X and Y position in the image, as well as its actual distance. We use a text-to-speech converter to output the type of object, its distance, and its relative location. The overall architecture of our system is depicted in Figure 5.

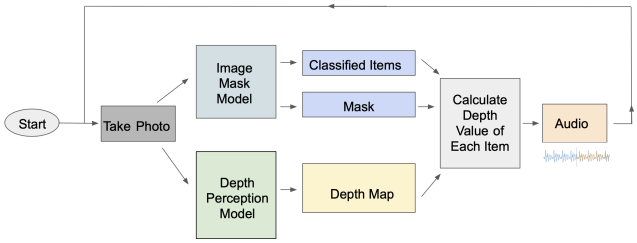

Figure 5: System overall architecture

Object detection and segmentation: in Figure 6. This model was originally trained on the COCO dataset, which had 90 classes + 1 background class. However, we have 50 classes + 1 background class, so in ordeFor our object detection and segmentation model, we chose to fine tune the pre-trained network: maskrcnn_resnet50_fpn_v2 as shown r to suit this, we changed the output size of the maskrcnn predictor and fastrcnn predictor from 91 to 51. We have also set the hidden layer size to be 200.

Implementation Details and results object: For training, we encountered issues with exceeding the GPU memory, and as a result, we were limited to using a training dataset of only approximately 500 out of the 2517 images, and to divert all memory into training, we decided to not have a validation or testing dataset, instead, we will be evaluating accuracy by hand. We have also frozen the resnet50 backbone weights, and as for the hyperparameters, we have used a batch size of 32, and an Adam optimizer with a learning rate of 1e-2 over 490 epochs.

Depth perception model: We utilized a Multi-scale Deep Network, modeled after the architecture proposed by Eigen et al. (2014), to address the problem at hand. The network comprises two main

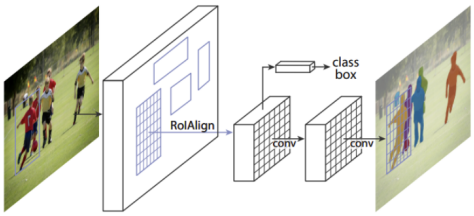

Figure 6: Mask RCNN framework He et al. (2018a)

components: coarseNet and fineNet. The coarseNet takes the input image and generates an initial depth map, while the fineNet takes both the image and the coarseNet's output to produce a full-resolution depth map. The coarseNet consists of 5 convolutional layers and 2 fully-connected layers, while the fineNet comprises 3 convolutional layers. The depth perception model's architecture is illustrated in Figure 7. In the complete model, the output of the coarseNet is concatenated with the fineNet's pooling layer output, forming the input for the subsequent stages.

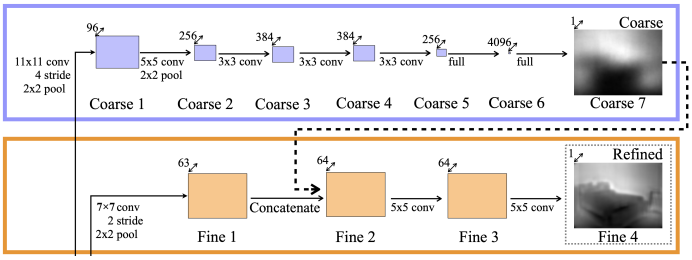

Figure 7: Architecture of depth model

5 Baseline Model

In our pursuit of comparative analysis alongside our deep learning architecture, we meticulously evaluated two distinct standard machine learning methodologies: a support vector machine (SVM) for obstacle categorization and a random forest for image depth estimation. The choice to employ an SVM framework for obstacle identification was driven by its applicability to multi-dimensional attributes like pixel characteristics. Simultaneously, we opted for a Random Forest Regression approach as a reference for approximating image depth due to its prowess in predicting intricate non-linear continuous data, such as depth values within images. Both these methodologies exhibited commendable accuracy during testing, making them apt for direct comparison with our deep learning results.

In the context of common indoor obstacle classification, the SVM implementation combined scikit-learn and TensorFlow. The model successfully employed the entire image range to detect distinct objects, but complexity limitations led us to focus solely on pre-classified common indoor obstacles. Training and validation encompassed over 50 object categories, resulting in a 42.6% accuracy rate, along with probability calculations for each category. By using preprocessed images before training the model, the random forest machine learning algorithm is able to take raw background image examples and estimate the maximum depth value to avoid user collisions with obstacles.

For image depth estimation and preempting user collisions with obstacles, we adopted a Random Forest Regression model. Focused on the lower 25% of images due to increased obstacle likelihood, the model extracted maximum depth values from this region. Background image preprocessing, feature extraction from training sets, and predicting maximum depth values constituted the model's design. Integrating depth-related pixel values from the bottom 25% and raw ImageNet-VGG16 features, the model achieved an RMS value of 8.947 for train-test error and 3.782 for train-validation error. Further clarification regarding the interpretation of RMS Error is provided in sections addressing the Depth Estimation Deep Learning Model.

6 Implementation Details and Results

Our system consists of two main part: a depth perception model and a object detection model. We have carried on extensive tests on each model and results are generally categorized to qualitative ones and quantitative ones.

6.1 Depth Perception Model

To train our depth perception model, we generate a subset of 8000 images from the original NYU Depth V2 dataset. We split the subset to train, validation, and test sets which contain 6000, 1000, 1000 images respectively. Training process was divided to two part for each of coarseNet and fineNet. For coarseNet, we train it for 150 epochs with batch size of 8. An Adam optimizer is used and the learning rate is set to 0.0001. For fineNet, we train it for 100 epochs with batch size of 16 and an Adam optimizer with learning rate set to 0.0001.

Qualitative Results: Figure 8 shows some representative examples that demonstrate the performance of our depth model on the test set. Notice that the model manages to capture the complexity of input images and provides excellent depth estimations.

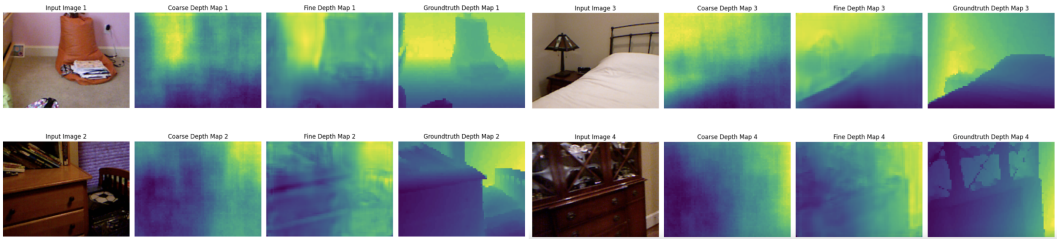

Figure 8: Qualitative results of depth perception model. For each example, from left to right are input images, coarseNet predicted depth maps, fineNet predicted depth maps, and gt depth maps.

Quantitative Results: We measure the performance of the model with three metrics: root mean square error (RMSE), RMSE-log, and accuracy with a threshold (δ_t) as recommended by Eigen et al. (2014). They are formulated using the following equations respectively:

$$RMSE = \sqrt{\frac{1}{|D|}\sum_{pred \in D} ||gt - pred||^2} \tag{1}$$

$$RMSE - log = \sqrt{\frac{1}{|D|}\sum_{pred \in D} ||\log(gt) - \log(pred)||^2} \tag{2}$$

$$\delta_t = \frac{1}{|D|}|\{pred \in D| \max(\frac{pred}{gt}, \frac{gt}{pred}) < 1.25^t\}| \times 100\% \tag{3}$$

In these equations, the $pred$ and gt denote predicted depth and ground truth, respectively. D represents the set of all predicted depths value for a single image, $|.|$ returns the number of the elements in each input set, and δ_t represents the threshold. From the high level, RMSE and RMSE-log measure the difference between model's predictions and ground truth labels. Accuracy with threshold represent how well the predictions of the model are. Table 1 shows our depth perception models' performance on the test set.

6.2 Object Detection Model

Qualitative Results: In figure 9 and figure 10, we present our qualitative results in image classification and segmentation. Depicting the complex image with substantial variations in lighting and object overlap, our model has successfully identified most of the objects in figure 9.

	RMSE	RMSE-log	δ_1	δ_2	δ_3
Coarse+Fine	0.94	0.44	0.370	0.734	0.933
Coarse	0.99	0.45	0.361	0.714	0.923

Table 1: Quantitative results of depth estimation model on test set. Lower is better for RMSE and RMSE-log. Higher is better for δ_1, δ_2, and δ_3.

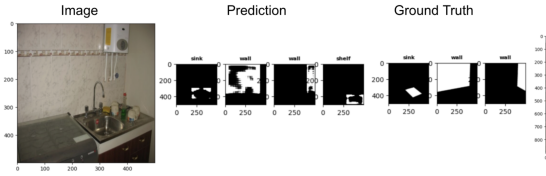

Figure 9: Demonstration of object detection.

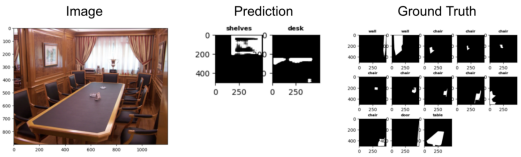

Figure 10: Demo of object detection failing.

However, our model presents inconsistent performance. As shown in figure 10, it occasionally fails to identify the majority of the objects. This inconsistency is further highlighted by different performance in identifying different objects. While the model has an impressive performance in identifying desks, walls, and cupboards, it almost always fails to identify chairs or sofas. Another example would be it frequently classifies a single object into both bookshelf and cupboard. This could be due to the limited dataset size and inconsistent labels in the ground truths.

Quantitative Results: We manually computed the accuracy by taking 12 random unseen images, calculating the total number of correctly labeled objects with accurate size and location for their masks (within 20% range of ground truth position), and dividing it by the total number of objects detected resulting in an accuracy of 77%. However, when compared to the ground truth labels, we divided the total amount of correctly classified objects with accurate size and location for their masks by the total amount of objects and masks resulting in an accuracy relative to the ground truth labels of 57%.

7 Generalization and Evaluation on New Data

When combining the image classifier, the depth estimation model, and the algorithms utilized for calculating the average depth of each image item, and ultimately producing audio signals for the users, several metrics come into play for evaluating the model's performance on novel and unseen indoor data. Numerous mathematical calculations were factored in, with three of the foremost evaluation metrics being computational efficiency, accuracy, and loss assessment. These metrics have proven vital in enabling us to fine-tune our model's hyperparameters effectively.

Software and Hardware Computational Efficiency: When continuously looping through the entire system, a new image data is first captured via the Raspberry Pi camera, which is later sent over the network to run remotely on Google Colabatory. The image is then passed into two distinct models: the image mask model and the depth estimation model. After the forward propagation pass, the classified items and depth map are then passed into the depth value calculation algorithm to generate an audio output eventually. Excluding the time required to output each sentence to the user, the process from image capture to audio output takes an average of 69.0ms. For instance, for each image with a shape of (1, 3, 384, 640), the system speed takes 1.8ms for preprocessing, 69.0ms for inference, and 1.4ms for postprocessing. Based on this result, the computational efficiency serves as a good representation for new data, ensuring excellent performance without latency when moving through indoor rooms. This ensures that our model is not overly complex and doesn't suffer from prolonged forward propagation times.

Accuracy, Loss, Reference performance assessment: To assess the performance of our models on newly unseen indoor data, we assembled a collection of 100 indoor images in collaboration with our team. These images encompass diverse scenarios such as kitchens, bedrooms, and others, presenting varying obstacle types and fluctuations in lighting conditions.

For the image classification model, we computed the proportion of accurately classified instances in 12 random unseen images, resulting in 77% accuracy. This is done by calculating the total number of correctly labeled objects with accurate size and location for their masks (within 20% range of ground truth position) and dividing it by the total number of objects detected. In a kitchen setting as shown in Figure 11, we can see that the model has successfully identified the cupboard and the counter with its relative positions.

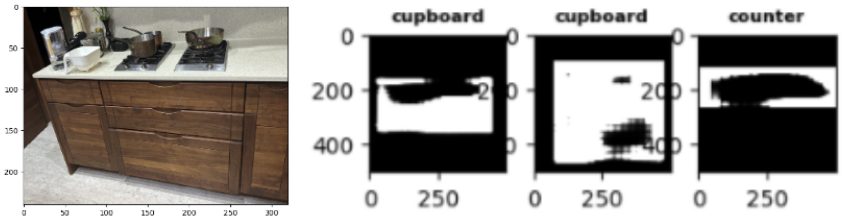

Figure 11: An Accurate and Confident Detection of Obstacles in the Kitchen

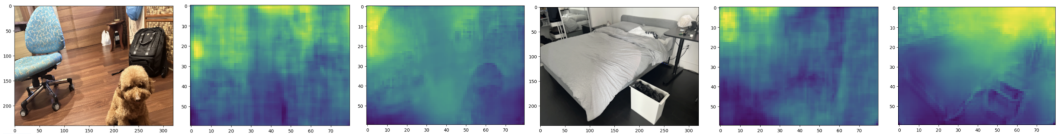

Figure 12: Depth Map Performance: Before (Middle) v.s. After (Right)

To evaluate the effectiveness of our depth estimation model with fresh data, we employed two distinct metrics: the square error loss for overall image depth and the accuracy of estimated depth within a 0.2-meter range from the actual depth. In our initial attempt to test the depth estimation model's performance while identifying all classified image items, the outcomes were notably unsatisfactory (Figure 12 Middle). An approximate 75% of all image categories exhibited a disparity exceeding 0.2 meters from the actual depth (as determined using an iPhone 14 Pro LiDAR Sensor as the reference standard). However, this setback prompted a reevaluation of our depth model, leading to the adjustment of pertinent hyperparameters to ultimately enhance its performance (Figure 12 Right). Subsequent to the training of over 10,000 image data instances, we achieved an accuracy of 82.3%. This improvement is evident in the distinct outlines and accurate depth values apparent within each classified category (Figure 13). In a broader context, our model demonstrates strong generalization capabilities for both our perception and image mask components.

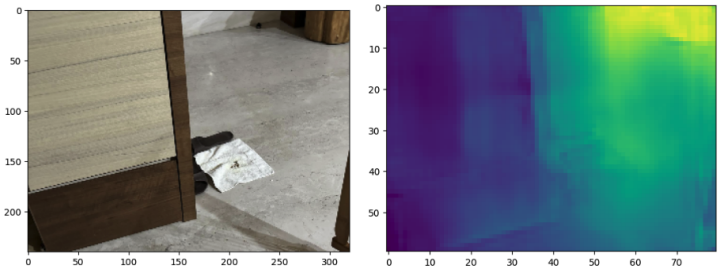

Figure 13: An Accurate Depth Estimation of Wall Obstacles Successfully Achieved

8 Discussion

8.1 Depth Model

Despite numerous difficulties faced during the architecture design, training, and generalization phases, our depth model performs wonderful on test set as well as on unseen images.

Performance Analysis: From the test, we notice that our model provides accurate estimations of the depth for 93.3% of all examples in the test set. Even if this result doesn't match with the state-of-art approaches such as P3Depth proposed by Patil et al. (2022), its performance perfectly meets our project requirements. This conclusion is verified after we deployed the whole system on a Raspberry Pi and tested it with new unseen data.

Training and Generalization Obstacles: NYU Depth V2 dataset contains images with densely labeled depth information taken in several scenes. This fact created extra difficulties when we were trying to create high-quality and representative training set at the very beginning. A training set can easily contain a couple of similar images taken in the same scenes. We initially created a training set with about 1800 examples and as a result, the model suffered terrible overfitting.

The limited computing power made the training even harder. With no local machine with suitable graphic card for training machine learning models, we used Google colab instead to carry out the experiments. Intermittent availability of high-RAM machines and lack of enough computing power limited the size of our training set to around 6000 images. More complicated data preprocessing skills were utilized to minimize the similarity between images in the training set and to ensure the quality of generalization. As a comparison, Eigen et al. (2014), from whom we got inspired and designed our model, trained their model on 120,000 unique images and thus generated better results.

Potential Better Model Architectures and Future Direction: The performance of the depth perception part of our system can be easily improved by substituting our current depth model with a better one. Candidates can be the P3Depth proposed by Patil et al. (2022) or the AdaBins proposed by Bhat et al. (2021), which are both trained on NYU Depth V2 dataset. To improve the generalization of our depth model, more explorations of better data augmentation skills will be done. In addition, low level optimizations of our training script can potentially fit a larger training set in the machine with the same RAM.

8.2 Object Detection and Segmentation

We have encountered issues with mislabeled ground truth labels and limited GPU memory.

Mislabeled ground truths: Having mislabeled ground truths as well as missing labels most likely confused the model during training. For example, in figure 14, an entire section of the image was just labeled as a wall when it should have been individually labeled as cupboards.

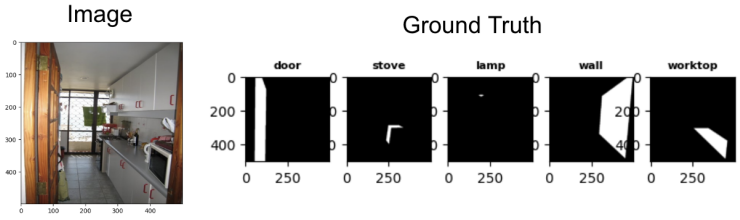

Figure 14: Image and Ground Truth. Torralba & Sinha (2009)

Issues with GPU memory: Since each image has a mask of 500x500 for each object in the ground truth, the data loader consumes a massive amount of GPU memory. As a result, we were limited to using a training dataset size of approximately 500 images. Due to mislabeled ground truths, missing labels, and memory limitations, the model failed to show performance that we are fully satisfied with. However, excellent results were demonstrated in some examples.

9 Ethical Consideration

In the process of developing a deep learning model for an Object Detection and Depth Perception System, our utmost focus lies in safeguarding individual privacy and adhering to privacy regulations such as GDPR by Paul et al. (2016) and PIPEDA by Policy & of the Office of the Privacy

Commissioner of Canada (2000). We have taken careful measures during our training process to avoid using potentially sensitive individual information. This has led us to implement robust protective measures to prevent unauthorized access. Rigorous data security protocols, including secure storage and access controls, are of paramount importance. Furthermore, a meticulous review of privacy policies and the acquisition of informed consent for data sharing and utilization were essential prerequisites before our model training. PIPEDA Additionally, when it comes to deploying the system, it is crucial to be vigilant about potential biases that could undermine the system's fairness and inclusivity. The exclusive introduction of the system in wealthier neighborhoods or the presence of bias in the training data could result in uneven outcomes and limited benefits for marginalized communities. Moreover, biases originating from the training data might lead to lower accuracy in detecting individuals from diverse ethnic backgrounds. To ensure equitable results, we have taken multiple perspectives into account and considered it of utmost importance to address and mitigate these biases. Our aim is to strive for an impartial and unbiased representation within the design and deployment of the system.

10 Project Difficulty and Quality

Our team has chosen an extraordinarily difficult project as we are trying to solve two complicated and classical problems at the same time. What makes it even harder is the requirement of deploying the system on a mobile device. Object detection and depth estimation without LiDAR have been long investigated from various angles. In total, we conducted a preliminary review of 16 papers and selected 5 papers that were particularly promising for further investigation. We have also spent more than 20 hours researching different object segmentation and detection models. Building on this foundation, we invested an incredible amount of effort into model training. As a result, we achieved impressive performance in depth estimation, and satisfactory results in object segmentation.

Depth estimation is an exceptionally challenging task, especially for people with no previous experiences in this field. Among all of its derivations, monocular depth estimation has always been interesting due to its low requirements on complicated equipment or professional techniques. Numerous tricks should be used otherwise training of the model can fail. For example, a depth perception model sometimes has to estimate the depth of close and distinct objects at the same time. In such cases, the level of the error of the model can be exceptionally sensitive. Being a few centimeters off in our estimation of depth for an object that is meters away is acceptable. However, it is a bigger mistake to be a few centimeters off if the object is only ten centimeters away. As a consequence, most work should be done in log space. Unable to gain such insights in the depth estimation problems has constructed uncountable obstacles for us. Extensive research beyond the coverage of this course was carried out in order to generate a valid and reasonable depth estimation model. Nonetheless, our final depth model met the requirements of our project and performed well when we tested it on new data as demonstrated in section 6 and 7.

Object detection and segmentation presents itself as yet another task of exceptional difficulty. The need to identify an unknown number of objects in an image as well as create a mask and bounding box for each object is not trivial. Not only this, but also in order to have a robust model, we would need to train the model on a massive dataset, which would take up considerably more GPU memory than we currently have available to us. Upon integrating our model with the Raspberry Pi, our initial intention was to execute it locally using the available hardware resources. However, we encountered challenges related to downloading dependencies due to limited bandwidth and observed excessive heat generation by the built-in processor. Consequently, we devised a solution by capturing images and transmitting them over the network to be processed remotely on a MacBook laptop. This transition posed its own set of difficulties. Despite these obstacles, our system demonstrated commendable performance, exhibiting an acceptable range of latency.

Overall, our team embarked on an exceptionally challenging project, and the performance of our model exceeded the expectations of this complex task. We have demonstrated learning that is well beyond the APS360's labs and tutorials, combining our deep learning models with appropriate software and hardware resources, and our tasks may even surpass typical undergraduate.

REFERENCES

Zuria Bauera, Alejandro Domingueza, Edmanuel Cruza, Francisco Gomez-Donosoa, Sergio Orts-Escolanoa, and Miguel Cazorlaa. Enhancing perception for the visually impaired with deep learning techniques and low-cost wearable sensors. *Pattern Recognition Letters*, 137:27–36, 2020.

Shariq Farooq Bhat, Ibraheem Alhashim, and Peter Wonka. AdaBins: Depth estimation using adaptive bins. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, jun 2021. doi: 10.1109/cvpr46437.2021.00400. URL https://doi.org/10.1109%2Fcvpr46437.2021.00400.

David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network, 2014.

Ross Girshick. Fast r-cnn, 2015.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn, 2018a.

Lei He, Guanghui Wang, and Zhanyi Hu. Learning depth from single images with deep neural network embedding focal length. *IEEE Transactions on Image Processing*, 27(9):4676–4689, sep 2018b. doi: 10.1109/tip.2018.2832296. URL https://doi.org/10.1109%2Ftip.2018.2832296.

Seiya Ito, Naoshi Kaneko, Yuma Shinohara, and Kazuhiko Sumi. Deep modular network architecture for depth estimation from single indoor images. In Laura Leal-Taixé and Stefan Roth (eds.), *Computer Vision – ECCV 2018 Workshops*, pp. 324–336, Cham, 2019. Springer International Publishing. ISBN 978-3-030-11009-3.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger (eds.), *Advances in Neural Information Processing Systems 25*, pp. 1097–1105. Curran Associates, Inc., 2012. URL http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf.

Pushmeet Kohli Nathan Silberman, Derek Hoiem and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012.

Vaishakh Patil, Christos Sakaridis, Alexander Liniger, and Luc Van Gool. P3depth: Monocular depth estimation with a piecewise planarity prior, 2022.

Voigt Paul, Bygrave Lee, Dibble Suzanne, Hitchen Brian, Besemer Leo, and Hijmans Hielke. Regulation (eu) 2016/679 of the european parliament and of the council of 27 april 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing directive 95/46/ec (general data protection regulation). *International Data Privacy Law*, 2016.

Policy and Research Group of the Office of the Privacy Commissioner of Canada. Personal information protection and electronic documents act. 2000.

Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection, 2016.

Antonio Torralba and Pawan Sinha. Recognizing indoor scenes. *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2009*, 06 2009. doi: 10.1109/CVPRW.2009.5206537.

CODE ACCESS

Link to GitHub repository: https://github.com/Spencer-16/APS360-AwareAI