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

Object detection and depth perception problems have been challenging and widely discussed for several years. With the advancements in deep neural networks (DNNs) and computer vision algorithms, there is a growing focus on refining learning strategies, model architectures, and their practical applications. In this paper, we propose AwareAI: an Object Detection and Depth Perception System for Visually Impaired People. Our system combines deep learning frameworks and a machine learning system with voice output to assist visually impaired individuals by providing information about the objects and obstacles in their view. By taking images as input, the system generates audio outputs that offer information about objects or obstacles in front of the user, including their distances. We utilize the NYU Depth V2 and ScanNet datasets to train the depth perception model and generate a depth map, and the ImageNet dataset to train the object detection model. We employ a support vector machine as our baseline model and make comparisons with our proposed deep learning model. Additionally, we present a comprehensive table outlining tasks, team members, deadlines, and a detailed project timeline. Collaboration methods and potential risks associated with the project are also discussed.

—-Total Pages: 9

1 INTRODUCTION

Many people around the world suffer from visual impairments and eye conditions, such as blurred vision, diabetic retinopathy, age-related macular degeneration, and other vision problems State (2012) . However, treatments for these visual impairments and eye conditions may not be accessible to everyone, and having constant assistance may not always be feasible. Without treatment or assistance, visually impaired individuals face a greater risk of injury due to collisions with obstacles like furniture, tripping on objects on the ground, or falling down stairs that they may not have seen. Our project, AwareAI, aims to address this issue.

AwareAI will assist the visually impaired by utilizing their smartphone camera, which is a relatively accessible tool, to record their environment and receive voice outputs about nearby objects and obstacles along with their respective distances. For example, the system may say, "There is a chair two meters away on your left." This will enable users to be more aware of their surroundings, thereby reducing the risk of collisions, tripping, and falling.

Deep learning is a suitable approach for developing this project because it is highly effective in object classification within images. Additionally, to ensure accessibility to as many people as possible, we will utilize only a single camera and refrain from using multiple cameras or LiDAR technology. Our model will take one picture at a time as input and accurately estimate distances using deep learning frameworks.

2 Background and Related Work

Several recent projects and studies have investigated both object detection and depth perception under different settings. Broadly, these approaches can be grouped into two categories. The first group of them relies on mathematical models and relationships. For example, they use known geometry information such as the size of the object in real life and compute the distance in between with mathematical formulas. Another kind of approach is learning-based, where most effort is spent on designing machine learning architectures and training a model that detects objects and estimates depths.

2.1 Triangle similarity for Object/Marker to Camera Distance

Rosebrock (2015) proposes an approach that assumes the availability of external information, such as the object's dimensions and the camera's focal length. Upon receiving the image, this approach utilizes a pre-trained model to detect the object's edges and subsequently calculates the object's width in pixels within the image (Figure 1). A relationship is then established between the real width of the object (W), the focal length (F), the object's width in the image (P), and the distance between the object and the camera (D) as follows:

$$F = (P * D)/W \tag{1}$$

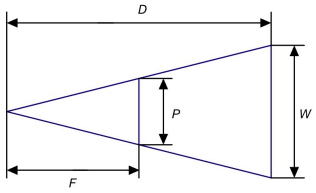

Figure 1: Geometric meaning of Equation 1

This method produces an estimation of 2.92 feet for an object located 3 feet away from the camera. However, this approach is flawed as its assumption regarding the availability of external information may not always be valid.

2.2 Perception with Deep Learning and Low-cost Sensors

Bauera et al. (2020) examines the feasibility of employing low-cost sensors to assist visually impaired individuals in comprehending their surroundings. The research group's system primarily comprises three components: local sensors, a remote server, and an output device, as illustrated in Figure 2.

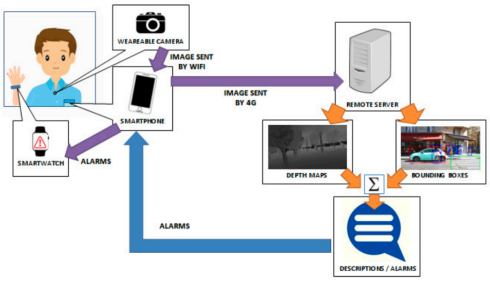

Figure 2: Architecture of the system proposed by Bauera et al. (2020)

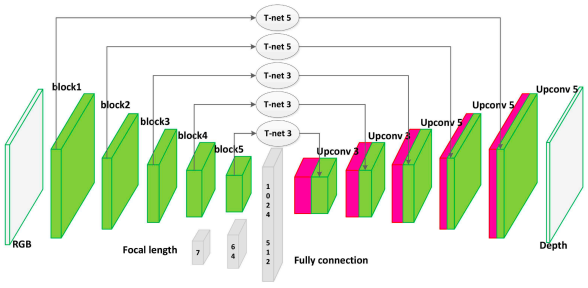

Figure 3: Architecture introduced by He et al. (2018) where focal length is taken as an input

This system captures an image using a wearable camera and transmits it to users' smartphones, which then send the image to a remote server. On the remote server, the image is processed by a deep neural network (DNN) based on He et al. (2015). The network generates a depth map and an

annotated image where detected objects are enclosed within bounding boxes. The depth map and annotated image are combined to calculate depth information for each detected object. The results are then sent back to users' smartphones, where descriptions and alarms are generated. This system achieves a detection rate of 87.99 for obstacles.

2.3 Depth Perception with know focal length

He et al. (2018) addresses depth perception by incorporating the focal length of the camera into their end-to-end trained deep neural network (DNN). They opt to include the focal length as an input to the model, aiming to enhance the accuracy of their depth estimation. Figure 3 illustrates the architecture of their approach, with the convolutional neural network (CNN) based on the VGG model proposed by Simonyan & Zisserman (2015).

In popular datasets for depth perception, such as NYU Depth V2 and ScanNet, only a single image is captured per scene, and images with varying focal lengths are not available. However, in the approach proposed by He et al. (2018), they generate images with different focal lengths by applying matrix operations to the original image. Figure 4 showcases two images with distinct focal lengths captured at the same scene. Subsequently, these newly generated images are passed through their deep neural network (DNN) to obtain depth maps as outputs. The method presented by He et al. (2018) surpasses the performance of state-of-the-art models in terms of prediction accuracy.

Figure 4: Two pictures taken at the same place with cameras with different focal lengths

2.4 Modular Depth Estimation from Single Indoor Images

Ito et al. (2019) introduces a modular network architecture for estimating indoor scenes, which was adapted from the GAN (Goodfellow et al. (2020)) and AlexNet (Krizhevsky et al. (2012)) architectures. In Figure 5, the architecture initially takes an RGB image as input and passes it through the layout network $\boldsymbol{N}_L$ and the object network $\boldsymbol{N}_O$. The values obtained from $\boldsymbol{N}_L$ and $\boldsymbol{N}_O$ are then combined to determine a reliability score. If this score exceeds a certain threshold, the image data is further processed for depth estimation. Instead of solely learning features implicitly within the network, this architecture incorporates a *discriminator network* that classifies each instance as either fake or real before proceeding to the depth estimation network.

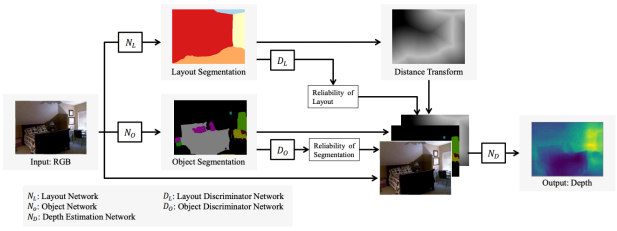

Figure 5: Architecture of the system proposed by Ito et al. (2019)

2.5 Different Model Architectures for learning-based approaches

According to Masoumian et al. (2022), depth estimation can be categorized into three divisions: monocular depth estimation (MDE), binocular depth (BDE), and multi-view depth estimation

(MVDE). The earliest significant advancement in MDE, which involves estimating depth from a single viewpoint, can be traced back to a research paper from Stanford (Liu et al. (2016)). Considering memory limitations, employing MDE is a suitable choice for deep learning (DL) applications on edge devices, as complex networks may not be practical for real-time applications. Architectures such as ResNet (He et al. (2015)) and DenseNet (Huang et al. (2017)) strike a balance between accuracy and the power consumption requirements. Additionally, Howard et al. (2017) introduce MobileNet, a DNN architecture designed for low computational cost in mobile and embedded vision applications. Moreover, efficientNet proposed by Tan & Le (2020) outperforms ResNet and DenseNet in terms of computational cost, particularly when the image resolution is high Tadepalli et al. (2021).

3 Data Preprocessing

Huggingface (Wolf et al. (2019)) offers a comprehensive collection of datasheets conveniently bundled in a library, providing convenient access. Therefore, the Nathan Silberman & Fergus (2012)'s /sayakpaul/nyu_depth_v2 datasheet from Huggingface was chosen for its exceptional compilation of one of the largest image/depth datasets and its user-friendly interface. Upon closer inspection, the dataset reveals two distinct fields.

1. Image : A PIL.PngImagePlugin.PngImageFile object with uint8 data type.
2. Depth Map: A PIL.TiffImagePlugin.TiffImageFile object with float32 data type.

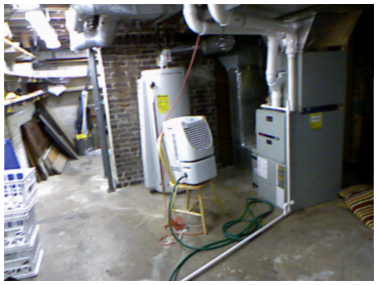

Figure 6: Image

Figure 7: Depth map visualization

Both datasets share a common resolution of 640 by 480. There is no need for data preprocessing in terms of resizing the data to a specific size. However, the following sections describe other data preprocessing steps that will be performed:

3.1 Visualization

To facilitate future debugging purposes, it would be beneficial to have a visualization of the depth map. Therefore, our team plans to devise a visualization mechanism for depth maps as part of the data preprocessing procedure. The procedure will begin by normalizing the depth map within a range of 0 to 1. Subsequently, the depth map will be converted into a numpy array with an RGB representation, which will have a shape of 640 by 480 by 3. This conversion will be achieved using the plt.cm.viridis function. Finally, by multiplying the resulting array by 255, we will obtain the desired visualization of the depth map, as depicted in Figure 7.

3.2 Data Augmentation

By utilizing data augmentation, we can effectively mitigate the risk of overfitting and enhance the robustness of our model. Therefore, we will generate an additional dataset by randomly selecting augmentations from a predefined list. These augmentations include operations such as horizontal flip, random cropping, brightness contrast adjustment, gamma adjustment, and saturation adjustment. Subsequently, we will apply these augmentations randomly to our collection of 2000 images and their corresponding depth maps, and stored the augmented results.

3.3 DATA ORGANIZATION

The original dataset downloaded from Huggingface consists of over 100GB, making it impractical to use the entire dataset. After careful consideration, it has been decided that our project dataset will consist of the first 2000 images, which amounts to a more manageable size of 9.2GB. This decision takes into account the storage limitations of our device, as expanding the dataset beyond this size could risk a potential system crash.

A design choice has been made to store all files as numpy arrays, as these files essentially contain 2D array information represented in various forms. By adopting a universal data type, significant future efforts can be saved.

The data has been organized into 5 folders: image, depth_map, depth_map_visualization, augmented_image and augmented_depth_map. Each folder contains the corresponding data. We have adopted a straightforward naming convention, for example: image_17.npy and depth_map_17.npy, where the number represents the index of the data.

4 MODEL ARCHITECTURE

In this section, we present the overall architecture of our project, along with the model architectures for object detection and depth map generation.

4.1 OVERALL PROJECT 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 detecting objects in the image, labeling them, and creating bounding boxes around them. The depth map generator takes the original images as input and generates corresponding depth maps. We then combine the image with the bounded objects and its depth map. This allows us to establish a one-to-one relationship between detected objects and their distances. Various algorithms can be employed to calculate this distance, such as averaging all depth points or considering the average of the closest 25% of depth points. 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. A visual representation of our project architecture is depicted in Figure 8.

4.2 OBJECT DETECTION ARCHITECTURE

We will utilize Convolutional Neural Networks (CNNs) for training the object detection model, taking inspiration from successful architectures like ResNet He et al. (2015) and AlexNet Krizhevsky et al. (2012). Specifically, we will enhance the ResNet architecture, originally designed for image classification on ImageNet, to incorporate object detection capabilities with bounding box generation. Through modifications and the inclusion of additional layers and techniques, our goal is to empower the model to effectively detect objects and generate accurate bounding box predictions.

4.3 DEPTH MAP GENERATION ARCHITECTURE

Our plan involves utilizing U-Net Ronneberger et al. (2015) for generating depth maps from input images. U-Net is a fast model that supports end-to-end training, even with small datasets. Additionally, we are considering replacing the encoder of U-Net with the DenseNet Huang et al. (2017) or ResNet He et al. (2015) architecture to achieve more accurate results while preserving efficiency.

5 BASELINE MODEL

We will use a support vector machine (SVM) as the baseline model for comparison with our neural network architecture. To begin, we will utilize the preprocessed dataset described in Section 3, which consists of two categories: RGB images paired with their corresponding depth map values, and labeled images representing different classes.

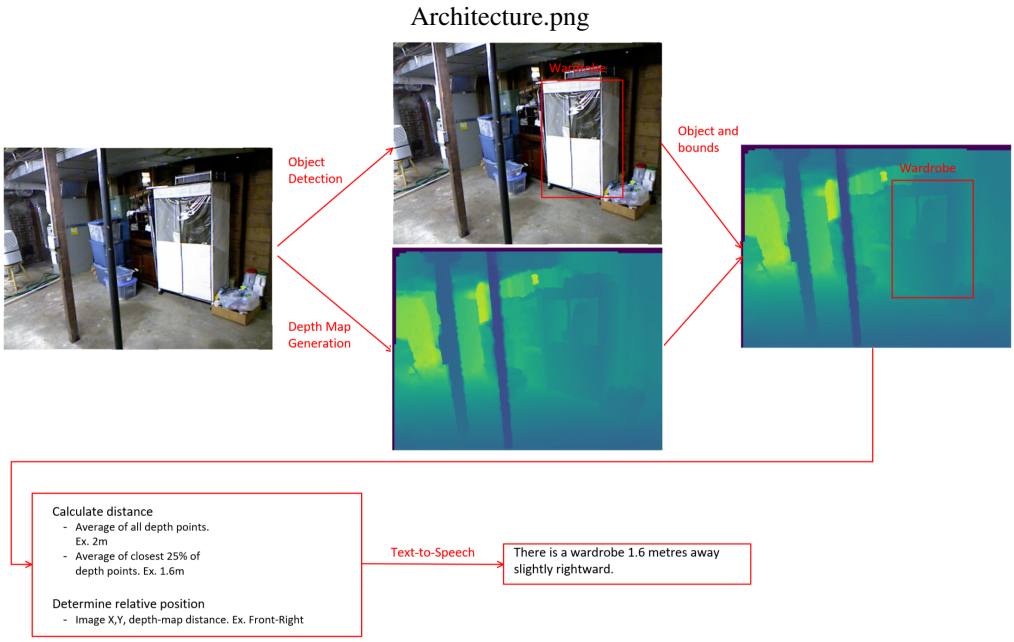

Figure 8: Visual representation of the project architecture. (Nathan Silberman & Fergus (2012))

For image classification, we will create a hash table containing all the objects we aim to classify from the RGB images, using the preprocessed data mentioned earlier. Next, we will extract handcrafted features from the RGB images to capture relevant information, including color histograms, text descriptors, edge-based features, and geometric features. These extracted features will be normalized accordingly. Using the labeled images, we will train the SVM for multi-class classification by tuning the hyperparameters (such as the regularization parameter C and kernel-specific parameters) through grid and random search. We will evaluate the accuracy of the SVM classification using metrics like root mean square or mean absolute error, both for training (roughly 75% split) and validation sets (roughly 25% split).

Similarly, for depth estimation, we will create a hash table containing the data objects and their corresponding regression values based on the preprocessed data. Handcrafted features (such as color histograms, text descriptors, edge-based features, and geometric features) will be extracted from the RGB images to capture relevant information. These features will be normalized, and class imbalance handling techniques will be applied if necessary. The SVM will be trained for multi-class classification using labeled images with values in a specific range. Hyperparameters will be tuned, and a suitable kernel function (such as linear or polynomial) will be selected through grid and random search. Accuracy evaluation for SVM regression will be performed using metrics like root mean square or mean absolute error, again for both training (roughly 75% split) and validation sets (roughly 25% split).

Finally, we will compare the accuracy of the SVM with our deep learning model to determine if our deep neural network outperforms the traditional machine learning approach. This evaluation will provide insights into the relative strengths of each method for both image classification and depth estimation tasks.

6 Project Plan

6.1 Ways of working and basic rules

Since the group was formed, we have had several discussions on roles of team members. The detailed roles and responsibilities are reported in Table 1. A Discord channel has been established for the group, where online meetings are held every Tuesday at 10 pm. Meetings with short notice,

upon the request of team members, are discussed and scheduled through the Discord channel. A one-day advance notice is required for absences during the weekly meetings. However, the weekly meeting can be rescheduled or canceled in case of midterms or unforeseen circumstances.

We utilize a GitHub repository to store relevant codes and project resources. Git version control tools will be used to ensure smooth collaboration among team members without affecting each other's code. The following coding rules are in place:

- After pushing, clear and concise commit messages, as well as comments in the code, are always required.
- For more complex feature developments, branches should be created.
- Only working code should be pushed, unless assistance is needed for debugging.

To manage tasks, we employ a group TODO list that is updated during the weekly meetings. The TODO list groups tasks into three categories based on their importance and urgency. Responsible members and deadlines are assigned based on task priority. High priority tasks are assigned to responsible team members with a deadline, typically within one week. Medium priority tasks are due within 1-2 weeks and may be moved to the high priority list as their deadlines approach. Medium priority tasks can be left without an assigned responsible member, but a deadline must still be set. Low priority tasks, such as personal background research, are also included in our TODO list, but they do not require a responsible member or a deadline. Table 2 presents our project timeline.

Team Member	Role	Responsibilities
Jason Tai	Team Learder	Establish the structure of the overall project and its goals
Rick Lin	Project Manager	Responsible for creating project plans, timelines, deliverables, and allocate DL/ML resources to the team
Danjie Tang	Data Management Lead	Lay the foundation for the group collaboration by preprocessing data in a speedy manner
Spencer Li	Architecture Lead	Carry on throughout research on state-of-art model architectures and provide insights on designing the model architecture for this project

Table 1: Team members and corresponding responsibilities

Tasks:		Team member:	Deadline:	Done:
Project Proposal (Hard deadline: Jun 16)	Create a google drive folder	Rick Lin	Jun 12	Jun 12
	Create a github repo	Spencer Li	Jun 12	Jun 12
	Abstract	Spencer / Jason	Jun 15	Jun 14
	Introduction	Jason	Jun 15	Jun 14
	Background & Related Work	Rick / All	Jun 15	Jun 16
	Data processing	Danjie / Rick	Jun 15	Jun 16
	Model architecture	Spencer / Jason	Jun 15	Jun 15
	Baseline model	Rick	Jun 15	Jun 15
	Project plan	All	Jun 15	Jun 16
	Ethical consideration	Rick	Jun 15	Jun 16
	Risk register	All	Jun 15	Jun 15
	Illustration / Figure	All	Jun 15	Jun 16
Data Processing	Processing scripts	Danjie	Jun 19	
Object Detection Implementation	Model Implementation and initial test	Spencer / Rick	Jul 1	
	Model training and parameter tuning	All	Jul 8	
	Final testing	Spencer / Rick	Jul 10	
Depth Perception Implementation	Model Implementation and initial test	Jason / Danjie	Jul 8	
	Model training and parameter tuning	All	Jul 15	

	Final testing	Jason / Danjie	Jul 17	
Project Process Report (Hard deadline: Jul 14)	Tasks and deadlines to be set	All	Jul 6	
Model Combining and Audio Output Generation	Model combining Audio generation System final test	Spencer / Rick Danjie / Jason All	Jul 19 Jul 19 Jul 23	
Project Presentation Video (Hard deadline: Aug 4)	Brainstroming and individual script writing Script combining and finalizing Video Recording Final video editing and extra clips recording	All All All All	Jul 27 Jul 28 Aug 1 Aug 3	
Project Final Report (Hard deadline: Aug 11)	Tasks and deadlines to be set	All	Aug 4	

Table 2: Project timeline

6.2 Measures of Success

We have determined the measures of success of our model such that; the resulting program describes the room well enough such that it gives the user a rough idea of where objects in the room are, and our model outperforms the baseline model (SVM) in accuracy and recall, with a lower Root Mean Squared Error (RMSE) as well.

7 Ethical Considerations

When developing our project for visually impaired individuals using deep learning, several critical ethical considerations must be addressed and carefully considered.

7.1 Privacy of Individuals

During the training of our deep learning model, the data used may contain sensitive information about individuals, such as location information and personal belongings. It is crucial for us to comply with relevant privacy regulations and guidelines, such as GDPR(General Data Protection Regulation) and PIPEDA(Personal Information Protection and Electronic Documents Act), to ensure robust measures are in place to protect the privacy of individuals (European Parliament & Council of the European Union (2016), Government of Canada (2000)).

7.2 Data Security, Consent, and Usage

Prior to training our model, it is essential to ensure that there is no unauthorized access or use of the data. Storing images in secure storage systems and preventing unauthorized access to storage areas are critical measures to uphold data security. Additionally, it is important to review the privacy policies regarding data collection in our dataset, ensuring that informed consent is obtained regarding data sharing and usage.

7.3 Deployment and Representational Biases

If the Object Detection and Depth Perception System designed to assist visually impaired individuals is deployed exclusively in affluent neighborhoods or areas with better infrastructure, either due to data bias or other considerations, it can result in biased outcomes and limit the benefits of the system for marginalized communities. Similarly, if the training data used for the system exhibits bias towards a particular group, it can lead to lower accuracy when detecting individuals from different ethnic backgrounds. It is crucial to address these biases and strive for fair and unbiased representation in our system.

8 RISK REGISTER

Throughout the development of our project, it is crucial to carefully consider and address the following potential risk factors that may arise.

8.1 GROUP MEMBER DROPS THE COURSE

If a group member decides to drop the course due to personal reasons or unforeseen circumstances, it is important for the member to communicate with the rest of the team before dropping. The member should discuss their intentions with the team, and the remaining members will assess the workload distribution. Depending on the circumstances, adjustments may need to be made to ensure that the remaining members can handle the additional responsibilities.

8.2 MODELS TAKE LONGER THAN EXPECTED

If the selected models exceed the team's original time expectations, several measures can be taken to address the situation. Firstly, the team may consider leveraging GPU acceleration by utilizing available GPU devices or exploring the purchase of GPU accelerators such as NVIDIA Graphics Cards. In cases where the model architecture proves to be highly complex, it is advisable for our team to reevaluate the hyperparameters and explore optimization techniques as needed. Techniques like early stopping or transfer learning can be applied during the model training process to mitigate lengthy training times. Early stopping helps prevent unnecessary iterations when further performance improvements are unlikely, while transfer learning leverages pre-trained models for specific tasks, significantly reducing development time. By considering these strategies, our team can effectively manage and overcome challenges related to longer model training times.

8.3 CHANGING PROJECT REQUIREMENTS

If any changes to the project requirements are proposed during the development, whether small or significant, it is crucial for team members to communicate within the team to ensure clear understanding of the reasons behind the modifications. It is essential to reassess the necessity of changes, taking into account the project's overall objectives and feasibility. In cases where the chosen model proves to be excessively complex or encounters training difficulties, the team should discuss with TAs or the professor to evaluate the model's performance and select the most optimal course of action. Their expertise can help identify potential solutions, such as adjusting the model architecture, fine-tuning hyperparameters, or exploring alternative approaches.

8.4 INTELLECTUAL DATA VIOLATION

If any intellectual data violation occurs during the model training process, the team must reassess data sources and license usage terms. It is necessary to replace any inappropriate data with suitable alternatives and update the model training accordingly. Ensuring compliance with intellectual property rights is crucial to maintain ethical and legal standards.

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.

European Parliament and Council of the European Union. 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, 2016. URL https://eur-lex.europa.eu/eli/reg/2016/679/oj.

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.

Government of Canada. Personal information protection and electronic documents act (s.c. 2000, c. 5), 2000. URL https://laws-lois.justice.gc.ca/eng/acts/p-8.6/index.html.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015.

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 2018. doi: 10.1109/tip.2018.2832296. URL https://doi.org/10.1109%2Ftip.2018.2832296.

Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications, 2017.

Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

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.

Fayao Liu, Chunhua Shen, Guosheng Lin, and Ian Reid. Learning depth from single monocular images using deep convolutional neural fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(10):2024–2039, oct 2016. doi: 10.1109/tpami.2015.2505283. URL https://doi.org/10.1109%2Ftpami.2015.2505283.

Armin Masoumian, Hatem A. Rashwan, Julin Cristiano, M. Salman Asif, and Domenec Puig. Monocular depth estimation using deep learning: A review. *Sensors*, 22(14), 2022. ISSN 1424-8220. doi: 10.3390/s22145353. URL https://www.mdpi.com/1424-8220/22/14/5353.

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

O. Ronneberger, P.Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, volume 9351 of *LNCS*, pp. 234–241. Springer, 2015. URL http://lmb.informatik.uni-freiburg.de/Publications/2015/RFB15a. (available on arXiv:1505.04597 [cs.CV]).

Adrian Rosebrock. Find distance from camera to object/marker using python and opencv. https://pyimagesearch.com/2015/01/19/find-distance-camera-objectmarker-using-python-opencv/, 2015. Accessed: 2023-06-15.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2015.

Agencies New York State. Types of vision problems, 2012.

Yasasvy Tadepalli, Meenakshi Kollati, Swaraja Kuraparthi, and Padmavathi Kora. Efficientnet-b0 based monocular dense-depth map estimation. *Traitement du Signal*, 38(5):1485–1493, 2021. doi: 10.18280/ts.380524. URL https://doi.org/10.18280/ts.380524.

Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks, 2020.

Thomas Wolf, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, and Josh Brew. Hugging face. https://huggingface.co, 2019.

CODE ACCESS

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