

**PROJECT PROPOSAL**

GENERAL PROJECT INFORMATION

| PROJECT NAME | | COURSE PROFESSOR | PROJECT PI/SPONSOR |
| --- | --- | --- | --- |
| Descriptive Vision Assistance Device | | Jeongkyu Lee |  |
| STUDENT | EMAIL | Role | |
| Tongle Yao | yao.to@northeastern.edu | Project Architect | |
| Siyi Gao | gao.siyi1@northeastern.edu | Data Curator | |
| Yufei Huang | huang.yufe@northeastern.edu | Validation Specialist | |
| Xuedong Pan | pan.xue@northeastern.edu | Model Trainer | |

1. **PROJECT OVERVIEW**

In this section, define the problem you aim to solve using computer vision techniques. Clearly explain why it is an important problem and what the real-world application of the solution will be.

| **Introduction** | According to the World Health Organization, there are over 285 million people who live with some form of visual impairment. Many of the individuals face significant challenges when navigating their daily life environments. Besides, current visual assistive technologies are often limited in providing real-time, accurate feedback in diverse settings, particularly outdoors. For our final project, we choose to do an engineering project. We aim to utilize AI and computer vision technology to create a real-time obstacle avoidance system for visually impaired people that can truly be taken outdoors and serve as an aid comparable to a white cane. |
| --- | --- |
| **Problem Statements** | For people with severe visual impairment, daily life activity could be challenging and risky. The Current known assistance includes White cane and special labor services that have been considered the mainstream methods for more than 100 years. However, traditional white canes can detect obstacles, but they do not provide detailed information about the nature, size, or how to navigate around them. An AI and computer vision (AI+CV) based assistive system is needed to offer real-time, accurate environmental information to enhance their mobility.   * Models like YOLO-v8-nano struggle with real-time processing on portable devices, achieving only 5-15 FPS for 640x640 video, which is insufficient for reliable obstacle detection. * Models often identify all visible objects they can detect regardless of their relevance, resulting in information overload. * Most models are trained on datasets like COCO, which contain only 80 classes and lack critical elements such as curbs, steps, and starirs. As a result, the range of supported scenarios is limited. * Models only provide information about the class of the object, lack of information about their size and distance, and people with visual impairment do not know how to avoid them. * They perform poorly in low-light conditions, limiting their effectiveness. |
| **Objective (Goals)** | Our application should be able to perform the following functionalities:   1. Enhance the real-time processing capability of the YOLO-v8-nano model on portable devices, ensuring at least 20 FPS on small-scale computing devices. 2. Develop a distance-based object filtering mechanism to identify and mark only the obstacles directly related to the user’s movement, reducing information overload and focusing on relevant front-facing obstacles. 3. Fine-tune the model using our customized dataset and other datasets we can find online to enrich its recognition capabilities and improve its practicality in the scenario of people with visual impairment traveling. 4. Enhance the model’s recognition accuracy in low-light conditions (e.g., indoor environments, nighttime, and cloudy days) to ensure the visual assistance system remains robust under various lighting scenarios. 5. Develop a simplified feedback system that converts detected obstacles into clear and comprehensible audio or tactile cues, assisting visually impaired users in making quick and informed responses. |
| **Deliverables** | 1. A fully optimized and fine-tuned model capable of real-time obstacle detection at a minimum of 20 FPS on portable devices. The model will be tailored to the specific needs of visually impaired users. 2. An implemented and tested distance-based filtering mechanism that prioritizes objects relevant to the user’s navigation, reducing unnecessary information, providing distance information, and focusing on immediate front-facing obstacles. 3. The datasets, including those collected online and created by ourselves for model fine-tuning. 4. The approaches used and the final implementation of data augmentation for fine-tuning the model to improve performance in low-light environments. 5. A feedback module that translates detected obstacles into simple, real-time audio or tactile cues. This module will be integrated with the model to provide a user-friendly experience for visually impaired individuals. 6. A functional prototype of the visual assistance system, which integrates the optimized model, filtering algorithm, and feedback mechanism. 7. All code implementations will be submitted in a GitHub repository, which will include detailed usage instructions and technical documentation. 8. A comprehensive project report documenting the problem, objectives, methodology, experiments, results, and conclusions. |

**2. DATA COLLECTION PLAN**

Describe how you plan to collect the data or where you will source it from. Outline any preprocessing that will be necessary to make the data usable.

| * **Data Source** | 1. BDD100K (Berkeley DeepDrive Dataset)  * An open-source dataset suitable for road object detection. * Contains labeled images from various cities * Covers a wide range of lighting conditions and weather scenarios (including rain, fog, and nighttime scenes) * <https://www.vis.xyz/bdd100k/>  1. AED20K  * Comprehensive scene parsing dataset released in 2017 by MIT's Computer Science and Artificial Intelligence Laboratory * It stands out for its diverse range of scenes, including both indoor and outdoor environments. * <https://ade20k.csail.mit.edu/>  1. Cityscapes  * A large-scale dataset focused on urban street scenes, released in 2016 by Daimler AG R&D in collaboration with several German institutions. * https://www.cityscapes-dataset.com/  1. Supplementary Custom Dataset:  * Focusing on addressing the limitations of existing datasets, such as missing classes or scenarios where the model underperforms based on our testing. |
| --- | --- |
| **Dataset Size & Characteristics** | 1. BDD100K:  * 100,000 HD video sequences, 40s each, including more than 100 million frames * 100,000 images * Diverse weather conditions and times of day  1. AED20K  * more than 27,000 images spanning 365 different scenes * 707,868 unique objects from 3,688 categories * It offers a diverse range of complex indoor and outdoor scenes, high-quality manual annotations, hierarchical object categorization, and rich metadata.  1. Cityscapes  * more than 25000 images * Cityscapes is a large-scale dataset focused on urban street scenes, featuring high-resolution images with fine-grained pixel-level annotations for 30 classes across 50 cities.  1. Supplementary Custom Dataset:  * Target size: 2,000 images * Variety: Different types of low-lying obstacles, temporary obstacles, and various lighting conditions |
| **Data Collection**  **(If applicable) /**  **Annotation** | Currently, we believe that the datasets described above are sufficient for our project. However, if we eventually discover that there are missing scenarios, missing targets, or unusable data in these datasets, we will consider collecting our own dataset. Here's our plan:   * **Plan Dataset**   + Determine what target objects or scenes we need to collect images of based on our requirements   + Decide how many images we need based on the size and performance of our model * **Data Collection**   + Use cameras with uniform pixel resolution (can be webcams or smartphones) to capture images * **Data Annotation**   + We currently believe that our annotation method should be bounding boxes   + We will use LabelImg to label the captured images * **Data Augmentation**   + This step is actually applicable to the data in our existing datasets as well   + Increase the model's generalization ability by scaling and transforming images   + Enhance the model's generalization ability by changing brightness, contrast, adding noise, or using techniques like mosaic |

**3. METHODS**

Outline the specific computer vision techniques and algorithms you plan to use.

| **Proposed Approach** | * Object Detection   + Use deep learning models to identify and locate obstacles in the environment. * Distance Estimation   + Using dual cameras to estimate the distance between obstacles and the user. * Image Enhancement   + Improve image quality under low-light conditions to enhance model performance in various environments. * Model Efficiency Optimization   + Increase the real-time processing capability of the model on portable devices. * Interactive Feedback   + Convert detected obstacle information into user-friendly audio or tactile feedback. * Transfer Learning   + Fine-tune pre-trained models for specific tasks. * Data Augmentation   + Expand and diversify the training dataset to improve the model’s generalization ability. |
| --- | --- |
| **Algorithms/**  **Models** | 1. Model we Choose:   We will use YOLO-v8-nano as our base model, evaluate the capability of the pre-trained model, and perform transfer learning based on it. We will fine-tune the model according to our specific needs and the dataset we use.   1. Data Processing:    1. We need to filter the selected dataset, as not all object classes are relevant to our task. We will select images that contain the classes we care about.    2. Different datasets may have different formats, so we need to standardize image dimensions, annotation formats, and class indices across the datasets.    3. We will apply image augmentation techniques such as geometric transformations, color and brightness adjustments, noise addition, blurring and sharpening, occlusion simulation, MixUp, Mosaic, etc., to improve the model’s generalization capability.    4. The dataset will be split into training, validation, and test sets. 2. Model Optimization:   After the model is trained, we will make lightweight adjustments based on its performance on edge devices.   * 1. Frames can be skipped during processing to maintain real-time performance.   2. We will perform model pruning using techniques like Network Slimming or L1-norm based channel pruning.   3. The model architecture can be adjusted, such as adopting MobileNet as the backbone.  1. Distance Estimation:    1. We plan to use the SGM (Semi-Global Matching) algorithm to estimate the distance between the user and obstacles.    2. Based on the distance and orientation of the obstacles, we will filter out which obstacles need to be alerted to the user. 2. User Feedback Interaction: 3. After filtering obstacles based on distance estimation, we will provide audio cues or TTS (Text-to-Speech) to notify the user of the type and distance of objects ahead. |
| **Software/Tools:** | * Programming Language: Python * Deep Learning Framework: PyTorch * YOLO-specific Tool: Ultralytics YOLOv8 * Data Processing and Augmentation: OpenCV, Albumentations, NumPy * Dataset Management and Processing: LabelImg, Roboflow * Edge Device Deployment: PyTorch Mobile * Stereo Vision and Distance Estimation: OpenCV * Audio Processing and TTS: pyttsx3 * Development Environment and Version Control: Google Colab, GitHub |

**4. EVALUATION PLAN**

Explain how you plan to evaluate the performance of your application or model, and compare it with existing solutions. Define metrics and benchmarks.

| **Evaluation Metrics** | 1. Object Detection Performance:    1. Mean Average Precision (mAP) at different IoU thresholds (e.g., 0.5, 0.75, 0.5:0.95)    2. Precision: TP / (TP + FP)    3. Recall: TP / (TP + FN)    4. F1 Score: 2TP / (2TP + FP + FN)    5. Performance of the above metrics in different lighting conditions, such as outdoors at dusk, low light indoors, and high-brightness environments. 2. Real-Time Performance:    1. Frame Rate (FPS): Processing speed on the target edge device    2. Inference Latency: Processing time per frame 3. Distance Estimation Accuracy:    1. Mean Absolute Error (MAE) |
| --- | --- |
| **Testing Plan**  **(if applicable)** | 1. Technical Performance Testing    1. Laboratory Condition Testing       1. Objective: Evaluate the basic performance of the system in a controlled environment       2. Use a standard test dataset to evaluate object detection performance       3. Place objects at different distances and angles to test distance estimation accuracy       4. Simulate various lighting conditions to test the system's performance under different brightness levels indoor    2. Hardware Performance Testing       1. Objective: Evaluate the system's operational efficiency on the target edge hardware       2. Test battery life       3. Monitor CPU, GPU, and memory usage       4. Test performance under different resolution and frame rate settings 2. Environmental Adaptability Testing    1. Indoor Testing       1. Scenarios: Home, office, mall, etc.       2. Tests: Furniture detection, obstacle detection, signal detection, step detection, etc.    2. Outdoor Testing       1. Scenarios: Streets, parks, intersections, etc.       2. Tests: Stairs detection, pedestrian avoidance, traffic signal recognition, etc.    3. Special Condition Testing       1. Low-light environments: Overcast days, dimly lit indoor spaces       2. Complex backgrounds: Narrow streets, cluttered indoor environments, crowded streets 3. User Experience Testing    1. Objective: Preliminary evaluation of the system's user-friendliness    2. Use the system while wearing a blindfold    3. Design specific tasks (e.g., walking to a designated location) |

**5. PRESENTATION PLAN**

Explain how your group will present your results, both visually and in writing. Include plans for a poster, a written report, and a short video.

| **Poster** | * Key Elements:   + Problem Statement: Challenges faced by visually impaired in navigation   + Methodology: YOLO-based object detection and distance estimation techniques   + Results: Performance metrics, comparison with baselines   + Conclusion: Key findings and future improvements * Visual Elements:   + System architecture diagram   + Sample detection results in various conditions   + Graphs comparing performance metrics   + Confusion matrix for object detection |
| --- | --- |
| **GitHup Repo** | https://github.khoury.northeastern.edu/panxuedong418/CS5330\_F24\_Group5\_Project |
| **Write Up** | * IEEE 2-column format * Sections:   1. Abstract   2. Introduction   3. Related Work   4. Methodology   5. Experiments and Results   6. Discussion   7. Conclusion and Future Work * Each team member will contribute to specific sections |
| **Video Clip** | 1-minute promotional video highlighting:   1. Problem statement (15 seconds) 2. Brief system overview (15 seconds) 3. Key features and benefits (15 seconds) 4. Demo of the system in action (15 seconds) |

**6. TENTATIVE SCHEDULE**

Provide a brief timeline, allocating approximate dates for each stage of the project.

| **KEY MILESTONE** | **START** | **FINISH** |
| --- | --- | --- |
| Form Project Team | 09/12/2024 | 09/20/2024 |
| Finalize Project Plan / Kick Off | 09/21/2024 | 10/10/2024 |
| Define Phase (e.g., Problem Definition, Dataset, Pre-processing, initial setting) | 10/11/2024 | 10/25/2024 |
| Development Phase (develop and test initial model) | 10/26/2024 | 11/10/2024 |
| Analysis Phase (model tuning, optimization, and evaluation) | 11/11/2024 | 11/20/2024 |
| Control Phase (finalize results) | 11/21/2024 | 11/30/2024 |
| Presentation Phase | 12/01/2024 | 12/12/2024 |

**REFERENCES**

1. J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in \*Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)\*, 2016, pp. 779–788.

2. A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal speed and accuracy of object detection," \*arXiv preprint arXiv:2004.10934\*, 2020.

3. C. Godard, O. Mac Aodha, and G. J. Brostow, "Unsupervised monocular depth estimation with left-right consistency," in \*Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)\*, 2017, pp. 270–279.

4. A. G. Howard, M. Zhu, B. Chen, et al., "MobileNets: Efficient convolutional neural networks for mobile vision applications," \*arXiv preprint arXiv:1704.04861\*, 2017.

5. Yu, F., Chen, H., Wang, X., Xian, W., Chen, Y., Liu, F., ... & Darrell, T. (2020). BDD100K: A diverse driving dataset for heterogeneous multitask learning. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 2636-2645).