Helping the Visually Impaired with Object Detection

Jack Arteaga

Brookdale Community College

May 6th, 2022

Advisors: Dr. Christopher Ochs, Professor Karina Ochs, Professor Ana Teodorescu

Email: arteagajack2@gmail.com

Abstract

Those who have visual impairments are at times restricted in movement due to the inability to distinguish objects in front of them. Object detection, a computer vision technique, can be used to detect and avoid objects. Using off the shelf hardware and a pre-trained object detection model, an object avoidance system was built to facilitate the navigation through a space for a visually impaired user. The system can process a live video feed where it can detect and calculate the distance of objects in the database from the camera/user. Once a certain distance threshold is crossed an alert is triggered communicating an object is ahead. This report evaluates the current approaches to object detection and demonstrates how it can be implemented in an object avoidance system to help the visually impaired. The objective is to determine if off the shelf hardware and object detection techniques can be used to provide live object avoidance of everyday household items to the visually impaired.

I. Introduction

According to the Centers for Disease Control and Prevention, approximately 12 million people, 40 years and over, in the United States have vision impairment, including 1 million who are blind [1]. Living in a dominantly seeing world creates limited access to mobility for the visually impaired who may not navigate the same as a seeing person. There are systems and accommodations currently in place to compensate for the difference in mobility. This includes assisted guidance such as a seeing-eye dogs or a personal care assistant. Object detection can be used as a tool to offer a more independent lifestyle for those who need assisted guidance to navigate a certain space.

Using object detection an object avoidance system can be built to help the visually impaired navigate a space more independently. There are current techniques to implementing object detection for a variety of purposes and similar domains. This project was a preliminary evaluation of those techniques and an attempt to implement them into a system to help the visually impaired navigate a space such as their home. Using off the shelf hardware and various open-source software an object avoidance system was built. To begin building that system object detection was first defined.

II. Background

A. Defining Object Detection

Object detection is a computer vision technique for locating instances of objects in images or videos [2]. To begin identifying objects they must first be defined. What is an object? An object will be defined by a use case. In this instance the objective is to detect and avoid everyday household items. The objects will be a broad number of common items found in a household that may be in the way of a visually impaired person when traversing a space in their home (i.e., a chair, couch, table, stool). The object must be classified and localized in an image for it to be detected. Object detection models can be used to classify objects. An object detection model is a collection of mathematical representation of data and an algorithm that can classify objects in an image, typically trained using thousands of images. They can be created and trained to fit certain use case. There are pre-trained object detection models which have been trained according to the needs of previous use cases and can be applied to use cases in similar domains.

B. Object Detection Models

To begin using an object detection model both approaches, creating and training a model and pre-trained models, must be evaluated.

Creating and training a model begins with identifying a use case. Once a use case is defined, the data that will be used to train the model for the use case must be collected and sorted. Depending on the use case the type of data may vary. This project's use case would mean collecting thousands of images of individual household items. Each image must then be classified, labeled, and sorted in a database. At this point the algorithm that will classify the objects in an image must be created. This is typically done using a neural network [3].

To use a pre-trained model a use case must once again be identified. After identifying the use case a pre-trained model that has a been trained in a similar domain can be used to detect objects. A pre-trained model is essentially "plug and play" because it has already been trained on previous data. Using a pre-trained model that sufficiently meets the use case that it will be used for may save time and resources that would otherwise be used to collect and train data.

Two pre-trained models were tested for this project, You Only Look Once (YOLO)v3 [4] and MobileNet-SSD (Single Shot Detectors) [5]. YOLOv3 is a real-time object detection algorithm that identifies specific objects in videos, live feeds, or images. YOLOv3 uses features learned by a deep convolutional neural network to detect an object [3]. The MobileNet-SSD model is a Single-Shot multibox Detection (SSD) network intended to perform object detection. This model is implemented using the Caffe* framework [5].

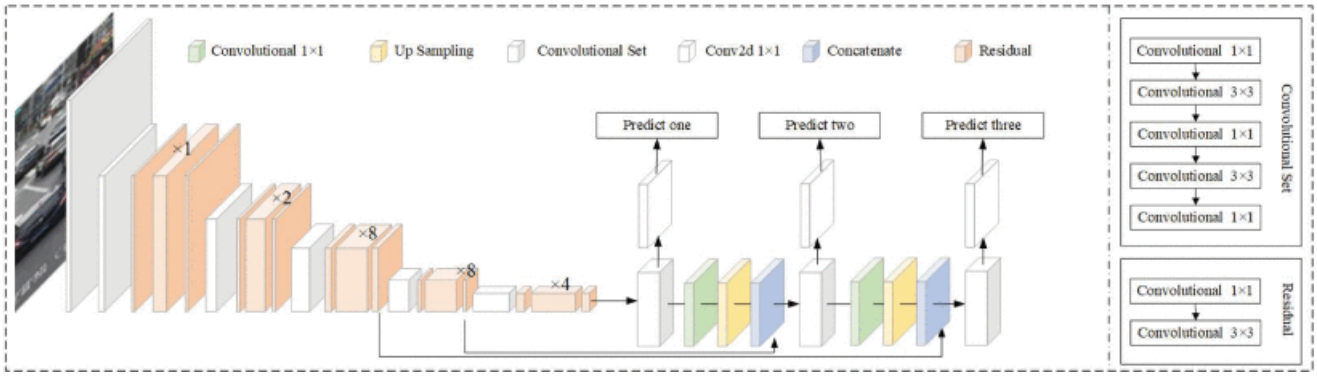

Figure 1: Structure detail of YOLOv3.It uses Darknet-53 as the backbone network and uses three scale predictions [4].

C. Current Technology

There are currently applications, products and techniques trying to assist the visually impaired with everyday tasks. Mobile apps such as Be My Eyes and TapTapSee use cellphones to describe objects in front of the user. Be My Eyes is a free app that connects blind and low-vision people with sighted volunteers and company representatives for visual assistance through a live video call [6]. TapTapSee, powered by the CloudSight Image Recognition API, utilizes cellphones camera and VoiceOver functions to take a picture or video of anything and identify it out loud for the user [7]. Other companies like biped are attempting to create an AI copilot for blind and visually impaired people. The biped smart harness predicts your surroundings a few seconds ahead and generates intuitive feedback through spatial sounds [8].

III. Research

A. Evaluating Approaches

Based on empirical analysis, a pre-trained model better suited this project. The objective is to detect everyday household items that may typically be in the way of a visually impaired person in their home. Training a model on thousands of household items would be impractical due to the data and computing requirements. For this project and time constraint, the time allocated for data

collection and model training would take up what would be used to build a complete object avoidance system. Pre-trained open-source models exist to detect everyday household items, achieving a similar result to creating and training a model. Based on this analysis a pre-trained model was chosen.

As mentioned previously, two pre-trained models were tested for this project, YOLOv3 and MobileNet-SSD. Scripts were written to process a live video feed and test the object detection of each model using every day household items. YOLOv3 detected a larger number of objects more accurately than MobileNet-SSD. YOLOv3 also detected objects at a faster frame rate than MobileNet-SSD. Based on these tests it was concluded that YOLOv3 was a faster and more accurate object detection model for everyday household items than MobileNet-SSD. YOLOv3 was chosen as the pre-trained model to be used in the object avoidance system.

B. Algorithm

Once the pre-trained object detection model was chosen the object avoidance system began being developed. The system must detect objects and calculate their distance from the user. To calculate distance from a two-dimensional image captured from a webcam a reference image must be used. A database of each object being calculated for distance must be created. The database must contain a reference image of that object with a known distance from the camera and the objects dimensions. This will later be used to calculate distance in a live feed for the algorithm. Once objects are detected and their distance calculated an alert can be triggered once a certain distance threshold is crossed.

Processing a Live Video Feed and Detect Objects

The first step is to process a frame from a live video feed to detect objects. The pre-trained model, in this case the YOLOv3 object detection model, will classify objects in the image using weight files. As seen in figure 1 the frame is divided into a grid shape by the model to begin classification.

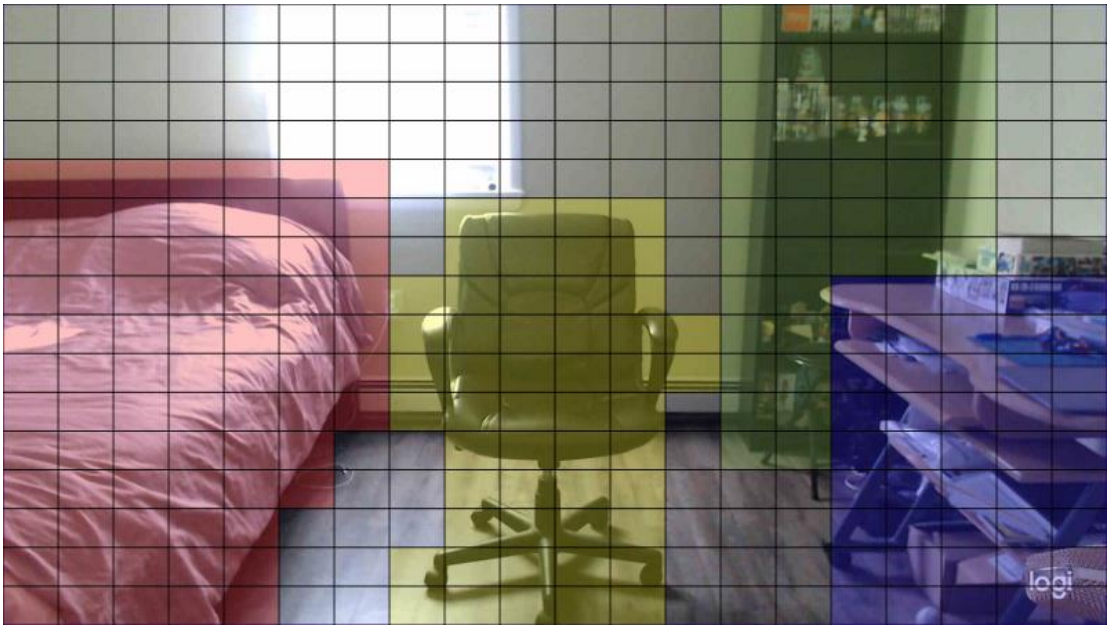

Figure 1: Frame from a video feed capturing a bedroom. Classification grid demonstrated.

Find Objects in Database

Once objects are classified only objects found in the database will be stored and filtered through to be further processed as seen in figure 2.

Figure 2: Database with numbered objects (right) and frame finding objects (left)

Locating Object in Frame

The location of the object in frame must be found. The YOLOv3 model while classifying will also produce coordinates as an estimation to where the object may be in the image. The object may be detected multiple times therefore multiple coordinates are produced as seen in in figure 1 (left). The system pulls the coordinates demonstrated in figure 1 (right), to further refine the location of the object.

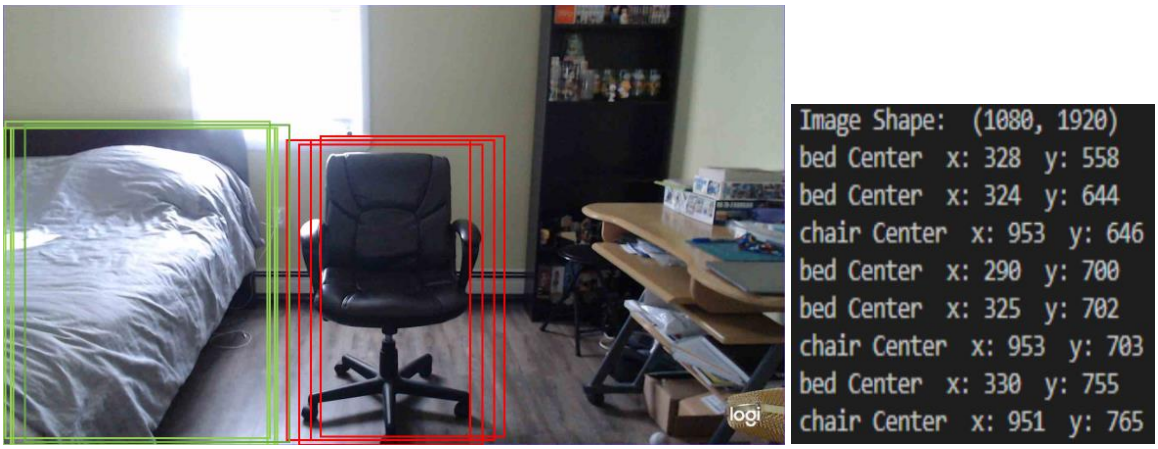

Figure 3: Estimation of objects location demonstrated (left) and output of coordinates for each detection of the objects (right)

Refine Location and Label

A single bounding box must be created for each object to be accurately represented using its found location. To achieve this non-maximum suppression (NMS) is used to pass detections if they haven't already been detected. Alongside NMS a label and confidence are attached to bounding box as shown in figure 4. The label is pulled from the database when the object is detected. The confidence is a score that the YOLOv3 model produces when detecting an object. It is an evaluation of how accurate the detection may be.

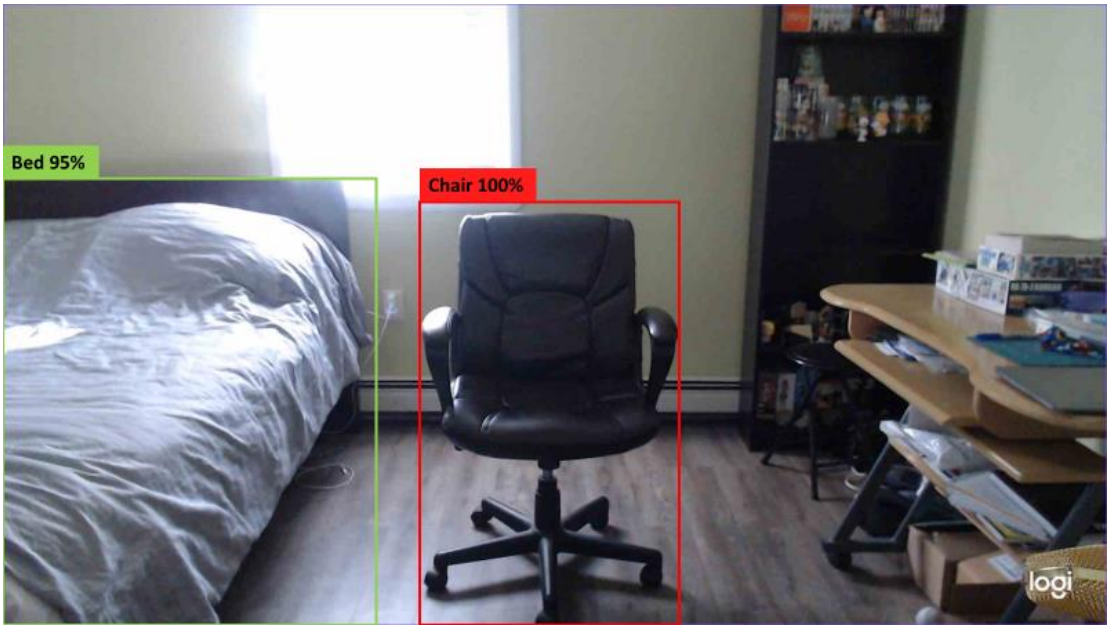

Figure 4: Objects detected displaying a bounding box, label and confidence

Calculate Distance

If the object being detected is a known object (known referring to its reference distance and dimensions known) than the distance from the user/camera is computed. To calculate distance from a two-dimensional image the fluctuation of pixel width that the object takes up in frame is compared to the reference image. There must be a compensation for the proportional difference of

the true size and size in frame of the object. This is the focal length. Focal length is determined with the reference as a constant. The focal length formula is:

$$\textbf{Focal } \boldsymbol{Length} = \frac{(\boldsymbol{widthInPixels} * \boldsymbol{measuredDistance})}{\boldsymbol{actualWidth}}$$

Figure 5: Chair measured to be 6ft away (431 pixels wide and 677 pixels tall in frame)

Using the figure 5 as the reference image for the chair the calculation is:

$$1293 \text{ pixels} = \frac{(431\ pixels * 72\ inches)}{24\ inches}$$

Once there is a change objects pixel width the distance formula can be used to calculate the new distance:

$$\boldsymbol{Distance} = \frac{(\boldsymbol{actualWidth} * \boldsymbol{focalLength})}{\boldsymbol{widthInPixels}}$$

If the pixel width of the object increases, it can be inferred that the object has come closer to the user therefore the distance between the object and user decreases. Inversely if the pixel width decreases, it is inferred the object has gone farther away from the user increasing the distance between each other.

Trigger an Alert

The final step is to determine if the user has crossed the distance threshold. If the user has, the system will trigger an alert.

Figure 6: Object crossing distance threshold triggering an alert (862 pixels wide in frame)

Using figure 6 and the distance formula mentioned previously, the distance for the chair can be calculated.

$$36\ inches = \frac{(24\ inches\ *\ 1293\ pixels)}{862\ pixels}$$

As stated before, the pixel width in the frame has increased therefore the distance from the object (the chair) has decreased from the user (the camera) from six feet to three. The threshold being set to four feet means the current distance of the chair would trigger an alert as demonstrated in the image above.

C. System Implementation

The algorithm was implemented into the system using various open source, off the shelf hardware and software. The frames are captured using Python programming language and OpenCV a Computer Vision library. Once captured each frame is processed using the pre-trained YOLOv3 algorithm to detect objects. Only objects with a confidence greater than 60% will display a bounding box. These parameters can be adjusted. This was done to further refine the detection of objects and make sure they are as accurate as possible without being too rigid in detection, as not all objects will be in the perfect condition to be detected.

Hardware components consisted of a Logitech C922pro Webcam which captured footage at 1080p, an AMD Ryzen 5 3600 CPU and a GTX 1660 Ti Graphics Card. All components ran on a desktop machine with Windows 10 as the operating system. Visual Studio Code was the code editor used to develop the object avoidance system. Source code for this object avoidance system can be found on GitHub at https://github.com/uandmeatadeli/ObjDetectionProject .

IV. Results

An off the shelf, pre-trained object detection model can be used to build an object avoidance system to help the visually impaired. As demonstrated through the algorithm subsection of section III, the system can process a live video feed and detect objects. It can calculate the size

and distance of known objects. Lastly the system can trigger and alert to avoid objects once a certain distance threshold is crossed.

Various limitations were uncovered during the project. One of the first limitations was calculating distance of an object from a two-dimensional image. The solution to that was to have objects previously measured for distance and size to then store that information in a database to use as reference later in the process. Hardware restriction also played a role in limitations, specifically the camera being used to capture the live video feed. Using a regular webcam restricted the depth and spatial capture of footage. Capturing depth would potentially increase the accuracy of detection as well as provide more data of the space being captured. There are a lot of "blind spots" in the system such as walls, objects not detected by the YOLOv3 algorithm, or simply an object approaching at a faster speed than the frames being processed.

V. Future Work

Although an object avoidance system was built this was a limited scope project and a preliminary evaluation to the techniques that can be used. Further evaluations must be done to develop a more sophisticated system. Using techniques like triangulation, capturing objects positions in three-dimensional space could potentially improve the system. Investigation of different hardware such as stereo vision cameras and lidar devices can be done in the future. Expanding the database to capture, detect and calculate distances from objects to the user is something that over time can be an addition to the system. As discussed in the background section, further evaluation can be done on creating and training an object detection model. Ultimately the goal is to develop a system that can be tested with users (people with visual impairments) safely.

VI. Conclusion

A collision avoidance algorithm and system were developed using a pre-trained object detection model, to help the visually impaired. Objects can be detected and their distance from the ser can be computed. The system still needs to detect objects more consistently and accurately before performing user tests. Further evaluation is still required to assess the utility of the system when implemented in the real world.

Acknowledgements

This work was supported by NASA's New Jersey Space Grant Consortium and Brookdale Community College. I would like to thank Dr. Christopher Ochs Professor, Karina Ochs, and Professor Ana Teodorescu for their continued assistance and support throughout this project.

References

1. Centers for Disease Control and Prevention. (2020, June 9). *Fast facts of common eye disorders*. Centers for Disease Control and Prevention. Retrieved May 5, 2022, from https://www.cdc.gov/visionhealth/basics/ced/fastfacts.htm#:~:text=Approximately%2012%20million%20people%2040,due%20to%20uncorrected%20refractive%20error

2. *What is object detection?* What Is Object Detection? - MATLAB & Simulink. (n.d.). Retrieved May 4, 2022, from https://www.mathworks.com/discovery/object-detection.html

3. Sultana, Farhana, Abu Sufian, and Paramartha Dutta. "A review of object detection models based on convolutional neural network." Intelligent computing: image processing based applications (2020): 1-16.

4. Meel, V. (2022, April 19). *Yolov3: Real-time object detection algorithm (what's new?)*. viso.ai. Retrieved May 4, 2022, from https://viso.ai/deep-learning/yolov3-overview/#:~:text=YOLOv3%20AI%20models-,What%20is%20YOLOv3%3F,Joseph%20Redmon%20and%20Ali%20Farhadi.

5. *Mobilenet-SSD - openvino™ toolkit*. OpenVINO. (n.d.). Retrieved May 4, 2022, from https://docs.openvino.ai/2021.3/omz_models_model_mobilenet_ssd.html

6. *Be my eyes - see the world together*. Be My Eyes - See the world together. (n.d.). Retrieved May 5, 2022, from https://www.bemyeyes.com/

7. *Taptapsee*. TapTapSee. (n.d.). Retrieved May 5, 2022, from http://taptapseeapp.com/

8. *"Biped Unveils an AI Copilot for Blind and Visually Impaired People at CES 2022." Designboom, 11 Jan. 2022, https://www.designboom.com/technology/biped-ai-copilot-blind-visually-impaired-people-ces-2022-01-10-2022/.*

9. *Naser, et al. "Yolo Object Detection with Opencv." PyImageSearch, 5 Feb. 2022, https://pyimagesearch.com/2018/11/12/yolo-object-detection-with-opencv/.*