Augmented Reality Object Detection and Labeling

CONNOR REINHOLDTSEN and ANDY STANCIU, University of Washington, USA

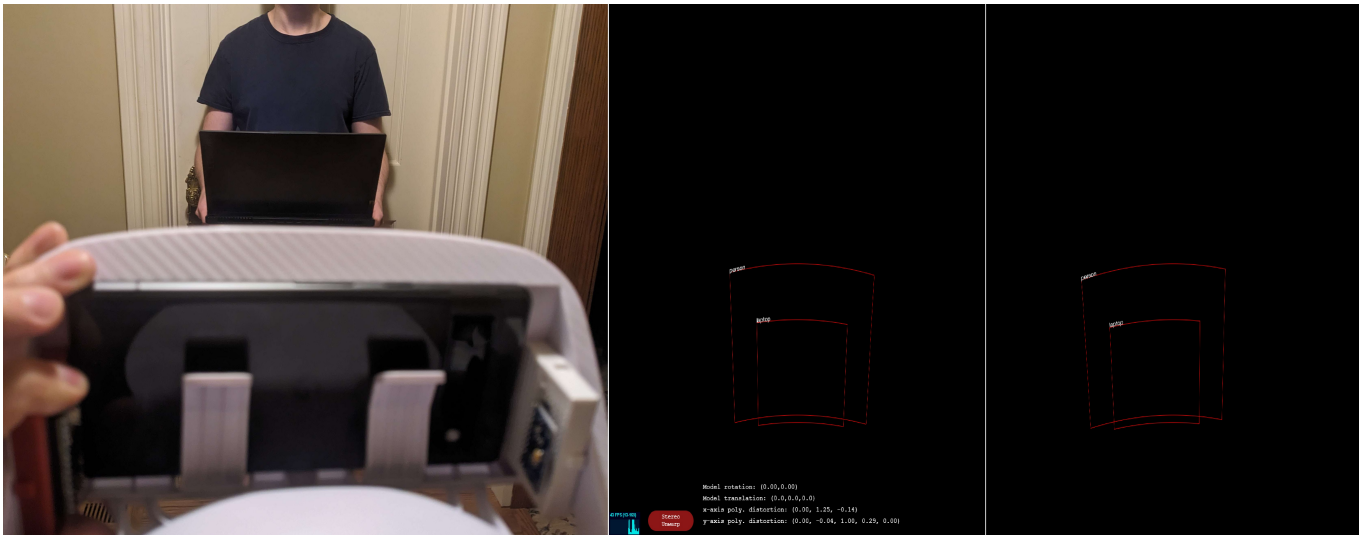

Fig. 1. Our CSE 493V AR headset detects and labels objects in real time using a front-facing camera. The system identifies a person and a laptop, displaying their bounding boxes and labels within the AR view, which applies distortion correction and reflects off the headset's lens for the viewer.

We integrate object detection and labeling into the CSE 493V AR headset, enabling real-time identification of objects within the user's field of view. Using YOLOv8, we detect objects and overlay bounding boxes and labels in augmented reality. YOLO's single-shot detection minimizes latency, allowing seamless integration of virtual elements into the physical environment. A front-facing camera captures the live feed, which is processed through the YOLO model. The final output is rendered to the AR headset with distortion correction to ensure alignment with the real world.

1 Introduction

State-of-the-art object detection models have rapidly evolved to meet the dual demands of high accuracy and real-time processing. Among these, YOLOv8 distinguishes itself as a promising candidate by building on the innovations of its predecessors while integrating cutting-edge techniques such as an optimized CSPDarknet53 backbone, advanced data augmentation, and novel training strategies often referred to as "bag of freebies" and "bag of specials." These enhancements allow YOLOv8 to achieve a superior balance between detection precision and speed, making it highly effective for applications ranging from autonomous vehicles to smart surveillance systems. Overall, YOLOv8's ability to deliver robust performance in real-time environments marks a significant milestone in the progression of object detection technology.

We propose to leverage the capabilities of YOLOv8 to develop an augmented reality (AR) object detection and labeling system. Our system is integrated into the CSE 493V AR headset, enabling real-time identification of objects within the user's field of view. By combining object detection with AR, we aim to enhance the user's perception of the physical environment by overlaying virtual elements that provide additional information. This approach could be used in a variety of applications, including:

- **Accessibility** – Assisting visually impaired users by identifying and labeling objects in their environment. Further extensions could involve narrating object descriptions to assist users in navigating their surroundings.
- **Education and Training** – Enhancing learning experiences by providing real-time annotations for objects in AR, useful in fields like medicine, engineering, and mechanics.
- **Retail and Shopping** – Helping users recognize and learn about products in stores by overlaying relevant information.
- **Manufacturing and Maintenance** – Assisting workers by identifying components.
- **Gaming and Interactive Experiences** – Enabling interactive AR experiences by integrating virtual overlays onto real world objects.

Our approach builds on the existing CSE 493V AR headset infrastructure, which provides the necessary hardware components, stereo rendering, and distortion correction. We utilize a front-facing camera to capture the live feed, which is processed through the YOLO model. The computed bounding boxes and labels are rendered to the AR headset with stereoscopic rendering and distortion correction to ensure alignment with the real world.

Authors' Contact Information: Connor Reinholdtsen, creinh@uw.edu; Andy Stanciu, andys22@uw.edu, University of Washington, Seattle, USA.

1.1 Contributions

- We utilized YOLOv8 to develop an AR object detection and labeling system integrated into the CSE 493V AR headset. This system enables real-time identification of objects within the user's field of view.
- We utilized a front-facing camera to capture the live feed, which is processed through the YOLO model. The final output is rendered to the AR headset with distortion correction to ensure alignment with the real world.
- We demonstrated the potential applications of our system in various domains, including accessibility, education, retail, manufacturing, and gaming.

2 Related Work

2.1 CSE 493V AR Headset

We built our AR object detection and labeling system on top of the CSE 493V codebase and AR headset [University of Washington 2025], which consists of the following components:

- **AR HMD enclosure** – Houses the other components, provides a mount for the display phone, and includes a lens for the user to view the AR scene.
- **Display phone** – Acts as the AR headset's screen and includes a front-facing camera for capturing the live feed.
- **IMU** – Tracking head movement.
- **Microcontroller** – Processing IMU data.

The CSE 493V infrastructure provided us with a solid foundation for developing our AR object detection, including the necessary hardware, stereo rendering, and distortion correction.

Fig. 2. Hardware components of the CSE 493V AR headset.

2.2 YOLOv8

YOLO (You Only Look Once) is a single-shot detection model known for its speed and accuracy [Jocher et al. 2023]. Unlike traditional region proposal-based approaches, YOLO directly predicts bounding boxes and class labels in a single pass through the network, making it well-suited for real-time applications. YOLOv8 introduces architectural improvements, including a more efficient backbone and improved anchor-free detection, enhancing both accuracy and inference speed.

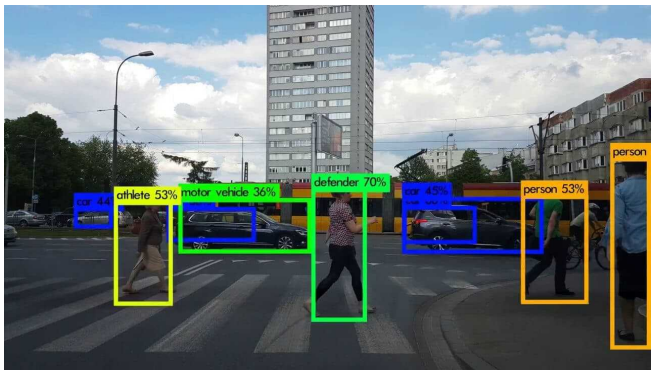

Fig. 3. YOLO computes bounding boxes and class labels for an input image.

2.3 MiDaS

MiDaS (Monocular Depth Estimation via Scale-Invariant Learning) is a deep learning model designed to estimate depth from a single RGB image [Birkl et al. 2023]. Unlike stereo-based or LiDAR depth estimation methods, MiDaS uses a convolutional neural network trained on diverse datasets to infer depth without requiring multiple viewpoints. It leverages a scale-invariant loss function, allowing it to generalize well across different environments and lighting conditions.

For AR applications, MiDaS offers an attractive solution for estimating scene depth without additional hardware. By predicting depth from a single camera feed, it could enable more accurate positioning of virtual objects within the physical space. However, MiDaS produces depth maps in relative terms rather than absolute metric measurements. This means that while it can infer the relative distance between objects in a scene, the actual scale must be manually calibrated. Frequent recalibration would be required to maintain accuracy in an AR environment, which adds complexity to user interactions. Due to this limitation, we opted to use a fixed affine transformation for mapping detected objects into the AR scene instead of relying on MiDaS for depth estimation.

Fig. 4. MiDaS estimates depth from a single RGB image.

3 Method and Implementation Details

We modified the CSE 493V codebase to initialize a second WebSocket which receives object detection results from the YOLO server (`object_detection` subproject). The sent bounding box coordinates are of the form $[x_{\min}, y_{\min}, x_{\max}, y_{\max}]$, and each coordinate is normalized to the range $[0, 1]$. The AR frontend then renders the detected objects with bounding boxes and labels in the AR view. To determine the final coordinates of the bounding box on the screen, the frontend applies an affine transformation to the bounding box coordinates and applies distortion correction. This affine transformation must be callibrated based on the specific front-facing camera to ensure the bounding boxes line up with real life objects seen through the lens. An example message sent from the YOLO server to the AR frontend is shown below:

```
[
  {
    "label": "person",
    "confidence": 0.98,
    "bbox": [0.2, 0.5, 0.8, 0.9]
  },
  {
    "label": "laptop",
    "confidence": 0.95,
    "bbox": [0.23, 0.45, 0.6, 0.6]
  }
]
```

These bounding boxes will be rendered in the AR view with the corresponding labels. The rendered bounding boxes and labels reflect off the headset's lens to surround the detected objects in the user's field of view.

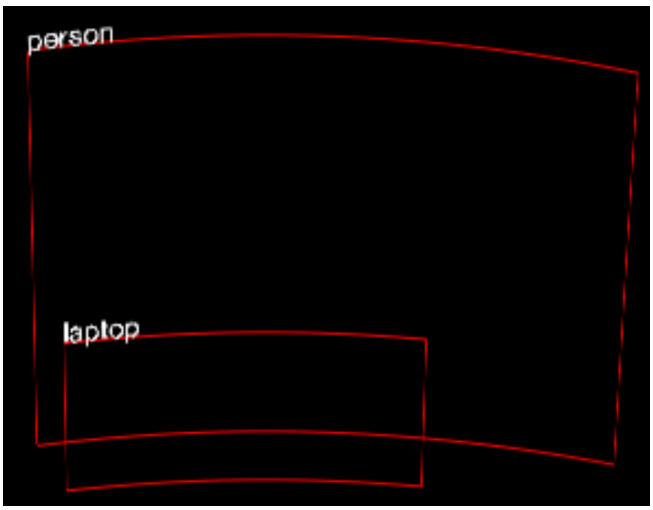

Fig. 5. Bounding boxes and labels for a person and laptop are rendered to the left lens.

3.1 Hardware

We simply used the CSE 493V AR headset [University of Washington 2025] hardware components and a display phone with a front-facing camera. The display phone runs the AR frontend and streams the camera feed to the YOLO server over HTTP.

3.2 System Architecture

Our AR object detection and labeling system consists of the following main components:

- **Camera Feed** – Captures live video frames from the AR headset's camera for processing and streams them to the YOLO object detection server over HTTP in Motion JPEG (MJPEG) format. We ran the IP Webcam app on the display phone to serve the camera feed.
- **YOLO Object Detection Server** – Runs the YOLOv8 model to detect objects in the video stream and outputs bounding boxes with class labels. Sends the detection results to the AR frontend via WebSockets.
- **CSE 493V IMU Server** – Collects inertial measurement unit (IMU) data and sends the computed head rotation to the AR frontend via WebSockets.
- **CSE 493V Frontend** – The user-facing AR interface that overlays detected object labels and bounding boxes in the augmented reality view with stereoscopic rendering and distortion correction. It receives object detection results and IMU data via WebSockets. The frontend display is streamed to the display phone with spacedesk.

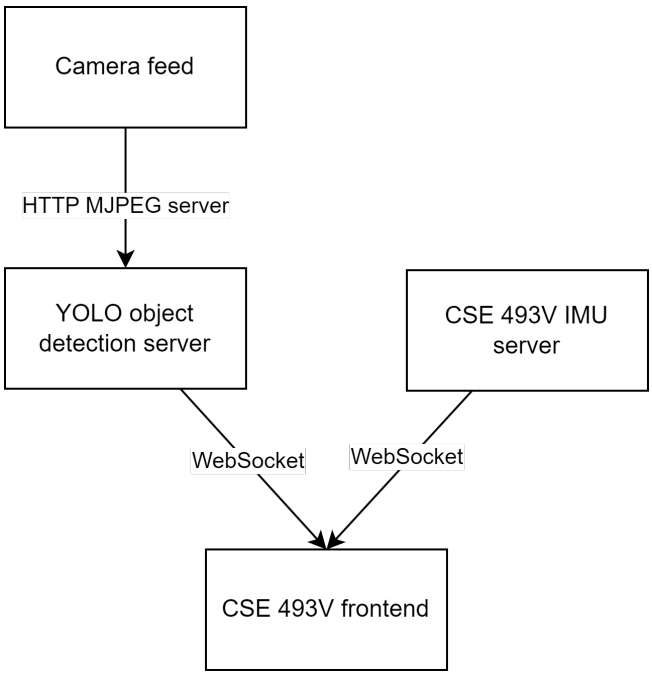

Fig. 6. System architecture of our AR object detection and labeling system.

4 Evaluation of Results

We evaluated the inference time for various YOLOv8 models on an NVIDIA RTX 3050 Ti GPU. Table 1 shows the average inference time for different YOLOv8 models. To ensure real-time performance, we decided to use the YOLOv8-nano model, which prioritizes speed over accuracy for real-time inference. If we had more powerful hardware, we would have liked to leverage a larger model for improved detection accuracy.

YOLO Model	Avg. Inference Time (ms)
YOLOv8-nano	43.7
YOLOv8-small	87.4
YOLOv8-medium	190.2

Table 1. YOLO Model Inference Time on RTX 3050 Ti

We found that streaming the camera feed over HTTP at 60 frames per second (FPS) introduced significant latency, which dramatically impacted the overall system performance. While we weren't able to measure the exact latency due to the app we used for streaming not providing this information, we found that reducing the camera feed streaming framerate to about 20 FPS dramatically improved the overall responsiveness of the system.

4.1 Limitations

Some of the key limitations of our system include:

- The camera occasionally captures reflections off the headset lens, leading to false detections where the system mistakenly identifies the wearer as a detected person. Using an external camera instead of the front-facing camera of the display phone could mitigate this issue. We simply didn't have time to build a custom mount for an external camera.
- Latency accumulates due to the time taken to send camera data to the server and receive processed results. The USB port on our display phone was blocked off by the HMD enclosure, so we were forced to stream the camera feed wirelessly over HTTP. A phone with an exposed USB port would allow for a wired connection, reducing latency compared to a wireless network setup.
- A tradeoff was necessary between model size and inference speed. While a smaller model ensures real-time performance, it comes at the cost of reduced bounding box and labeling accuracy.
- The system does not utilize a stereo camera setup, limiting bounding boxes to 2D projections rather than full 3D spatial positioning. A stereo camera would improve depth estimation and enable more accurate object placement in the AR environment.
- Aliasing effects make text and bounding boxes harder to read in the headset. Implementing antialiasing techniques into the distortion correction could enhance clarity and visual quality.
- We currently use an affine transformation calibrated based on the camera to map detected objects into the AR scene. A stereo camera setup would allow for more precise 3D placement of bounding boxes without relying on a precomputed transformation.

5 Future Work

While our current implementation integrates YOLOv8-based object detection with the CSE 493V AR headset, several areas remain for future exploration and improvement.

5.1 Depth Estimation and 3D Object Placement

One limitation of our system is the absence of accurate depth estimation. While we explored MiDaS for monocular depth estimation, its reliance on relative depth rather than absolute measurements posed challenges. Future work could incorporate stereo depth sensing or integrate LiDAR for precise depth estimation. This would allow bounding boxes to be properly placed around detected objects in the 3D AR environment.

5.2 Multimodal Interaction and Feedback

Integrating multimodal interaction methods, such as hand gestures, could enhance usability. Future iterations could enable users to query detected objects using natural language processing models or interact with virtual overlays via gesture recognition.

5.3 Personalized and Context-Aware Detection

Our current model provides generic object detection. Future enhancements could include user-specific customization, allowing individuals to prioritize specific object categories. Additionally, integrating contextual awareness—such as detecting objects relevant to the user's task—could improve the system's practical applications in fields like education, retail, and manufacturing. For example, in a retail setting, the system could highlight products on a shopping list or provide additional information about items in the user's vicinity.

5.4 Extended Dataset and Fine-Tuning

While YOLOv8 is trained on a general dataset, fine-tuning it on domain-specific datasets could improve detection accuracy for specialized applications. For instance, datasets tailored for medical, industrial, or accessibility-focused use cases could significantly enhance performance in those areas.

5.5 User Studies and Usability Evaluation

Conducting user studies would provide insights into how our system performs in real-world scenarios. Evaluating usability, comfort, and effectiveness across different applications could guide refinements and future iterations.

6 Conclusion

Our project integrates real-time object detection and labeling into the CSE 493V AR headset using YOLOv8. The system overlays detected objects with bounding boxes and labels within the augmented reality display, which could have applications in enhancing users' perception of their environment. We explored the feasibility of depth estimation with MiDaS but ultimately opted for a fixed affine transformation due to the challenges of relative depth scaling without

stereo cameras. Our results demonstrate that real-time object detection in AR offers potential applications in accessibility, education, retail, and beyond.

References

Reiner Birkl, Diana Wofk, and Matthias Müller. 2023. MiDaS v3.1 – A Model Zoo for Robust Monocular Relative Depth Estimation. *arXiv preprint arXiv:2307.14460* (2023).

Glenn Jocher, Jing Qiu, and Ayush Chaurasia. 2023. *Ultralytics YOLO.* https://github.com/ultralytics/ultralytics

University of Washington. 2025. *CSE 493V GitLab Repository.* https://gitlab.cs.washington.edu/cse493v/cse493v-25wi Accessed: 2025-03-18.