Augmented Reality Object Detection and Labeling

PAUL G. ALLEN SCHOOL OF COMPUTER SCIENCE & ENGINEERING

Connor Reinholdtsen, Andy Stanciu

Problem

We propose an AR object detection system using YOLOv8 to enable real-time object labeling on the CSE 493V AR headset. This enhances user perception with applications in accessibility, education, retail, manufacturing, and gaming. By combining fast detection with AR visualization, we create an intuitive tool for real-world interaction.

Related work

CSE 493V AR headset

This system, developed in class, contains a display phone, front-facing camera, IMU for head tracking, a microcontroller, as well as stereo rendering and distortion correction.

YOLOv8

YOLO is a fast, single-shot object detection model that predicts bounding boxes and class labels in one pass, making it ideal for real-time applications. Specifically, YOLOv8 improves accuracy and speed with a more efficient backbone and anchor-free detection.

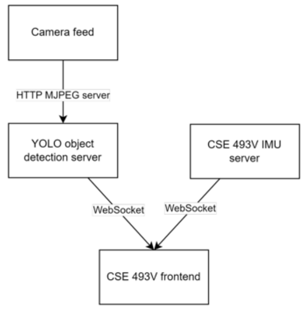
MiDaS

MiDaS estimates depth from a single RGB image using a CNN trained on diverse datasets. Unlike stereo or LiDAR methods, it infers relative depth without additional hardware. Since its depth maps requires significant manual calibration, we opted to use a fixed affine transformation for object placement instead.

Methodology

We modified the CSE 493V codebase, adding a second WebSocket that receives YOLO object detection results, with bounding boxes in the normalized format [x_{min}, y_{min}, x_{max}, y_{max}]. The AR frontend applies an affine transformation and distortion correction to align detected objects with real-world views. Bounding boxes and labels are then rendered in the AR view, reflecting off the headset's lens. An example message from the YOLO server:

```
[
  {
    "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]
  }
]
```



System architecture

- Camera feed – captures video from the AR headset and streams it via MJPEG using the IP Webcam app.
- YOLO object detection server – runs YOLOv8 to detect objects and sends results via WebSockets.
- CSE 493V IMU server – collects IMU data and sends head rotation updates via WebSockets.
- CSE 493V frontend – overlays labels and bounding boxes in AR with stereoscopic rendering and distortion correction, receiving data from WebSockets and streaming the display via spacedesk.

Results

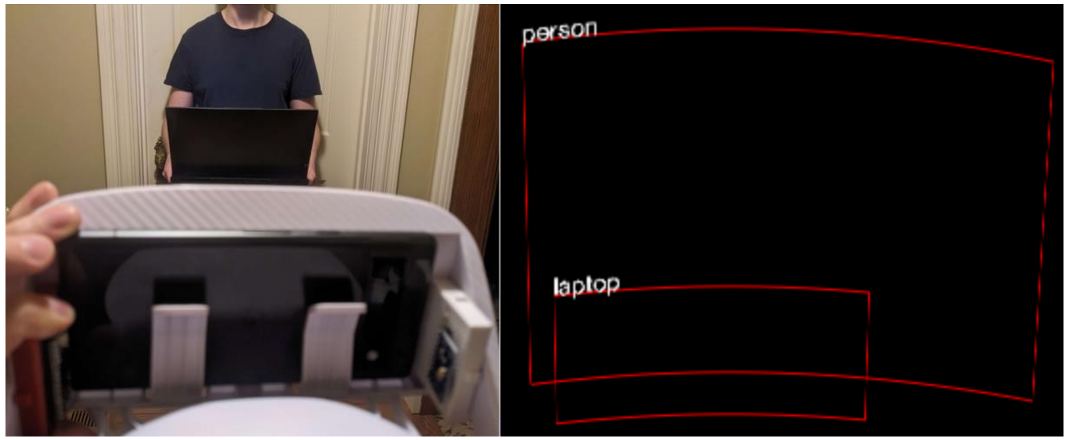


Fig. 1. Image (person and laptop), as well as bounding boxes and labels for a person and laptop (left lens render).

Limitations

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

- Reflections – the headset lens occasionally reflects the wearer, causing false detections; an external camera could help.
- Latency – wireless camera streaming adds delay; a wired connection via an exposed USB port would reduce this.
- Model tradeoff – smaller models improve speed but reduce accuracy.
- Visual clarity – aliasing affects text and bounding boxes; antialiasing could improve readability.
- Affine transformation – relies on manual calibration; a stereo camera would enable more precise 3D placement.

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.