Lab Scene Understanding

Ryan Page

May 12, 2025

Abstract

This document gives an overview of my approach / thoughts to the task of lab scene understanding. It gives an overview of my example. General observations for each component, which I have broken into: imaging, dataset generation and augmentation, model architecture and training and model execution. I tried to give details on where I would go next as I did not have time to get to the final MVP.

1 Running the Example

The process can be split into four parts.

- Run data extraction from original video. This pulls out ROIs for the bottle and perti-dish class and full images for the hand.
- Run YOLO data generation. This generates a series of images with associated labels in the file and dir format used to train the YOLO series of models. In particular, it generates composite images using the ROIs and a background image of the scene.
- Run YOLOv8n fine tune. Freeze the layers used to generate the features and fine tune the detection head for the three classes.
- Run inference. Run the model over each frame using the additional tracking module to identify tracks and then identify events in the scene, for example when a bottle enters or leaves. This moves the system towards an event based approach where subscribers are alerted of key changes in a scene.

2 Notes on Imaging

Watching the video it is clear that a wide-angle lens is being used, as a result there is barrel distortion; this can effect a systems ability to generalise across the FOV due to radial distortion. If the camera intrinsics could be extracted at either runtime or via a calibration a different projection could be used. It might be that using a linear projection helps a CNN generalise across the FOV better. The image has worse sharpness on the right-hand side (from the camera's point of view), this might make tracking across a boundary harder, for example. It could be a defective lens, which could be spotted at construction time to remove the additional complexations of needing to generalise across different spatial resolutions. Lighting is consistent over the course of the video and likely lit from external source so not likely to change over the course of a day. There is strong shadowing, if cameras can be fitted with a lighting ring that might help reduce the effect. It mabybe this is impractical and the data just has to include directional lighting. A final note, quick hand motions are blurred, it maybe with a shorter exposure and larger gain that can be improved, but there seems to be little hand object interactions that are fast enough to warrant that in this clip.

3 Data Generation and Augmentation

The idea was to split the video into 2 or 3 sections, starting with objects that were not interacting with the hands, approximately the first 2000 frames. Then develop the dataset from there. Unfortunately,

I only got as far as the first run. Using CVAT I generated three classes with bounding boxes and attributes, for example, count for number of petri dishes in a stack (although those were not used and were going to be part of the next dataset / model iteration). Those were used to extract an ROI that could then be overlayed across the FOV on the background scene (first image in the video). I randomised how many were in the scene, balancing the number of labels for each class so as not to create an uneven dataset. The augmentation step is very crude, a simple improvement would be to add multi class images. Going further, it would be interesting to look at methods to generate images with the objects in different parts of the FOV with the correct shadowing. Either a full 3D scene, or using a model based approach. Each would give long term benefits for handling different environments in future.

The next short term dataset iteration, was going to be:

- Different numbers of perti dishes with different amounts of liquid
- Bottles in different orientations and with different amounts of liquid.
- labeling objects when held.

The plan was to continue to use ROIs, including hands this time and overlaying them onto a background scene. This would control class labels ensuring balanced datasets. Then increase the number of augmentations, such as parity flipping for hands and objects, rotations and feathering to avoid boarder effects. In addition to controlling the label balance, it would enable me to keep some footage for testing.

4 Model Architecture and Training

I choose to use a YOLOv8 model [Yas24], these are good for detection tasks with a focus on real time performance, and have models with different capacities so I could experiment with accuracy and speed in the future. The example uses the nano model which has 3.2 million parameters and is the fastest, with a single forward pass using GPU acceleration around 5ms on my machine (NVIDIA GeForce RTX 3080 Ti Laptop GPU, 16117MiB), giving room to experiment with larger models. There are 22 layers, with only the last detect head being trained.

The next steps could be to look at a larger model, however I suspect the performance issues are in the dataset, that would be the place to start first before moving onto a different model. It could be interesting to look at adding another head to predict class attributes as the dataset becomes more complex.

Going further, looking at some of the latest transformers, like the Visual Geometry Grounded Transformer [WCK+25] could be interesting as a backbone. This could enable exploration into 3D pose of hands and objects allowing for more sophisticated understanding of actions, for example pouring liquid from a bottle, as the hand and object pose share a common transform during the action as they are essentially a single rigid body. This does not help for objects which deform or more subtle actions, there has been some interesting work looking at affordances of objects and hand poses [JLL+23] which could be worth exploring.

5 Inference

This example is currently run in python, the next step is to export the model to an intermediate representation (I believe Reach use OpenVINO) and run in C++. The general flow being:

- Open the capture device, either video or live feed.
- Open the model.
- Run any image preprocessing - warping or resizing for example.
- Populate the tensor and run a forward pass.
- Distribute output to subscribers.

- Generate events based on changes in scene, for example perti dish changes from empty to full

Currently event generation is basic and fires when a new item from the three classes appears or disappears. The tracking module used in the current example is bytetrack; however, the detection is not stable enough and the tracking keeps getting lost, so the event stream is very noisy. Data iterations should lead to more stable detection and better tracking. This would enable simple heuristics to be developed to generate hand-on object events. For example, when both hand and object bounding box centroid vectors are moving in similar directions, close to parallel, for example, and are within a certain centroid-to-centroid distance. This can all be done in pixel space. It would be interesting to go further and use a metric space in meters, so sizes and distances have more meaning, this is perhaps where VGGT could be useful.

6 Conclusions and Future Work

There is still much to explore and I could easily spend another week or two with this dataset. In the short term I would spend time on getting better data and having stable tracking, and longer term looking at different dataset generation and augmentation strategies, as well as looking toward 3D scene understanding to get a richer set of detectable actions.

References

[JLL+23] Juntao Jian, Xiuping Liu, Manyi Li, Ruizhen Hu, and Jian Liu. Affordpose: A large-scale dataset of hand-object interactions with affordance-driven hand pose, 2023.

[WCK+25] Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer, 2025.

[Yas24] Muhammad Yaseen. What is yolov8: An in-depth exploration of the internal features of the next-generation object detector, 2024.