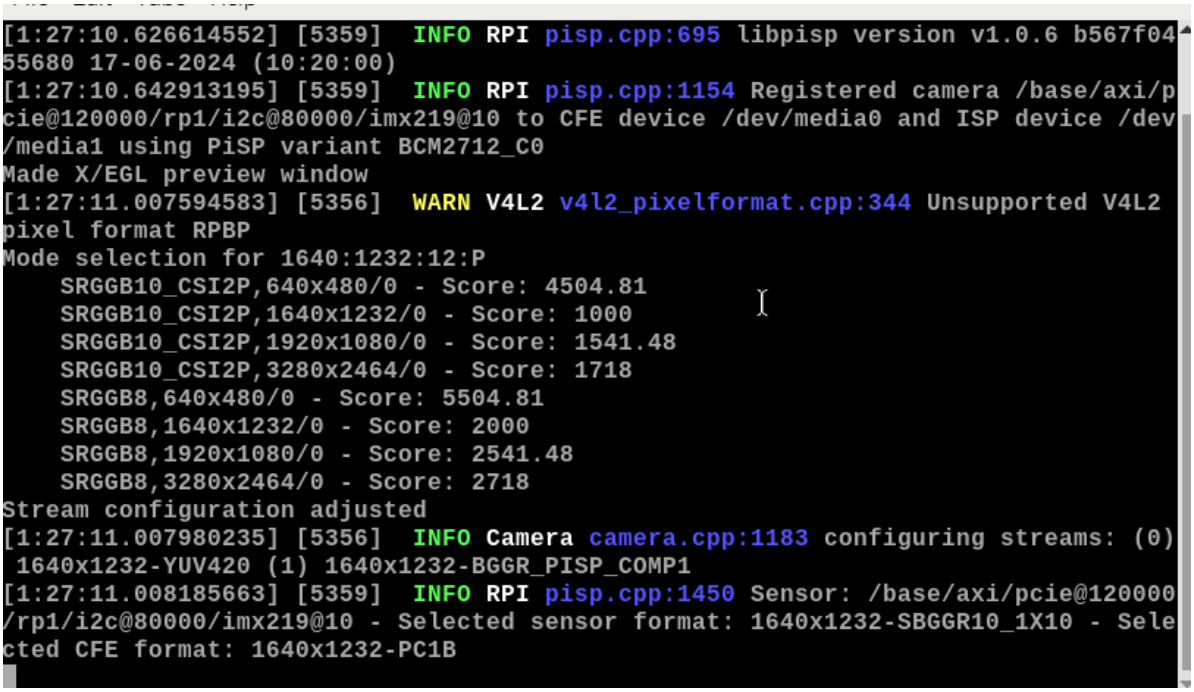
rpicam-apps

The official rpicam-apps camera driver natively supports the AI module and automatically uses the npu to run compatible post-processing tasks.

1. Test the camera

To ensure that the camera is running properly, run the following command:

```
rpicam-hello -t 10s
```



This will start the camera and display a preview window for 10 seconds. Once you have confirmed that everything is installed correctly, you can run some demos.

2. Demonstration

The camera application suite implements a post-processing framework. This section contains several demonstration post-processing stages that highlight some of the features of the AI suite. `rpicam-apps`

The following demonstration uses, by default, a preview window is displayed. However, you can use other methods instead, and you may need to add or modify some command line options to make the demonstration commands compatible with other applications. `rpicam-apps`

First, download the post-processing JSON files required by the demo. These files determine which post-processing stages to run and configure the behavior of each stage. For example, you can enable, disable, increase or decrease the strength of temporal filtering in the object detection demo. Or, you can enable or disable output mask drawing in the segmentation demo.

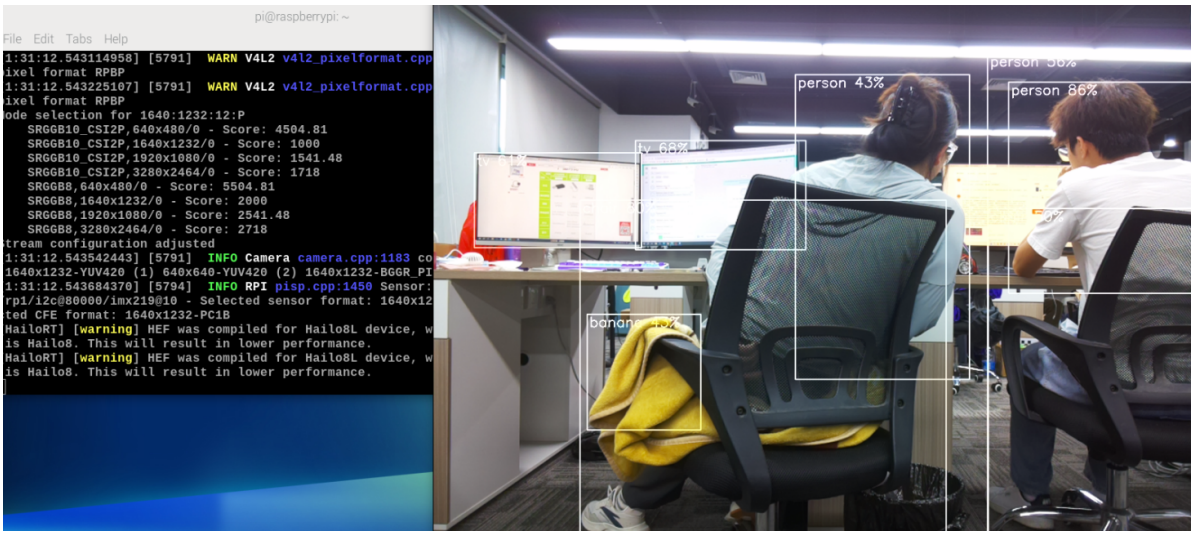
To download the entire set of post-processing JSON files, download the github repository. Run the following command: **(No download required when using the yahboom version image)**

```
git clone --depth 1 https://github.com/raspberrypi/rpicam-apps.git ~/rpicam-apps
```

3. Object Detection

This demo shows bounding boxes around objects detected by the neural network. To disable the viewfinder, use the `-n` flag. To return plain text output describing the detected objects, add that option. Run the following command to try the demo on a Raspberry Pi: `-v 2`

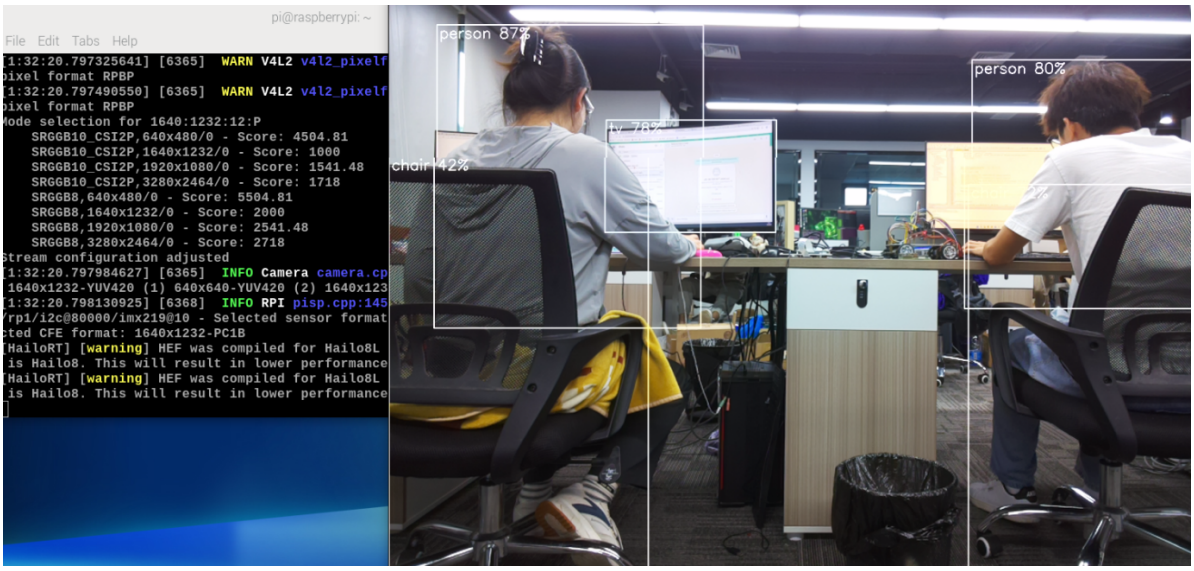
```
$ rpicam-hello -t 0 --post-process-file ~/rpicam-
apps/assets/hailo_yolov6_inference.json --lores-width 640 --lores-height 640
```



Alternatively, you can try other models that offer different tradeoffs in performance and efficiency.

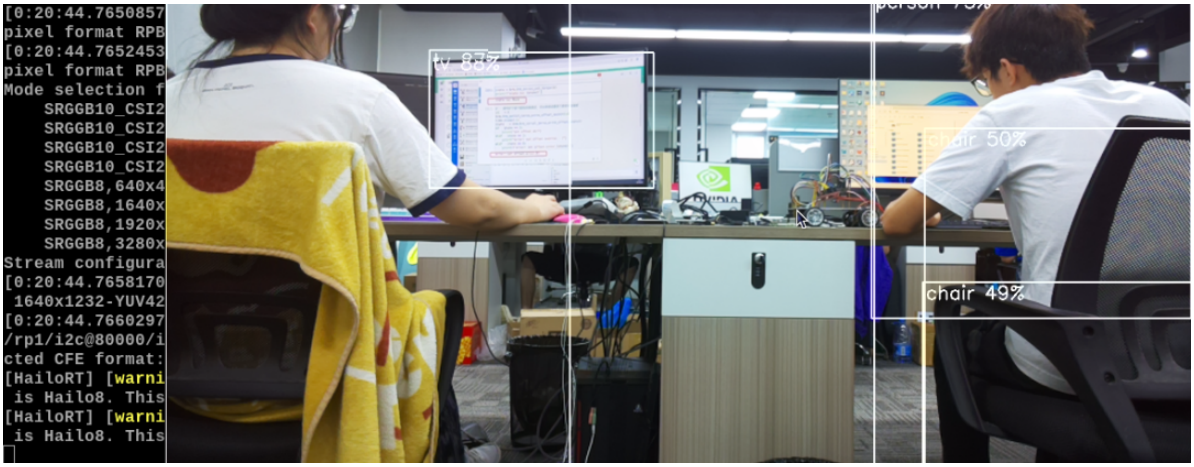
To run the demo with the Yolov8 model, run the following command:

```
$ rpicam-hello -t 0 --post-process-file ~/rpicam-
apps/assets/hailo_yolov8_inference.json --lores-width 640 --lores-height 640
```



To run the demo with the YoloX model, run the following command:

```
$ rpicam-hello -t 0 --post-process-file ~/rpicam-
apps/assets/hailo_yolox_inference.json --lores-width 640 --lores-height 640
```



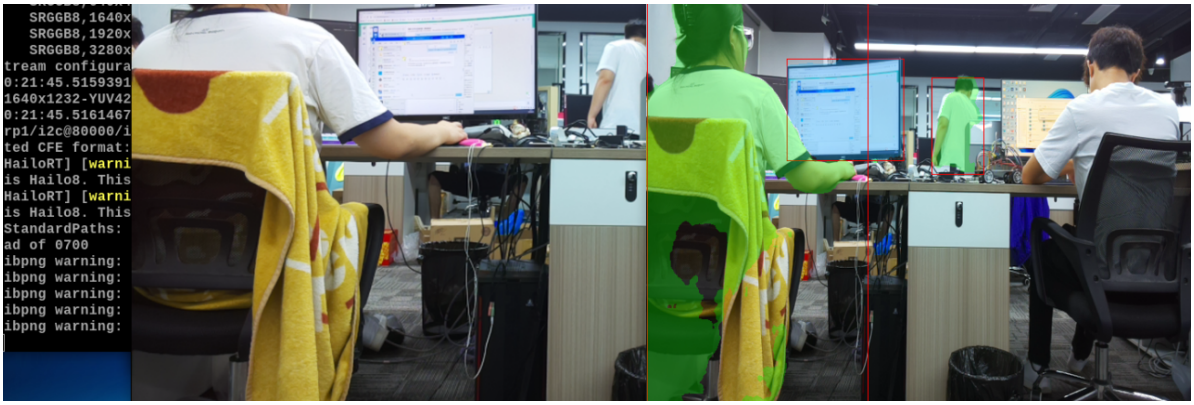
To run the demo using the Yolov5 Person and Face model, run the following command:

```
$ rpicam-hello -t 0 --post-process-file ~/rpicam-
apps/assets/hailo_yolov5_personface.json --lores-width 640 --lores-height 640
```

4. Image Segmentation

This demo performs object detection and segments objects by drawing color masks on the viewfinder image. Run the following command to try the demo on a Raspberry Pi:

```
rpicam-hello -t 0 --post-process-file ~/rpicam-
apps/assets/hailo_yolov5_segmentation.json --lores-width 640 --lores-height 640 -
-framerate 20
```



5. Pose Estimation

```
rpicam-hello -t 0 --post-process-file ~/rpicam-apps/assets/hailo_yolov8_pose.json
--lores-width 640 --lores-height 640 ``` ![image-20240812095609309](image-
20240812095609309.png)
```