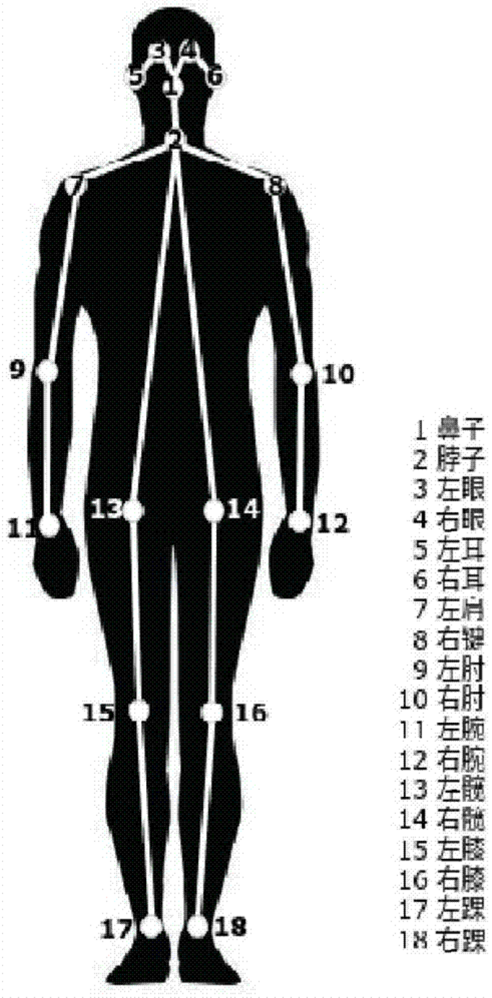
2.Human Pose Estimation

For PI5/JETSON NANO controllers, you must first enter the Docker container. For Orin boards, this is not necessary.

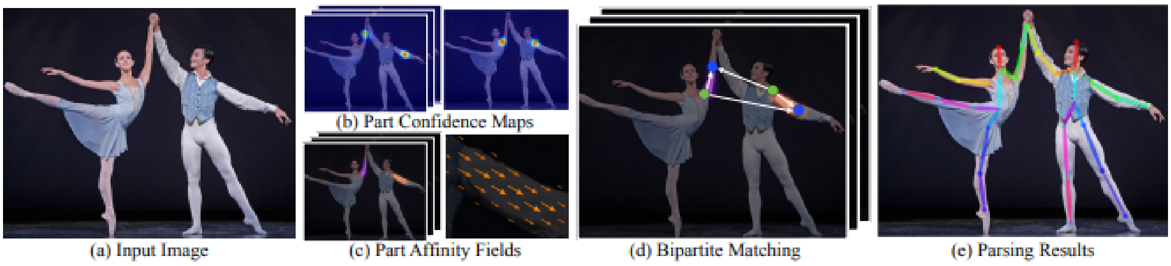
Source code path: ~/yahboomcar_ros2_ws/yahboomcar_ws/src/yahboomcar_vision

2.1 Overview

Human Posture Estimation (HPE) estimates human pose by correctly linking key points detected in an image. Key points typically correspond to joints with a certain degree of freedom, such as the neck, shoulder, elbow, wrist, waist, knee, and ankle, as shown in the figure below.



2.2 Principle



(a) Input Image (b) Part Confidence Maps (c) Part Affinity Fields (d) Bipartite Matching (e) Parsing Results

An image is input, and features are extracted through a convolutional network, resulting in a set of feature maps. This is then split into two paths, using a CNN to extract Part Confidence Maps and Part Affinity Fields, respectively.
After obtaining these two pieces of information, we use bipartite matching from graph theory to find part associations, connecting the joints of the same person. Due to the vector nature of the PAF, the resulting bipartite matching is accurate, ultimately merging into a single person's overall skeleton.
Finally, multi-person parsing is performed based on the PAFs—converting the multi-person parsing problem into a graph problem—Hungarian Algorithm (Hungarian Algorithm)
(The Hungarian Algorithm is the most common algorithm for partial graph matching. Its core is to find augmenting paths, which are used to find the maximum matching of a bipartite graph.)

2.3. Startup

Start the camera.

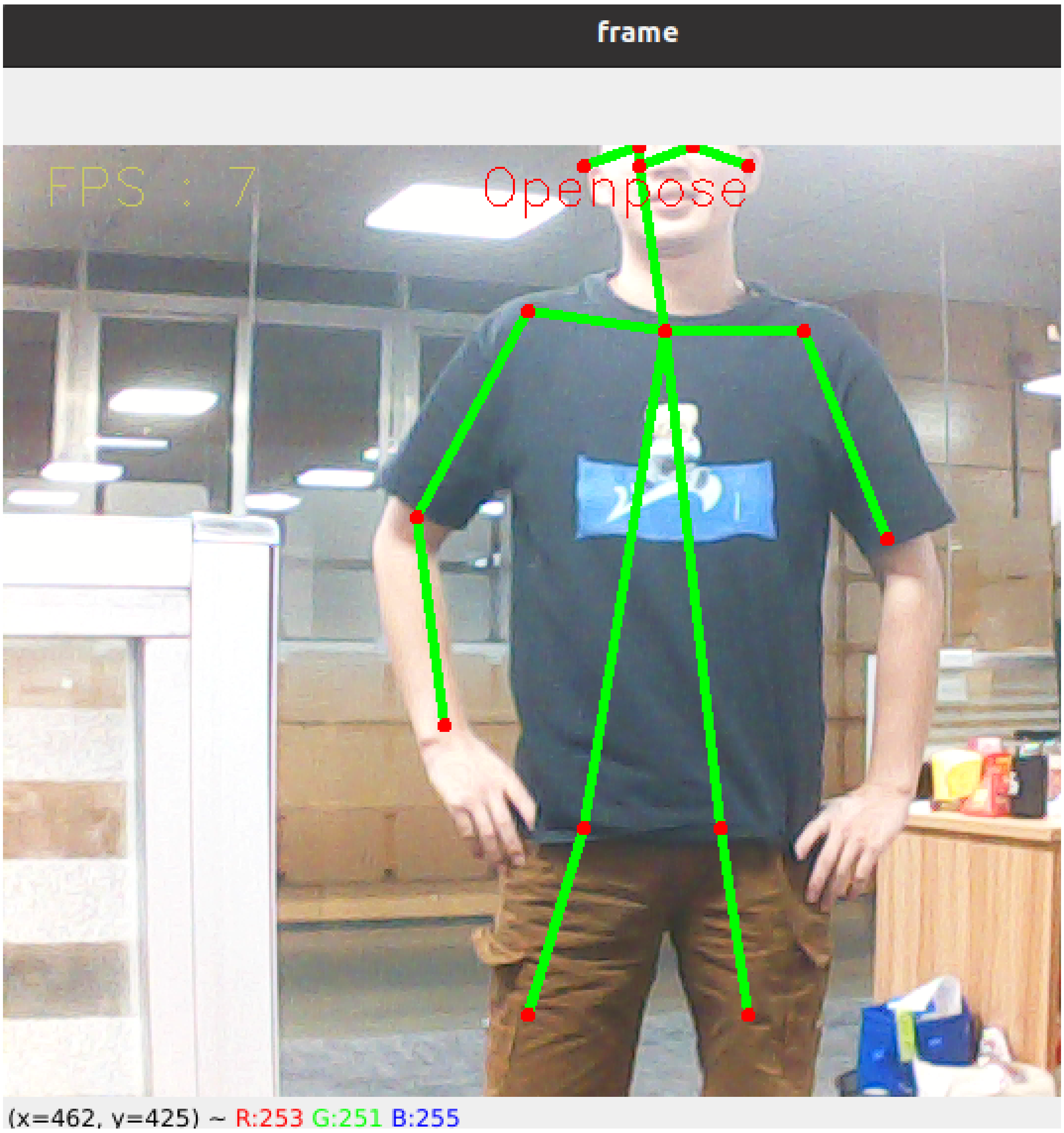
```
#usb camera
ros2 launch usb_cam camera.launch.py
#nuwa camera
ros2 launch ascamera hp60c.launch.py
```

```
ros2 run yahboomcar_vision detect_pose
```

Input Image

Output Image

You can see the actual camera image.



(x=462, y=425) ~ R:253 G:251 B:255