

Mediapipe

This tutorial takes usb no-drive camera as an example. All codes run in docker. When entering docker, you need to mount the camera. For details, please refer to docker tutorial [5、 Enter the docker Container].

1、 Introduction to Mediapipe

- MediaPipe is a data stream processing machine learning application development framework developed by Google and open source. It is a graph based data processing pipeline that enables the construction of various forms of data sources, such as video, frequency, sensor data, and any time series data. MediaPipe is cross platform and can be run on embedded platforms (such as Raspberry Pi), mobile devices (iOS and Android), workstations, and servers, while supporting mobile GPU acceleration. MediaPipe provides cross platform, customizable ML solutions for real-time and streaming media. The core framework of MediaPipe is implemented in C++ and provides support for languages such as Java and Objective C. The main concepts of MediaPipe include Packets, Streams, Calculators, Graphs, and Subgraphs.

Characteristics of MediaPipe :

- End-to-end acceleration: The built-in fast ML inference and processing can be accelerated even on ordinary hardware;
- Build once, deploy anytime, anywhere: Unified solution suitable for Android, iOS, desktop/cloud, web, and IoT;
- Instant Solution: A cutting-edge ML solution that showcases all functionalities of the framework;
- Free open source: Framework and solution under Apache 2.0, fully scalable and customizable.

2、 Use Cases

2.1、 Program Running Example

Source code path reference (in docker)

```
/root/yahboomcar_ros2_ws/yahboomcar_ws/src/yahboomcar_mediapipe/yahboomcar_media  
pipe
```

docker终端输入,

```
#Hand detection
ros2 run yahboomcar_mediapipe 01_HandDetector

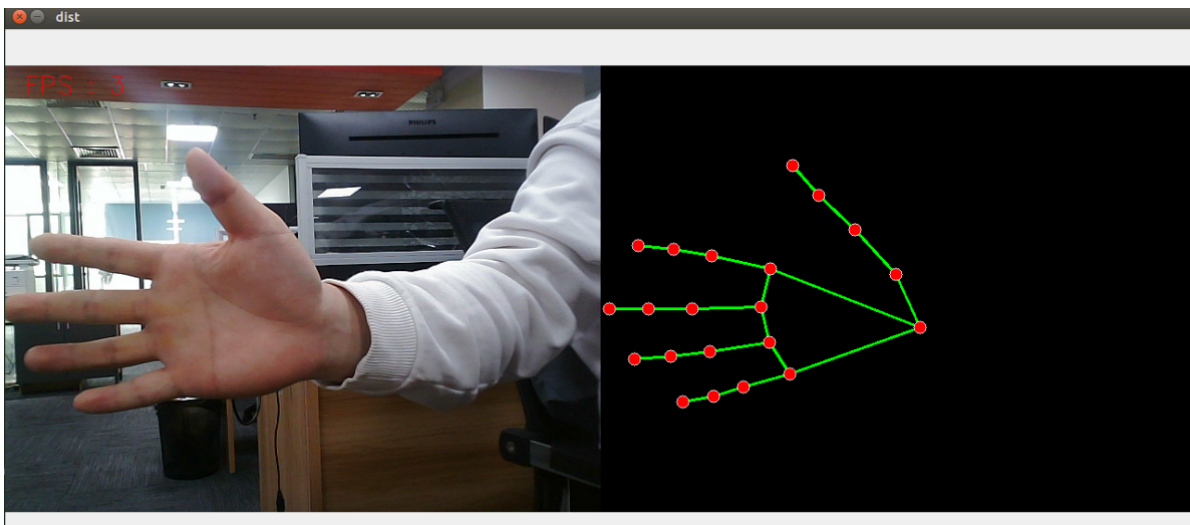
# attitude detection
ros2 run yahboomcar_mediapipe 02_PoseDetector

# Overall inspection
ros2 run yahboomcar_mediapipe 03_Holistic

# Internal inspection
ros2 run yahboomcar_mediapipe 04_FaceMesh

# Facial recognition
ros2 run yahboomcar_mediapipe 05_FaceEyeDetection
```

Taking hand detection as an example, the screenshot of the operation is as follows

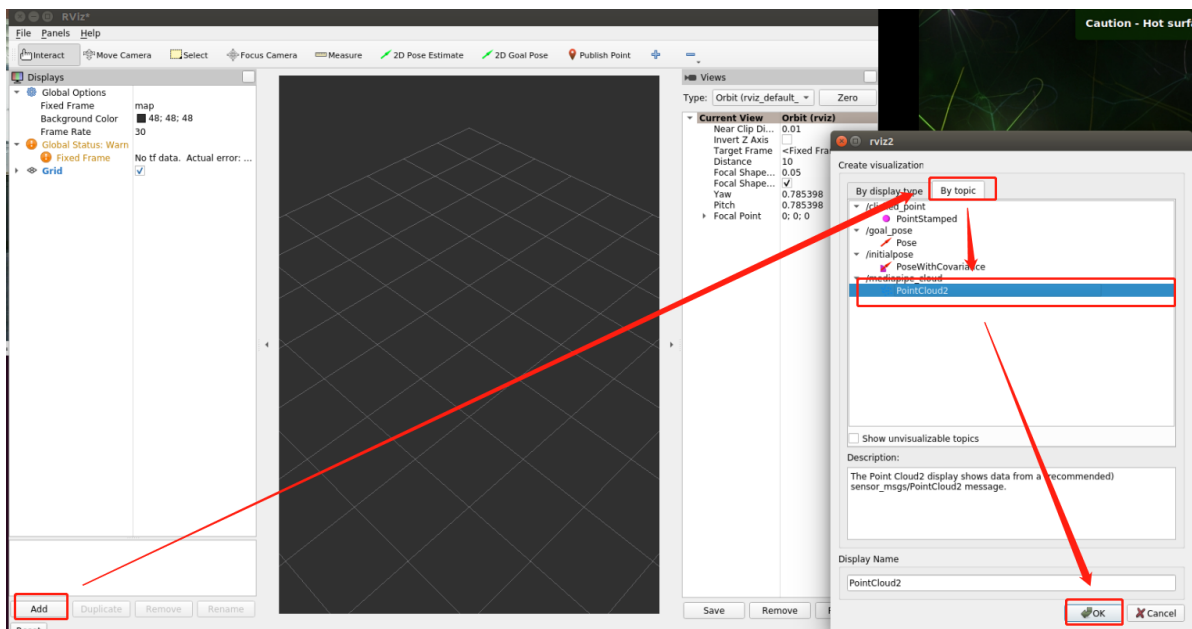


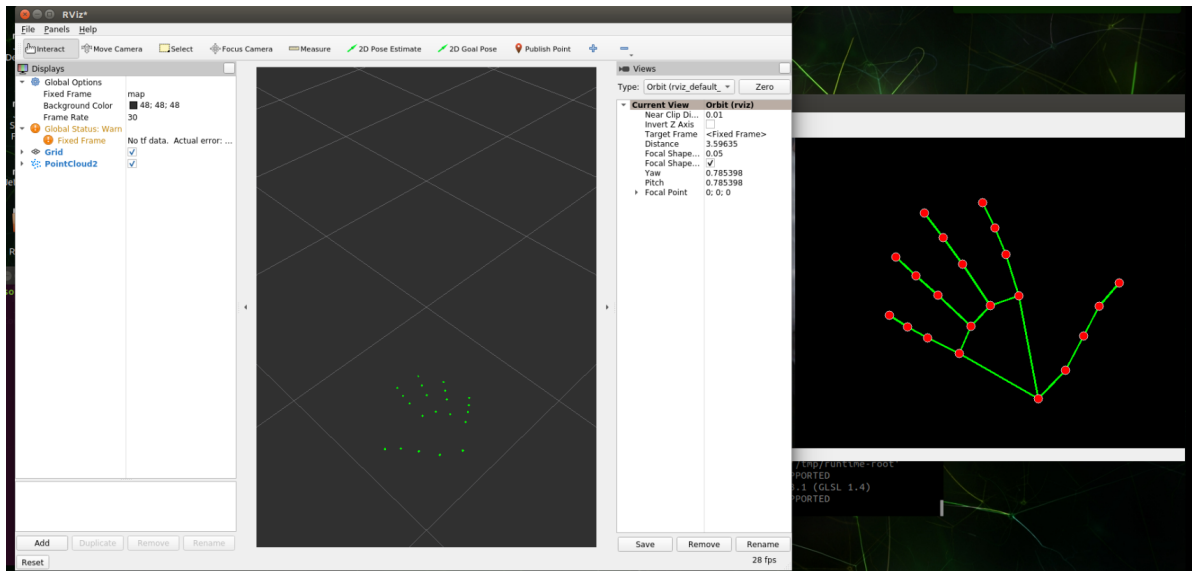
In addition, you can also view point cloud data and input it through the Docker termina

```
# Run point cloud publishing program
ros2 run yahboomcar_point pub_point

# Enable rviz to view point clouds
rviz2
```

Follow these steps to add a point cloud topic to rviz



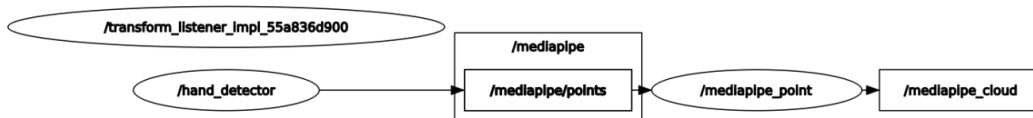


Above is running palm detection, which is 01_ Point cloud diagram of HandDetector program.

Point cloud viewing only supports demos from 01 to 04.

You can use rqt_graph View Node Topic Communication Graph

```
ros2 run rqt_graph rqt_graph
```



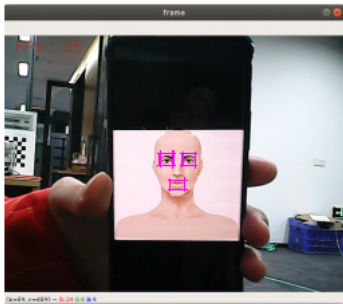
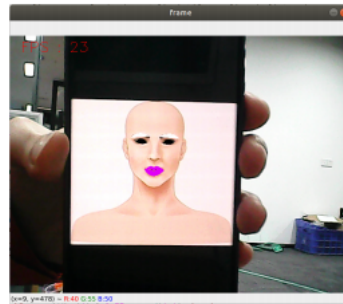
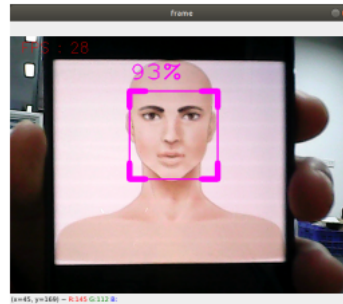
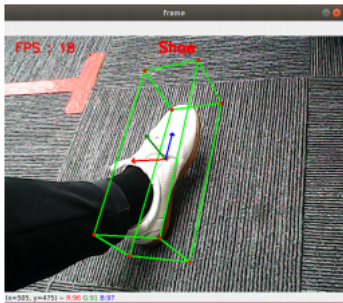
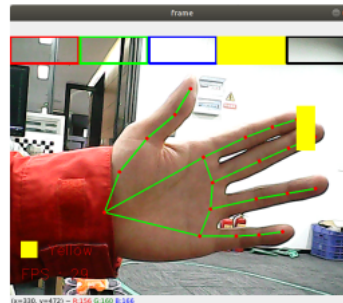
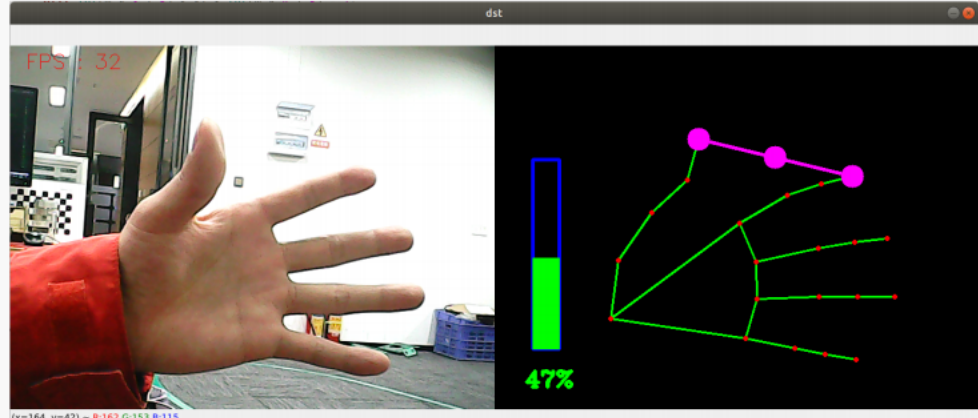
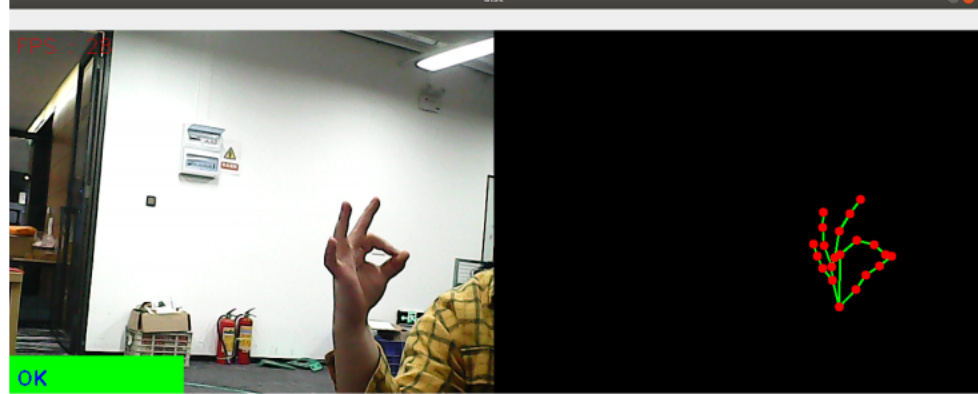
2.2、 Other fun gameplay

Docker terminal input

```

cd
/root/yahboomcar_ros2_ws/yahboomcar_ws/src/yahboomcar_mediapipe/yahboomcar_mediapipe

# Face effect
python3 06_FaceLandmarks.py
# Face detection
python3 07_FaceDetection.py
# 3D object recognition
python3 08_Objectron.py
# painting brush
python3 09_VirtualPaint.py
# Finger control
python3 10_HandCtrl.py
# Gesture recognition
python3 11_GestureRecognition.py
  
```

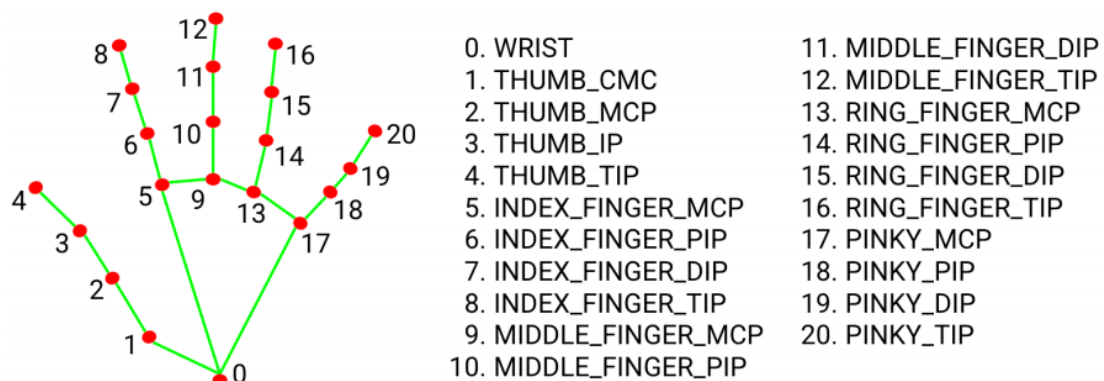
<p>05. Face recognition</p> 	<p>06. Face special effects</p> 	<p>07. Face detection</p> 
<p>08. 3D object recognition</p> 		<p>09. Brush</p> 
<p>10. Finger control</p>		
<p>11. Gesture recognition</p>		

3、Mediapipe Hands

MediaPipe Hands is a high-fidelity hand and finger tracking solution. It uses machine learning (ML) to infer the 3D coordinates of 21 hands from a single frame.

After palm detection is performed on the entire image, accurate key point positioning is performed on the 21 3D hand joint coordinates in the detected hand region by regression according to the hand marker model, that is, direct coordinate prediction. The model learns consistent internal hand pose representations and is robust to even partially visible hands and

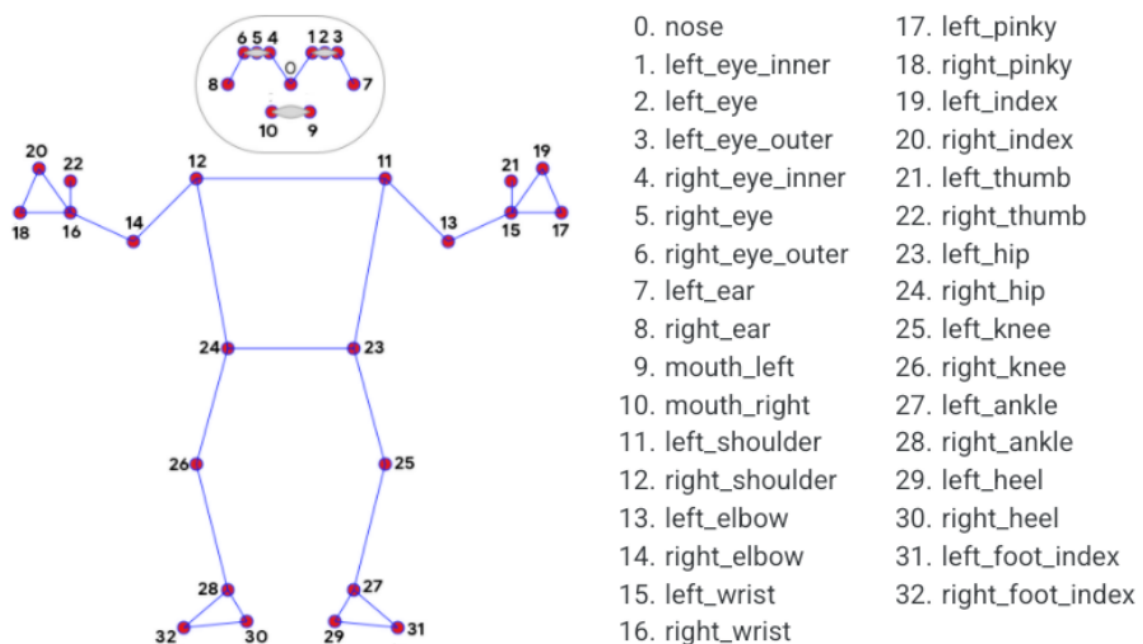
self-occlusion. To obtain ground truth data, about 30K real-world images were manually annotated with 21 3D coordinates as shown below (Z-values are obtained from the image depth map, if there is a Z value for each corresponding coordinate). To better cover the possible hand poses and provide additional supervision on the nature of the hand geometry, high-quality synthetic hand models in various contexts are also drawn and mapped to the corresponding 3D coordinates.



4、 Mediapipe Pose

MediaPipe Pose is an ML solution for high-fidelity body pose tracking that infers 33 3D coordinates and a full-body background segmentation mask from RGB video frames using the BlazePose research that also powers the ML Kit pose detection API.

The landmark model in MediaPipe pose predicts the location of 33 pose coordinates (see figure below).



5、 dlib

The corresponding case is the face effect.

DLIB is a modern C++ toolkit containing machine learning algorithms and tools for creating complex software in C++ to solve real-world problems. It is widely used by industry and academia in fields such as robotics, embedded devices, mobile phones, and large-scale high-performance computing environments .

The dlib library uses 68 points to mark important parts of the face, such as 18-22 points to mark the right eyebrow and 51-68 points to mark the mouth. Use the `get_frontal_face_detector` module of the dlib library to detect the face, and use the `shape_predictor_68_face_landmarks.dat` feature data to predict the face feature value.