

Depth camera ranging

This lesson uses the Raspberry Pi 5 as an example. For Raspberry Pi and Jetson Nano boards, you need to open a terminal on the host computer and enter the command to enter the Docker container. Once inside the Docker container, enter the commands mentioned in this lesson in the terminal. For instructions on entering the Docker container from the host computer, refer to **[01. Robot Configuration and Operation Guide]** -- **[5. Enter Docker (For JETSON Nano and RPi 5)]**.

For Orin boards, simply open a terminal and enter the commands mentioned in this lesson.

1. Program Functionality

After starting the program, click a point in the pseudo-color image with your mouse. Within the depth camera's valid data range, the screen displays the distance between that point and the depth camera.

2. Program Code Reference Path

```
~/yahboomcar_ros2_ws/yahboomcar_ws/src/yahboomcar_depth/yahboomcar_depth/Basic/get_center_dis.py
```

- get_center_dis.py

The main program uses the center coordinates of the mouse click to obtain the distance of the changed coordinates and display it on the screen.

3. Program Startup

For the Raspberry Pi PI5 controller, you must first enter the Docker container. The Orin controller does not need to enter this container.

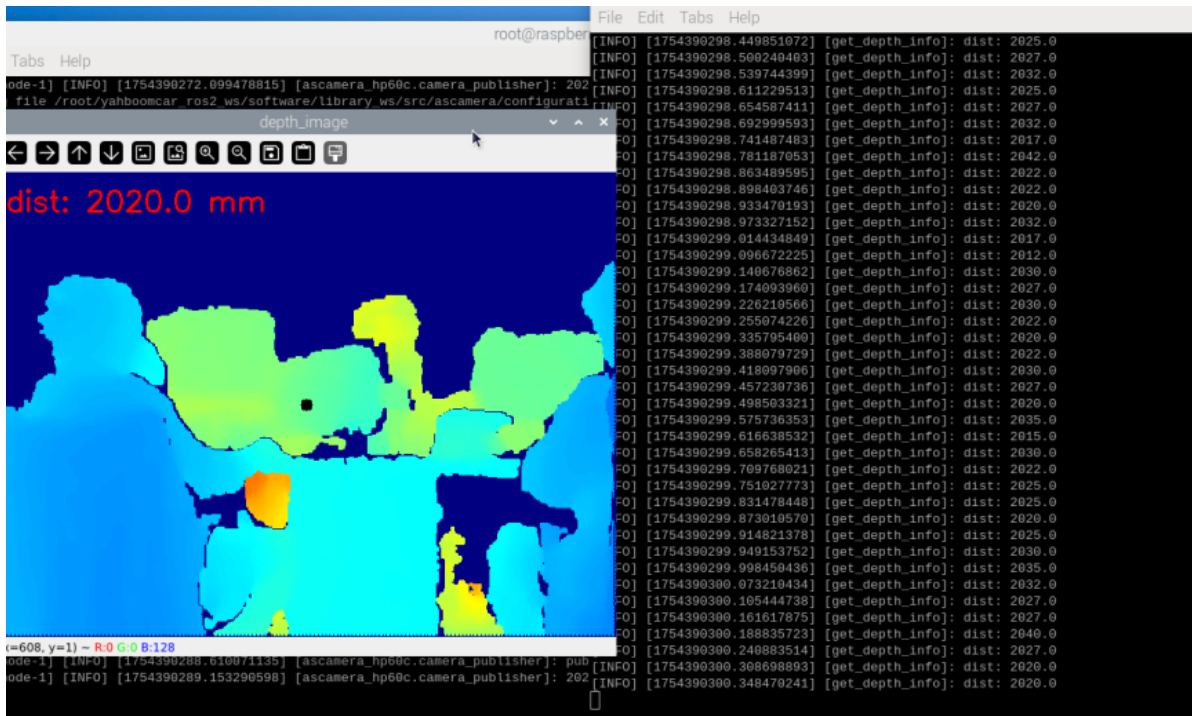
Enter the Docker container (for steps, see [Docker Course] --- [4. Docker Startup Script]).

All the following Docker commands must be executed from the same Docker container. **(For steps, see [Docker Course] --- [3. Docker Submission and Multi-Terminal Access]).**

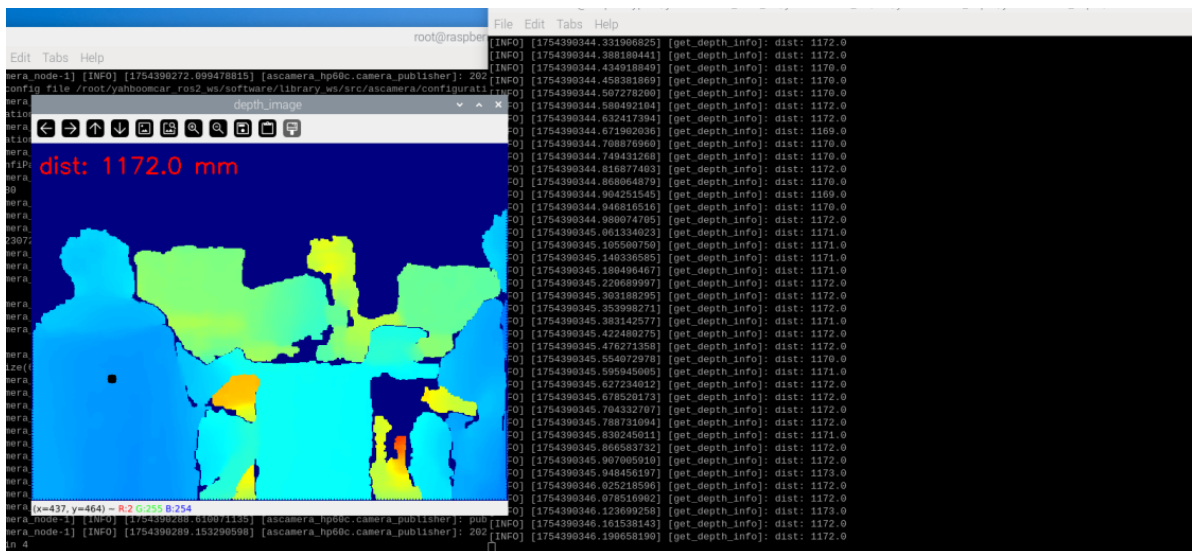
Enter the terminal:

```
ros2 launch ascamera hp60c.launch.py
ros2 run yahboomcar_depth get_center_dis
```

Upon startup, the program retrieves the distance to the image center by default. If the distance is outside the depth camera's range, an error may occur. A normal startup is shown below.



Click any point within the pseudo-color image frame with your mouse. A black dot will appear, indicating the current location. The distance to the selected point will be displayed in the image and on the terminal simultaneously.



In this demonstration, the distance to the adjacent chair is 117.2 cm, or 1172 mm.

4. Core Code

get_center_dis.py

This program has the following main functions:

- When the user clicks the left mouse button in the depth image window, it returns the distance to the current point.
- It updates the coordinates (x, y) of the current detection point.
- It returns the depth distance, and the window displays the detailed information.

Some of the core code is as follows.

```
def topic_callback(self, msg):
    # 1. ROS图像转OpenCV格式 1. Convert ROS images to OpenCV format
    depth_image = self.depth_bridge.imgmsg_to_cv2(msg, encoding[1])
```

```

# 2. 图像预处理 2. Image Preprocessing
frame = cv.resize(depth_image, (640, 480))
depth_image_info = frame.astype(np.float32)

# 3. 获取当前点的深度值 Get the depth value of the current point
dist = depth_image_info[self.y, self.x]

# 4. 创建彩色深度图像 Creating a color depth image
depth_image_colored = cv.applyColorMap(
    cv.convertScaleAbs(depth_image, alpha=0.06),
    cv.COLORMAP_JET
)

# 5. 深度值处理和记录 Depth value processing and recording
self.get_logger().info(f'dist: {dist}')
dist = round(dist, 3)
dist_text = f'dist: {dist} mm'

# 6. 在图像上添加文本 6. Add text to the image
cv.putText(depth_image_colored,
            dist_text,
            (10, 40),
            cv.FONT_HERSHEY_SIMPLEX,
            1.0,
            (0, 0, 255),
            2)

# 7. 标记当前检测点 7. Mark the current detection point
cv.circle(depth_image_colored,
           (self.x, self.y),
           1,
           (0, 0, 0),
           10)

# 8. 显示图像并设置鼠标回调 8. Display the image and set the mouse callback
cv.imshow(self.window_name, depth_image_colored)
cv.setMouseCallback(self.window_name, self.click_callback) # 关联鼠标事件 #
Associate mouse events
cv.waitKey(1)

```

