9. ROS+Opencv Application

9.1. Overview

Wiki: http://wiki.ros.org/opencv_apps

Source code: https://github.com/ros-perception/opencv_apps.git

Most of the code was originally taken from https://github.com/Itseez/opencv/tree/master/samples/cpp

Function package: opencv_apps

```
sudo apt install ros-noetic-opencv-apps
```

The topic subscribed by this function package is [/image]. What we need to do is to open the camera node and write a node to convert the camera topic into [/image] and publish the [/image] topic.

The path of the node that opens the camera and publishes the [/image] topic:

```
/home/yahboom/YBAMR-COBOT-EDU-
00001/src/yahboom_navrobo_other/yahboom_navrobo_visual/scripts/pub_image.py
```

The opencv_apps program provides various nodes that run opencv functions internally and publish the results to ROS topics. When using the opencv_apps program, you only need to run a launch file according to your own business needs, so you don't have to write program code for these functions.

The ROS Wiki has relevant node analysis, topic subscription and topic publishing of the corresponding node, and related parameter introductions. For details, please see the ROS WiKi.

Contents

9.2, Use

9.2.1, Start

Step 1: Start the camera

```
sudo supervisorctl stop ChassisServer
#Indoor version of NAVROBO-astra_pro2 camera executes this command
/home/yahboom/YBAMR-COBOT-EDU-00001/start/OBColorViewer #Release color stream
video100
#[info][722318][Pipeline.cpp:251] Start streams done!
#[info][722318][Pipeline.cpp:234] Pipeline start done!
#[warning][722318][Pipeline.cpp:327] Wait for frame timeout, you can try to
increase the wait time! current timeout=100
roslaunch yahboom_navrobo_visual opencv_apps.launch img_flip:=false
```

- img_flip parameter: whether the image needs to be flipped horizontally, the default is false.

Add the [web_video_server] node to the [usb_cam-test.launch] file as required. After it is turned on, you can use the [IP:8080] web page to view the image in real time.

Step 2: Start the function of Opencv_apps

```
roslaunch opencv_apps face_recognition.launch # Face recognition
roslaunch opencv_apps corner_harris.launch # Harris corner detection
roslaunch opencv_apps camshift.launch # Target tracking algorithm
```

```
roslaunch opencv_apps contour_moments.launch # Contour moments
roslaunch opencv_apps convex_hull.launch # Polygonal contour
roslaunch opencv_apps discrete_fourier_transform.launch # Discrete Fourier
transform algorithm
roslaunch opencv_apps edge_detection.launch # Edge detection algorithm
roslaunch opencv_apps face_detection.launch # Face detection algorithm
roslaunch opencv_apps fback_flow.launch # Optical flow detection algorithm
roslaunch opencv_apps find_contours.launch # Contour detection
roslaunch opencv_apps general_contours.launch # General contour detection
roslaunch opencv_apps goodfeature_track.launch # Feature point tracking
roslaunch opencv_apps hls_color_filter.launch # HLS color filtering
roslaunch opencv_apps hough_circles.launch # Hough circle detection
roslaunch opencv_apps hough_lines.launch # Hough line detection
roslaunch opencv_apps hsv_color_filter.launch # HSV color filtering
roslaunch opencv_apps lk_flow.launch # LK optical flow algorithm
roslaunch opencv_apps people_detect.launch # Human body detection algorithm
roslaunch opencv_apps phase_corr.launch # Phase correlation displacement
detection
roslaunch opencv_apps pyramids.launch # Image pyramid sampling algorithm
roslaunch opencv_apps rgb_color_filter.launch # RGB color filtering
roslaunch opencv_apps segment_objects.launch # Clear background detection
algorithm
roslaunch opencv_apps simple_flow.launch # Simplified optical flow algorithm
roslaunch opencv_apps smoothing.launch # Simple filter
roslaunch opencv_apps threshold.launch # Threshold image processing
roslaunch opencv_apps watershed_segmentation.launch # Watershed segmentation
algorithm
```

Almost every functional case will have a parameter [debug_view], Boolean type, whether to use Opencv to display the image, the default display.

If you don't need to display, set it to [False], for example

```
roslaunch opencv_apps contour_moments.launch debug_view:=False
```

However, after starting in this way, some cases may not be displayed in other ways, because in the source code, some [debug_view] is set to [False], which will turn off image processing.

9.2.2, Display method

- rqt_image_view

Enter the following command and select the corresponding topic

```
rqt_image_view
```

- opencv

The system displays by default, and no processing is required.

- Web viewing

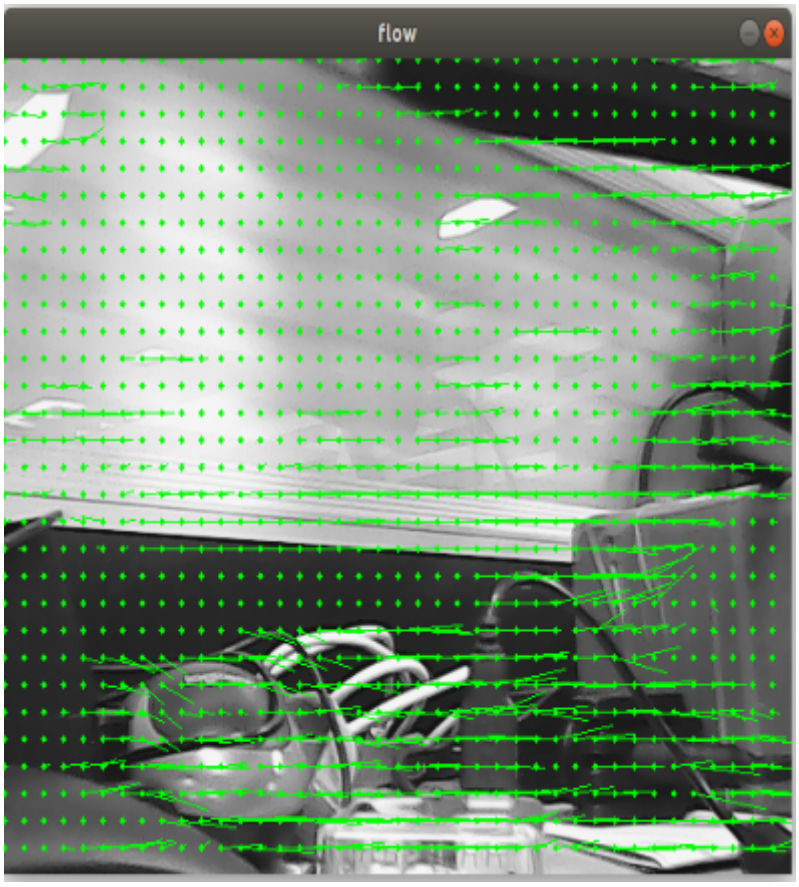
(Under the same LAN) Enter IP+port in the browser, for example:

```
192.168.2.79:8080
```

9.2.3, Effect display

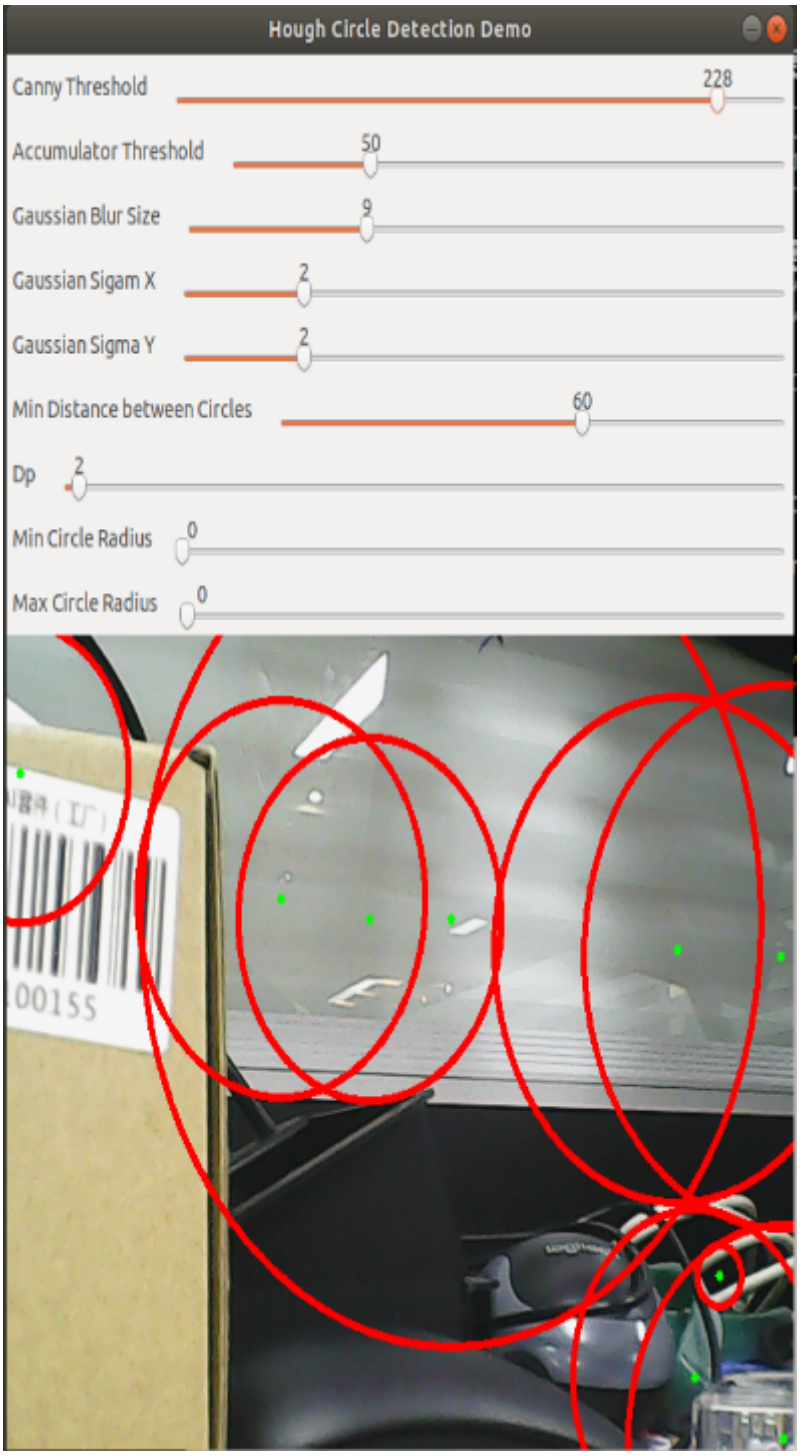
- Optical flow detection algorithm

Move the screen and observe the phenomenon.



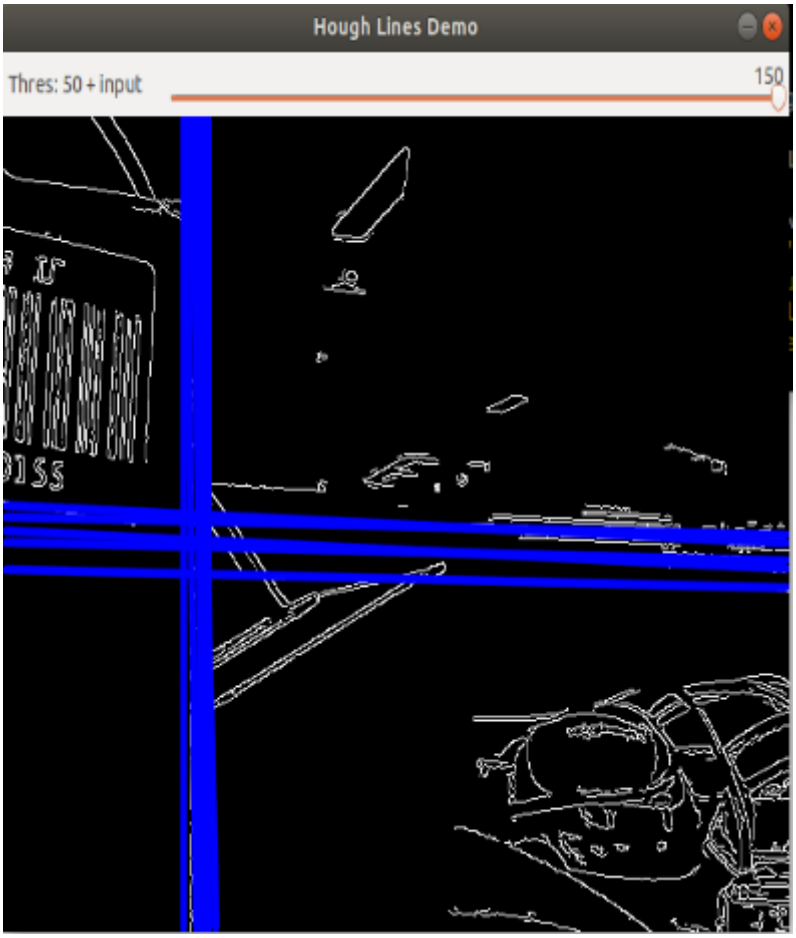
- Feature point tracking

- Hough circle detection



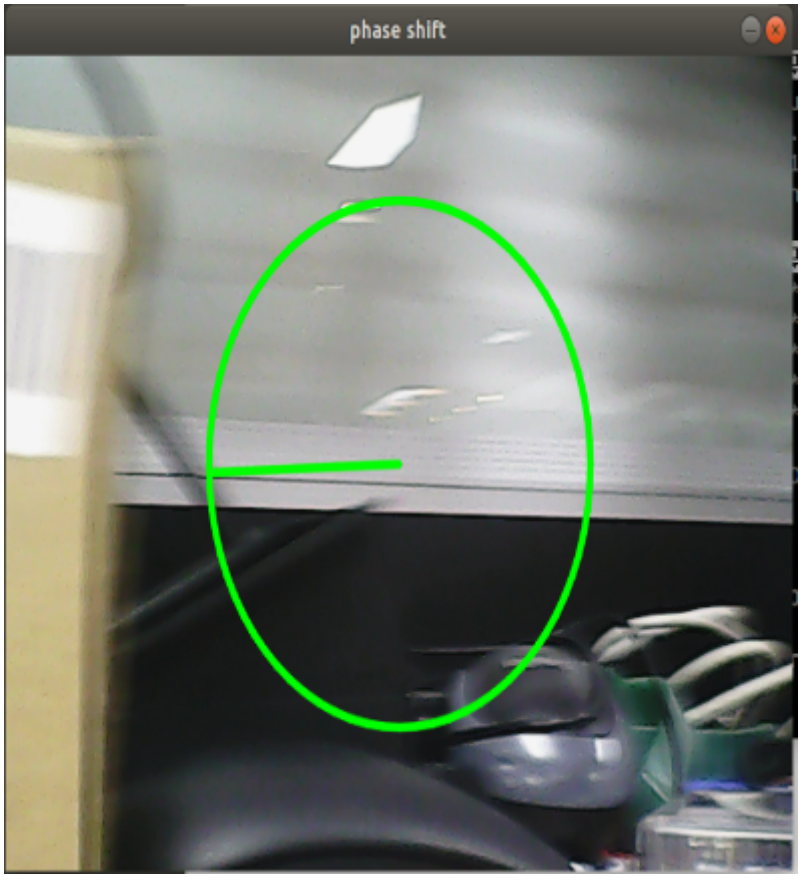
- Hough line detection

The lower the threshold, the more lines there are, and the easier it is for the picture to get stuck.



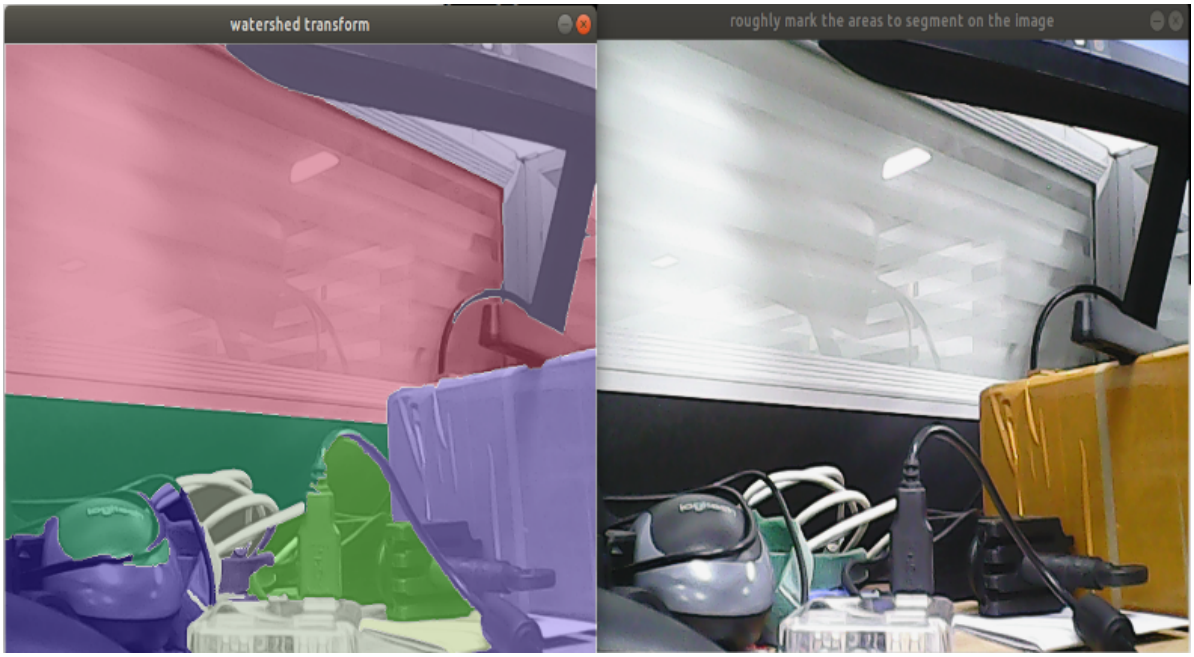
- Phase-correlated displacement detection

The faster the camera moves, the larger the radius of the circle.



- Watershed segmentation algorithm

Use the mouse to select different objects, and the system will automatically distinguish them.

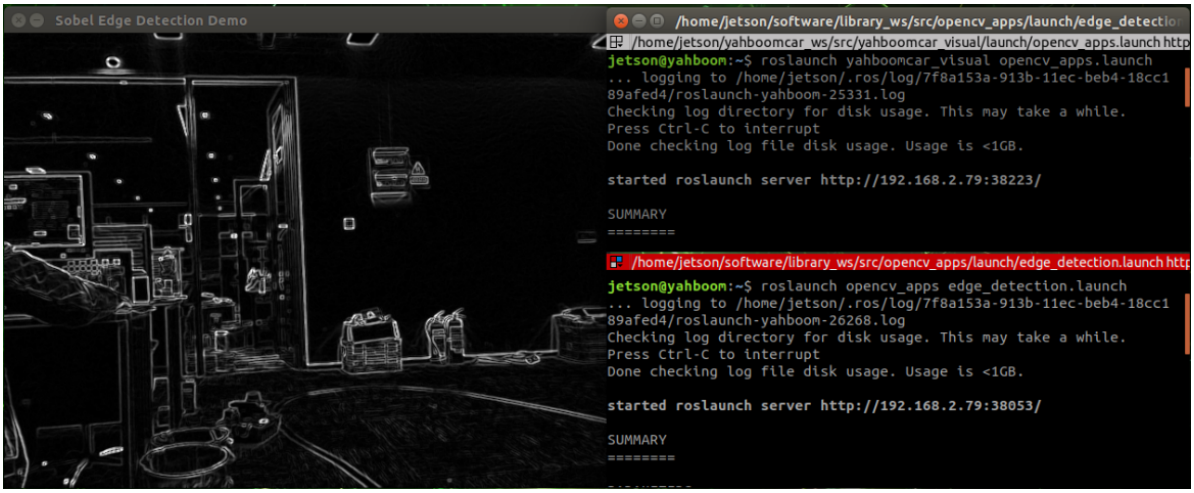


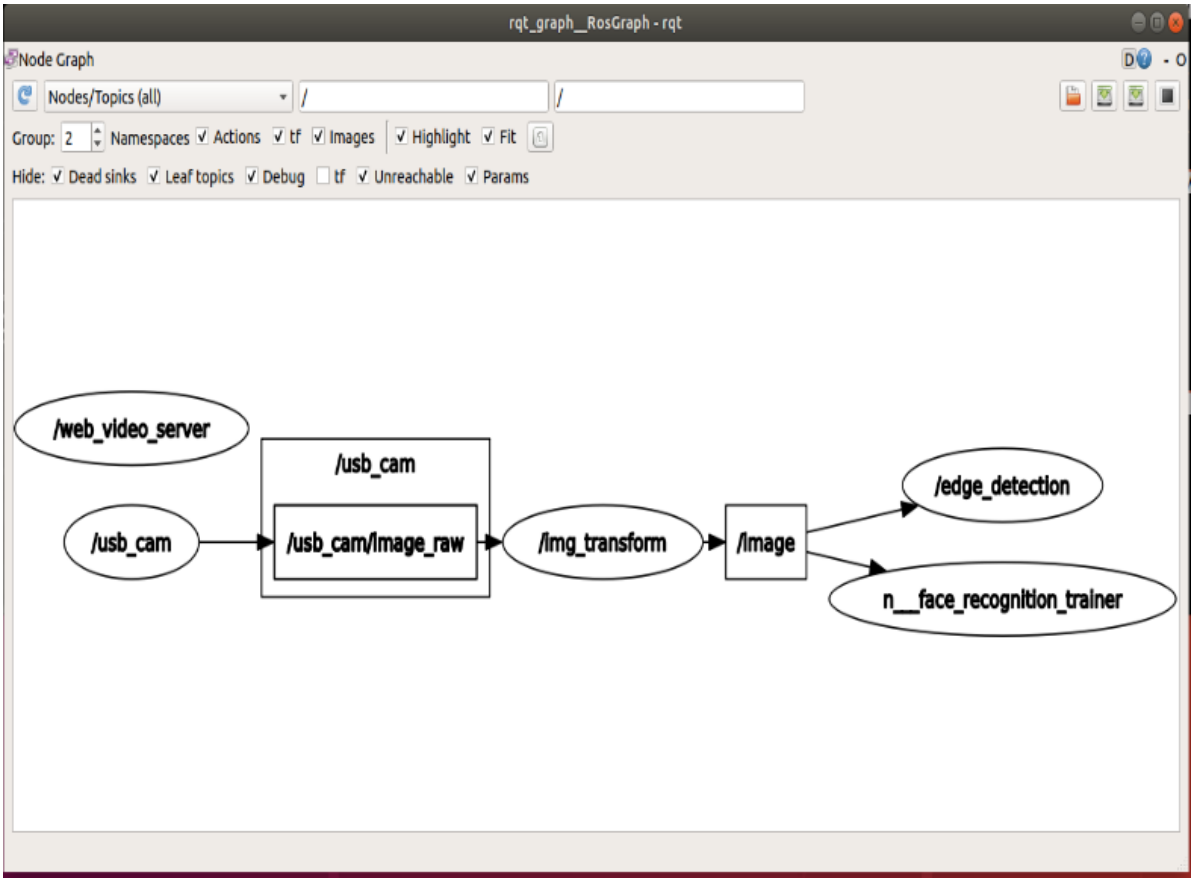
9.3, Node

Each case in this section will have a topic for subscribing to and publishing images.

9.3.1, Edge Detection Algorithm

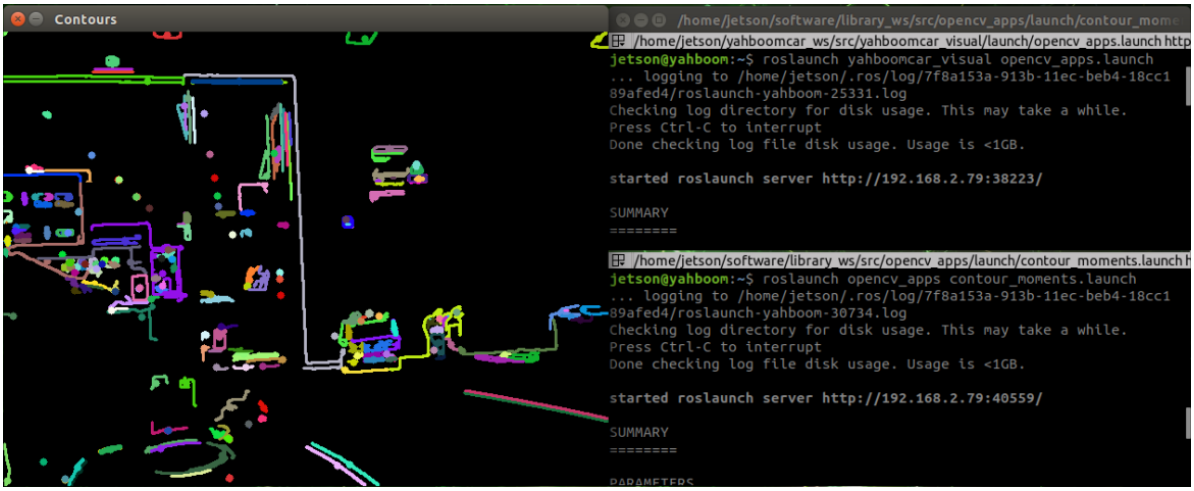
Parameter	Type	Default	Analysis
~use_camera_info	bool	true	Subscribe to the topic [camera_info] to obtain the default coordinate system ID, otherwise use the image information directly.
~debug_view	bool	false	Whether to create a window to display the node image
~edge_type	int	0	Specify edge detection method: 0: Sobel operator, 1: Laplacian operator, 2: Canny edge detection
~canny_threshold1	int	100	Specify the second canny threshold
~canny_threshold2	int	200	Specify the first canny threshold
~apertureSize	int	3	The aperture size of the Sobel operator.
~apply_blur_pre	bool	True	Whether to apply blur() to the input image
~postBlurSize	double	3.2	Input image aperture size
~apply_blur_post	bool	False	Whether to apply GaussianBlur() to the input image
~L2gradient	bool	False	canny parameters
~queue_size	int	3	queue size





9.3.2, contour moment

parameter	type	default	analysis
~use_camera_info	bool	true	Subscribe to the topic [camera_info] to obtain the default coordinate system ID, otherwise use the image information directly.
~debug_view	bool	false	Whether to create a window to display the node image
~canny_low_threshold	int	0	Canny edge detection low threshold
~queue_size	int	3	Queue size



9.3.3, Face recognition

This case is to collect people's images for self-training and real-time recognition, and the steps are slightly complicated.

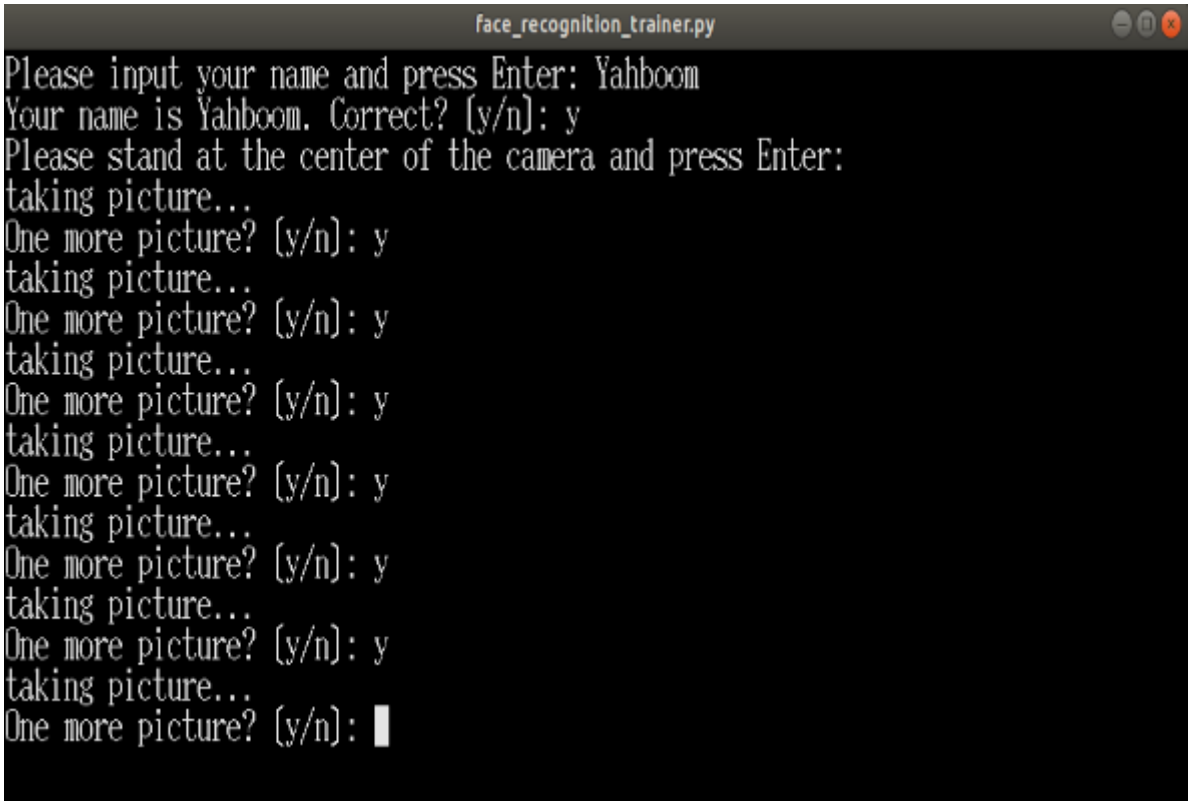
Parameter	Type	Default	Analysis
~approximate_sync	bool	false	Subscribe to the topic [camera_info] to obtain the default coordinate system ID, otherwise use the image information directly.
~queue_size	int	100	Queue size for subscribing to topics
~model_method	string	"eigen"	Face recognition method: "eigen", "fisher" or "LBPH"
~use_saved_data	bool	true	Load training data from ~data_dir
~save_train_data	bool	true	Save training data to ~data_dir for retraining
~data_dir	string	"~/opencv_apps/face_data"	Path to save training data
~face_model_width	int	190	Width of training face images
~face_model_height	int	90	Height of training face images
~face_padding	double	0.1	Padding ratio for each face
~model_num_components	int	0	The number of components of the face recognizer model (0 is considered unlimited)
~model_threshold	double	8000.0	Face recognition model threshold
~lbph_radius	int	1	Radius parameter (only for LBPH method)
~lbph_neighbors	int	8	Neighborhood parameter (only for LBPH method)

Parameter	Type	Default	Analysis
~lbph_grid_x	int	8	Grid x parameter (only for LBPH method)
~lbph_grid_y	int	8	Grid y parameter (only for LBPH method)
~queue_size	int	100	Image subscriber queue size

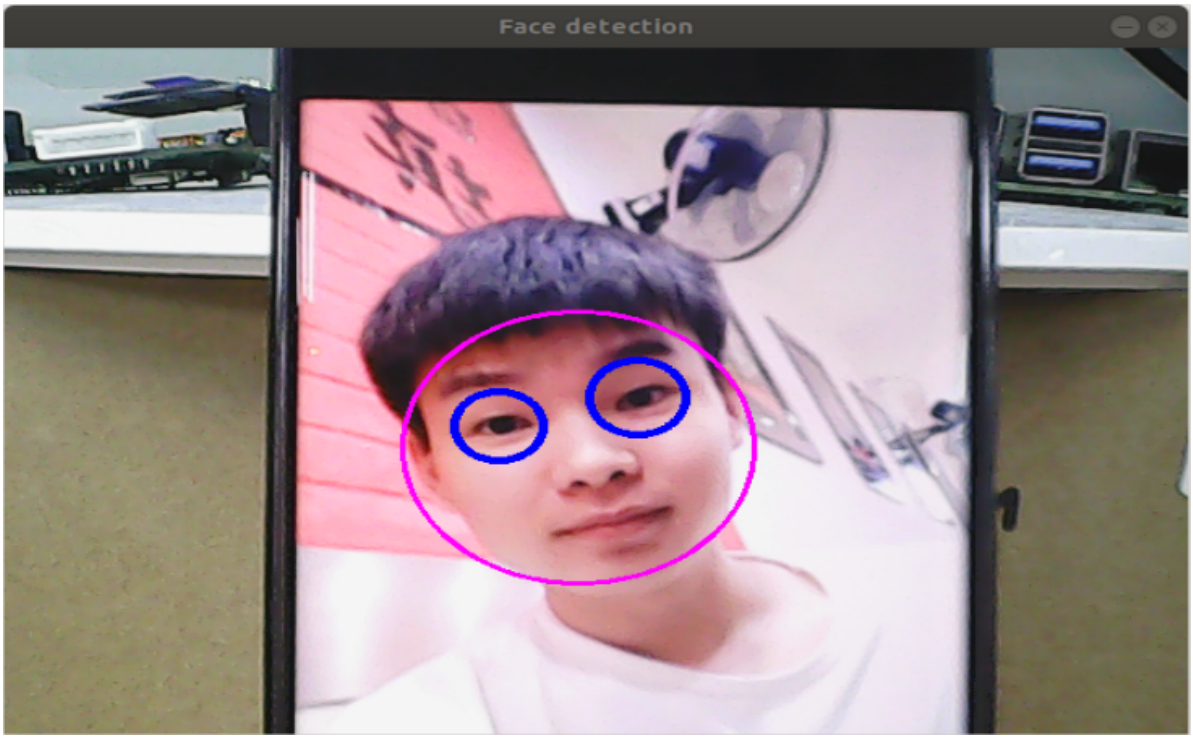
Steps:

1. First, enter the name of the person after the colon in the figure below: Yahboom
2. Confirm the name: y
3. Then put the face in the center of the image and click Confirm.
4. Loop to add a photo: y, click Confirm.
5. End the image collection, enter: n, and click OK.
6. Close the launch file and restart.

If you need to enter the recognition, loop 1 to 5 in sequence until all the recognized people are entered, and then execute step 6.



Step 3: Ensure that the face can be recognized



Final recognition effect