4. ROS+Opencv application

4.1. 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

Feature pack: ~/software/library_ws/src/opencv_apps

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

The path to the node that enables the camera and publishes the [/image] topic:

```
~/yahboomcar_ws/src/yahboomcar_visual/scripts/pub_image.py
```

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

ROS Wiki has related node analysis, topic subscription and topic publishing of corresponding nodes, introduction of related parameters, etc. See the ROS WiKi for details.

Contents

4.2. Use

4.2.1. Start

Step 1: Activate the camera

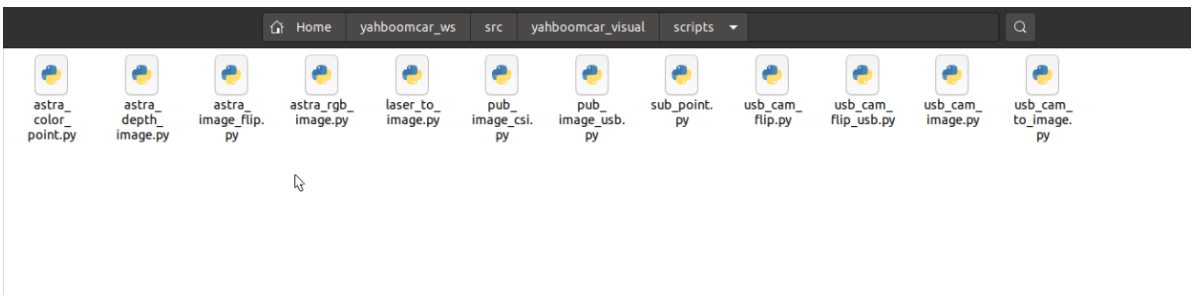
```
roslaunch yahboomcar_visual opencv_apps.launch img_flip:=false
```

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

The [usb_cam-test.launch] file opens the [web_video_server] node by default, and you can directly use the [IP:8080] webpage to view the image in real time.

- If use a monocular camera or Raspberry PI CSI camera, change 【pub_image_usb.py】 to 【pub_image.py】

  If use a Jetson camera, you'll need to change 【pub_image_csi.py】 to【 pub_image.py】



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  convex_hull.launch                  # polygon outline
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 filter
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 filter
roslaunch  opencv_apps  lk_flow.launch                      # LK optical flow
algorithm
roslaunch  opencv_apps  people_detect.launch                # human 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 filter
roslaunch  opencv_apps  segment_objects.launch              # clear background
detection 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 picture, which is displayed by default.

Set to [False] if no display is required, for example

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

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

4.2.2. Display method

- rqt_image_view

Enter the following command to select the corresponding topic

```
rqt_image_view
```

- opencv

The system defaults to display, no need to do anything.

- web viewing

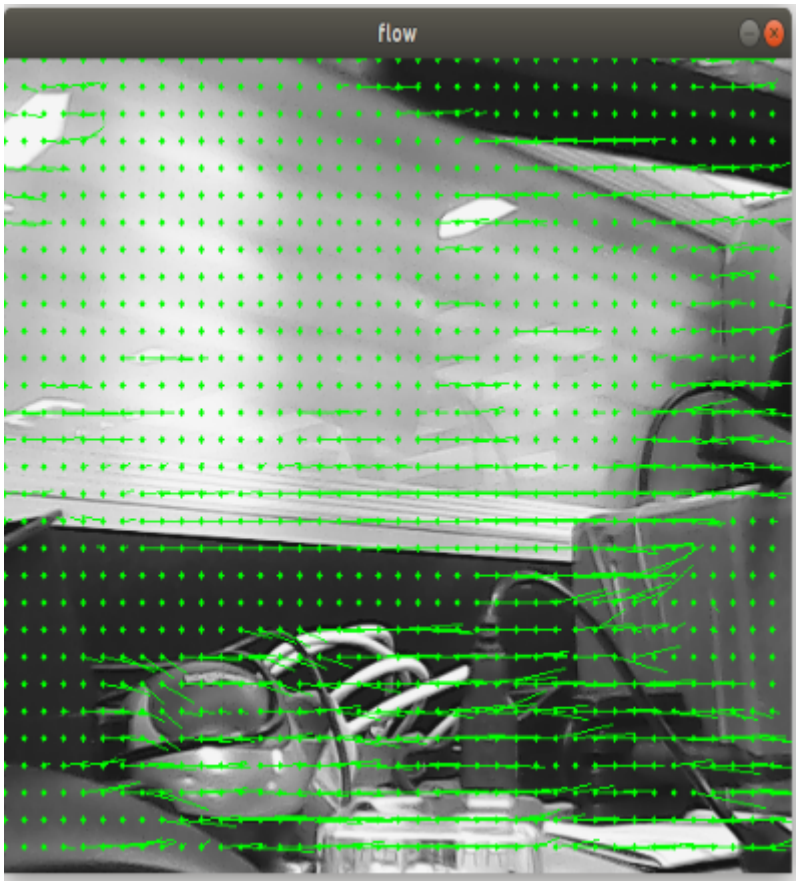
(same as the local area network) Enter IP+port in the browser, for example:

```
192.168.2.79:8080
```

4.2.3. Effect display

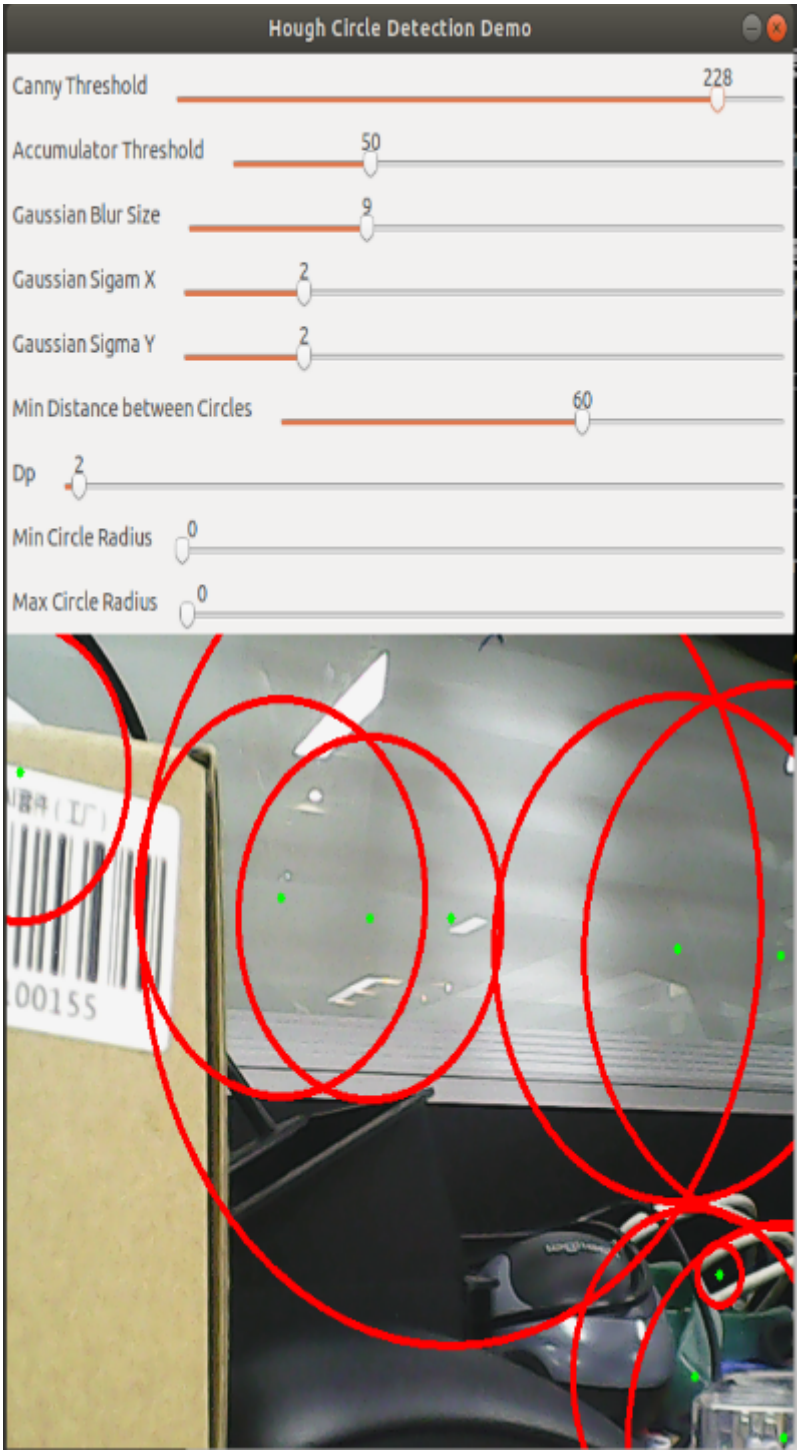
- Optical flow detection algorithm

Move the screen and observe the phenomenon.



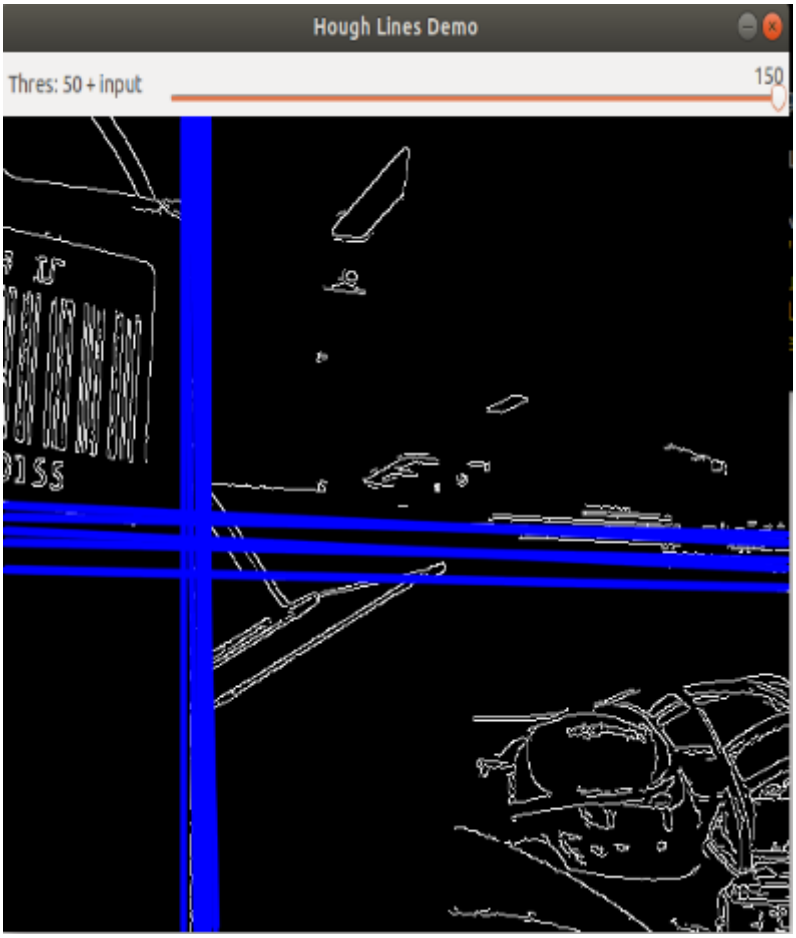
- Feature point tracking

- Hough circle detection



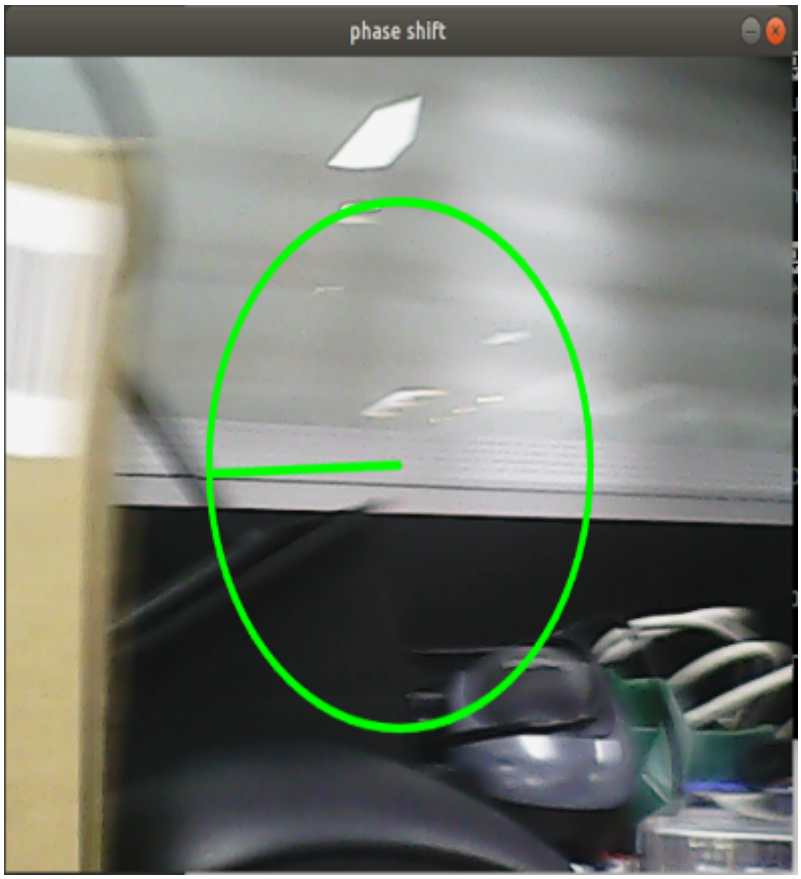
- Hough Line Detection

The lower the threshold, the more lines, and the easier the picture gets stuck.



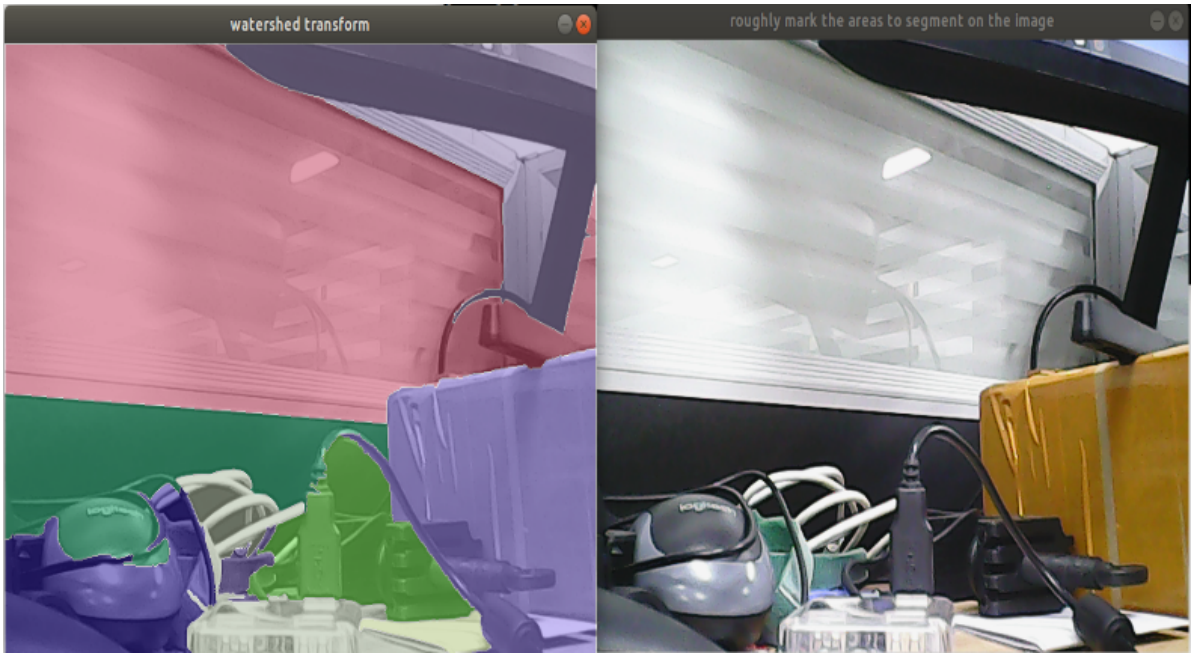
- Phase-dependent displacement detection

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.

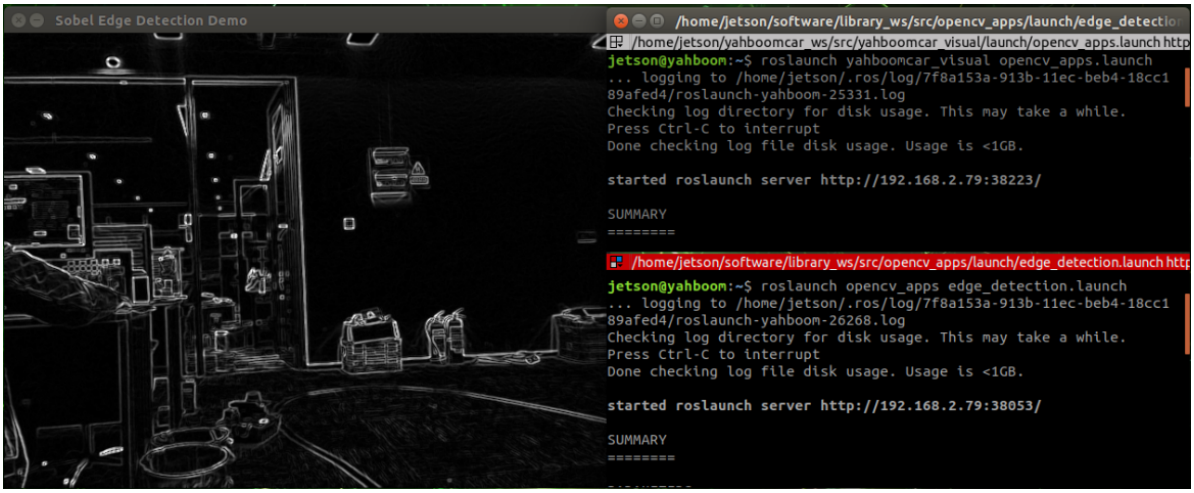


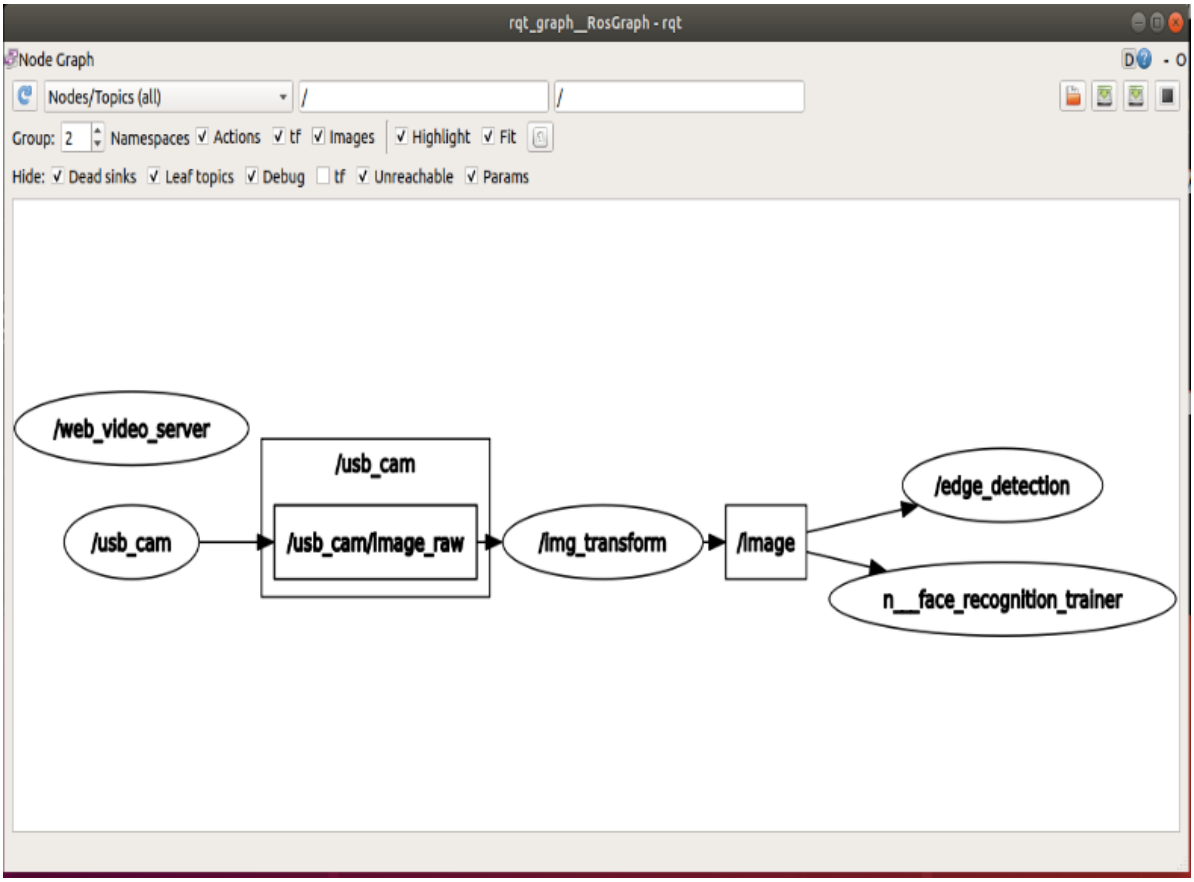
4.3. Node

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

4.3.1. edge detection algorithm

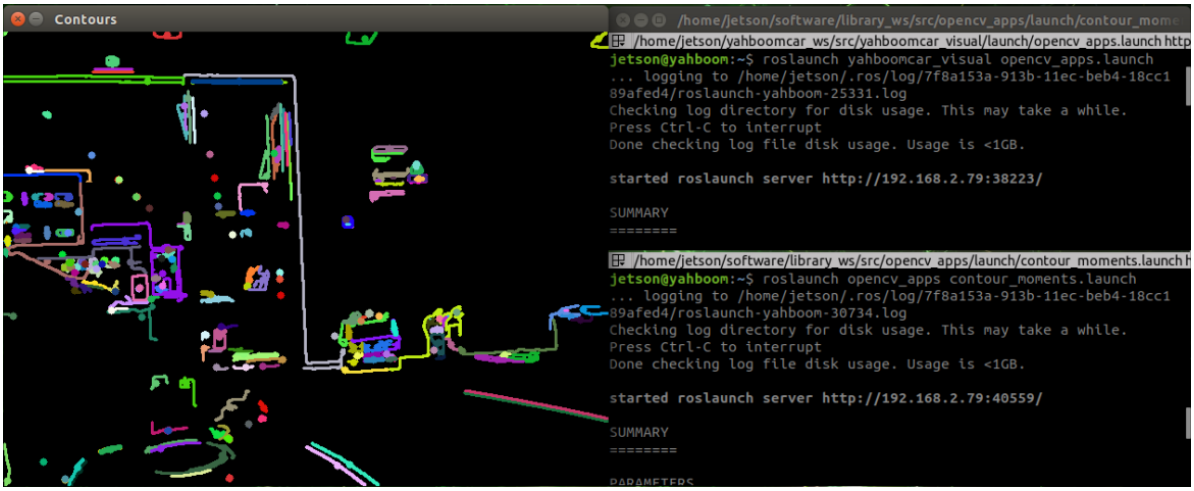
parameter	type	default	Parse
~use_camera_info	bool	true	Subscribe to the topic [camera_info] to get the default coordinate system ID, otherwise use the image information directly.
~debug_view	bool	false	whether to create a window to display the node image
~edge_type	int	0	Specify the edge detection method: 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	Enter the image aperture size
~apply_blur_post	bool	False	whether to apply GaussianBlur() to the input image
~L2gradient	bool	False	Parameters of canny
~queue_size	int	3	queue size





4.3.2. Contour moment

parameter	type	default	Parse
~use_camera_info	bool	true	Subscribe to the topic [camera_info] to get 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



4.3.3. Face Recognition

This case is self-training and real-time recognition through real-time collection of human images, and the steps are slightly complicated.

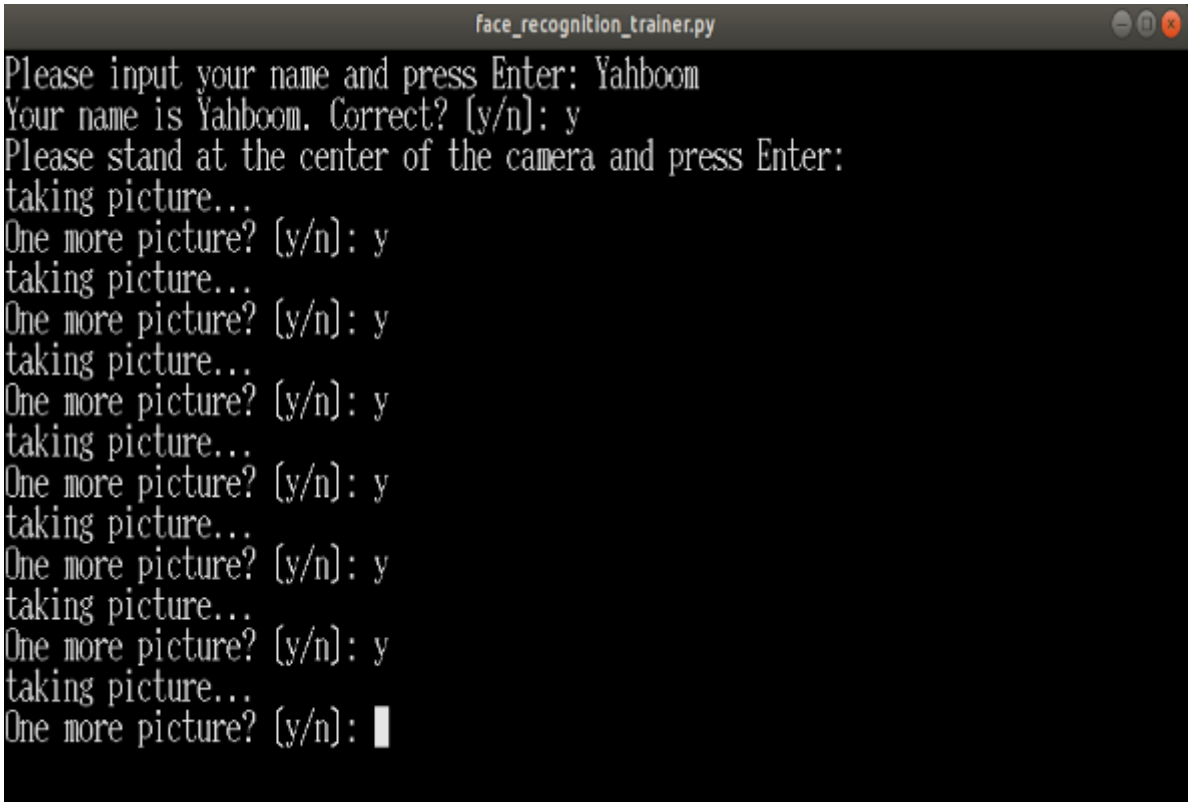
parameter	type	default	Parse
~approximate_sync	bool	false	Subscribe to the topic [camera_info] to get the default coordinate system ID, otherwise use the image information directly.
~queue_size	int	100	Queue size for subscribing to topics
~model_method	string	"eigen"	Methods of face recognition: "eigen", "fisher" or "LBPH"
~use_saved_data	bool	true	Load training data from ~data_dir path
~save_train_data	bool	true	Save the training data to the ~data_dir path for retraining
~data_dir	string	"~/opencv_apps/face_data"	Save training data path
~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	Fill ratio for each face

parameter	type	default	Parse
~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)
~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

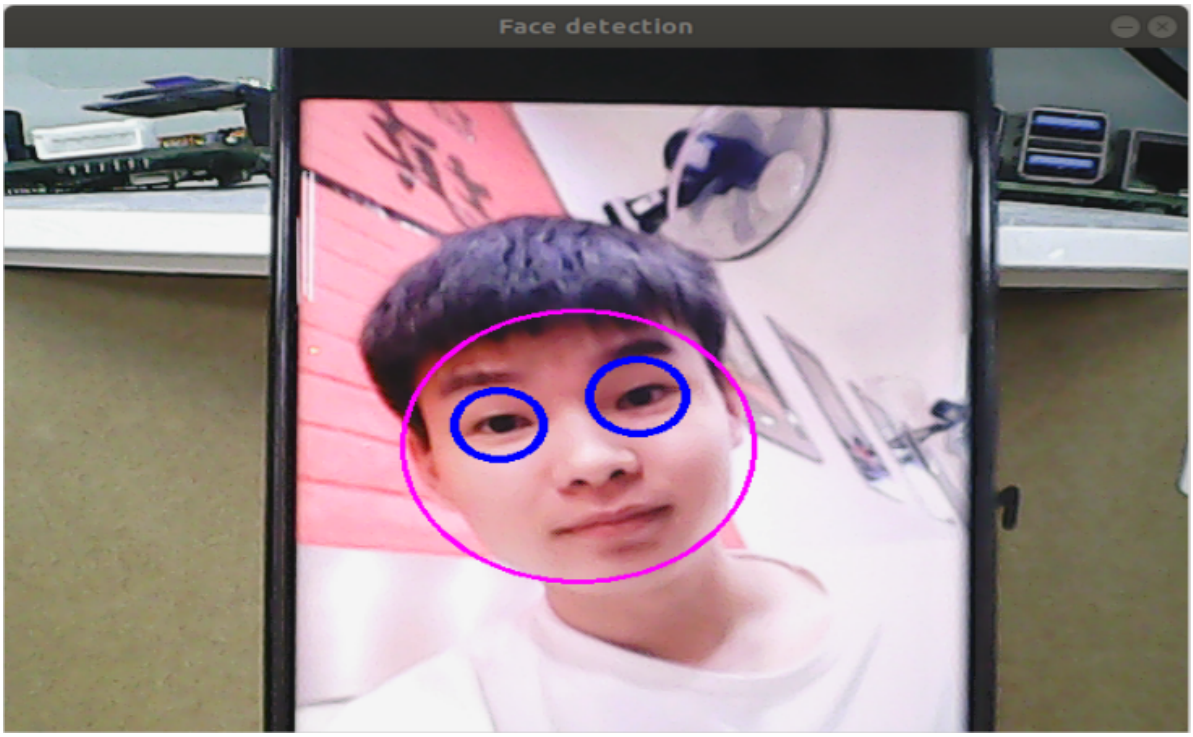
Steps:

1. First, enter the character name after the colon in the following figure: Yahboom
2. Confirm the name: y
3. Then place the face in the center of the image and click OK.
4. Add a photo cyclically: y, click OK.
5. To end the image collection, enter: n, and click OK.
6. Close the launch file and restart.

If you need to enter the recognized recognition, cycle 1 to 5 in turn, until all recognition personnel are completed, and then perform the sixth step.



Step 3: To ensure that the face can be recognized



The final recognition effect


/face_recognition/debug_image
Yahboom (1720.20)