Edge detection algorithm

1. Use

Source code launch file path：/opt/ros/noetic/share/opencv_apps/launch

Step 1: Start the camera

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

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

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

Step 2: Start the corner detection function of Opencv_apps

```
roslaunch opencv_apps edge_detection.launch                # Edge detection
algorithm
```

Each functional case will have a parameter [debug_view], Boolean type, whether to use Opencv to display images, which is displayed by default.

If no display is required, set it to [False], for example

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

However, after starting in this way, 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.

2. Display method

- rqt_image_view

Enter the following command to select the corresponding topic

```
rqt_image_view
```

- opencv

The system displays it by default, no need to do anything.

- Web viewing

(Same as under LAN) Enter IP+port in the browser, for example:

```
192.168.2.116:8080
```

For specific IP, use your current virtual machine IP.

3. Effect display

There will be a topic for subscribing images and publishing images.

Parameter	Type	Default	Analyze
~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	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's parameters
~queue_size	int	3	Queue size

Effect picture:

Node picture: