The Canny edge detection algorithm is composed of 5 steps:

1) Noise reduction: - Image convolution technique is applied with a Gaussian Kernel (3x3). The kernel size depends on the expected blurring effect. Basically, the smallest the kernel, the less visible is the blur. In our example, we will use a 5 by 5 Gaussian kernel.

2) Gradient calculation: - Smoothened image is then filtered with a Sobel kernel in both horizontal and vertical direction to get first derivative in horizontal direction (G\_x) and vertical direction (G\_y). From these two images, we can find edge gradient and direction for each pixel as follows:

Edge\_Gradient (G) = sqrt{G\_x^2 + G\_y^2}

Angle (theta) = tan^{-1}({G\_y}/{G\_x})

Gradient direction is always perpendicular to edges. It is rounded to one of four angles representing vertical, horizontal and two diagonal directions.

3) Non-maximum suppression: - After getting gradient magnitude and direction, a full scan of image is done to remove any unwanted pixels which may not constitute the edge. For this, at every pixel, pixel is checked if it is a local maximum in its neighborhood in the direction of gradient. Point A is on the edge ( in vertical direction). Gradient direction is normal to the edge. Point B and C are in gradient directions. So point A is checked with point B and C to see if it forms a local maximum. If so, it is considered for next stage, otherwise, it is suppressed ( put to zero).

In short, the result you get is a binary image with “thin edges”.

4) Double threshold: - The double threshold step aims at identifying 3 kinds of pixels: strong, weak, and non-relevant:ggggg

a) Strong pixels are pixels that have intensity so high that we are sure they contribute to the final edge.

b) Weak pixels are pixels that have an intensity value that is not enough to be considered as strong ones, but yet not small enough to be considered as non-relevant for the edge detection.

c) Other pixels are considered as non-relevant for the edge.

Now you can see what the double thresholds hold for:

1) High threshold is used to identify the strong pixels (intensity higher than the high threshold)

2) Low threshold is used to identify the non-relevant pixels (intensity lower than the low threshold)

3) All pixels having intensity between both thresholds are flagged as weak and the Hysteresis mechanism (next step) will help us identify the ones that could be considered as strong and the ones that are considered as non-relevant.

5) Edge Tracking by Hysteresis: - Based on the threshold results, the hysteresis consists of transforming weak pixels into strong ones, if and only if at least one of the pixels around the one being processed is a strong one.

OpenCV puts all the above in single function, cv2.Canny()

Conclusion: - This project outputs the edges found in an image by using the opencv features. The image is captured using the cv::Videocapture api provided by OpenCV Library. The image is then used as a sensors\_msgs/Image message on ROS,the cv\_bridge library was used to transform an OpenCV image to a ROS one,finally the ROS image is published (using image\_transport library) so it can be used by other nodes. The Image published by custom\_cv\_camera is subscribed by canny\_edge\_node and converted to OpenCV format so it can be processed. The Canny Filter is applied and then the resulting image is converted again to a ROS image so it can be published into a topic and be used by other nodes.

Output: - Below is the generated output image.

