Week3 Task

Explanation

The Canny Edge detection method uses linear filtering with a Gaussian kernel to smooth noise and then computes the edge strength and direction for each pixel in the smoothed image. Candidate edge pixels are identified as the pixels that survive a thinning process called non-maximal suppression.

In this process, the edge strength of each candidate edge pixel is set to zero if its edge strength is not larger than the edge strength of the two adjacent pixels in the gradient direction. Thresholding is then done on the thinned edge magnitude image using hysteresis. In hysteresis, two edge strength thresholds are used. All candidate edge pixels below the lower threshold are labeled as non-edges and all pixels above the low threshold that can be connected to any pixel above the high threshold through a chain of edge pixels are labeled as edge pixels.

To implement Canny Edge Detection using OpenCV

we model each of the various steps using functional programming method. The following steps are required to complete the Canny Edge Detection.

1. Noise reduction using gaussian filter
2. Gradient Calculation using Sobel filter
3. Thinning of the edges using non maximal suppression matrix
4. Double thresholding to identify strong and weak pixels along the edges
5. Edge tracking a.k.a hysteresis for actual edge tracking

From the coding point of view, the following functions were defined:

1. def read_image(image_path) to read an image

2. def gray_scale_image(image) to convert image to gray scale

3. def gaussian_filter (image, filter_size, sigma) for noise reduction using gaussian filter. Filte_size here is the dimension of the filter matrix, sigma is the standard deviation

4. def sobel_filter(any_image) for gradient filtering using Sobel filter which detect image edges

5. def non_maximum_suppression (image, angles) for Non maximum suppression matrix calculation which thins out the edges from the image out of the Sobel filter.

6. def double_threshold_hysteresis (image) for Double thresholding and hysteresis which will identify strong and weak pixels for the edges and then track the edges to maintain continuity of the connecting points along the edges.

Workflow

1. Read the image.

2. The returned read image is passed into the function to convert to gray scale

3. The image is further converted into black, white, and saved. A black and white image is the basis for the image manipulation in Canny Edge detection.
4. The gray scale imaged is passed into a gaussian filter function to reduce its noise component and a copy of the image with noise reduction is saved.

5. The image with noise reduction is further passed into the Sobel filter function to detect the edges in the image and a copy of the Sobel output image is saved.

6. The image from the Sobel filter function is once again passed into a non-maximum suppression matrix in order to thin the edges. This is done because the edges produced from the Sobel filtering are too big or heavily highlighted with light intensity. The output image from this process is saved to be used in the final function

7. Finally, to achieve a canny edge detected image, the output from the non-maximum separation is passed into double threshold hysteresis function where the image's weak and strong pixels are identified and replaced, then hysteresis is used to track the edges. The output is the final Canny edge detection.

Results

From the saved image result, we see that the original image was properly converted from the BGR to Grayscale successfully and the filter reduced the noise within the image.

The Sobel filter produced an image that has high white light intensity and blurs the image a little bit. This is particularly evident looking at the edges around the car window and the car edges in general.

The image from non-maximum suppression actually thins these edges down and the heavy white light around the car edges are greatly reduced as seen in the picture saved. This makes it possible to see a finer thin line of the car edges. The benefit of this is that a proper reading of the pixel value is possible at those edges via a camera or sensor.

The final output from the canny edge detector is a finer image with defined edges that are all linked and continuous throughout the outline of the image itself.

Please find link to the complete task folder here: Week3_Task