Challenge 1 for Hw2.

b.

In this part, I set 'rho_num_bins' to 450 and 'theta_num_bins' to 360, that is there are a total of 450 candidate ρ and 360 candidate θ.

The main idea in this part is to

i. Transform the 'edge_image' into an edge point dictionary.
ii. go through all the points of 'hough_img', for each point, go through all the dictionary to check if the point lies on the corresponding sinuous function of the dictionary, if is, value of point in 'hough_img' plus one, else continue.

Pseudo code:

$Transform\ edge\ image\ into\ dictionary$

$for\ i\ in\ theta\ bins$:

    $for\ j\ in\ rho\ bins$:

        $for\ k\ in\ dictionary$:

            $if\ (theta = i, rho = j) is\ on\ sinous\ function\ xsin(\theta) - ycos(\theta) + \rho = 0$:

                $hough(i,j) += 1$

            $else$:

                $continue$

The reason for using this polling scheme is that it is simple and pretty expressive. Although this polling scheme will not give satisfying result itself, I wrote some preprocessing part to modify the data.

The value of 'rho_num_bins' and 'theta_num_bins' are chosen by experiment.

c.

In this part, I first did a preprocessing to the 'hough_img' to remove duplicate lines. As 'rho_num_bins' and 'theta_num_bins' are relatively large in **b**, there might be more than one pair of (θ, ρ) for each line in 'edge_image'. Thus, I selected a block and let this block go over all the 'hough_img' and suppressed the value of every point exceeded threshold except the maximum value within this block.

One thing to note is that, as the range of θ is $[0, \pi)$, I also removed all the points around π in order to avoid duplication.

Pseudo code:

$for\ each\ point\ in\ hough\ image > threshold$:

    $if\ current\ point\ is\ maximum\ value\ within\ the\ block$:

        $set\ all\ points\ value\ in\ the\ block\ except\ current\ point\ to\ 0$

    $else$:

        $set\ current\ point\ value\ to\ 0$

As the preprocessing is done, there will be no duplication of lines, the next part is simply to select the point exceeding threshold and to draw the correspond line on origin image.

Pseudo code:

$for\ i\ in\ theta\ bins$:

    $for\ j\ in\ rho\ bins$:

        $if\ value\ of\ (theta = i, rho = j) > threshold$:

            $draw\ the\ corresponding\ line\ in\ origin\ image$

The value of 'threshold' is chosen by experiment.

d.

The preprocessing part used in **c** is also used here.

Then, unlike **c** where I draw the whole line, here, I will only draw part of the line where there are edge points nearby. To do this, I also chose a block, and for each point on the line drawn in **c**, I searched if there are any edge points within the block, if there is, I keep the point, if there is not, I drop the point.

One thing to note, as some lines in image2 and image3 are vertical, I need to do the line segment search in two ways. If the theta is in range $\left[0, \frac{\pi}{4}\right)\ or\ [\frac{3}{4}\pi, \pi)$ the search is done on x-axis basis, else is in y-axis basis.

Here I will only give the pseudo code in x-axis basis, as they are very similar to each other.

Pseudo code:

$for\ i\ in\ theta\ bins$:

    $for\ j\ in\ rho\ bins$:

        $if\ value\ of\ (theta = i, rho = j) > threshold$:

            $initialize\ points\ coordinate\ on\ x - axis$

            $calculate\ points\ coordinate\ on\ y - axis$

            $for\ each\ point\ in\ points\ set$:

                $if\ there\ is\ edge\ points\ within\ block$:

                    $save\ this\ point$

                $else$:

                    $continue$

            $draw\ line\ segment\ based\ on\ saved\ points$

The size of block is chosen by experiment.