1 CS410-Homework-1

- Student Name: Hanlin Cai, 蔡汉霖
- MU Student ID: 20122161
- FZU Student ID: 832002117

1.1 Q3.

3. (a) With the aid diagrams, detail the steps involved in detecting linear features in an image using the Hough transform. Provide a pseudo-code description of the algorithm, and explain the key data structures that it uses.

    How can the gradient orientation be utilised to improve the efficiency of the standard Hough transform?

   (b) The figure below shows an image of a checkboard on the left and its corresponding Hough space on the right. Demonstrate your understanding of the Hough transform by explaining how the geometric structures in the input image are manifested in the resultant Hough space.

- ## 1.2 Q3(a-1)

Step 1: Edge Detection

Before applying the Hough transform, we need to detect edges in the image using techniques like Canny edge detection or other edge detection algorithms.

Step 2: Initialize the Hough Accumulator

Create an empty 2D array called the Hough accumulator, which will be used to store votes for potential lines. The size of the accumulator depends on the desired resolution and range of angles and distances we want to consider. Typically, it's a grid covering the parameter space of lines.

Step 3: Voting

- Loop through a range of possible angles (usually from -90 degrees to +90 degrees) and calculate the corresponding distance (rho) from the origin to the line along that angle.
- Increment the corresponding cell in the Hough accumulator for the (rho, angle) pair.

```
for each edge pixel (x, y):
    for each angle in range(-90 to +90 degrees):
        rho = x * cos(angle) + y * sin(angle)
        increment accumulator cell at (rho, angle)
```

Step 4: Peak Detection

Search for local maxima or peaks in the Hough accumulator. These peaks represent the most likely lines in the image.

Step 5: Interpret Peaks

- Convert (rho, angle) back to the line's equation in Cartesian coordinates (y = mx + b).
- Draw the detected line on the original image.

```
for each peak in accumulator:
    angle = peak.angle
    rho = peak.rho
    m = -cos(angle) / sin(angle)  # Slope of the line
    b = rho / sin(angle)         # Intercept of the line
    draw line y = mx + b on the original image
```

Key Data Structures:

- **Hough Accumulator**: A 2D array used to accumulate votes for potential lines, where each cell corresponds to a specific (rho, angle) pair.
- **Edge Map**: A binary image highlighting edge pixels in the original image.
- **Peak List**: A list or data structure to store detected peaks in the accumulator.

- ## 1.3 Q3(a-2) How can the gradient orientation be utilised to improve the efficiency of the standard Hough transform?

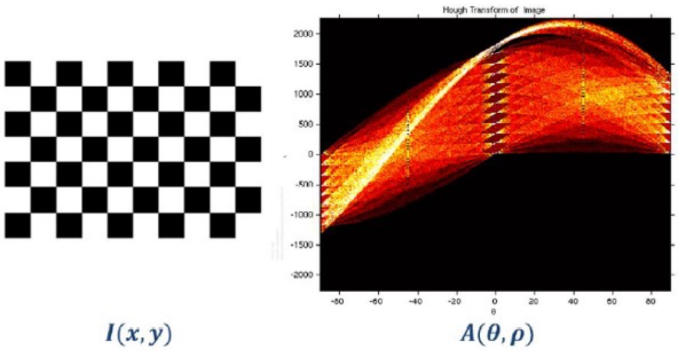
The gradient orientation can be utilized to improve the efficiency of the standard Hough transform by reducing the parameter space to be searched and focusing on potential line directions.

Here's how it can be done:

1. **Edge Detection and Gradient Computation**
2. **Thresholding or Non-Maximum Suppression**
3. **Accumulator Resolution**
4. **Voting Based on Gradient Orientation**
5. **Peak Detection and Line Interpretation**

By incorporating gradient orientation information into the Hough transform, we narrow down the search space, reducing computational complexity and focusing on the likely orientations of the lines in the image. This can significantly improve the efficiency of line detection, particularly when dealing with complex images with many non-linear features.

- Q3(b)



How would the Hough space change if the checkerboard was rotated by 45° clockwise in the plane of the image?

If we rotate the checkerboard pattern by 45 degrees clockwise, the lines that were vertical will now be at a -45 degrees orientation relative to their original position, and the lines that were horizontal will now be at a 45 degrees orientation. In Hough space, this rotation results in a shift of the peaks of the sinusoidal curves. Essentially, for each line in the image, its corresponding peak in Hough space will move along a circular path, corresponding to the rotation of the line in the image space.

So, the peaks that were at 0 degrees for vertical lines will now move to -45 degrees, and the peaks that were at 90 degrees for horizontal lines will now move to 45 degrees. The sinusoidal curves will still intersect at points that correspond to the intersections of lines in the image space, but the angles at which these curves peak will be rotated by 45 degrees.

In conclusion, the effect of rotating the checkerboard image by 45 degrees clockwise would be to rotate the orientation of the peaks in the Hough space by the same amount, while preserving the relative distances between them. This means that the pattern of curves in Hough space would appear rotated, mirroring the rotation of the lines in the image space.

- Q3(c)

(c) Imagine a computer vision system that will be presented with the above checkerboard under varying orientations and scales. Using your answer to question (b), sketch an algorithm to robustly and efficiently detect the position, scale and orientation of the board in the input image.

Note that you may assume that checkerboard will remain parallel to the image plane.

To detect the checkerboard's position, scale, and orientation in a computer vision system, we can follow these steps:

1. **Grayscale Conversion**: Convert the image to grayscale to reduce complexity.
2. **Gaussian Blur**: Apply a Gaussian blur to smooth out the image, reducing noise and improving the accuracy of subsequent edge detection.

3. **Edge Detection**: Use an edge detector, like the Canny edge detector, to find edges in the image.
4. **Hough Transform for Lines**: Apply the Hough transform to detect lines within the image.
5. **Line Sorting**: Sort detected lines based on orientation to identify two perpendicular sets corresponding to the checkerboard pattern.
6. **Corner Detection**: Find intersections of perpendicular lines to identify potential corners of the checkerboard squares.
7. **Pattern Recognition**: Match the pattern of intersections to the expected checkerboard layout to confirm the presence and position of the board.
8. **Orientation Determination**: Calculate the average angle of lines in each set to determine the board's orientation.
9. **Scale Estimation**: Measure the distances between parallel lines in each set to estimate the scale of the checkerboard.
10. **Output**: Return the checkerboard's position, scale, and orientation in the image.

1.4 Q2.

2 (a) With the aid of diagrams and pseudo-code, explain how the Hough transform can be used to detect lines in images.

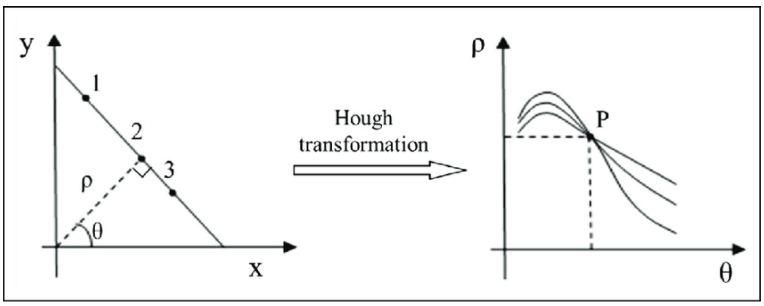
(b) With reference to part (a), comment on how the gradient orientation can be used to improve the efficiency of Hough transform.

In your answer you should detail the changes required to the pseudo-code to take advantage of the gradient orientation.

(c) Given the following image of a skewed checkerboard, select the corresponding Hough space from the 4 Hough spaces shown in the table that follows. Provide an explanation justifying your choice

- **1.5 Q2(a)**

The diagram of the Hough Transform is:



The pseudo-code of the Hough Transform is given below:

```
init accumulator H to all zeros
for each edge point (x,y) in the image
  for θ = 0 to 180:
    ρ = x cos θ + y sin θ
    H(θ, ρ) = H(θ, ρ) + 1
  end
end

find the value of (θ, ρ) where H(θ, ρ) == max
```

Hough Transform is actually based on two spaces, **Image Space** and **Hough Parameter Space**.

In the context of the **Polar-form Hough Transform**, if two edge points lay on the same line, their corresponding cosine curves will intersect each other on a specific (ρ, θ) pair. Actually, two points are already enough for detecting the line.

Accurately, if we pick multiple points on a single line and define a threshold, then their corresponding curves in Hough Parameter Space will intersect on a specific point (ρ, θ), as we expected. And if the number of intersections is beyond the threshold, then we could say the line has been detected.

Therefore, the Hough Transform algorithm detects lines by finding the (ρ, θ) pairs that have a number of intersections larger than a certain threshold.

- ### 1.6 Q2(b)

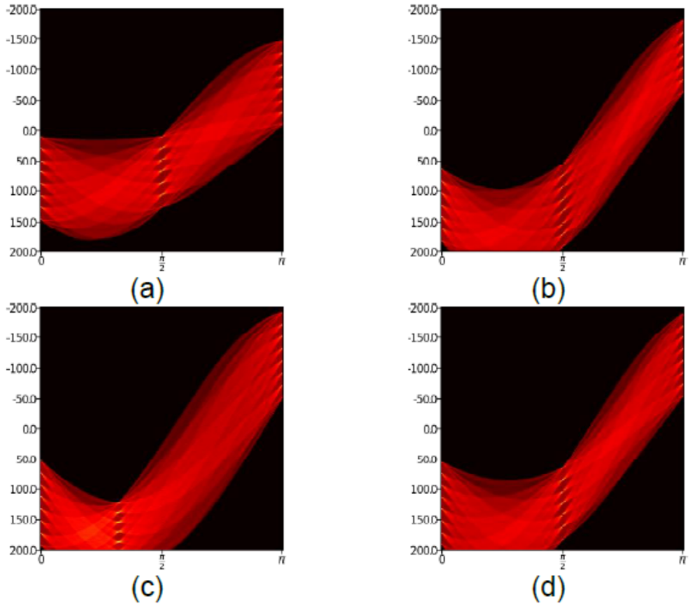
When we detect an edge point, we also know its gradient direction, which means the line is uniquely determined. Therefore, originally the algorithm needs to iterate θ from 0 to 180°. Now, it is updated to only consider θ of the gradient orientation.

The modified pseudo-code is

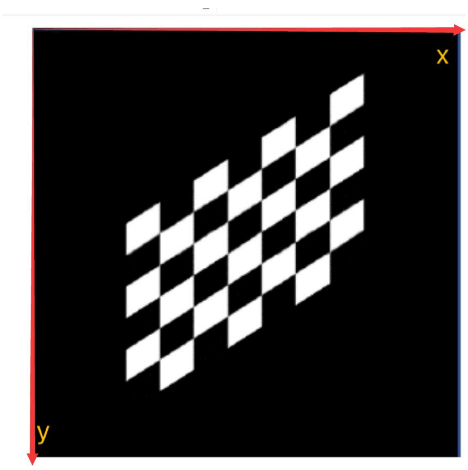
```
init accumulator H to all zeros

// Changes that compared to the original code:
For each edge point (x,y)
  θ = gradient orientation at (x,y)
  ρ = x cos θ + y sin θ
  H(θ, ρ) = H(θ, ρ) + 1
end

find the value of (θ, ρ) where H(θ, ρ) == max
```

- ## 1.7 Q2(c)

The answer is (c).



Combining the characteristics of the four options (a) (b) (c) (d), it can be inferred that the coordinate system is roughly constructed in the following manner:



For the lines perpendicular to the X-axis, there are 8 lines, so there should be 8 points at $\theta = 0$ in the Hough Plane. Then, (b) can be excluded. For the lines whose slopes are roughly -1 on this coordinate system, there are 7 lines. Therefore, there should be 7 points at $\theta = 45°$. Then, (a) and (d) are both excluded.

For option (c), seemingly there are only 6 points on the transformed Hough Plane. The reason is the last point is hidden due to the size of the image. Actually, it has 7 points. Based on the above analysis, the answer should and only be (c).