Key for the corrections

An exploration and implementation of implementing the **Hough Transform** to Identify Straight Lines

Hough transform is a robust tool for **detecting straight lines** in images, making it essential for computer vision tasks involving structure recognition. They involve the steps seen:

1) Load the input image
2) Initialize the accumulator array for Hough Transform parameters.
3) For each pixel :

    if white: skip.

    if black:

      for each θ:

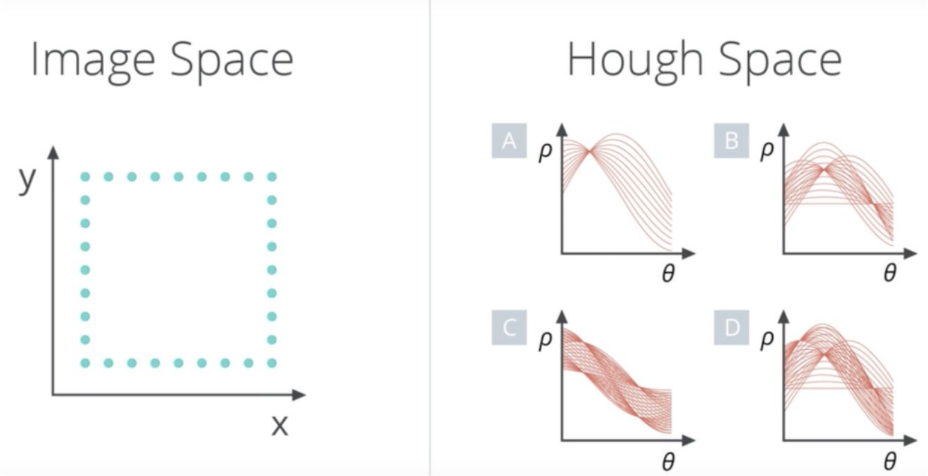
        $\rho = x \cdot \cos(\theta) + y \cdot \sin(\theta)$.

        normalise ρ to image space.

        update the Hough matrix at (θ, ρ).

4) Identifying the maxima of the Hough Transform (isolate spots, enlarge spots, locate centroid of the spots)
5) Draw straight lines in the original image

What will this image look like in Hough space? Choose the correct plot.



Click to write the which

- A)
- B)
- C)
- D)
- E) I dont' know

What is a common issue faced when using the Hough Transform for shape detection in images?

- A) Overfitting to specific shapes
- B) High computational cost compared to edge detection
- C) Inability to detect colors to comparison to black and white
- D) Loss of image quality during the transform processing
- E) I don’t know

What happens when you increase the `r_dim` and `theta_dim` in the Hough space matrix?

- A) The computational time decreases but the accuracy increases.
- B) The computational time increases and the resolution of detected lines improves.
- C) The image size increases proportionally.
- D) It decreases the sensitivity to lines in the original image.
- E) I don't know

Creating the hough transform matrix step by step

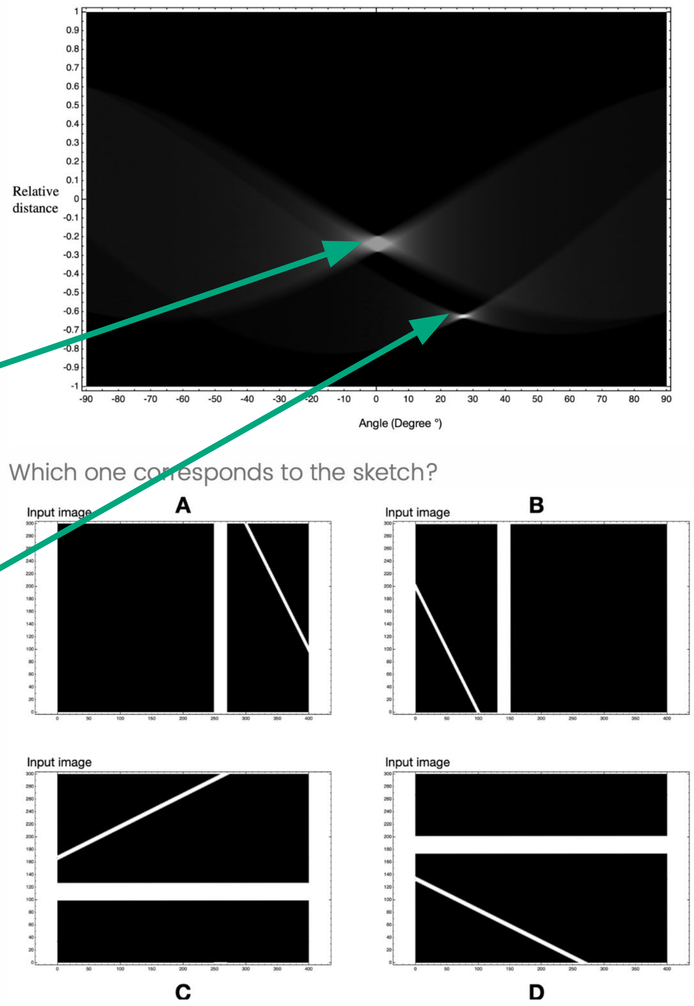
1. We start by creating a numpy array of arbitrary dimensions.

```
# We choose to have a final matrix of the following
# dimensions. This is an arbitrary decision

r_dim = 200
theta_dim = 300
```

The larger gray spot corresponds to a thicker vertical line, with a relative distance 0.2 of the half image from the center, in a vertical direction.

The bright, small spot at $\varphi = 28^\circ$ corresponds to a thin bright line perpendicular to the projection line at $\varphi = 28^\circ$, i.e. with an angle of 62°, with a relative distance of 0.6 from the center of the image.



Why was it necessary to shift and scale the computed ρ (=r) values before indexing into the Hough accumulator array (with $\theta \in [0, \pi)$?

- A) To normalize by image size for scale invariance.
- B) To increase the numerical accuracy of the accumulator.
- C) To map ρ into non-negative values for the array indices
- D) To convert ρ to [0,1]
- E) I don't know

$$r = x \cdot \cos(\theta) + y \cdot \sin(\theta)$$

1. **Scaling ρ to the Hough Space Index:**
   - The computed r value from the equation $r = x \cdot \cos(\theta) + y \cdot \sin(\theta)$ can range from $-r_{\max}$ to $r_{\max}$.
   - However, array indices must be non-negative integers.
   - To map r to a valid index (`ir`) in the Hough space array (`hough_space`), we scale and shift r accordingly.

Suppose you are using a Hough transform to do line fitting, but we notice that **our system is detecting 2 lines** where **there is actually 1** line in the image. Which of the following is most likely to alleviate this problem?

- A) Increase the size of the bins in the Hough transform .
- B) Decrease the size of the bins in the Hough transform.
- C) Sharpen the image.
- D) Make the image larger (e.g., through interpolation).
- E) I don't know