Hough Transform for Sudoku Line Detection

Image Segmentation:

Image segmentation divides an image into meaningful regions or objects (e.g., lines, shapes). In the context of a Sudoku puzzle, segmentation involves identifying the grid lines to extract the puzzle structure.

Hough Transform Theory:

The Hough Transform is a feature extraction technique used to detect parametric shapes (e.g., lines, circles) in images.

Line Detection:

Represents lines in a parameter space (e.g., Hough space using rho and theta). For a point (x, y) in the image, all possible lines passing through it are mapped to a sinusoidal curve in Hough space: rho = x * cos(theta) + y * sin(theta). Accumulator bins in Hough space count votes for line parameters. Peaks in the accumulator indicate likely lines.

Application:

1. Sudoku Solver: Detects grid lines to segment cells for digit recognition.
2. Document Analysis: Identifies table lines or text alignment.
3. Robotics and Autonomous Vehicles: Detects lane lines or structural features in environments.
4. Medical Imaging: Identifies linear structures (e.g., blood vessels).

```
import cv2
import numpy as np
import matplotlib.pyplot as plt

# Load image from Google Drive
img = cv2.imread('/content/drive/MyDrive/Colab Notebooks/imageprocessingimages/sudoku.jpg')


# Convert to grayscale
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

# Apply Canny edge detection
edges = cv2.Canny(gray, 20, 100, apertureSize=3)

# Apply erosion to refine edges
kernel = np.ones((5, 5), np.uint8)
edges = cv2.erode(edges, kernel, iterations=1)

# Apply Hough Transform to detect lines
lines = cv2.HoughLines(edges, 1, np.pi/180, 150)
print('Number of Hough lines:', len(lines) if lines is not None else 0)

# Draw detected lines on a copy of the image
img_lines = img.copy()
if lines is not None:
    for line in lines:
        rho, theta = line[0]
        a = np.cos(theta)
        b = np.sin(theta)
        x0 = a * rho
        y0 = b * rho
        x1 = int(x0 + 1000 * (-b))
        y1 = int(y0 + 1000 * (a))
        x2 = int(x0 - 1000 * (-b))
        y2 = int(y0 - 1000 * (a))
        cv2.line(img_lines, (x1, y1), (x2, y2), (0, 0, 255), 2)

# Save output image
cv2.imwrite('Hough.jpg', img_lines)

# Display images using Matplotlib
plt.figure(figsize=(15, 5))
plt.subplot(1, 3, 1)
plt.imshow(gray, cmap='gray')
plt.title('Gray')
plt.axis('off')
```

```
plt.subplot(1, 3, 2)
plt.imshow(edges, cmap='gray')
plt.title('Erode')
plt.axis('off')

plt.subplot(1, 3, 3)
plt.imshow(img_lines[:, :, ::-1])  # Convert BGR to RGB for Matplotlib
plt.title('Hough Lines')
plt.axis('off')

plt.show()
```

```
Number of Hough lines: 0
```