ЛАБОРАТОРНАЯ РАБОТА №10
ПРЕОБРАЗОВАНИЕ ХАФА ДЛЯ ОБНАРУЖЕНИЯ ЛИНИЙ И ФОРМ В PYTHON

Цель работы: Изучение и практическое применение преобразования Хафа для обнаружения линий и форм на изображениях с использованием языка программирования Python.

Необходимое программное обеспечение:

1. Python
2. Jupyter Notebook
3. Библиотеки: OpenCV, NumPy, Matplotlib

Шаги выполнения:

1. Введение:

- Ознакомьтесь с принципами преобразования Хафа и его применением в компьютерном зрении для обнаружения линий и форм.

2. Загрузка и визуализация изображения:

- Загрузите изображение, на котором будут проводиться эксперименты.
- Визуализируйте изображение.

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

# Загрузка изображения
image_path = 'path/to/your/image.jpg'
image = cv2.imread(image_path)

# Перевод изображения в оттенки серого
gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

# Визуализация изображения
plt.imshow(gray_image, cmap='gray')
plt.title('Original Image')
plt.axis('off')
plt.show()
```

3. Преобразование Хафа для обнаружения линий:

- Напишите функцию для применения преобразования Хафа для обнаружения линий на изображении.
- Примените функцию к вашему изображению и визуализируйте результаты.

```
def hough_transform_lines(image, threshold):
    # Преобразование Хафа для обнаружения линий
    lines = cv2.HoughLines(image, 1, np.pi / 180, threshold)

    # Копирование изображения для визуализации
    image_with_lines = image.copy()

    # Визуализация обнаруженных линий
    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(image_with_lines, (x1, y1), (x2, y2), (0, 0, 255), 2)

    # Визуализация изображения с линиями
    plt.imshow(image_with_lines, cmap='gray')
    plt.title('Image with Detected Lines')
    plt.axis('off')
    plt.show()

# Применение преобразования Хафа для обнаружения линий
hough_transform_lines(gray_image, threshold=100)
```

4. Преобразование Хафа для обнаружения форм:

- Расширьте функцию для обнаружения не только линий, но и форм (кругов, прямоугольников и т.д.).
- Примените функцию к изображению и визуализируйте результаты.

```
def hough_transform_shapes(image, min_radius, max_radius, threshold):
    # Преобразование Хафа для обнаружения форм
    circles = cv2.HoughCircles(image, cv2.HOUGH_GRADIENT, dp=1, minDist=20,
                               param1=50, param2=30, minRadius=min_radius,
maxRadius=max_radius)

    # Копирование изображения для визуализации
    image_with_shapes = image.copy()

    # Визуализация обнаруженных форм
    if circles is not None:
        circles = np.uint16(np.around(circles))
        for i in circles[0, :]:
```

```
            # Рисование кругов
            cv2.circle(image_with_shapes, (i[0], i[1]), i[2], (0, 255, 0), 2)

    # Визуализация изображения с формами
    plt.imshow(image_with_shapes, cmap='gray')
    plt.title('Image with Detected Shapes')
    plt.axis('off')
    plt.show()

# Применение преобразования Хафа для обнаружения форм
hough_transform_shapes(gray_image, min_radius=10, max_radius=50, threshold=50)
```

5. Эксперименты с параметрами:

- Проведите эксперименты с различными параметрами преобразования Хафа, такими как порог обнаружения, минимальная длина линии, минимальный и максимальный радиусы форм.
- Визуализируйте, как изменение параметров влияет на результаты.

6. Анализ результатов:

- Сравните результаты обнаружения линий и форм с использованием преобразования Хафа.
- Проанализируйте, какие параметры лучше всего подходят для конкретных типов линий и форм.

7. Дополнительное задание:

- Примените преобразование Хафа к нескольким изображениям с различным содержимым.
- Сравните результаты и определите, какие изображения лучше всего подходят для обнаружения с использованием преобразования Хафа.

8. Отчет:

- Составьте отчет, описывающий каждый этап работы, включая примеры кода, визуализацию результатов и выводы.