МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра математического обеспечения и применения ЭВМ

ОТЧЕТ

по лабораторной работе №2
по дисциплине «Цифровая обработка изображений»
Тема: Поиск особенностей в изображении

Студент гр. 7381 _____ Минуллин М.А.

Преподаватель _____ Черниченко Д.А.

Санкт-Петербург
2021

Постановка задачи.

Реализовать на языке Python с использованием библиотеки OpenCV программу, выполняющее поиск окружностей в изображении путем преобразования Хо (Hough).

Входные данные.

1. Цветное изображение в формате bmp, jpg
2. Минимальное расстояние между центрами окружностей;
3. Минимальный радиус окружности;
4. Максимальный радиус окружности.

Выходные данные.

Цветное изображение в формате bmp (result.bmp)

Выполнение работы.

Входные данные будем принимать в качестве параметров командной строки:

```
filename = sys.argv[1]
min_distance = int(sys.argv[2])
min_radius = int(sys.argv[3])
max_radius = int(sys.argv[4])
```

Загрузим изображение из файла и отобразим на экране (результат представлен на рис. 1):

```
image = cv2.imread(filename)
cv2.imshow("Original image", image)
```

Рисунок 1 – Исходное изображение

Преобразуем изображение в чёрно-белое и отобразим на экране (результат представлен на рис. 2):

```
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
cv2.imshow("Gray image", gray)
```

Рисунок 2 – Черно-белое изображение

Для упрощения дальнейшего распознавания окружностей можно наложить эффект размытия, что сделает объекты на изображении более монотонного цвета и более гладкими (результат представлен на рис. 3):

```
gray = cv2.medianBlur(gray, 5)
cv2.imshow("Blurred gray image", gray)
```

Рисунок 3 – Размытое изображение

На размытом чёрно-белом изображении осуществим поиск окружностей путем преобразования Хо:

```
circles = cv2.HoughCircles(gray, cv2.HOUGH_GRADIENT, 1.7, param1=80, param2=57, minDist=min_distance, minRadius=min_radius, maxRadius=max_radius)
```

Найденные окружности наложим на исходное изображение и отобразим изображение на экране (результат представлен на рис. 4):

```
if circles is not None:
    circles = np.round(circles[0, :]).astype("int")
    for (x, y, r) in circles:
        cv2.circle(image, (x, y), r, (0, 255, 0), 4)
        cv2.rectangle(image, (x - 5, y - 5), (x + 5, y + 5), (0, 128, 255), -1)
cv2.imshow("Circles", image)
```

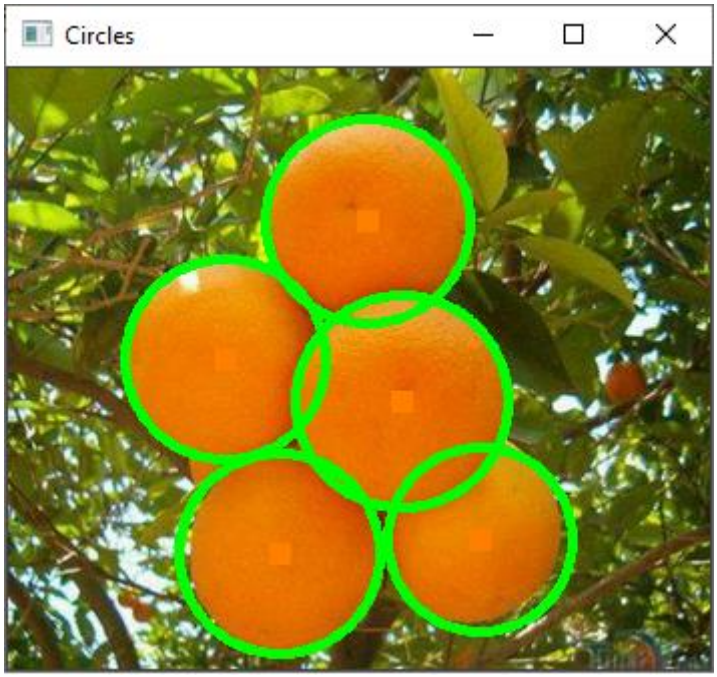

Рисунок 4 – Размеченное изображение

Сохраним полученное изображение с именем «result.bmp»:

```
cv2.imwrite('result.bmp', image)
```

Для исходного изображения достаточно оптимальными значениями оказались следующие: минимальная дистанция между центрами окружностей – 50 пикселей, минимальный радиус окружности – 30 пикселей, максимальный радиус окружностей – 60 пикселей.

Полный код программы представлен в приложении А.

Выводы.

В ходе выполнения лабораторной работы была реализована на языке Python с использованием библиотеки OpenCV программа, выполняющая поиск окружностей в изображении путем преобразования Хо (Hough).

ПРИЛОЖЕНИЕ А

КОД ПРОГРАММЫ

```
import cv2
import numpy as np
import sys

filename = sys.argv[1]
min_distance = int(sys.argv[2])
min_radius = int(sys.argv[3])
max_radius = int(sys.argv[4])

image = cv2.imread(filename)
cv2.imshow("Original image", image)

gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
cv2.imshow("Gray image", gray)

gray = cv2.medianBlur(gray, 5)
cv2.imshow("Blurred gray image", gray)

circles = cv2.HoughCircles(gray, cv2.HOUGH_GRADIENT, 1.7, param1=80,
param2=57, minDist=min_distance, minRadius=min_radius,
maxRadius=max_radius)

if circles is not None:
    circles = np.round(circles[0, :]).astype("int")
    for (x, y, r) in circles:
        cv2.circle(image, (x, y), r, (0, 255, 0), 4)
        cv2.rectangle(image, (x - 5, y - 5), (x + 5, y + 5), (0, 128,
255), -1)
cv2.imshow("Circles", image)
cv2.imwrite('result.bmp', image)
cv2.waitKey(0)
```