

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

ОТЧЕТ
по лабораторной работе №2
по дисциплине «Информатика»
ТЕМА: ВВЕДЕНИЕ В АРХИТЕКТУРУ КОМПЬЮТЕРА

Студент гр. 2381

Кузнецов И.И.

Преподаватель

Шевская Н.В.

Санкт-Петербург

2022

Цель работы.

Изучить библиотеку Pillow. Применить полученные знания для решения задач, представленных в работе.

Задание.

1) Рисование пентаграммы в круге

Необходимо написать функцию solve(), которая рисует на изображении пентаграмму в окружности.

Функция solve() принимает на вход:

Изображение (img)

координаты центра окружности (x,y)

радиус окружности

Толщину линий и окружности (thickness)

Цвет линий и окружности (color) - представляет собой список (list) из 3-х целых чисел

Функция должна изменить исходное изображение и вернуть его изображение.

Примечание:

Вершины пентаграммы высчитывать по формуле:

$$\text{phi} = (\pi/5) * (2*i + 3/2)$$

$$\text{node_i} = (\text{int}(x_0 + r * \cos(\text{phi})), \text{int}(y_0 + r * \sin(\text{phi})))$$

x_0, y_0 - координаты центра окружности, в который вписана пентаграмма

r - радиус окружности

i - номер вершины от 0 до 4

2) Поменять местами участки изображения и поворот

Необходимо реализовать функцию solve, которая меняет местами два квадратных, одинаковых по размеру, участка изображений и поворачивает эти участки на 90 градусов по часовой стрелке, а затем поворачивает изображение на 90 градусов по часовой стрелке.

Функция solve() принимает на вход:

Квадратное изображение (img)

Координаты левого верхнего угла первого квадратного участка(x_0, y_0)

Координаты левого верхнего угла второго квадратного участка(x_1, y_1)

Длину стороны квадратных участков (width)

Функция должна сначала поменять местами переданные участки изображений. Затем повернуть каждый участок на 90 градусов по часовой стрелке. Затем повернуть всё изображение на 90 градусов по часовой стрелке.

Функция должна вернуть обработанное изображение, не изменяя исходное.

3) Средний цвет

Необходимо реализовать функцию solve, которая заменяет цвет каждого пикселя в области на средний цвет пикселей вокруг (не считая сам этот пиксель).

Функция solve() принимает на вход:

Изображение (img)

Координаты левого верхнего угла области (x_0, y_0)

Координаты правого нижнего угла области (x_1, y_1)

Функция должна заменить цвета каждого пикселя в этой области на средний цвет пикселей вокруг.

Пиксели вокруг :

8 самых близких пикселей, если пиксель находится в центре изображения

5 самых близких пикселей, если пиксель находится у стенки

3 самых близких пикселя, если пиксель находится в угле

Функция должна вернуть обработанное изображение, не изменяя исходное.

Средний цвет - берется целая часть от среднего каждой компоненты из rgb.
($\text{int}(\text{sum}(r)/\text{count}), \text{int}(\text{sum}(g)/\text{count}), \text{int}(\text{sum}(b)/\text{count})$)

Выполнение работы.

Из библиотеки Pillow импортируем модули Image, ImageDraw.
Импортируем Numpy, с псевдонимом np.

Функция pentagram(img, x, y, r, thickness, color):

Функция принимает на вход изображение, координаты центра окружности, толщину и цвет линий.

Создает объект `draw` для рисования. Вычисляет координаты прямоугольника (квадрата), в который вписана окружность. Затем рисует окружность с заданными параметрами. Пробегаясь по циклу, вычисляет координаты вершин пентаграммы и рисует линии, соединяющие вершины, с помощью метода `line`.

Функция изменяет исходное изображение, рисуя на нем вписанную в окружность пентаграмму, и возвращает его.

Функция `swap(img, x0, y0, x1, y1, width)`:

Функция получает на вход изображение, координаты верхних левых углов квадратных участков и длину их стороны.

В переменные `area1`, `area2` сохраняются обрезанные с помощью метода `crop` и повернутые на 90 градусов по часовой стрелке куски скопированного с помощью метода `copy` копии исходного изображения. Копирует исходное изображение в `new_img`. Используя метод `paste` вставляет `area1`, `area2` в `new_img`. Поворачивает изображение на 90 градусов по часовой стрелке.

Функция возвращает обработанное изображение, не изменяя исходное.

Функция `avg_color(img, x0, y0, x1, y0)`:

Функция получает на вход изображение и координаты верхнего левого и нижнего правого углов прямоугольной области, на которой необходимо заменить цвета пикселей.

В `new_img` копируется исходное изображение. В переменную `size` записывается кортеж, содержащий размеры исходного изображения. Двойным вложенным циклом пробегается по всем пикселям задаваемой области. Для каждого пикселя создается кортеж из координат всех возможных близлежащих пикселей. С помощью функции `filter` фильтруются кортежи, оставляя только те, которые содержат в себе координаты пикселей, не выходящих за границы изображения. В `sum_rgb` записываются суммы цветов по компонентам каждого пикселя. Вычисляется значение среднего цвета и с помощью метода `putpixel` исходный цвет заменяется на средний.

Функция возвращает обработанное изображение, не изменяя исходное.

Разработанный программный код см. в приложении А.

Результаты тестирования см. в приложении Б.

Выводы.

Была освоена библиотека Pillow. Полученные знания были применены на практике, для разработки программы, состоящей из трех функций: функция `pentagram` рисует пентаграмму в заданном месте на данном изображении, `swap` меняет расположение частей изображения и поворачивает их, `avg_color` меняет цвет пикселя на средний цвет окружающих его пикселей.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main_lb2.py

```
import numpy as np
from PIL import Image, ImageDraw

def swap(img, x0, y0, x1, y1, width):
    area1 = img.copy().crop((x0, y0, x0 + width, y0 + width)).rotate(-90)
    area2 = img.copy().crop((x1, y1, x1 + width, y1 + width)).rotate(-90)
    new_img = img.copy()
    new_img.paste(area2, (x0, y0))
    new_img.paste(area1, (x1, y1))
    return new_img.rotate(-90)




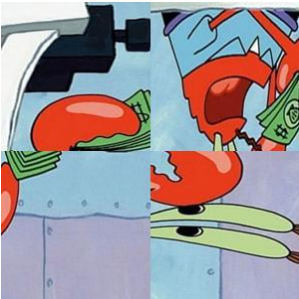


def avg_color(img, x0, y0, x1, y1):
    new_img = img.copy()
    size = img.size
    for x in range(x0, x1 + 1):
        for y in range(y0, y1 + 1):
            sum_rgb = [0, 0, 0]
            pixels_around = ((x + 1, y - 1), (x + 1, y), (x + 1, y + 1),
(x, y - 1), (x, y + 1), (x - 1, y - 1), (x - 1, y), (x - 1, y + 1))
            pixels_around = tuple(filter(lambda c: (0 <= c[0] <= size[0]
- 1) and (0 <= c[1] <= size[1] - 1), pixels_around))
            for i in pixels_around:
                pixel = img.getpixel(i)
                for j in range(3):
                    sum_rgb[j] += pixel[j]
            length = len(pixels_around)
            new_color = (int(sum_rgb[0] / length), int(sum_rgb[1] /
length), int(sum_rgb[2] / length))
            new_img.putpixel((x, y), new_color)
    return new_img

def pentagram(img, x, y, r, thickness, color):
    draw = ImageDraw.Draw(img, "RGB")
    x1 = x - r
    x2 = x + r
    y1 = y - r
    y2 = y + r
    draw.ellipse(((x1, y1), (x2, y2)), fill = None, width = thickness,
outline= tuple(color))
    coord = []
    for i in range(5):
        phi = (np.pi / 5) * (2 * i + 3 / 2)
        node_i = (int(x + r * np.cos(phi)), int(y + r * np.sin(phi)))
        coord.append((node_i))
    coord = tuple(coord)
    for i in range(5):
        draw.line((coord[i], coord[(i + 2) % 5]), fill = tuple(color), width
= thickness)
    return img
```

ПРИЛОЖЕНИЕ Б

ТЕСТИРОВАНИЕ

Таблица Б.2 - Примеры тестовых случаев

№ п/п	Входные данные	Выходные данные	Комментарии
1.			pentagram(img,130, 130, 120, 7, (255, 100, 50))
2.			swap(img,0, 0,150,150, 150)
3.			avg_color(img,0,0,299 , 299)