**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра математического обеспечения и применения ЭВМ**

отчет

**по лабораторной работе №1**

**по дисциплине «Цифровая обработка изображений»**

**Тема: Геометрические преобразования изображений**

|  |  |  |
| --- | --- | --- |
| Студентка гр. 7381 |  | Алясова А.Н. |
| Преподаватель |  | Черниченко Д.А. |

Санкт-Петербург

2021

**Постановка задачи.**

Реализовать на языке Python с использованием библиотеки OpenCV программу, выполняющее геометрические преобразования изображения.

**Входные данные**.

1. Цветное изображение в формате bmp, jpg.
2. Угол поворота - целое число.
3. Коэффициент - рациональное число.
4. Признак используемой интерполяционной схемы (0 – нулевого порядка, 1 – первого порядка, 2- третьего порядка).

**Выходные данные.**

Цветное изображение в формате bmp (result.bmp), являющиеся результатом следующих преобразований:

1. Исходное изображение поворачивается на градусов против часовой стрелки.
2. Из получившегося изображения вырезается прямоугольник максимальной площади, вписанный в повернутое изображение.
3. Данное изображение увеличивается по оси в раз и уменьшается по оси в раз.
4. Результат с использованием интерполяции соответствующего порядка записывается в файл result.bmp.

**Выполнение работы.**

Загрузим цветное изображение в формате .jpg (рис.1).

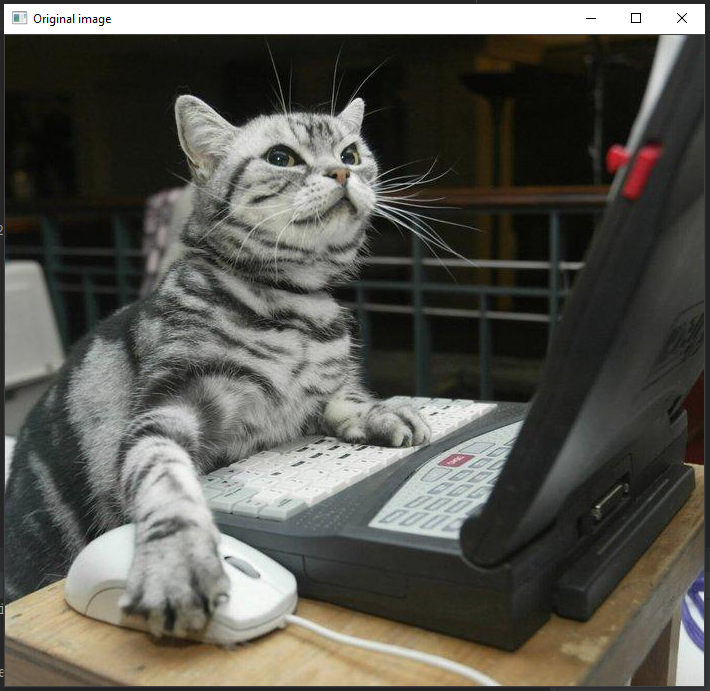


Рисунок 1

Зададим угол поворота (рис. 2).

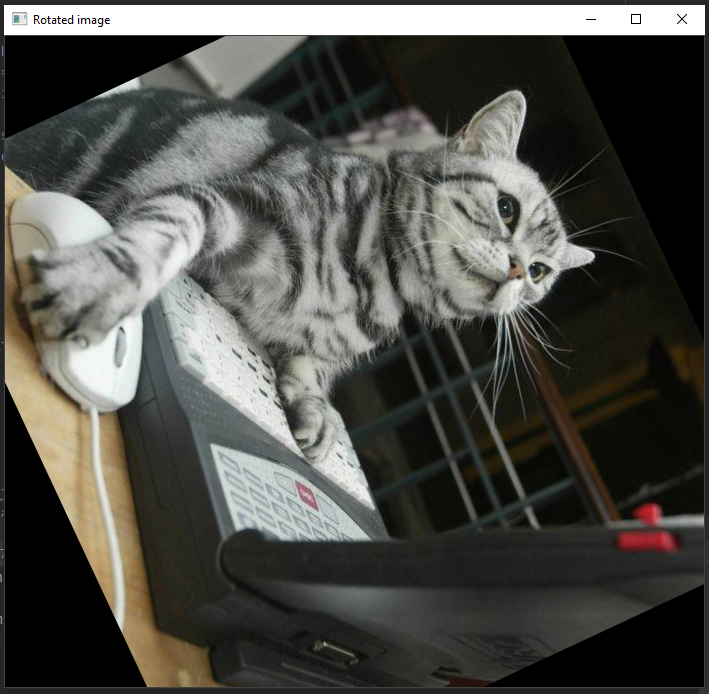


Рисунок 2

Коэффициент возьмем .



Рисунок 3

Из изображения, представленного на рис. 2 вырежем прямоугольник наибольшей площади, вписанный в повёрнутое изображение. Отобразим полученный результат на экране (результат представлен на рис. 4). Признак используемой интерполяционной схемы выберем третьего порядка (равный 2). Результат представлен на рис. 4.

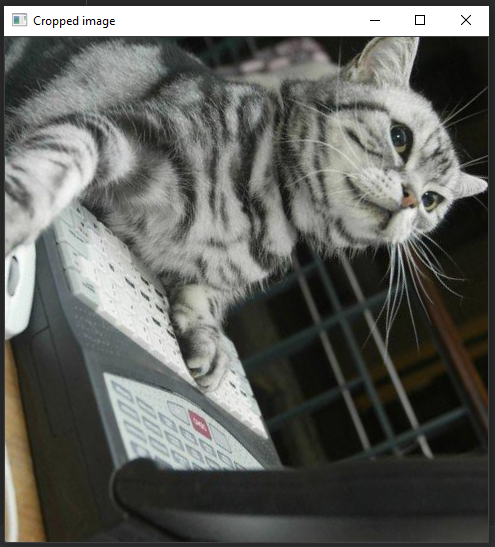


Рисунок 4

**Выводы.**

В ходе выполнения лабораторной работы была реализована программа на языке Python с использованием библиотеки OpenCV, выполняющая геометрические преобразования изображения.

**ПРИЛОЖЕНИЕ А**

**ИСХОДНЫЙ КОД**

import cv2 as cv

import math

import sys

def crop\_around\_center(image, width, height):

image\_size = (image.shape[1], image.shape[0])

image\_center = (int(image\_size[0] \* 0.5), int(image\_size[1] \* 0.5))

if width > image\_size[0]:

width = image\_size[0]

if height > image\_size[1]:

height = image\_size[1]

x1 = int(image\_center[0] - width \* 0.5)

x2 = int(image\_center[0] + width \* 0.5)

y1 = int(image\_center[1] - height \* 0.5)

y2 = int(image\_center[1] + height \* 0.5)

return image[y1:y2, x1:x2]

def largest\_rotated\_rect(w, h, angle):

quadrant = int(math.floor(angle / (math.pi / 2))) & 3

sign\_alpha = angle if ((quadrant & 1) == 0) else math.pi - angle

alpha = (sign\_alpha % math.pi + math.pi) % math.pi

bb\_w = w \* math.cos(alpha) + h \* math.sin(alpha)

bb\_h = w \* math.sin(alpha) + h \* math.cos(alpha)

gamma = math.atan2(bb\_w, bb\_w) if (w < h) else math.atan2(bb\_w, bb\_w)

delta = math.pi - alpha - gamma

length = h if (w < h) else w

d = length \* math.cos(alpha)

a = d \* math.sin(alpha) / math.sin(delta)

y = a \* math.cos(gamma)

x = y \* math.tan(gamma)

return (bb\_w - 2 \* x, bb\_h - 2 \* y)

def func(interpol):

return {

0: cv.INTER\_NEAREST,

1: cv.INTER\_LINEAR,

2: cv.INTER\_CUBIC

}[interpol]

filename = sys.argv[1]

alpha = int(sys.argv[2])

k = float(sys.argv[3])

interpol = int(sys.argv[4])

original\_image = cv.imread(filename)

cv.imshow('Original image', original\_image)

(rows, cols) = original\_image.shape[:2]

M = cv.getRotationMatrix2D(((cols - 1) / 2.0, (rows - 1) / 2.0), alpha, 1)

rotated\_image = cv.warpAffine(original\_image, M, (cols, rows))

cv.imshow('Rotated image', rotated\_image)

cropped\_image = crop\_around\_center(rotated\_image, \*largest\_rotated\_rect(cols, rows, math.radians(alpha)))

cv.imshow('Cropped image', cropped\_image)

resized\_image = cv.resize(cropped\_image, None, fx=k, fy=1.0/k, interpolation=func(interpol))

cv.imshow('Resized image', resized\_image)

cv.imwrite('result.bmp', resized\_image)

cv.waitKey(0)