МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования   
**«Нижегородский государственный университет им. Н.И. Лобачевского»**

**Национальный исследовательский университет**

**Институт информационных технологий, математики и механики**

**Кафедра: Программной инженерии**

Направление подготовки: «Программная инженерия»

«Обработка изображений»

**Отчёт по лабораторной работе**

|  |  |
| --- | --- |
|  | Выполнил:  студенты группы 381908-3  ФИО  Черемушкин Кирилл  Игоревич\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись)  Бражник Дмитрий Алексеевич\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись)  Козел Светлана  Александровна\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись)  Проверила:  Гетманская А.А  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) |

Нижний Новгород  
2021 г.

Содержание

[Задача 3](#_heading=h.gjdgxs)

[Работа 3](#_heading=h.30j0zll)

[Результаты и сравнения 4](#_heading=h.1fob9te)

[Код main.py 6](#_heading=h.2et92p0)

[Вывод 8](#_heading=h.tyjcwt)

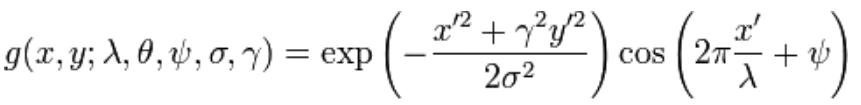
# Задача

Сделать обработку отпечатка пальца с помощью фильтров Габора.

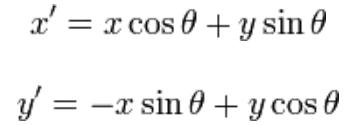
# Работа

В данной работе была создана программа для улучшения изображения отпечатка пальца при помощи фильтра Габора.

## **Представление функции Габора**



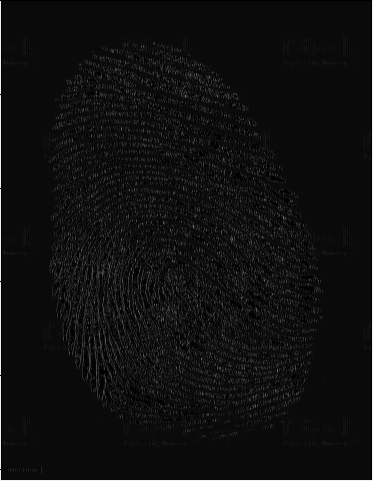
#### среди них:



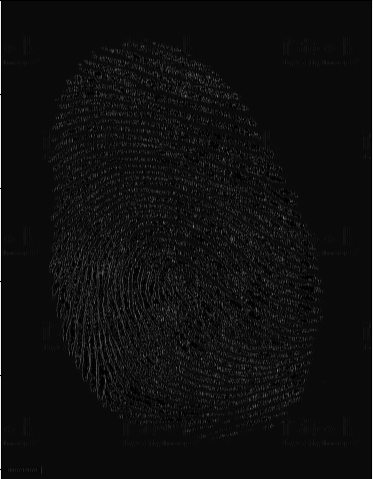
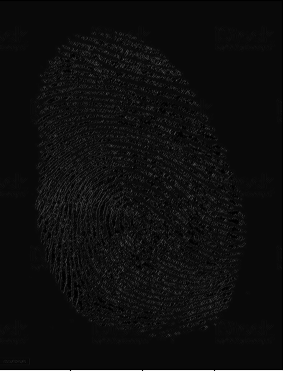
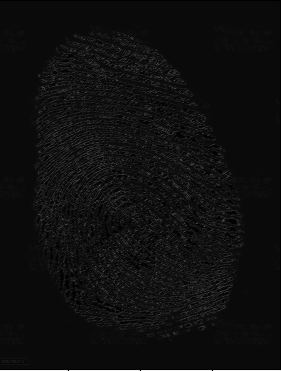
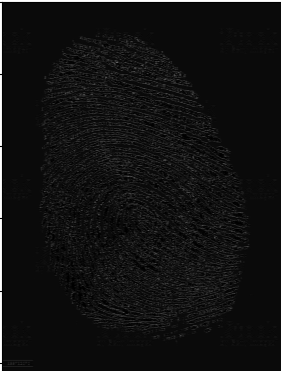
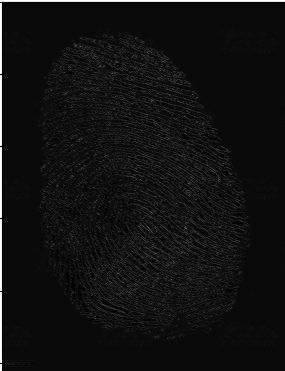
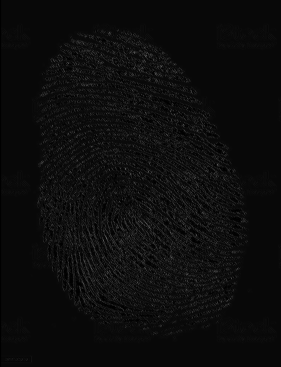
# Результаты и сравнения

Проблема с этой работой заключается в сложности подбора углов для разных отпечатков

Входящее изображение Фильтр Габора Бинарное изображение

Проход фильтра Габора под 6-ю разными углами (0, 30, 60, 90, 120, 150)

# Код main.py

import cv2

import numpy as np

import matplotlib.pyplot as plt

def BGR2GRAY(img):

gray = 0.2126 \* img[..., 2] + 0.7152 \* img[..., 1] + 0.0722 \* img[..., 0]

return gray

def Gabor\_filter(K\_size=111, Sigma=7, Gamma=1.2, Lambda=10, Psi=0, angle=0):

d = K\_size // 2

gabor = np.zeros((K\_size, K\_size), dtype=np.float32)

for y in range(K\_size):

for x in range(K\_size):

px = x - d

py = y - d

theta = angle / 180. \* np.pi

\_x = np.cos(theta) \* px + np.sin(theta) \* py

\_y = -np.sin(theta) \* px + np.cos(theta) \* py

gabor[y, x] = np.exp(-(\_x \*\* 2 + Gamma \*\* 2 \* \_y \*\* 2) / (2 \* Sigma \*\* 2)) \* np.cos(

2 \* np.pi \* \_x / Lambda + Psi)

#[y, x] = np.exp(-(x \*\* 2 + y \*\* 2) / (2 \* Sigma \*\* 2)) \* np.cos(2 \* np.pi \* Lambda) \

#\* (np.sin(theta) \* px + np.cos(theta) \* py)

gabor /= np.sum(np.abs(gabor))

return gabor

def Gabor\_filtering(gray, K\_size=111, Sigma=7, Gamma=1.2, Lambda=10, Psi=0, angle=0):

H, W = gray.shape

out = np.zeros((H, W), dtype=np.float32)

gabor = Gabor\_filter(K\_size=K\_size, Sigma=Sigma, Gamma=Gamma, Lambda=Lambda, Psi=0, angle=angle)

out = cv2.filter2D(gray, -1, gabor)

out = np.clip(out, 0, 255)

out = out.astype(np.uint8)

return out

def Gabor\_process(img):

H, W, \_ = img.shape

gray = BGR2GRAY(img).astype(np.float32)

#As = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 121, 122, 123, 124, 125, 126, 127, 128, 129, 130, 131, 132, 133, 134, 135, 136, 137, 138, 139, 140, 141, 142, 143, 144, 145, 146, 147, 148, 149, 150, 151, 152, 153, 154, 155, 156, 157, 158, 159, 160, 161, 162, 163, 164, 165, 166, 167, 168, 169, 170, 171, 172, 173, 174, 175, 176, 177, 178, 179]

#As = [25, 65, 70, 80, 90, 115, 120, 125, 130, 135, 140, 145]

As = [0, 30, 60, 90, 120, 150]

#As = [0, 45, 90, 135, 150, 165]

#As = [0, 15, 30, 45, 60, 75, 90, 105, 120, 135, 150, 165]

#As = [0, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60, 65, 70, 75, 80, 85, 90, 95, 100, 105, 110, 115, 120, 125, 130, 135, 140, 145, 150, 155, 160, 165, 170, 175]

plt.subplots\_adjust(left=0, right=1, top=1, bottom=0, hspace=0, wspace=0.2)

out = np.zeros([H, W], dtype=np.float32)

for i, A in enumerate(As):

\_out = Gabor\_filtering(gray, K\_size=11, Sigma=1.5, Gamma=1.2, Lambda=3, angle=A)

plt.imshow(\_out, cmap='gray')

plt.show()

out += \_out

out = out / out.max() \* 255

out = out.astype(np.uint8)

return out

img = cv2.imread(r'C:\Users\kirill\PycharmProjects\pythonProject10\tmp2.JPG').astype(np.float32)

img1 = cv2.imread(r'C:\Users\kirill\PycharmProjects\pythonProject10\tmp2.JPG')

def main():

cv2.imshow("orig", img1)

out = Gabor\_process(img)

cv2.imshow("result", out)

thresh = 36

img\_binary = cv2.threshold(out, thresh, 255, cv2.THRESH\_BINARY)[1]

cv2.imshow("result\_b-w", img\_binary)

cv2.waitKey(0)

cv2.destroyAllWindows()

if \_\_name\_\_ == '\_\_main\_\_':

main()

# Вывод

Нашей командой был применен фильтр Габора для улучшение изображения. На нем отчетливо видно линии отпечатков без шумов. Под каждый отпечаток нужен свой набор углов, сигм и лямд.