

**Московский государственный технический
университет им. Н.Э. Баумана**

Факультет «Информатика и системы управления»
Кафедра ИУ5 «Системы обработки информации и управления»

Курс «ПиКЯП»

Отчет по ДЗ

«Распознавание лиц»

Выполнил:

студент группы ИУ5-36Б

Илюхин И. Д.

Подпись и дата: 18.12.2024

Проверил:

преподаватель каф. ИУ5

Нардид А. Н.

Подпись и дата:

Москва, 2024 г.

Аннотация

Программа написана на языке Python и имеет следующий функционал:

1. Просмотр загруженного изображения
2. Ручное выделение области на изображении для размытия
3. Распознавание возраста, пола и эмоции человека
4. Автоматическое размытие лица(программа обнаруживает лицо/лица самостоятельно)

Код программы

Файл 'main.py':

```
import cv2
from Face_detect import detect_faces, detect_and_process_faces
from Face_function import pixelate_face
from Face_analize import analyze_face

def main_menu():
    image_path = "image/"
    image_path = image_path + input("Введите имя файла: ")
    while True:
        print("\nВыберите действие:")
        print("1. Просмотр изображения")
        print("2. Ручное выделение области лица")
        print("3. Распознавание лица (возраст, пол, эмоция)")
        print("4. Размытие лица")
        print("5. Выход")

        choice = input("Введите номер действия: ")

        if choice == '1':
            image = cv2.imread(image_path)
            if image is None:
                print("Не удалось загрузить изображение. Проверьте путь к файлу.")
            else:
                cv2.imshow("Image", image)
                cv2.waitKey(0)
                cv2.destroyAllWindows()

        elif choice == '2':
            detect_and_process_faces(image_path)

        elif choice == '3':
            analyze_face(image_path)

        elif choice == '4':
            image = cv2.imread(image_path)
            if image is None:
                print("Не удалось загрузить изображение. Проверьте путь к файлу.")
            else:
```

```

        original_image = image.copy()
        faces = detect_faces(image)
        if faces.any():
            for (x, y, w, h) in faces:
                pixelated_image = pixelate_face(original_image, x, y, x + w, y
+ h)
                cv2.imshow("Blurred Image", pixelated_image)
                cv2.waitKey(0)
                cv2.destroyAllWindows()
        else:
            print("На изображении не найдено ни одного лица для размытия.")

    elif choice == '5':
        print("Выход из программы.")
        break

    else:
        print("Неверный выбор. Пожалуйста, выберите из предложенных вариантов.")

if __name__ == '__main__':
    main_menu()

```

файл 'Face_analize.py'

```

from deepface import DeepFace

def analyze_face(image_path):
    analysis = DeepFace.analyze(img_path=image_path, actions=['age', 'gender',
'emotion'])

    if isinstance(analysis, list):
        for i, face in enumerate(analysis):
            print(f"Лицо {i + 1}:")
            print(" Возраст:", face['age'])
            print(" Пол:", face['dominant_gender'])
            print(" Эмоция:", face['dominant_emotion'])
    else:
        print("Возраст:", analysis['age'])
        print("Пол:", analysis['dominant_gender'])
        print("Эмоция:", analysis['dominant_emotion'])

```

файл “Face_detect.py”

```

import cv2

face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades +
'haarcascade_frontalface_default.xml')

roi_start = None

```

```
roi_end = None
cropping = False

def mouse_crop(event, x, y, flags, param):
    global roi_start, roi_end, cropping

    if event == cv2.EVENT_LBUTTONDOWN:
        roi_start = (x, y)
        cropping = True

    elif event == cv2.EVENT_LBUTTONUP:
        roi_end = (x, y)
        cropping = False
        cv2.rectangle(param, roi_start, roi_end, (0, 255, 0), 2)
        cv2.imshow("Image", param)

def detect_faces(image):
    gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    faces = face_cascade.detectMultiScale(gray_image, scaleFactor=1.1, minNeighbors=5,
minSize=(30, 30))
    return faces

def pixelate_region(image, x1, y1, x2, y2, pixel_size=10):
    if x1 < 0 or y1 < 0 or x2 > image.shape[1] or y2 > image.shape[0] or x1 >= x2 or
y1 >= y2:
        print("Некорректная область для пикселизации.")
        return image

    roi = image[y1:y2, x1:x2]
    if roi.size == 0:
        print("Выбранная область пустая. Пропускаем пикселизацию.")
        return image

    roi = cv2.resize(roi, (pixel_size, pixel_size), interpolation=cv2.INTER_LINEAR)
    roi = cv2.resize(roi, (x2 - x1, y2 - y1), interpolation=cv2.INTER_NEAREST)
    image[y1:y2, x1:x2] = roi
    return image

def detect_and_process_faces(image_path):
    global roi_start, roi_end

    image = cv2.imread(image_path)
    if image is None:
        print("Не удалось загрузить изображение. Проверьте путь к файлу.")
        return

    original_image = image.copy()
    faces = detect_faces(image)

    for (x, y, w, h) in faces:
        cv2.rectangle(image, (x, y), (x + w, y + h), (255, 0, 0), 2)
```

```

cv2.imshow("Image", image)
cv2.setMouseCallback("Image", mouse_crop, image)
cv2.waitKey(0)
cv2.destroyAllWindows("Image")

if roi_start and roi_end:
    x1, y1 = roi_start
    x2, y2 = roi_end
    x1, y1, x2, y2 = min(x1, x2), min(y1, y2), max(x1, x2), max(y1, y2)

pixelated_image = pixelate_region(original_image, x1, y1, x2, y2)
cv2.imshow("Processed Image", pixelated_image)
cv2.waitKey(0)
cv2.destroyAllWindows("Processed Image")

save_choice = input("Сохранить изменения в старый файл? (y/n): ")
if save_choice.lower() == 'y':
    try:
        cv2.imwrite(image_path, pixelated_image)
        print("Изображение сохранено.")
    except Exception as e:
        print(f"Ошибка сохранения изображения: {e}")

if __name__ == "__main__":
    image_path = input("Введите путь к изображению: ")
    detect_and_process_faces(image_path)

```

файл “Face_function.py”

```

import cv2

def pixelate_face(image, x1, y1, x2, y2, pixel_size=10):
    if x1 < 0 or y1 < 0 or x2 > image.shape[1] or y2 > image.shape[0] or x1 >= x2 or
y1 >= y2:
        print("Некорректная область для пикселизации.")
        return image

    roi = image[y1:y2, x1:x2]

    roi_small = cv2.resize(roi, (pixel_size, pixel_size),
interpolation=cv2.INTER_LINEAR)

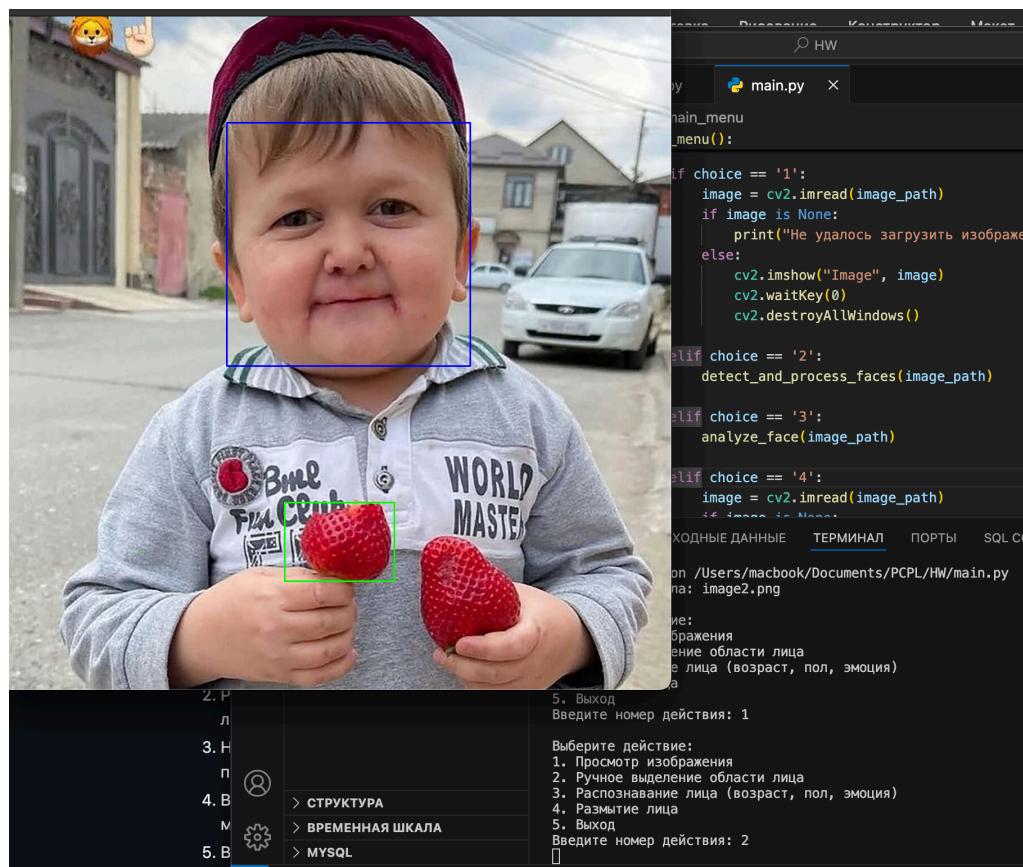
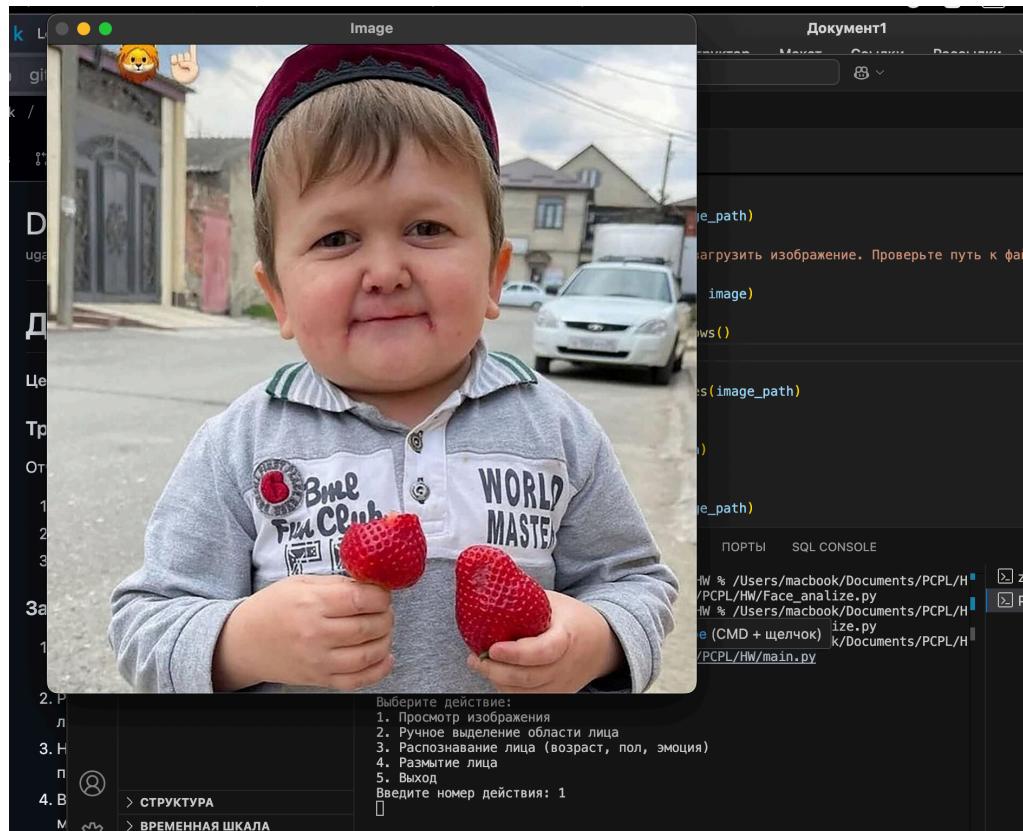
    roi_pixelated = cv2.resize(roi_small, (x2 - x1, y2 - y1),
interpolation=cv2.INTER_NEAREST)

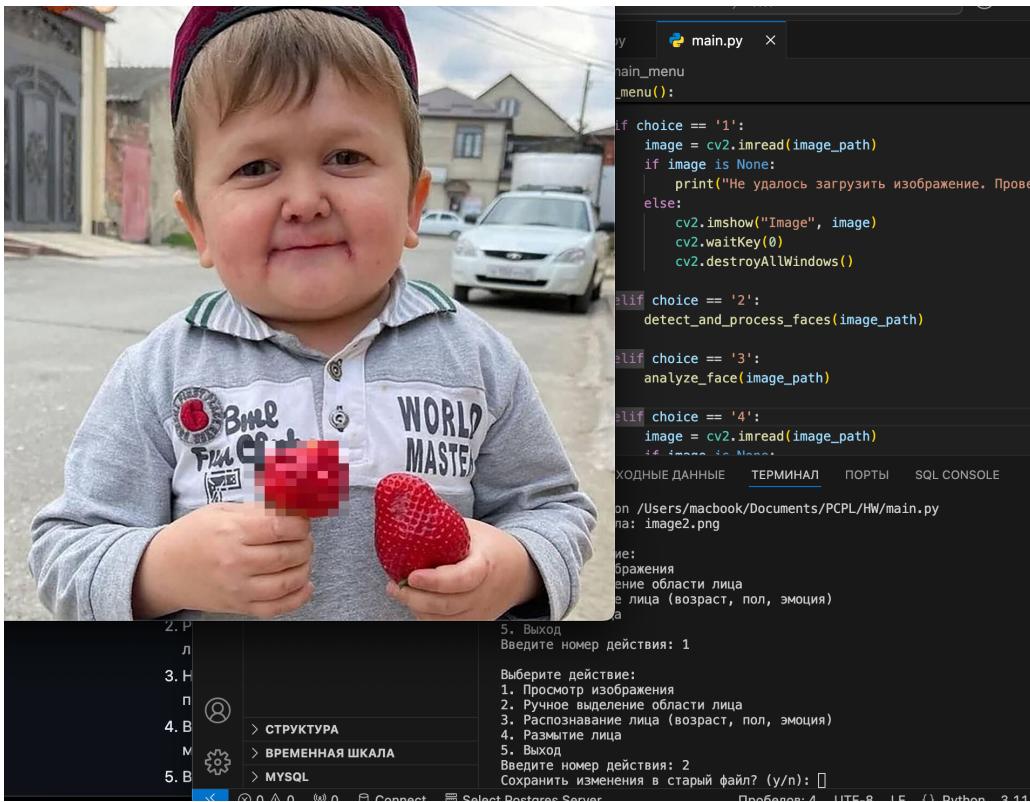
    image[y1:y2, x1:x2] = roi_pixelated

    return image

```

Пример работы:





```
Сохранить изменения в старый файл? (y/n): n

Выберите действие:
1. Просмотр изображения
2. Ручное выделение области лица
3. Распознавание лица (возраст, пол, эмоция)
4. Размытие лица
5. Выход

Введите номер действия: 3
Action: emotion: 100%|██████████| 3/3 [00:03<00:00, 1.07s/it]
Лицо 1:
Возраст: 24
Пол: Ман
Эмоция: neutral

Выберите действие:
1. Просмотр изображения
2. Ручное выделение области лица
3. Распознавание лица (возраст, пол, эмоция)
4. Размытие лица
5. Выход

Введите номер действия: █
```

