Минобрнауки России

Юго-Западный государственный университет

Кафедра программной инженерии

КУРСОВАЯ РАБОТА (ПРОЕКТ)

по дисциплине « Методы и алгоритмы обработки изображений »
(наименование дисциплины)

на тему « Системы распознавания на основе изображений лиц и карты пропуска »

Направление подготовки (специальность) 09.03.04
(код, наименование)

Программная инженерия

Автор работы (проекта) А.В. Боев _____
(инициалы, фамилия) (подпись, дата)

Группа ПО-11б

Руководитель работы (проекта) Р.А. Томакова _____
(инициалы, фамилия) (подпись, дата)

Работа (проект) защищена _____
(дата)

Оценка _____

Члены комиссии _____ _____
(подпись, дата) (инициалы, фамилия)

_____ _____
(подпись, дата) (инициалы, фамилия)

_____ _____
(подпись, дата) (инициалы, фамилия)

Курск 2024 г.

Минобрнауки России

Юго-Западный государственный университет

Кафедра программной инженерии

ЗАДАНИЕ НА КУРСОВУЮ РАБОТУ (ПРОЕКТ)

Студент А.В. Боев шифр 21-06-0138 группа ПО-11б
(инициалы, фамилия)

1. Тема Системы распознавания на основе изображений лиц и карты пропуска

2. Срок представления работы (проекта) к защите « ____ » _____ 20 ____ г.

3. Исходные данные (*для проектирования, для научного исследования*):

Определяются требованиями пользователя.

4. Содержание пояснительной записки курсовой работы (проекта):

4.1 ВВЕДЕНИЕ

4.2 Анализ и моделирование предметной области приложения

4.3 Техническое задание

4.4 Технический проект

4.5 Рабочий проект

4.6 ЗАКЛЮЧЕНИЕ

4.7 СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

4.8 ПРИЛОЖЕНИЕ А

5. Перечень графического материала:

диаграмма прецедентов, диаграмма компонентов, внешний вид интерфейса

Руководитель работы (проекта) Р.А. Томакова _____
(инициалы, фамилия) (подпись, дата)

Задание принял к исполнению А.В. Боев _____
(инициалы, фамилия) (подпись, дата)

РЕФЕРАТ

Данный текстовый документ имеет объем 32 страницы и включает в себя 9 рисунков, 1 приложение, 10 библиографических источников.

Перечень ключевых слов: изображение, обнаружение объектов, распознавание лиц, машинное зрение, каскады Хаара, LBPH-модель.

Целью проекта является разработка программного инструмента, позволяющего выполнять распознавание на основе изображений лиц и карты пропуска.

В процессе разработки были реализованы следующие ключевые элементы: распознавание лиц и карты пропуска, с последующей проверкой для авторизации.

Для реализации программы использовался язык программирования Python, а также сторонние библиотеки для чтения и обработки изображения.

Разработанное приложение успешно прошло этапы тестирования и продемонстрировало свою эффективность при распознавании лиц.

ABSTRACT

This text document has a volume of 32 pages, the number of illustrations is 9, applications - 1. During the work on this course work 10 sources were used.

List of keywords: image, object detection, face recognition, computer vision, Haar cascades, LBPH model.

The aim of the project is to develop a software tool capable of performing recognition based on facial images and access cards.

During the development process, the following key elements were implemented: face and access card recognition, followed by verification for authorization purposes.

The program was implemented using the Python programming language, along with third-party libraries for reading and processing images.

The developed application successfully passed the testing phase and demonstrated its effectiveness in face recognition.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

На сегодняшний день цифровые изображения широко используются во всех сферах жизни. Обработка изображений, в частности их анализ и распознавание, играет важную роль в различных областях, включая безопасность, медицину, маркетинг и другие. Одной из актуальных задач обработки изображений является автоматическое распознавание лиц, что позволяет создавать системы контроля доступа, мониторинга и идентификации пользователей.

Целью данной курсовой работы является разработка системы для распознавания лиц и авторизации пользователей на основе изображений лиц и карты пропуска.

Для достижения поставленной цели необходимо решить следующие задачи:

– изучить основные подходы и методы распознавания лиц;

– разработать алгоритм, совмещающий распознавание лиц и проверку карты пропуска для авторизации пользователей;

– реализовать программное решение на основе Python;

– провести тестирование разработанного решения.

Данный отчет состоит из введения, 4 разделов основной части, заключения, списка использованных источников, приложения.

Во введении сформулирована цель работы, поставлены задачи разработки, приведено краткое содержание каждого из разделов.

В первом разделе приводится обзор алгоритмов распознавания лиц.

Во втором разделе рассматриваются требования к программному решению и особенности его реализации.

В третьем разделе описывается программная реализация распознавания лиц.

В четвертом разделе представлены результаты разработки и тестирования программы.

В заключении излагаются основные результаты работы.

В приложении А представлен фрагмент исходного кода программы.

1 Анализ предметной области

1.1 Распознавание лиц

Распознавание лиц — это технология, основанная на анализе и сравнении уникальных характеристик лица человека. Она является одной из ключевых задач компьютерного зрения, поскольку лицо, как биометрический признак, обладает уникальными и стабильными чертами, которые можно использовать для идентификации личности.

1.2 Основные принципы работы каскадов Хаара

Обнаружение объектов с использованием каскадных классификаторов на основе признаков Хаара — это эффективный метод обнаружения объектов, предложенный Полом Виолой и Майклом Джонсом. Это подход, основанный на машинном обучении, где каскадная функция обучается на основе множества положительных и отрицательных изображений. Затем эта функция используется для обнаружения объектов на других изображениях. Первоначально, алгоритм требует довольно много положительных изображений (изображений лиц) и отрицательных изображений (изображений без лиц) для обучения классификатора. Далее необходимо извлечение из него особенности. Для этого используются признаки Хаара, показанные на рисунке ниже.

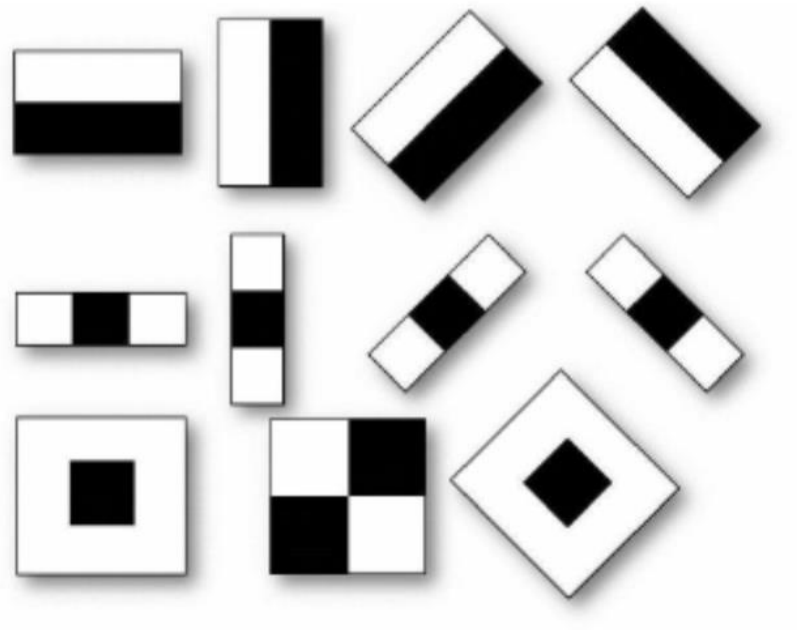

Рисунок 1.1 – признаки Хаара

Каждый объект представляет собой одно значение, полученное путем вычитания суммы пикселей под белым прямоугольником из суммы пикселей под черным прямоугольником.

Признаки Хаара состоят из смежных прямоугольных областей. Они позиционируются на изображении, далее суммируются интенсивности пикселей в областях, после чего вычисляется разность между суммами. Эта разность и будет значением определенного признака, определенного размера, определенным образом, помещенным на изображение.

Идея этого метода состоит в том, чтобы проверять каждую область изображения одновременно. Затем для каждой области получается одно значение путем вычитания суммы пикселей под белым прямоугольником из суммы пикселей под черным прямоугольником. [1]

Основная идея распознавания лиц на основе каскадной стратегии заключается в быстром исключении большинства фоновых областей на ранней стадии для того, чтобы уменьшить объем вычислений на более поздней стадии.

1.3 Основные принципы работы LBPH-модели

Гистограммы локальных бинарных шаблонов (LBPH) — это алгоритм для распознавания лиц, который представляет локальные объекты, такие как края на лицах, и поэтому не так подвержен ошибкам, вызванным различными условиями освещения. LBPH предназначен для распознавания людей в различных условиях освещения и на заднем фоне, ему не нужно много изображений одного и того же человека со всеми возможными условиями, которые могут возникнуть. Конкретные черты человека в этом алгоритме представлены с помощью простого вектора данных.

Для обучения алгоритма требуется набор данных с несколькими изображениями лиц человека, которых нужно распознать, с разным расположением и выражением. Объем изображений для каждого человека существенно влияет на точность. Алгоритм преобразовывает изображение в оттенки серого, где каждое изображение представлено значением интенсивности в диапазоне от 0 до 255. Затем для каждого окна размером 3x3 пикселя он сравнивает центральное значение с другими 8 значениями, если значение на краю ниже, то устанавливает 0, иначе 1. Затем удаляется центральное значение. Окно теперь содержит только двоичные значения 0 и 1, поэтому можно объединить эти 8 значений в одно новое значение, которое можно рассматривать как двоичное число, и это число помещается как значения для пикселя в центре.

На следующем шаге изображение с последнего шага делится на несколько сеток, количество сеток зависит от требуемой точности, скорости вычислений или требований к памяти. Для каждой сетки мы создаем гистограмму (значения от 0 до 255, поскольку мы работаем в градациях серого), показывающую, как часто данное значение / интенсивность пикселей встречается в сетке. Объединяя каждую гистограмму, мы получаем новую большую гистограмму, которая представляет конкретную особенность человека.

2 Техническое задание

2.1 Основание для разработки

Основанием для разработки программного продукта является задание для курсовой работы по дисциплине «Методы и алгоритмы обработки изображений».

2.2 Цель и назначение разработки

Основной задачей данной курсовой работы является разработка системы распознавания на основе изображений лиц и карты пропуска с помощью каскадов Хаара на языке программирования Python.

Основной целью данной работы является изучение методов распознавания, а также реализация одного из них на практике для решения задачи распознавания на основе изображений лиц и карты пропуска.

Задачами данной разработки являются:

– реализация алгоритма распознавания лиц с помощью каскадов Хаара;

– реализация алгоритма, совмещающего распознавание лиц и проверку карты пропуска для авторизации пользователей;

– тестирование полученного приложения для проверки корректности работы алгоритмов.

2.3 Требования к программе или программному изделию

Программа должна включать в себя:

– отображение изображения с камеры;

– возможность определения лица на изображении с камеры;

– возможность распознавания лица на изображении с камеры;

– возможность определения карты пропуска на изображении с камеры;

– возможность распознавания карты пропуска на изображении с камеры;

– отображение результатов распознавания на изображении с камеры;

2.4 Моделирование вариантов использования

2.4.1 Диаграмма прецедентов

Для разрабатываемого приложения была создана модель, обеспечивающая наглядное представление различных вариантов использования программы. Данная модель помогает в анализе и проектировании функциональных возможностей разрабатываемого приложения, а также при выявлении взаимосвязей между алгоритмами обработки.

При построении диаграммы вариантов использования применяется унифицированный язык визуального моделирования UML.

На основании анализа предметной области в программе должны быть реализованы следующие прецеденты:

1. Сканирование лица;
2. Сканирование карты пропуска;
3. Добавление нового пользователя.

Рисунок 2.1 – Диаграмма прецедентов

2.4.2 Сценарии прецедентов программы

1. Сценарий для прецедента «Сканирование лица»:

(a) основной исполнитель: пользователь;

(b) заинтересованные лица и их требования: пользователю необходимо предоставить изображения лица для распознавания;

(c) предусловие: приложение запущено, пользователь имеет возможность предоставить камере изображения лица;

(d) основной успешный сценарий:

– пользователь передает изображения лица через камеру;

– программа обнаруживает лицо для дальнейшего распознавания;

– программа распознает лицо пользователя в соответствии с имеющейся информацией;

– в окне программы рядом с распознанным изображением лица появляется информация о распознанном пользователе.

2. Сценарий для прецедента «Сканирование карты пропуска»:

(a) основной исполнитель: пользователь;

(b) заинтересованные лица и их требования: пользователю необходимо предоставить карту пропуска для распознавания;

(c) предусловие: приложение запущено, пользователь имеет возможность предоставить камере карту пропуска;

(d) основной успешный сценарий:

– пользователь передает карту пропуска через камеру;

– программа обнаруживает карту пропуска для дальнейшего распознавания;

– программа считывает карту пропуска в соответствии с имеющейся информацией;

– в окне программы рядом с обнаруженной картой пропуска появляется информация об обнаруженной карте пропуска.

3. Сценарий для прецедента «Добавление нового пользователя»:

(a) основной исполнитель: пользователь;

(b) заинтересованные лица и их требования: пользователю необходимо сохранить изображения своего лица для внесения себя в базу данных системы;

(c) предусловие: приложение запущено, пользователь имеет возможность предоставить камере изображения лица;

(d) основной успешный сценарий:

– пользователь передает изображения лица через камеру;

– программа сохраняет изображения лица пользователя для внесения в базу данных;

2.5 Требования к программной документации

Разработка программной документации и программного изделия должна производиться согласно ГОСТ19.102-77 и ГОСТ34.601-90. Единая система программной документации.

3 Технический проект

3.1 Общая характеристика организации решения задачи

Необходимо спроектировать и разработать систему, которая должна выполнять распознавание на основе изображений лиц и карты пропуска.

Система представляет собой программу, позволяющую пользователю выполнять распознавание лиц и сканирование карты пропуска для авторизации. Интерфейс включает в себя окно, отображающее видеопоток с камеры, результаты распознавания лица и карты пропуска, а также статус авторизации пользователя. Программа автоматически определяет лицо и карту пропуска в кадре, сопоставляет данные и отображает результаты выполнения программы на экране.

3.2 Описание используемых библиотек и языков программирования

В процессе разработки программы используется язык программирования Python, а также сторонние библиотеки OpenCV, Pyzbar, NumPy и Python Imaging Library.

3.2.1 Язык программирования Python.

Язык программирования Python представляет собой высокоуровневый язык общего назначения, отличающийся лаконичным синтаксисом и высокой читаемостью кода. Он нашел широкое применение в разработке программного обеспечения различного назначения, включая веб-приложения, настольные программы, системы обработки данных, научные вычисления и задачи машинного обучения. Python обеспечивает возможность реализации объектно-ориентированного, функционального и процедурного стилей программи-

рования. Значительное преимущество языка заключается в его обширной стандартной библиотеке, а также множестве сторонних библиотек, упрощающих разработку программных решений, включая графические интерфейсы, сетевые взаимодействия и обработку изображений.

3.2.2 Библиотека OpenCV

OpenCV представляет собой мощную библиотеку компьютерного зрения и обработки изображений, предназначенную для выполнения широкого спектра задач, связанных с анализом изображений и видео. Она предоставляет высокоуровневые и низкоуровневые средства для работы с изображениями, включая функции для обработки, фильтрации, распознавания объектов и анализа движения. Основное преимущество OpenCV заключается в её высокой производительности, гибкости и поддержке широкого спектра алгоритмов компьютерного зрения. Благодаря интеграции с языком Python, OpenCV позволяет быстро разрабатывать приложения, связанные с обработкой изображений и видео, и является популярным инструментом как для учебных, так и для полноценных исследовательских и промышленных проектов.

3.2.3 Библиотека Pyzbar

Pyzbar представляет собой библиотеку Python, предназначенную для декодирования штрих-кодов и QR-кодов. Она предоставляет удобные средства для считывания данных из изображений, содержащих одно- и двумерные коды. Основное преимущество Pyzbar заключается в её простоте использования и поддержке множества популярных форматов кодов, таких как QR, EAN, UPC, Code 128 и других. Благодаря интеграции с языком Python, Pyzbar позволяет быстро добавлять функциональность распознавания штрих-кодов и QR-кодов в приложения.

3.2.4 Библиотека NumPy

NumPy представляет собой фундаментальную библиотеку для численных вычислений в Python, обеспечивающую поддержку многомерных массивов и высокоуровневых математических операций над ними. Благодаря высокоэффективным операциям с массивами и встроенным линейным алгебраическим методам, NumPy позволяет обрабатывать изображения больших размеров с минимальными затратами ресурсов, что делает её незаменимой в задачах, связанных с сжатием, декомпрессией и преобразованием графических данных.

3.2.5 Python Imaging Library

Python Imaging Library, сокращенно PIL — это библиотека для работы с растровыми изображениями, обеспечивающая поддержку различных форматов, таких как JPEG, PNG, BMP и других. PIL предоставляет широкий набор функций для загрузки, сохранения, обработки и преобразования изображений, что упрощает различные взаимодействия с файлами изображений, такие как сохранение и загрузка изображений, поэтому она является удобным инструментом для предварительной и финальной обработки графических данных.

3.3 Структура проекта

3.3.1 Диаграмма компонентов

На рисунке 3.1 представлена диаграмма компонентов разрабатываемой системы.

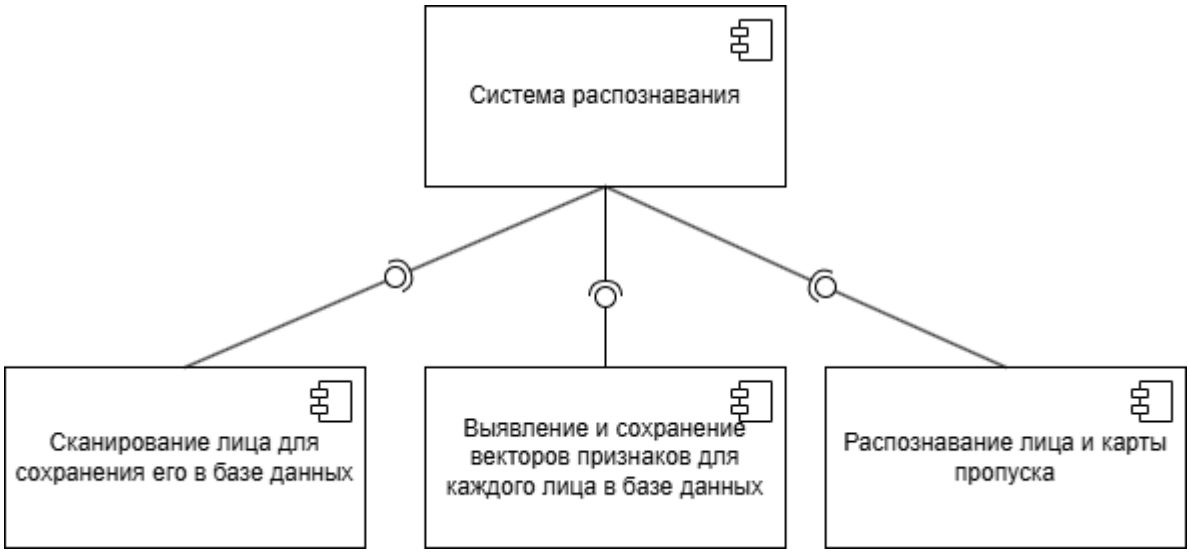

Рисунок 3.1 – Диаграмма компонентов разрабатываемой системы

Компонент «Сканирование лица для сохранения его в базе данных» представляет собой программу для сохранения лица для дальнейшего его распознавания.

Компонент «Выявление и сохранение векторов признаков для каждого лица в базе данных» записывает вычисления векторов признаков на основе множества изображений лиц. Результат – модель с сохраненными данными о лицах.

Компонент «Распознавание лица и карты пропуска» представляет собой программу для распознавания лиц и карты пропуска, отображения информации о них и проверки данных.

3.3.2 Подготовка изображения

Для успешного обнаружения лиц на изображении необходимо предварительно обработать его, убрав ненужные цвета. Важным шагом является конвертация цветного изображения в полутоновое. Это позволяет уменьшить объем данных, облегчая дальнейшую обработку. Каждый пиксель изображе-

ния преобразуется в уровень яркости с учетом взвешенной суммы его RGB-компонент: *Gray = 0.3R + 0.59G + 0.11B*. После создается новое изображение, где все пиксели представлены оттенками серого. Для каждого пикселя оригинального изображения рассчитывается значение серого и устанавливается в выходное изображение [6].

3.3.3 Работа алгоритма распознавания лица

Программа распознавания лиц начинает свою работу с чтения изображения, переданного пользователем, и преобразования полученных данных в массив. Извлекается и сохраняется имя файла для дальнейшей обработки. Затем изображение конвертируется в оттенки серого для упрощения дальнейшей обработки.

Далее начинается работа алгоритма распознавания лиц. Программа использует каскады Хаара для обнаружения лиц на изображении. Изображение сканируется с применением классификаторов, обученных на примерах лиц и фона, чтобы выделить области, содержащие лица.

После этого программа начинает извлечение признаков лица с помощью модели LBPH (Local Binary Patterns Histograms). Изображение лица делится на небольшие блоки, для каждого из которых вычисляются бинарные паттерны, представляющие текстуру. Эти паттерны собираются в гистограммы, которые образуют вектор признаков, описывающий уникальные характеристики лица.

Затем для каждого распознанного лица программа сравнивает полученные признаки с базой данных известных лиц, используя методы классификации. Если сходство между вектором признаков лица и одним из записанных в базе данных превышает определённый порог, лицо идентифицируется как соответствующее одному из известных. На рисунке 3.2 представлена схема алгоритма распознавания лица.

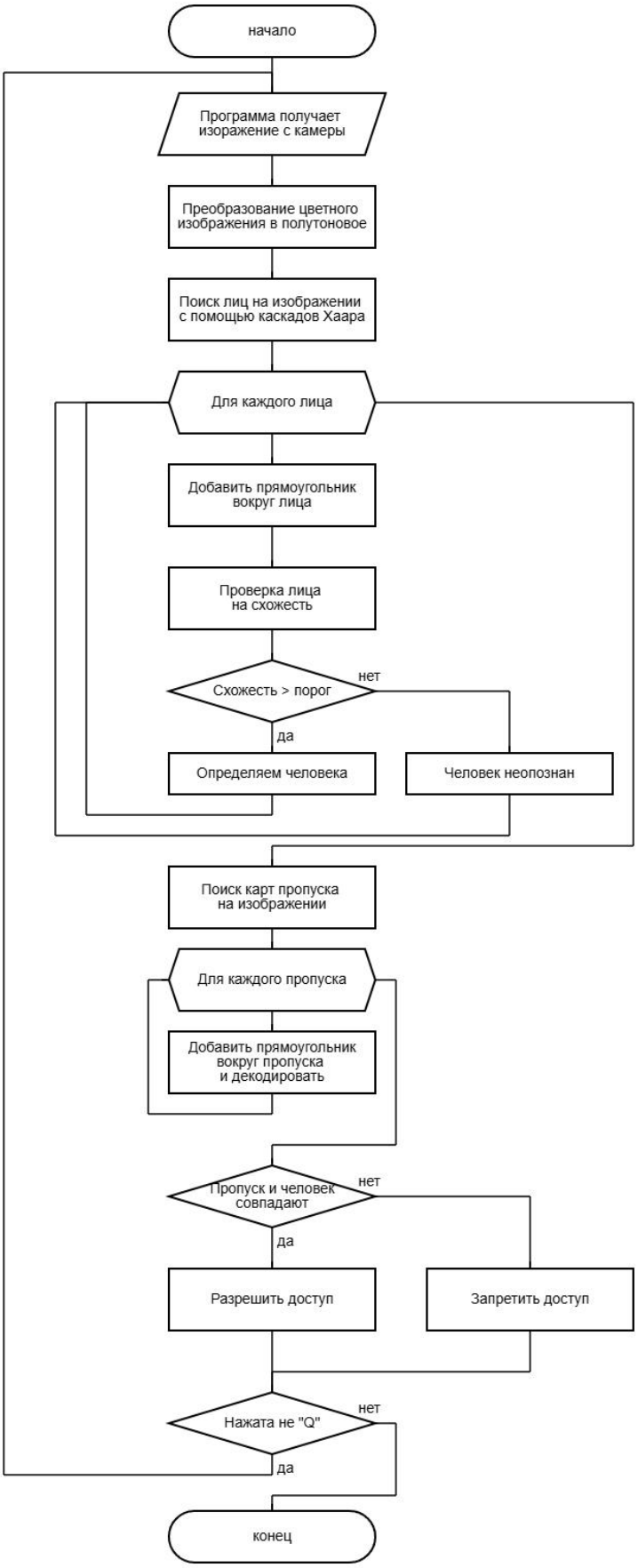

Рисунок 3.1 – Диаграмма компонентов разрабатываемой системы

4 Рабочий проект

4.1 Описание объектов интерфейса программы

На основе требований к системе в разделе 2.3 технического задания, с помощью языка программирования Python и IDE Visual Studio Code был создан интерфейс системы.

На рисунке 4.1 представлен данный интерфейс, отображающий информацию.

Программная реализация данного интерфейса представлена в виде исходного кода программы в ПРИЛОЖЕНИИ А.

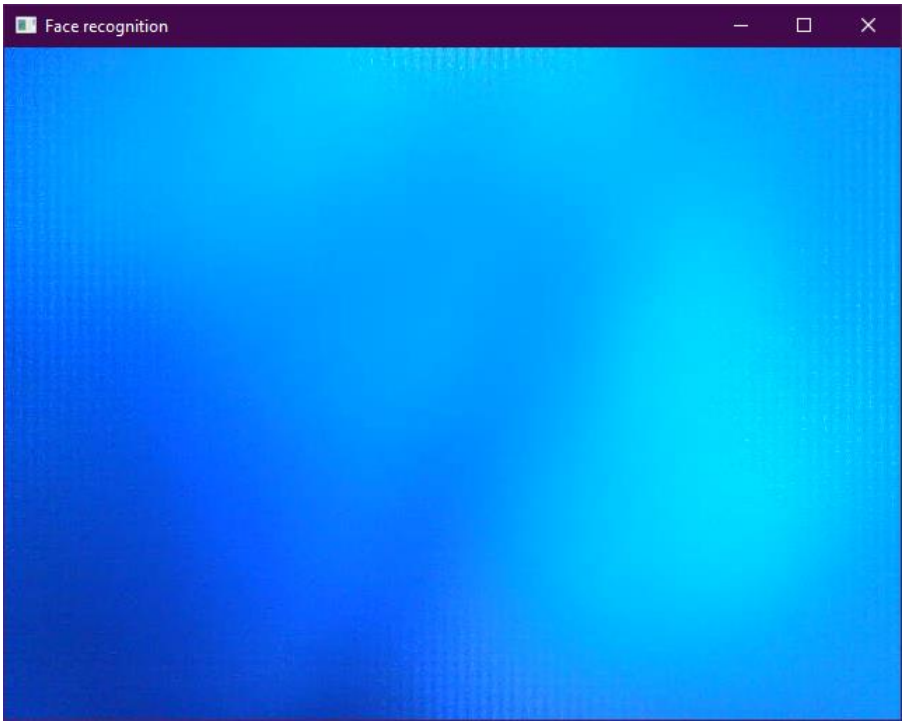

Рисунок 4.1 – Интерфейс взаимодействия пользователя и программы

4.2 Тестирование программной системы

Для отладки работы программы разработаны следующие тестовые наборы:

1 Случай использования: появление изображения лица

Предусловие: программа запущена;

Тестовый случай: перед камерой появляется изображение лица;

Ожидаемый результат: программа определяет лицо;

Результат представлен на рис. 4.2

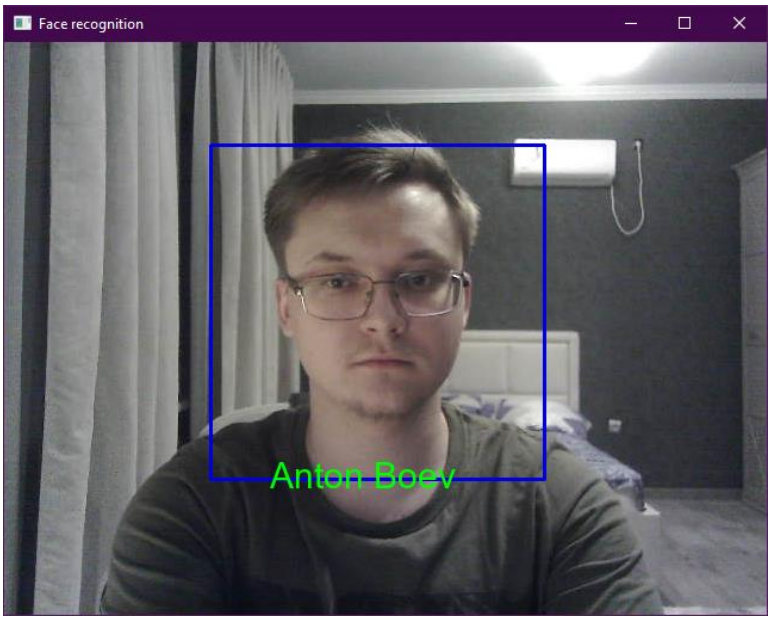

Рисунок 4.2 – Определение лица на изображении

2 Случай использования: появление карты пропуска

Предусловие: программа запущена;

Тестовый случай: перед камерой появляется карта пропуска;

Ожидаемый результат: программа определяет карту пропуска;

Результат представлен на рис. 4.3

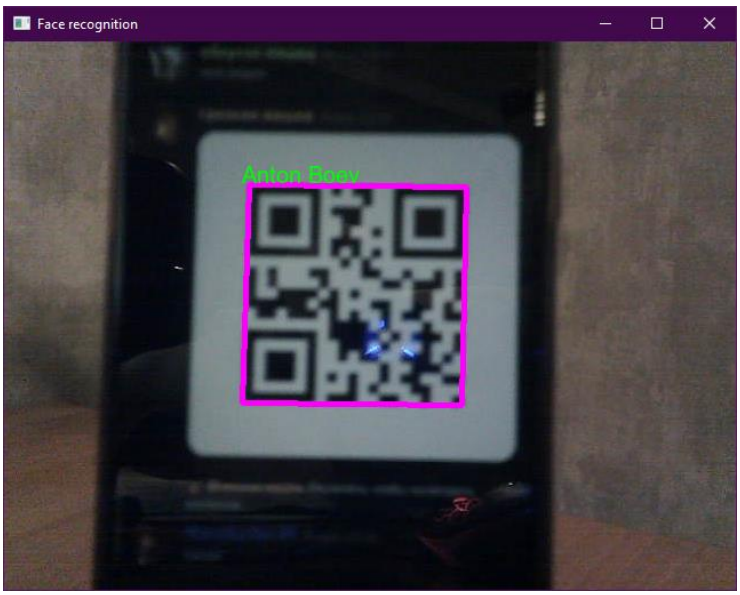

Рисунок 4.3 – Определение карты пропуска

3 Случай использования: несоответствие лица и карты пропуска

Предусловие: программа запущена;

Тестовый случай: перед камерой появляются несоответствующие друг другу изображение лица и карты пропуска;

Ожидаемый результат: система выведет сообщение о запрете доступа.

Результат представлен на рис. 4.4

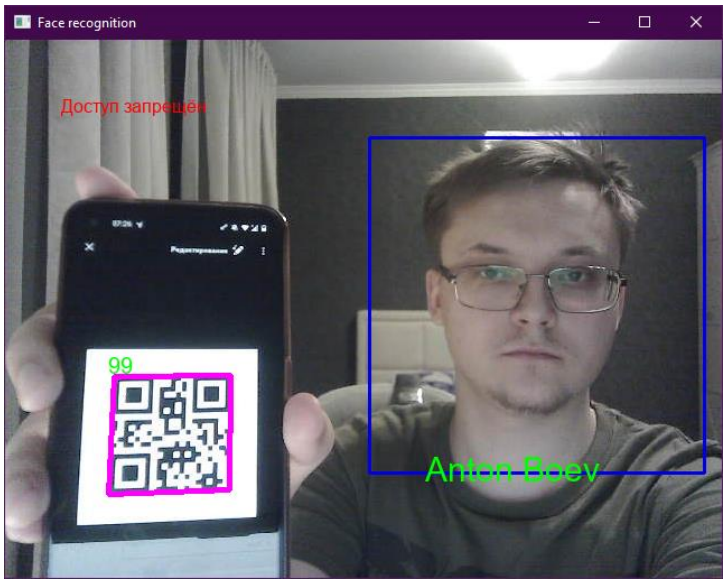

Рисунок 4.4 – сообщение «Доступ запрещён»

4 Случай использования: соответствие лица и карты пропуска

Предусловие: программа запущена;

Тестовый случай: перед камерой появляются соответствующие изображения лица и карты пропуска;

Ожидаемый результат: система выведет сообщение о разрешенном доступе.

Результат представлен на рис. 4.5

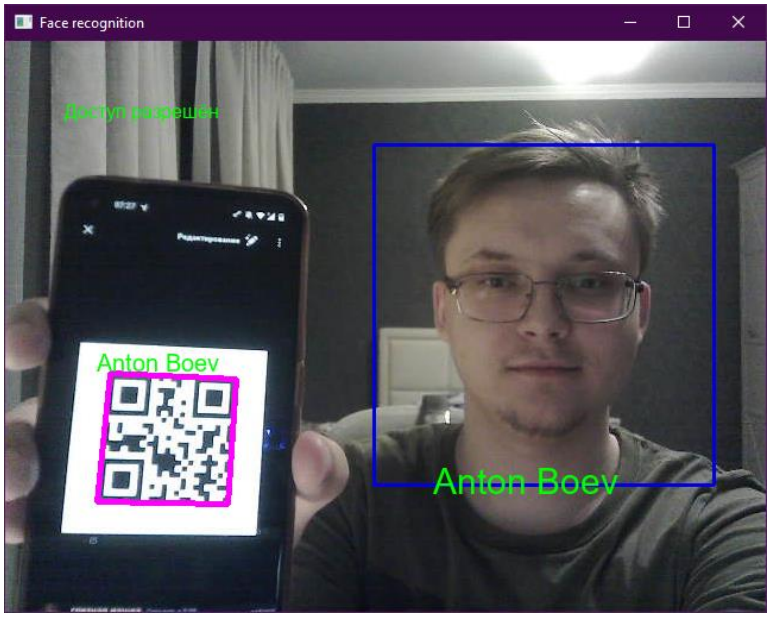

Рисунок 4.5 – сообщение «Доступ разрешён»

ЗАКЛЮЧЕНИЕ

В рамках данной работы была реализована система распознавания на основе изображений лиц и карты пропуска.

Разработанная программа позволяет распознавать лица и карты пропуска на изображениях с камеры и наблюдать на экране отображение информации о распознанных объектах.

В процессе выполнения курсовой работы на основе исследования предметной области были определены требования к программе. Для реализации этих требований была разработана система команд для управления процессом распознавания; спроектирована архитектура системы; реализованы алгоритмы и программные модули для распознавания лиц и проверки QR-кодов; проведено функциональное тестирование системы для проверки корректности работы всех модулей.

Все требования, объявленные в техническом задании, были полностью реализованы в данном программном продукте.

Все задачи, поставленные в начале разработки проекта, были решены.

Результатом работы является программное решение, которое может использоваться для распознавания лиц и проверки карты пропуска в целях авторизации пользователей. Это приложение обладает широкими возможностями для дальнейшего применения в задачах контроля доступа, включая использование в системах безопасности, учета рабочего времени и автоматизированных пропускных системах.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Амеличев Г. Э., Панина В. С., Белов Ю. С. Распознавание лиц с использованием каскадов Хаара — 2020. [Электронный ресурс]. URL: https://cyberleninka.ru/article/n/raspoznavanie-lits-s-ispolzovaniem-kaskadov-haara (дата обращения: 29.12.2024).

2. Е. А. Белых Обучение каскадов Хаара. Выпуск 1(22), 2017. [Электронный ресурс]. URL: https://cyberleninka.ru/article/n/obuchenie-kaskadov-haara (дата обращения: 19.12.2024).

3. Земцов А. Алгоритмы распознавания лиц. — М.: Издательский дом «Интеллект», 2023. — 320 с. ISBN 9783845426693. — Текст: непосредственный.

4. Сысоев И. В. Python. Разработка приложений. — М.: БХВ-Петербург, 2022. — 400 с. ISBN 978-5-9775-7891-1. — Текст: непосредственный.

5. Мартин, Р. Чистая архитектура. Искусство разработки программного обеспечения / Р. Мартин; пер. с англ. А. Кисилева. – Санкт-Петербург: Питер, 2018. – 351 с. – ISBN 978-5-4461-0772-8. – Текст: непосредственный.

6. Прэтт, У. Цифровая обработка изображений: основы и практические аспекты / У. Прэтт. – Санкт-Петербург: Питер, 2011.– 784 с.– ISBN 978-5 459-00445-9. – Текст: непосредственный.

7. Сысоева М. В., Сысоев И. В. Алгоритмизация и программирование на языке Python. В двух частях. Часть 1. — М.: Базальт СПО; МАКС Пресс, 2023. — 200 с. ISBN 978-5-317-06945-2. — Текст: непосредственный.

8. Сысоева М. В., Сысоев И. В. Алгоритмизация и программирование на языке Python. В двух частях. Часть 2. — М.: Базальт СПО; МАКС Пресс, 2023. — 184 с. ISBN 978-5-317-06947-6. — Текст: непосредственный.

9. Хамдамов У.Р., Умаров М.А., Умаров Х.А. Методы определения объектов на изображении, 2019. [Электронный ресурс]. URL:

https://www.researchgate.net/publication/340583958_METODY_OPREDELENIA_OBEKTOV_NA_IZOBRAZENII (дата обращения: 11.12.2024).

10. Баланов А. Н. Биометрия. Разработка и внедрение систем идентификации. Учебное пособие для вузов. — СПб.: Лань, 2024. — 228 с. ISBN 978-5-507-49167-4. — Текст: непосредственный.

ПРИЛОЖЕНИЕ А

Исходный код программы

face_test.py

```
import cv2
import os
import pickle  # Для загрузки словаря
from pyzbar.pyzbar import decode  # Для декодирования QR-кодов
import numpy as np
from PIL import Image, ImageDraw, ImageFont # Для отображения кириллицы

path = os.path.dirname(os.path.abspath(__file__))
recognizer = cv2.face.LBPHFaceRecognizer_create()
recognizer.read(path + r'/trainer/trainer.yml') # модель распознавания
faceCascade = cv2.CascadeClassifier(path + r'/trainer/haarcascade_frontalface_default.xml')

# загружаем словарь ID -> имена
with open(path + '/trainer/labels_to_names.pkl', 'rb') as f:
    labels_to_names = pickle.load(f)

cam = cv2.VideoCapture(0) # доступ к камере
font = cv2.FONT_HERSHEY_SIMPLEX # шрифт для вывода подписей

threshold = 70

qr_user_data = None
face_user_name = None

# функция для добавления текста с кириллицей
def put_text_with_pillow(img, text, position, font_size=30, color=(0, 255, 0)):
    pil_img = Image.fromarray(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
    draw = ImageDraw.Draw(pil_img)
    try:
        font = ImageFont.truetype("C:/Windows/Fonts/arial.ttf", font_size)  # Путь к шрифту
    except IOError:
        font = ImageFont.load_default()

    draw.text(position, text, font=font, fill=color)
    img = cv2.cvtColor(np.array(pil_img), cv2.COLOR_RGB2BGR)
    return img

while True:
    ret, im = cam.read()
    gray = cv2.cvtColor(im, cv2.COLOR_BGR2GRAY)
    faces = faceCascade.detectMultiScale(gray, scaleFactor=1.2, minNeighbors=5, minSize=(100, 100),
flags=cv2.CASCADE_SCALE_IMAGE)

    qr_user_data = None
    face_user_name = None

    for (x, y, w, h) in faces:
        nbr_predicted, coord = recognizer.predict(gray[y:y+h, x:x+w])
```

```
        print(coord)
        if coord > threshold:
            name = "Unknown"
        else:
            name = labels_to_names.get(nbr_predicted, "Unknown")
            face_user_name = name
        cv2.rectangle(im, (x-50, y-50), (x+w+50, y+h+50), (225, 0, 0), 2) # прямоугольник вокруг лица
        im = put_text_with_pillow(im, name, (x, y + h + 30), font_size=30, color=(0, 255, 0))

    qr_codes = decode(im)  # Декодируем QR-коды
    for qr_code in qr_codes:
        qr_data = qr_code.data.decode('utf-8')
        # прямоугольник вокруг QR-кода
        rect_points = qr_code.polygon
        if len(rect_points) == 4:
            pts = rect_points
        else:
            pts = qr_code.rect
            pts = [pts]
        pts = cv2.convexHull(np.array(pts, dtype=np.int32))
        cv2.polylines(im, [pts], True, (255, 0, 255), 3)
        im = put_text_with_pillow(im, qr_data, (qr_code.rect[0], qr_code.rect[1] - 20), font_size=20, col-
or=(0, 255, 0))
        qr_user_data = qr_data

    if qr_user_data and face_user_name:
        if qr_user_data == face_user_name:
            im = put_text_with_pillow(im, "Доступ разрешён", (50, 50), font_size=16, color=(0, 255, 0))
        else:
            im = put_text_with_pillow(im, "Доступ запрещён", (50, 50), font_size=16, color=(255, 0, 0))

    cv2.imshow('Face recognition', im)
    cv2.imshow('Gray', gray)

    if cv2.waitKey(10) & 0xFF == ord('q'):  # 'q', чтобы выйти
        break

cam.release()
cv2.destroyAllWindows()
```

face_gen.py

```
import cv2
import os, qr_gen

path = os.path.dirname(os.path.abspath(__file__))
detector = cv2.CascadeClassifier(path + r'/trainer/haarcascade_frontalface_default.xml')

i=0 # счётчик изображений
offset=50 # расстояния от распознанного лица до рамки
name=input('Введите имя пользователя: ')
video=cv2.VideoCapture(0)
```

```
while True:
    ret, im =video.read()
    gray=cv2.cvtColor(im,cv2.COLOR_BGR2GRAY)
    faces=detector.detectMultiScale(gray, scaleFactor=1.2, minNeighbors=5, minSize=(100, 100))
    # Обработка лиц
    for(x,y,w,h) in faces:
        # Защита от выхода за границы изображения
        x_start = max(0, x - offset)
        y_start = max(0, y - offset)
        x_end = min(im.shape[1], x + w + offset)
        y_end = min(im.shape[0], y + h + offset)

        face_region = gray[y_start:y_end, x_start:x_end]

        # Сохранение лица
        if face_region.size > 0:
            i=i+1
            cv2.imwrite(path + r'/Faces/face-' + name + '.' + str(i) + ".jpg", face_region)
            cv2.rectangle(im, (x_start, y_start), (x_end, y_end), (225, 0, 0), 2)
            cv2.imshow('im', face_region)
            cv2.waitKey(50)  # пауза перед новым кадром
    # если у нас хватает кадров
    if i>100:
        qr_gen.generate_qr(name, name)
        video.release()
        cv2.destroyAllWindows()
        break
```

face_train.py

```
import cv2

import os
import numpy as np
from PIL import Image
import pickle  # Для словаря

path = os.path.dirname(os.path.abspath(__file__))
recognizer = cv2.face.LBPHFaceRecognizer_create()
faceCascade = cv2.CascadeClassifier(path + r'/trainer/haarcascade_frontalface_default.xml')
dataPath = path + r'/Faces' # путь к датасету

# Получить изображения и метки (числовые)
def get_images_and_labels(datapath):
    image_paths = [os.path.join(datapath, f) for f in os.listdir(datapath)]
    images = []
    labels = []
    labels_to_names = {}  # Словарь id -> имя
    next_id = 0  # Уникальный ID для каждого имени

    for image_path in image_paths:
        # Чтение изображения и преобразование в оттенки серого
        image_pil = Image.open(image_path).convert('L')
        image = np.array(image_pil, 'uint8')
```

```
        # Извлечение имени из названия файла
        name = os.path.split(image_path)[1].split(".")[0].replace("face-", "")

        # Если имя новое, добавляем его в словарь
        if name not in labels_to_names.values():
            labels_to_names[next_id] = name
            label = next_id
            next_id += 1
        else:
            label = list(labels_to_names.keys())[list(labels_to_names.values()).index(name)]

        # Обнаружение лица
        faces = faceCascade.detectMultiScale(image)
        for (x, y, w, h) in faces:
            images.append(image[y: y + h, x: x + w])
            labels.append(label)
            cv2.imshow("Adding faces to training set...", image[y: y + h, x: x + w])
            cv2.waitKey(10)

    # Сохранение словаря id -> имя
    with open(os.path.join(path, 'trainer/labels_to_names.pkl'), 'wb') as f:
        pickle.dump(labels_to_names, f)

    return images, labels

# Получение изображений и меток
images, labels = get_images_and_labels(dataPath)
labels = np.array(labels, dtype='int')

# Обучение модели
recognizer.train(images, labels)
if not os.path.exists(os.path.join(path, 'trainer')):
    os.makedirs(os.path.join(path, 'trainer'))
recognizer.save(os.path.join(path, 'trainer/trainer.yml'))

cv2.destroyAllWindows()
```