

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
(ФГБОУ ВО «АмГУ»)

Институт компьютерных и инженерных наук
Кафедра информационных и управляющих систем
Направление подготовки 09.04.04 Программная инженерия
Направленность (профиль) образовательной программы Управление разработкой программного обеспечения

ДОПУСТИТЬ К ЗАЩИТЕ
Зав. кафедрой
_____ А.В. Бушманов
«__» _____ 2025 г.

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

на тему: Проектирование и реализация системы распознавания лиц

Исполнитель студент группы 3105-ом2	_____	Ван Цзяньюй
	(подпись, дата)	

Руководитель доцент, канд. техн. наук	_____	Т.А. Галаган
	(подпись, дата)	

Руководитель научного содержания программы магистратуры профессор, доктор техн. наук	_____	И.Е. Ерёмин
	(подпись, дата)	

Нормоконтроль инженер кафедры	_____	В.Н. Адаменко
	(подпись, дата)	

Рецензент старший преподаватель, канд. техн. наук	_____	Д.В. Фомин
	(подпись, дата)	

Благовещенск, 2025

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
(ФГБОУ ВО «АмГУ»)

Институт компьютерных и инженерных наук
Кафедра информационных и управляющих систем

УТВЕРЖДАЮ
Зав. кафедрой
_____ А.В. Бушманов
«___» _____ 2025 г.

З А Д А Н И Е

К магистерской диссертации студента группы 3105-ом2 Ван Цзяньюй

1. Тема магистерской диссертации: Проектирование и реализация системы распознавания лиц

(Утверждено приказом от 06.03.2025 № 609-уч)

2. Срок сдачи студентом законченной работы (проекта): 10.06.2025

3. Исходные данные к магистерской диссертации: документация разработчиков, интернет-ресурсы, учебная литература, отчёт по практической подготовке

4. Содержание магистерской диссертации (перечень подлежащих разработке вопросов): проектирование алгоритма решения задачи, работа с базой данных

5. Перечень материалов приложения (наличие чертежей, таблиц, графиков, схем, программных продуктов, иллюстративного материала и т.п.): нет

6. Рецензент магистерской диссертации: Фомин Д.В., старший преподаватель, канд. техн. наук

7. Дата выдачи задания 17.01.25

8. Руководитель выпускной квалификационной работы : _____

Т.А. Галаган, доцент, канд. техн. наук
(фамилия, имя, отчество, должность, уч.степень, уч.звание)

Заявление принял к исполнению _____

РЕФЕРАТ

Магистерская диссертация содержит 83 страницы, 39 рисунков, 44 источника, 6 таблиц

РАСПОЗНАВАНИЕ ЛИЦ, КОМПЬЮТЕРНОЕ ЗРЕНИЕ, NEURAL NETWORKS, PYTHON, OPENCV, БЕЗОПАСНОСТЬ

Целью данной магистерской диссертации является разработка системы сбора лиц с высокой точностью распознавания при минимальных затратах вычислительных ресурсов.

Проект направлен на создание программного решения, способного идентифицировать пользователей в режиме реального времени, что может быть применено в системах безопасности, контроля доступа и автоматизации.

Процесс реализации проекта включал следующие этапы:

- изучение предметной области и детализация объекта исследования;
- разработка аппаратной архитектуры на базе современных процессоров и сенсоров;
- интеграция алгоритмов РСА и сверточных нейронных сетей;
- тестирование и оптимизация системы для использования в реальных условиях.

Разработанная система обеспечивает надежную идентификацию лиц и может быть адаптирована под различные сценарии использования – от малых офисных систем до масштабных решений для предприятий. Результаты работы подтверждают эффективность применения нейросетевых алгоритмов в задачах биометрической идентификации, а также демонстрируют потенциал внедрения подобных решений в сфере информационной безопасности.

СОДЕРЖАНИЕ

Введение	7
1 Предметная область исследования	8
1.1 Справочная информация по теме	8
1.2 Анализ современного состояния проблемы исследования	9
1.2.1 Современные технологии разработки системы распознавания лиц	9
1.2.2 Состояние исследований в области распознавания лиц	11
1.3 Цель и задачи исследования	14
1.4 Описание методов и инструментов разработки программ	15
1.4.1 Анализ современных технологий распознавания лиц	17
1.5 Обзор существующих методов решения аналогичных задач	18
2 Алгоритмическое и программное обеспечение решения задачи распо- знавания лиц	24
2.1 Предлагаемый алгоритм компьютеризированного решения задачи	24
2.1.1 Встроенное устройство Raspberry Pi	25
2.1.2 Алгоритм распознавания лиц PCA	26
2.1.3 Сравнение интегральных схем	27
2.1.4 Схема функционального модуля	28
2.1.5 Алгоритм MTCNN для определения лиц	29
2.2 Анализ потребностей	31
2.2.1 Функциональные требования	32
2.2.2 Требования к характеристикам	34
2.3 Обзор возможностей профильного программного обеспечения	36
2.3.1 Библиотеки и платформы для обработки изображений и ма- шинного обучения	37
2.3.2 Технологии разработки	39
2.3.3 Типы данных и механизм сообщений	41

2.4 Характеристика выбранного программно-технического обеспечения	44
2.4.1 Аппаратная платформа: Raspberry Pi 3B+	44
2.4.2 NVIDIA Jetson TX2	45
2.4.3 Альтернативные платформы: Coral USB Accelerator	46
3 Программная реализация предлагаемого алгоритма решения задачи распознавания лиц	47
3.1 Основные этапы практической разработки программного продукта	47
3.1.1 Проектирование архитектуры системы	47
3.1.2 Проектирование и разработка базы данных	59
3.1.3 Реализация многопоточной обработки	63
3.2 Пример фактического тестирования программного продукта	68
3.2.1 Функции интерфейса	68
3.2.2 Структура базы данных и пользовательский интерфейс	78
Заключение	82
Библиографический список	83

ВВЕДЕНИЕ

В современную эпоху цифровых технологий автоматизация процессов анализа и обработки визуальной информации выходит на новый уровень. Одним из наиболее востребованных направлений в данной области является технология распознавания лиц, которая находит широкое применение в системах безопасности, контроля доступа, видеонаблюдения и интеллектуального анализа поведения.

Актуальность настоящего исследования обусловлена устойчивым ростом потребности в системах биометрической идентификации, способных работать в реальном времени, обеспечивая точное и надёжное распознавание лиц в различных условиях. Развитие алгоритмов машинного обучения и появление мощных библиотек, таких как TensorFlow, а также высокопроизводительных вычислительных архитектур, открыло возможности для построения адаптивных, масштабируемых и интеллектуальных систем компьютерного зрения.

В ходе реализации проекта особое внимание уделялось построению эффективной архитектуры программного обеспечения, интеграции компонентов многопоточной обработки данных, разработке удобного пользовательского интерфейса и обеспечению взаимодействия с базами данных. Для этого использовался гибридный подход, сочетающий язык программирования C++ с возможностями Python и библиотекой TensorFlow, что позволило достичь высокой производительности при реализации алгоритмов распознавания лиц.

Объектом исследования выступают технологии биометрической идентификации на основе изображений лиц.

Предметом исследования является процесс проектирования и реализации программной системы для распознавания лиц с использованием методов машинного обучения и инструментов кросс-языковой интеграции (C++/Python).

ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

БД	база данных
ИБ	информационная безопасность
ИИ	искусственный интеллект
НФ	нормальная форма
ОС	операционная система
ПК	персональный компьютер
ПО	программное обеспечение
СУБД	система управления базами данных
CNN	сверточная нейронная сеть
GAN	генеративно-сопоставительные сети
LDA	линейный дискриминантный анализ
MTCNN	многозадачная каскадная сверточная нейронная сеть
ORL	оптические возвратные потери
PCA	метод главных компонент
ROS	операционная система для программирования роботов

1 ПРЕДМЕТНАЯ ОБЛАСТЬ ИССЛЕДОВАНИЯ

1.1 Справочная информация по теме

Со времени третьей научно-технической революции наука и техника добились быстрого прогресса, и человечество вступило в информационный век. Благодаря огромным научным и технологическим достижениям в машинах, компьютерах, материалах и других отраслях производительность труда резко возросла. В то же время то, что следующая технологическая революция вызовет новые волны, вызывает серьезные опасения по поводу быстрого социального и экономического прогресса.

Одной из важнейших задач ИИ является обработка изображений, а в ее основе – распознавание лиц. Эта технология нашла применение в самых разных сферах: от обеспечения безопасности до персонализации пользовательского опыта. Например, системы видеонаблюдения используют распознавание лиц для идентификации подозрительных лиц в режиме реального времени, а мобильные устройства – для безопасной аутентификации пользователей.

Системы сбора данных лиц обеспечивают основу для эффективного функционирования алгоритмов распознавания. Такие системы позволяют не только идентифицировать человека, но и анализировать его выражения лица, что открывает новые перспективы для использования в маркетинге, медицине, образовании и других областях. Актуальность этих технологий возрастает с учетом растущего спроса на автоматизацию процессов и повышения уровня безопасности в общественных и частных секторах.

Мировые лидеры признают, что искусственный интеллект становится ключевым направлением научно-технологического прогресса. В различных странах разрабатываются национальные стратегии по внедрению и развитию ИИ, что открывает значительные возможности для научных исследований и практического применения. Важно как можно быстрее освоить основные технологии и направления в области ИИ, чтобы занять ведущие позиции в этой стратегически важной сфере. Это позволит не только продвигать научные и технологические

достижения, но и способствовать экономическому росту, улучшая качество жизни людей.

Публичные торги по искусственному интеллекту Google Alpha Go, публичный конкурс Google по искусственному интеллекту AlphaGo, онлайн-платежи, считывание лица и загрузка большего количества физического оборудования, передача посадки, беспилотный автомобиль Tesla и другие продукты, полностью продемонстрировали признание и направление исследований на рынке.

Когда дело доходит до полной реализации искусственного интеллекта, обработка изображений является первым шагом на пути внедрения цифровых данных в общую систему. В основе лежит признание и признание. Процесс распознавания лиц технически требует завершения всех процессов распознавания и распознавания. Поэтому эта технология играет ключевую роль в исследовании и реализации обработки изображений. В то же время исследования систем распознавания лиц очень важны. Ускорение идентификации и идентификации людей может быть применено к таким различным отраслям, как транспорт, безопасность и развлечения. Например, мониторинг безопасности играет важную роль в обеспечении личной и имущественной безопасности, а также социальной защищенности.

В большинстве случаев системы наблюдения должны выполнять задачу сопровождения целей. В процессе идентификации на транспорте контролеры билетов должны постоянно отслеживать идентификационную информацию субъекта. Используемые сегодня методы в основном основаны на ручном поиске или аутентификации, однако точность ручной работы зависит от множества факторов, которые очень нестабильны и требуют больших человеческих и материальных ресурсов.

1.2 Анализ современного состояния проблемы исследования

1.2.1 Современные технологии разработки системы распознавания лиц

Человеческое лицо покрывает множество органов человеческого тела, таких как глаза, нос, рот, брови. Формы, размеры и структуры этих органов различаются, что делает каждое лицо уникальным в мире. Следовательно, это может

быть основной характеристикой лицевого тестирования. В соответствии с эквивалентностью площади, угла и расстояния между чертами лица информация о чертах лица отличается от информации о чертах лица. В этой области достигнут некоторый прогресс, однако показатели сбора данных далеки от идеальных. Простой и быстрый метод распознавания лиц, основанный на геометрических особенностях, но выражение лица нестабильно и точность обнаружения невысока. На этой основе был достигнут некоторый прогресс, но сбор данных далёк от идеала. Простой и быстрый метод распознавания лиц, основанный на геометрических особенностях, но выражение лица будет меняться, а точность обнаружения будет не очень высокой.

Основанный на анализе главных компонент (РСА), также известном как преобразование KL, это один из алгоритмов обнаружения лиц, предложенных Терком и Пентландом в начале 1990-х годов. РСА является широко используемым методом анализа данных. Он использует методы статистической вероятности для извлечения атрибутов изображения, аппроксимации изображений и извлечения основных элементов изображения, которые представляют собой векторы признаков, называемые охватом лица. Использование краев объекта для уменьшения размера изображения и отображения изображения с минимальной информацией уменьшает объем вычислений в процессе сбора данных и ускоряет скорость сбора данных.

DeepFace – набор алгоритмов. Эти алгоритмы используют различные алгоритмы машинного обучения для решения проблем с изображением, звуком и текстом в многослойных искусственных нейронных сетях, получения функций на каждом уровне нейронной сети и решения важных задач в функциях ручного проектирования. Он основан на нейронных сетях и включает в себя ряд важных алгоритмов, таких как эволюционная нейронная сеть, многоуровневая сеть обратной связи, сеть глубокого доверия и автоматическое кодирование.

CNN – модель обучения с учителем, а DBN является моделью обучения без учителя. DBN предоставляет необязательный алгоритм обучения слоев, обеспечивающий основу для решения глубоких сетевых проблем. Углубленные

исследования позволили добиться прогресса во многих областях, таких как доступ к изображениям, языку и тексту, что привело к новым технологическим революциям.

Научные исследования в области распознавания лиц ведутся уже несколько-ко десятилетий. Важным шагом стало появление алгоритма Principal Component Analysis (PCA), предложенного Терком и Пентландом, который использует математические преобразования для уменьшения размерности данных и выделения основных характеристик лиц. Современные методы значительно продвинулись, включая использование сверточных нейронных сетей (CNN) для автоматического извлечения признаков и повышения точности распознавания.

Системы глубокого обучения, такие как модели DeepFace от Facebook и FaceNet от Google, используют миллионы изображений для обучения и демонстрируют выдающиеся результаты в задачах идентификации и верификации лиц. Кроме того, разработка мобильных платформ, таких как процессоры Qualcomm Snapdragon, интегрирующих модули ИИ, позволяет переносить вычисления на устройства, делая системы распознавания более быстрыми и автономными.

В последние годы также наблюдается рост интереса к использованию генеративных моделей для улучшения качества распознавания лиц. Например, генеративно-состязательные сети (GAN) могут создавать реалистичные изображения лиц, что помогает в обучении моделей и улучшении их устойчивости к различным видам атак и манипуляций.

Таким образом, современные технологии разработки систем распознавания лиц продолжают эволюционировать, сочетая математические методы, глубокое обучение и инновационные аппаратные решения для достижения высокой точности и надежности в различных приложениях.

1.2.2 Состояние исследований в области распознавания лиц

В последние годы наблюдается бурное развитие технологий распознавания лиц. Лаборатории, такие как SenseTime и Megvii, добились значительных успехов в разработке алгоритмов и оборудования. Применение методов машинного обучения, включая каскадный классификатор на основе AdaBoost и сверточные

нейронные сети, позволило улучшить точность и надежность распознавания даже в сложных условиях, таких как плохое освещение или частичная окклюзия лиц.

На базе таких технологий создаются не только приложения для обеспечения безопасности, но и платформы для анализа эмоций и поведения пользователей. Например, в образовательных учреждениях применяются системы мониторинга, анализирующие внимание и эмоции учащихся, что помогает педагогам корректировать подходы к обучению.

Метод, предложенный Сюй Чжэньхуэем, позволяет определять небольшие углы поворота лица в сложных условиях. Этот метод сочетает в себе оттенки серого и цвета лица, эффективно уменьшая экспонированную область. Наконец, чтобы создать каскадный классификатор для обнаружения этих маленьких кутикул, используется алгоритм. Исследователи предложили два набора инновационных идей, которые могли бы расширить прямоугольник, и предоставили вычислительные методы. Он обучался работе с базой данных изображений, включая лица и профили. Эксперименты показали, что этот метод имеет более высокие характеристики управления для людей с углами отклонения. Чтобы найти правильный метод распознавания лиц в видео, отсутствие распознавания лиц в сложных ситуациях находится на более высоком уровне. Например, чтобы уменьшить степень утечки, алгоритм Royal Marshal AdaBoost объединяется с алгоритмом GA. Машинное обучение более слабых классификаторов с помощью вспомогательных векторов.

При обучении мощного классификатора генетический алгоритм используется для оптимизации распределения его компонентов, в итоге создавая мощный классификатор, способный извлекать различные признаки. Исследователи пытаются изучить и найти алгоритм AdaBoost в сочетании с характеристиками интегрированных каналов. В системе реального времени, использующей программное обеспечение HD ROS.

Чтобы исправить эту ошибку был предложен улучшенный метод, который определяет границы отмеченных областей посредством морфологии до того, как

AdaBoost использует алгоритмы распознавания лиц. Этот метод можно использовать в программном обеспечении на платформе ROS. Ранее была объединена нейронная сеть BP для точного определения цвета кожи, а затем использовали сеть AdaBoost для определения цвета кожи. То, как он собирает людей, лучше всего проявляется на изображениях или видео с низким разрешением. Затем использовали ковариационные функции вместо прямоугольных функций, что значительно ускорило обучение алгоритма AdaBoost.

По сравнению с обычными компьютерными системами программное обеспечение платформы ROS имеет преимущества небольшого размера, высокой специфичности и высокой стоимости. Программная система платформы ROS также имеет свои недостатки, такие как медленная скорость. Скорость вычислений и системные ресурсы. Меньше, более сложные операционные среды и так далее. Существуют также программные комплексы платформы ROS, к которым предъявляются строгие требования по энергопотреблению, объему, стоимости и т. д.

Таким образом, предметом исследования этой темы является сбор лиц на платформе программной системы платформы ROS. Эта комплексная система сбора лиц подходит для устранения вышеупомянутых недостатков при сборе лиц в небольших масштабах. Некоторые смартфоны имеют 4 ядра ЦП, и каждое ядро ЦП может достигать частоты 1 ГГц, что показывает, что скорость разработки программной системы платформы ROS очень высока. Благодаря постоянному развитию и совершенствованию технологий также появятся системы сбора больших объемов данных. Будущие продукты программного обеспечения ROS для сбора лиц будут базироваться на платформах ПК.

В настоящее время быстрое развитие программной системы платформы ROS постепенно привело к появлению платформы ROS. Некоторые компании имеют свои собственные программные продукты для сбора лиц на платформе ROS, такие как отдел кадров Feirui, Hanwang и Zhongkong. Продаваемые на рынке программные продукты для сбора лиц на платформе ROS имеют высокую стоимость. Это процессоры цифровых сигналов, FPGA и прочие. Основная платформа программного комплекса платформы ROS.

Кроме того, активное развитие технологий глубокого обучения открывает новые горизонты для распознавания лиц. Использование трансформеров и других современных архитектур нейронных сетей позволяет значительно повысить точность распознавания и адаптивность систем к различным условиям эксплуатации. Важно отметить, что эти достижения также поднимают вопросы этики и конфиденциальности, требуя разработки новых стандартов и регуляций в области использования технологий распознавания лиц.

1.3 Цель и задачи исследования

Основной целью работы является разработка аппаратно-программной системы сбора лиц, которая обеспечивала бы высокую точность распознавания при минимальных затратах вычислительных ресурсов. Для достижения цели были сформулированы следующие задачи:

- анализ существующих технологий сбора и распознавания лиц;
- разработка аппаратной архитектуры на базе современных процессоров и сенсоров;
- интеграция программного обеспечения для обработки данных лиц, включая алгоритмы PCA и сверточные нейронные сети;
- тестирование и оптимизация системы для использования в реальных условиях.

Скоординированная разработка программного обеспечения для распознавания лиц осуществляется после завершения показателей эффективности системы на основе низкой стоимости и низкого использования ресурсов, а также анализа ресурсов программного обеспечения и распределения всей системы, каждая система соответствует уровню программного обеспечения. распределение высокоскоростных функций и баланс между легко заменяемыми функциями, быстрой заменой и сложностью.

В этой статье сначала кратко представлена история развития программной системы платформы ROS и обработки изображений, затем подробно представлены распространенные методы сбора лиц, а затем подробно представлена аппаратная структура платформы ROS NVIDIA TK1, а также операционной системы

Linux, и система программного обеспечения платформы ROS. Базовые среды, такие как файловая система программного обеспечения платформы ROS, были адаптированы, скомпилированы и перенесены. Алгоритмы распознавания лиц в основном включают в себя алгоритмы распознавания лиц и алгоритмы распознавания лиц. Поскольку обучение алгоритмов распознавания и идентификации лиц требует обширной обработки изображений, в видео использовалась основная часть анализа для идентификации лицевых целей и удаления частей. PCA используется для получения векторов признаков и сравнения их в базе данных. Наконец он был установлен на платформу ROS NVIDIA TK1. Для съемки изображений использовалась USB-камера, которая была серой и яркой. Среди них распознавание лиц осуществляется с помощью алгоритма PCA, а сбор лиц – с помощью алгоритма PCA, что делает платформу ROS более удобной и эффективной для практического применения системы сбора лиц.

Широко используется сбор выражений лица. В области электронных игр выражения пользователей могут собираться для игр и взаимодействия человека с компьютером; в образовании и преподавании эмоции учащихся могут собираться для мониторинга успеваемости учащихся в классе; в общественных местах пользователь Выражения людей могут быть обнаружены в режиме реального времени. Выражения людей могут обнаруживать ненормальные человеческие эмоции в режиме реального времени; в сфере покупок выражения клиентов могут быть обнаружены в режиме реального времени, и подходящие продукты могут быть своевременно рекомендованы; в области безопасной транспортировки водитель Выражение может быть собрано для определения состояния водителя и предотвращения несчастных случаев.

1.4 Описание методов и инструментов разработки программ

QT – кроссплатформенная среда разработки приложений с графическим пользовательским интерфейсом C++, разработанная компанией Qt. Его можно использовать для разработки как программ с графическим интерфейсом, так и программ без графического интерфейса, таких как консольные инструменты и серверы. Qt – объектно-ориентированная среда. Используя специальные

расширения для генерации кода и некоторые макросы, Qt легко расширяется и позволяет осуществлять истинное компонентное программирование.

MySQL – одна из самых популярных систем управления реляционными базами данных, разработанная шведской компанией MySQL AB.

SQL – наиболее распространенный стандартный язык доступа к базам данных.

Микропроцессор STM32 – первый 32-битный стандартный RISC-процессор на базе архитектуры ARMv7-M. Он не требует операционной системы и может быть запрограммирован на языке Keil C.

Соединение между базой данных и оборудованием:

- интерфейс JDBC, основанный на наборе инструкций SQL открытого X-модуля. Он состоит из набора классов и интерфейсов, написанных на Java, которые позволяют программистам создавать независимые от базы данных и платформы программные интерфейсы для создания приложений баз данных;

- инструмент SQL, реализующий обеспечение связи и связи с различными базами данных. Это язык по умолчанию для систем управления базами данных. В настоящее время большинство родственных систем баз данных поддерживают стандарт языка SQL.

Проектирование системы будет осуществляться с использованием языка программирования C++ и среды разработки QT. Для хранения данных лиц применяется реляционная база данных MySQL, которая обеспечивает быстрое и надежное управление информацией. Аппаратная часть включает в себя микроконтроллер STM32, который используется для управления сенсорами и передачи данных.

Связь между компонентами системы реализуется с помощью интерфейсов SQL и JDBC. Для оптимизации вычислений применяются алгоритмы машинного обучения, такие как PCA и CNN, которые будут протестированы на реальных наборах данных.

1.4.1 Анализ современных технологий распознавания лиц

Современные технологии распознавания лиц быстро развиваются

благодаря достижениям в области компьютерного зрения и машинного обучения.

Классификация нейронных сетей представлена на рисунке 1.



Рисунок 1 – Классификация нейронных сетей

Одним из ключевых направлений является использование глубоких нейронных сетей, таких как свёрточные нейронные сети (CNN), которые обеспечивают высокую точность при распознавании лиц даже в сложных условиях. Например, алгоритмы, основанные на архитектурах ResNet или Inception, широко применяются для извлечения глубоких признаков лиц. Кроме того, используются подходы с привлечением генеративно-сопоставительных сетей (GAN) для улучшения качества данных, что особенно актуально при работе с ограниченными наборами изображений.

В области аппаратных технологий активно используются высокопроизводительные процессоры, такие как NVIDIA Jetson, которые позволяют обрабатывать данные в реальном времени. Камеры высокого разрешения с инфракрасной подсветкой и сенсоры глубины, такие как Intel RealSense, обеспечивают дополнительные параметры для анализа лиц, включая определение глубины сцены и различных освещений. Для реализации системы встраиваемого типа популярны

платформы на основе микроконтроллеров STM32 или Raspberry Pi, которые обладают достаточной вычислительной мощностью для базовой обработки данных и взаимодействия с периферийными устройствами.

В таблице 1 представлен анализ всех популярных аппаратных технологий.

Таблица 1 – Аппаратные технологии

Технология/Устройство	Особенности	Преимущества	Недостатки
1	2	3	4
NVIDIA Jetson	высокопроизводительный процессор для обработки данных в реальном времени	- поддержка алгоритмов глубокого обучения - высокая вычислительная мощность - энергоэффективность	- высокая стоимость - требует продвинутых знаний для настройки
Intel RealSense (камера)	камеры высокого разрешения с поддержкой инфракрасной подсветки и сенсоров глубины	- возможность анализа глубины сцены - высокое качество съёмки в сложных условиях освещения	- ограниченная область применения - высокая цена
STM32 (микроконтроллер)	32-битный микроконтроллер с архитектурой ARM Cortex-M	- низкая стоимость - энергоэффективность - подходит для встраиваемых систем	- ограниченные вычислительные ресурсы - необходимость доп. оборудования
Raspberry Pi	многофункциональная мини-компьютерная платформа для общего применения	- низкая стоимость - поддержка множества периферийных устройств - широкое сообщество пользователей	- средняя производительность - подходит только для базовых задач

Традиционные методы, такие как анализ главных компонент (PCA) или дискриминантный анализ Лапласа (LDA), продолжают использоваться для снижения размерности данных и начальной обработки изображений. Однако они постепенно вытесняются более современными методами глубокого обучения. При этом выбор технологии зависит от требований к системе, таких как точность, скорость обработки, энергопотребление и стоимость.

Таким образом, современные технологии распознавания лиц демонстрируют стремительное развитие благодаря достижениям в области компьютерного

зрения и машинного обучения. Использование глубоких нейронных сетей, в частности сверточных нейронных сетей (CNN), обеспечивает высокую точность распознавания даже в сложных условиях, а архитектуры, такие как ResNet и Inception, активно применяются для извлечения глубоких признаков лиц. Генеративно-состязательные сети (GAN) становятся важным инструментом для улучшения качества данных, что особенно полезно при работе с ограниченными наборами изображений.

Хотя традиционные методы, такие как анализ главных компонент (PCA) и дискриминантный анализ Лапласа (LDA), всё ещё используются для начальной обработки изображений и снижения размерности данных, они постепенно уступают место более современным методам глубокого обучения. В итоге выбор технологии распознавания лиц зависит от конкретных требований системы, таких как точность, скорость обработки, энергопотребление и стоимость. Это разнообразие подходов и технологий открывает новые горизонты для применения систем распознавания лиц в различных областях, от безопасности до пользовательских интерфейсов.

1.5 Обзор существующих методов решения аналогичных задач

Геометрические методы основаны на выделении ключевых точек лица (глаза, нос, рот) и их взаиморасположении. Система анализирует геометрические параметры, такие как углы между чертами лица, пропорции и расстояния. Применяется в ранних версиях систем распознавания лиц.

Преимущества:

- простота реализации;
- низкие вычислительные требования.

Недостатки:

- низкая точность при изменении выражения лица, углов обзора или освещения.

Алгоритмы анализа главных компонент (PCA) – метод Principal Component Analysis (PCA) преобразует изображения в набор ортогональных компонент, называемых «лицевыми пространствами» (Eigenfaces). Используется в

системах распознавания лиц для сравнения лиц с эталонными данными.

Преимущества:

- уменьшение размерности данных;
- высокая скорость работы с малым количеством данных.

Недостатки:

- зависимость от освещения и выражения лица;
- ограниченная точность при работе с большими объемами данных.

Суть метода PCA. Вначале весь обучающий набор лиц преобразуется в одну общую матрицу данных, где каждая строка представляет собой один экземпляр изображения лица, разложенного в строку. Все лица обучающего набора должны быть приведены к одному размеру и с нормированными гистограммами.

Суть метода PCA представлена на рисунке 2.

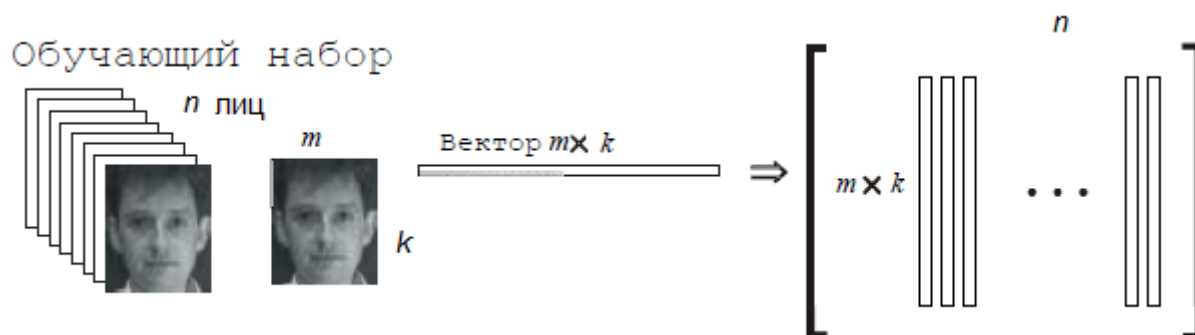


Рисунок 2 – Суть метода PCA

Затем производится нормировка данных и приведение строк к 0-му среднему и 1-й дисперсии, вычисляется матрица ковариации. Для полученной матрицы ковариации решается задача определения собственных значений и соответствующих им собственных векторов (собственные лица).

Пример первых десяти собственных векторов (собственных лиц), полученных на обучаемом наборе лиц представлен на рисунке 3.

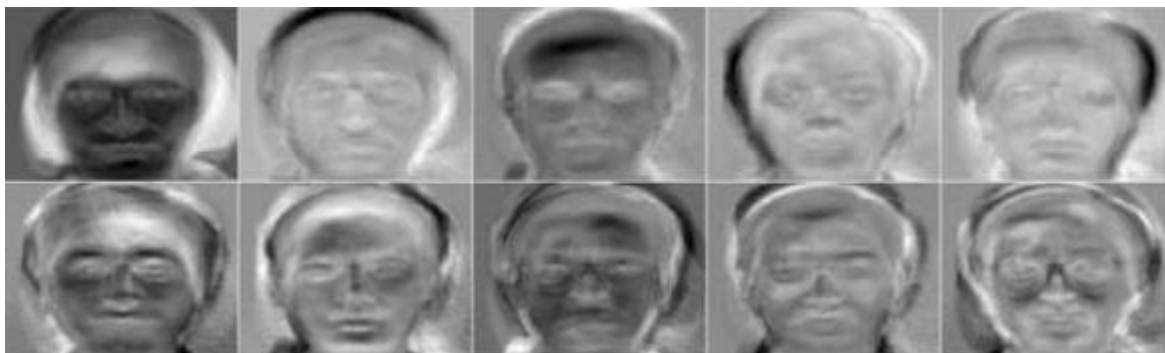


Рисунок 3 – Пример первых десяти собственных векторов, полученных на обучаемом наборе лиц.

Локальные бинарные шаблоны (LBP) – этот метод анализирует текстуры лица, представляя их в виде набора локальных шаблонов. Применяется в распознавании лиц на изображениях с фиксированным углом и расстоянием.

Преимущества:

- устойчивость к изменениям освещения;
- простота вычислений.

Недостатки:

- зависимость от качества изображения;
- плохая производительность в динамических условиях (например, движение).

Глубокие нейронные сети (DNN) – современные методы, такие как Convolutional Neural Networks (CNN), применяются для анализа лицевых изображений. CNN автоматически извлекают признаки лица, обучаясь на больших наборах данных. Применяется при реализации сложных систем распознавания лиц в реальном времени.

Преимущества:

- высокая точность;
- устойчивость к изменениям углов обзора, выражений лица и освещения.

Недостатки:

- высокие вычислительные требования;
- необходимость большого объема данных для обучения.

Отличительными особенностями CNN являются локальные рецепторные поля (обеспечивают локальную двумерную связность нейронов), общие веса (обеспечивают детектирование некоторых черт в любом месте изображения) и иерархическая организация с пространственным сэмпингом (spatial subsampling). Благодаря этим нововведениям CNN обеспечивает частичную устойчивость к изменениям масштаба, смещениям, поворотам, смене ракурса и прочим искажениям.

Схематичное изображение CNN представлено на рисунке 4.

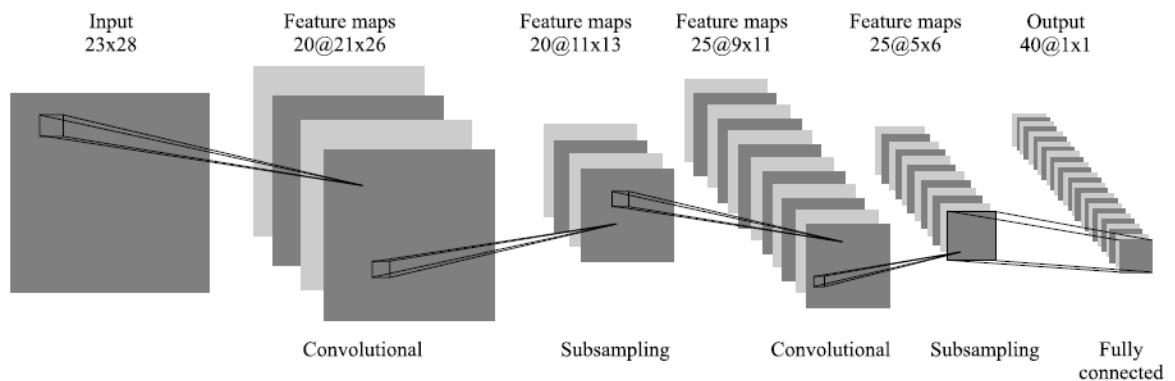


Рисунок 4 – Схематичное изображение CNN

Тестирование CNN на базе данных ORL, содержащей изображения лиц с небольшими изменениями освещения, масштаба, пространственных поворотов, положения и различными эмоциями, показало 96% точность распознавания.

Гибридные методы – сочетание традиционных алгоритмов (например, PCA) с современными подходами, такими как CNN или машинное обучение. Применяется для решения задач в условиях ограниченных ресурсов.

Преимущества:

- сбалансированность между точностью и производительностью;
- гибкость в настройке для различных условий.

Недостатки:

- более высокая сложность реализации.

Методы на основе сенсоров глубины – используются камеры с поддержкой глубины, такие как Intel RealSense, для анализа трехмерной структуры лица. Необходимы для биометрической идентификации и работы в условиях сложного освещения.

Преимущества:

- высокая точность;
- возможность работы в реальном времени.

Недостатки:

- высокая стоимость оборудования;
- ограниченная доступность в некоторых приложениях.

2 АЛГОРИТМИЧЕСКОЕ И ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ РЕШЕНИЯ ЗАДАЧИ РАСПОЗНОВАНИЯ ЛИЦ

2.1 Предлагаемый алгоритм компьютеризированного решения задачи

Благодаря быстрому развитию технологии программного обеспечения платформы ROS, скорость и точность программного обеспечения платформы ROS были значительно улучшены, что позволяет создавать портативные и удобные системы распознавания лиц. В сочетании с программным обеспечением платформы ROS и технологией распознавания лиц была создана система распознавания лиц на основе программного обеспечения ROS.

В этой теме основное внимание уделяется выявлению и определению целей и внедрению платформы ROS.

Анализ целостности системы делится на этапы:

- разработка платформы ROS, операционной системы Ubuntu;
- создание платформы TensorFlow на основе операционной системы Windows;
- внедрение и тестирование системы.

Разработка платформы ROS, операционной системы Ubuntu с загрузчиком и портирование корневой файловой системы для программного обеспечения NVIDIA TK1. Компиляция и установка библиотеки компьютерного зрения opencv, портирование USB-камеры и сопоставление драйверов образуют идеальную платформу для разработки системы.

Создание платформы TensorFlow на основе операционной системы Windows, используется алгоритм MTCNN для идентификации лиц в видео, раннего распознавания, сравнения и воспроизведения лиц.

Система внедрена на практике, протестирована для достижения целей тестирования, предложен метод предотвращения мошенничества с фотографиями со вспышкой и протестирована в различных условиях на популяции, соответствующей проектным требованиям. Разрабатываемая система распознавания лиц должна быть способна удовлетворить потребности быстрого распознавания.

В конструкции используется панель разработки NVIDIA TK1, процессор NVIDIA-A72 расположен в центре управления, USB-камера является модулем распознавания лиц, а данные о лицах хранятся в базе данных SDCA. Сначала база данных компьютерного зрения OpenCV вызывает веб-камеру для чтения видео. MTCNN проверил лица на видео в режиме реального времени и ввел их в алгоритм FaceNet, который сопоставил их с лицами в базе данных.

Типичная схема автономного вождения включает в себя два чипа Tesla FSD для реализации режима резервирования процессора двойной нейронной сети. Два процессора независимы друг от друга. Даже если у одного возникнет проблема, другой все равно может работать как обычно.

Каждый процессор может обрабатывать 1 ТБ данных в секунду при номинальной производительности 36TOPS, что в сумме составляет 72TOPS. Потребляемая мощность составляет около 250 Вт на милью, что составляет лишь одну седьмую от решения NVIDIA, а стоимость также составляет лишь одну седьмую. В качестве основного процессора используется 12-ядерный процессор ARM Cortex-A72 с частотой 2,2 ГГц, а процессор нейронной сети имеет 32 МБ SRAM и два ядра. Каждое ядро имеет производительность 36 TOPS на частоте 2 ГГц, поэтому двухъядерный процессор имеет общую производительность 72TOPS.

2.1.1 Встроенное устройство Raspberry Pi

Встроенное устройство должно оценивать контуры лица и положение изображения, собираемого камерой, особенно в автономном режиме. Они могут использовать черты лица и биологические особенности, поэтому им нужен мощный компьютерный процессор. Чтобы улучшить функции отладки и расширения, встроенное устройство также должно иметь достаточные интерфейсы управления, а также иметь небольшую емкость и высокую стоимость. Согласно приведенному выше анализу, разрешенные окончательные результаты включают Raspberry Pi 3B и NVIDIA TX2.

Nvidia TX2 имеет лучшие характеристики, чем Raspberry Pi 3B, но стоит дороже, чем Raspberry Pi 3B, из-за более интегрированной системы глубокого обучения и машинного обучения. Учитывая производительность и стоимость, в

качестве аппаратного терминала в итоге был выбран Raspberry Pi 3B.

2.1.2 Алгоритм распознавания лиц PCA

PCA – классический статистический метод, который не может описать тонкие различия в метках. Его основной принцип заключается в том, что данные высокой размерности можно проецировать в другое пространство через матрицу собственных векторов, чтобы выявить тестовые данные низкой размерности. Алгоритм сохраняет как можно большую часть исходного кода, если потеряно лишь небольшое количество исходной информации.

Распознавание людей на основе алгоритма PCA начинается с присвоения изображений людей в базе данных пространству лиц и расчета набора векторов после измерений Джона. Идентифицируемые изображения лиц также проецируются в пространство лица для расчета евклидова расстояния между изображением лица и базой данных лиц. Сравните расстояние и порог, чтобы завершить процесс распознавания лиц.

Алгоритм анализа главных компонент (PCA) является классическим методом снижения размерности данных, применяемым для выделения ключевых признаков изображений лиц.

Основные этапы работы алгоритма:

- формирование базы данных изображений, лицевые изображения преобразуются в градации серого и нормализуются по размеру;
- вычисление среднего изображения, усредненный вектор лиц из базы данных;
- построение ковариационной матрицы, отклонение каждого изображения от среднего;
- собственные векторы и значения, главные компоненты, которые представляют наиболее значимые вариации в данных;
- проекция новых изображений, тестовое изображение проецируется в пространство главных компонент, после чего вычисляется евклидово расстояние до ближайшего вектора в базе данных.

На рисунке 5 представлена последовательность шагов в алгоритме PCA.

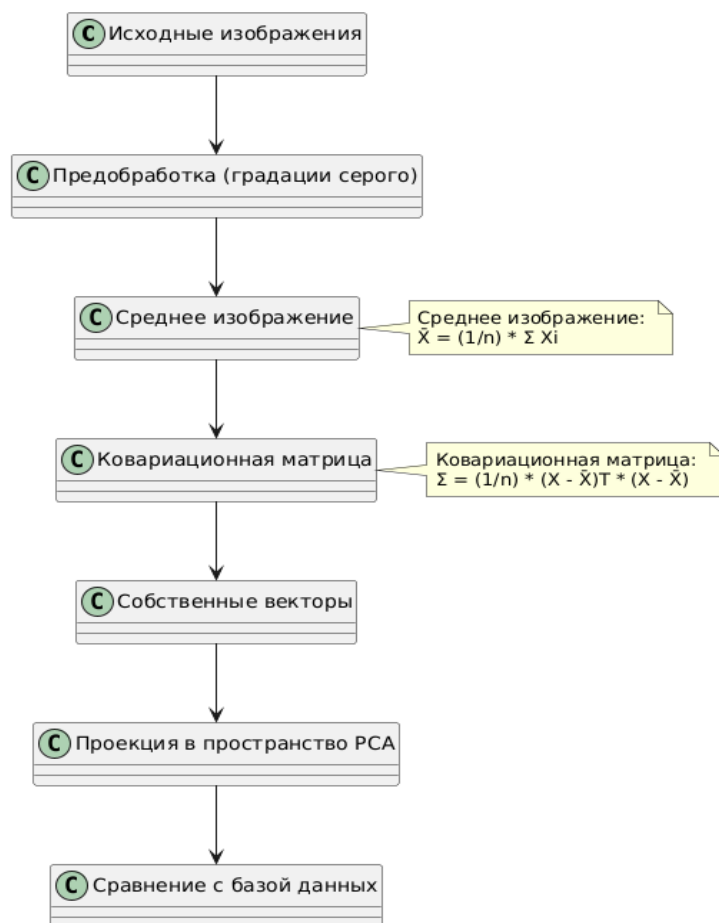


Рисунок 5 – Последовательность шагов в алгоритме PCA

Преимущества PCA:

- низкие вычислительные затраты;
- эффективность при ограниченном наборе данных.

Ограничения:

- чувствительность к изменениям освещения и угла поворота лица.

2.1.3 Сравнение интегральных схем

Долгое время моделирование и цифровой дизайн считались важным сегментированным рынком электронной промышленности. Компании-производители электроники разрабатывают аппаратное обеспечение для своих продуктов на основе своих стандартов и протоколов, что затрудняет конечным пользователям перенастройку устройств в соответствии со своими потребностями. Эта потребность в аппаратном обеспечении привела к разработке множества новых интегральных схем, адаптированных к местным условиям. В последние годы сектор

интегральных схем и связанного с ним программного обеспечения быстро развивался, что позволило ASIC, FPGA или SOC разрабатывать высокопроизводительные микросхемы.

По сравнению с процессорами, FPGA обладают уникальной способностью обрабатывать и передавать большие объемы данных посредством параллельных данных. С другой стороны, FPGA и ASIC быстрее, чем графические и центральные процессоры, поскольку они представляют собой вычислительные машины. Задержка очень важна для приложений, которым требуются функции реального времени, такие как обнаружение и проверка трафика видеообъектов. По сравнению с SOC, FPGA очень гибка и может реконструироваться с помощью различных структур. Он может даже перенастроить часть чипа, в то время как другие части чипа все еще работают.

2.1.4 Схема функционального модуля

Путем анализа конструкции каждого модуля распознавания лиц получена схема проектирования функции распознавания лиц на основе аппаратной структуры ROS-платформы NVIDIA TK1, показанная на рисунке 6.

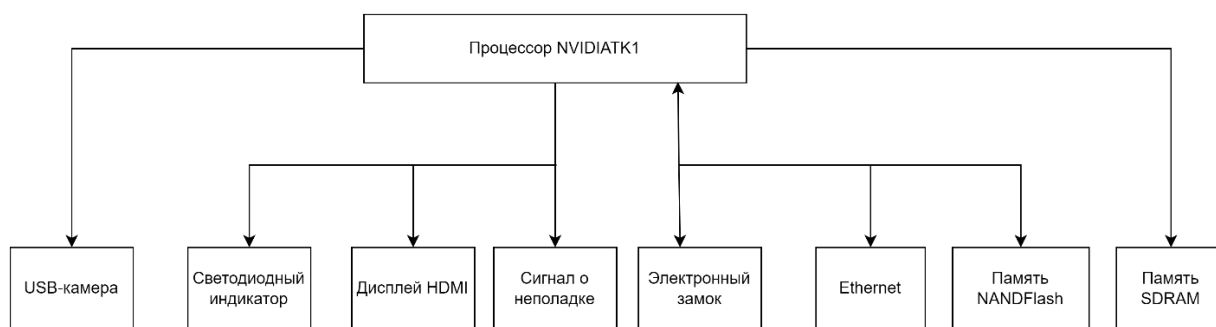


Рисунок 6 – Структурная схема модуля аппаратного обеспечения распознавания лиц

На рисунке изображена структура аппаратного обеспечения распознавания лиц. Структура включает следующие основные компоненты:

- процессор, центральный элемент системы, отвечающий за обработку данных и выполнение алгоритмов распознавания лиц;
- устройство USB-камеры для захвата изображений лиц, которые будут анализироваться системой;
- светодиодный индикатор, используемый для визуального отображения

статуса системы, например готовности к работе или наличия неполадок;

- дисплей (НОМ) устройство для вывода информации, такой как результаты распознавания или системные уведомления;

- сигнал о неполадке, механизм оповещения о технических проблемах в системе;

- электронный замок, устройство, управляемое системой для контроля доступа на основе результатов распознавания лиц;

- интерфейс Ethernet для подключения системы к сети, что позволяет удаленное управление и интеграцию с другими системами;

- память, компоненты для хранения данных, включая изображения и базы данных лиц, а также для временного хранения данных во время обработки.

Эти компоненты взаимодействуют друг с другом, обеспечивая захват, обработку и анализ изображений для распознавания лиц. Процессор управляет всеми операциями, получая данные с камеры, обрабатывая их и выводя результаты на дисплей или управляя электронным замком.

2.1.5 Алгоритм MTCNN для определения лиц

MTCNN (Multi-task Cascaded Convolutional Networks) – современный алгоритм, основанный на каскадных сверточных сетях, который широко используется для детектирования лиц и определения ключевых точек на изображениях. Алгоритм состоит из трех основных этапов, каждый из которых выполняет определенную задачу: обнаружение кандидатов в лица, уточнение координат и определение ключевых точек.

Этапы работы:

- быстрое сканирование изображения для обнаружения кандидатов в лица P-Net (Proposal Network). P-Net состоит из сверточных слоев (Conv, ReLU, MaxPool), которые позволяют эффективно обрабатывать изображение и выделять области, потенциально содержащие лица. P-Net работает на низком разрешении, что позволяет быстро обрабатывать большие изображения и находить кандидатов в лица;

- уточнение координат лиц и отсев ложных срабатываний R-Net

(Refinement Network). R-Net использует более глубокие слои с фильтрами, что позволяет более точно определить границы лиц и отфильтровать ложные срабатывания, обнаруженные P-Net. R-Net работает с меньшим количеством кандидатов, что повышает точность детектирования;

- точное определение ключевых точек (глаза, нос, рот) и границ лица O-Net (Output Network). O-Net включает выходные слои, которые определяют ключевые точки лица (обычно 5 точек) и окончательно уточняют границы лица. O-Net обеспечивает высокую точность даже при частичных окклюзиях (например, если лицо частично закрыто).

Преимущества MTCNN:

- высокая точность, алгоритм обеспечивает высокую точность детектирования лиц даже в сложных условиях, таких как частичные окклюзии, изменения освещения и различные углы поворота лица;

- реальное время, благодаря каскадной архитектуре, MTCNN может обрабатывать видео в реальном времени, что делает его пригодным для использования в системах видеонаблюдения и других приложениях, требующих быстрой обработки;

- поддержка устройств с ограниченными ресурсами, алгоритм оптимизирован для работы на устройствах с ограниченными вычислительными ресурсами, таких как Raspberry Pi, что делает его универсальным решением для различных платформ.

MTCNN интегрирован с библиотекой OpenCV, что позволяет легко использовать его в проектах, связанных с обработкой изображений и видео. OpenCV предоставляет удобные интерфейсы для работы с MTCNN, что упрощает процесс детектирования лиц и определения ключевых точек.

Пример использования:

- детектирование лиц на изображении, алгоритм MTCNN может быть использован для обнаружения лиц на статических изображениях. Например, в системах безопасности для идентификации людей на фотографиях;

- обработка видео в реальном времени, благодаря поддержке OpenCV,

MTCNN может быть использован для обработки видео в реальном времени, что полезно в системах видеонаблюдения или приложениях для анализа эмоций;

- определение ключевых точек, MTCNN может быть использован для определения ключевых точек лица, что полезно в приложениях для анализа мимики, распознавания эмоций или создания 3D-моделей лиц.

На рисунке 7 показаны этапы работы алгоритма MTCNN. Каждая сеть выполняет свою задачу, что позволяет алгоритму эффективно детектировать лица и определять ключевые точки.



Рисунок 7 – Этапы работы алгоритма MTCNN.

Алгоритм MTCNN является мощным инструментом для детектирования лиц и определения ключевых точек. Его каскадная архитектура, высокая точность и поддержка работы в реальном времени делают его одним из наиболее популярных решений в области компьютерного зрения. Интеграция с OpenCV и поддержка устройств с ограниченными ресурсами расширяют возможности его применения в различных проектах.

2.2 Анализ потребностей

В данном разделе рассматриваются функциональные требования и требования к характеристикам системы распознавания лиц. Эти требования формируются на основе анализа задач, которые должна решать система, а также ожиданий

пользователей и технических ограничений.

2.2.1 Функциональные требования

Функциональные требования описывают, что система должна делать, то есть её основные функции и возможности. Ниже приведены ключевые функциональные требования к системе распознавания лиц:

- уметь обнаруживать лица на изображениях и видео в реальном времени;
- поддерживать работу с изображениями различного качества и разрешения;
- идентифицировать лица, сравнивая их с базой данных известных лиц;
- обрабатывать лица под различными углами и при частичных окклюзиях;
- определять ключевые точки лица (например, глаза, нос, рот) для повышения точности распознавания;
- поддерживать хранение и обновление базы данных лиц;
- экспортировать и импортировать данные для интеграции с другими системами;
- обрабатывать видео в реальном времени с минимальной задержкой;
- поддерживать работу на устройствах с ограниченными ресурсами (например, Raspberry Pi);
- предоставлять уведомления обнаружения или распознавания лиц;
- генерировать отчеты о результатах работы системы;
- поддерживать интеграцию с другими приложениями и сервисами через API;
- поддерживать протоколы передачи данных HTTP, WebSocket и другие.

Для того чтобы показать, как пользователи взаимодействуют с системой для выполнения различных задач, связанных с распознаванием лиц необходимо сделать диаграмму вариантов использования.

На рисунке 8 представлена диаграмма вариантов использования системы.

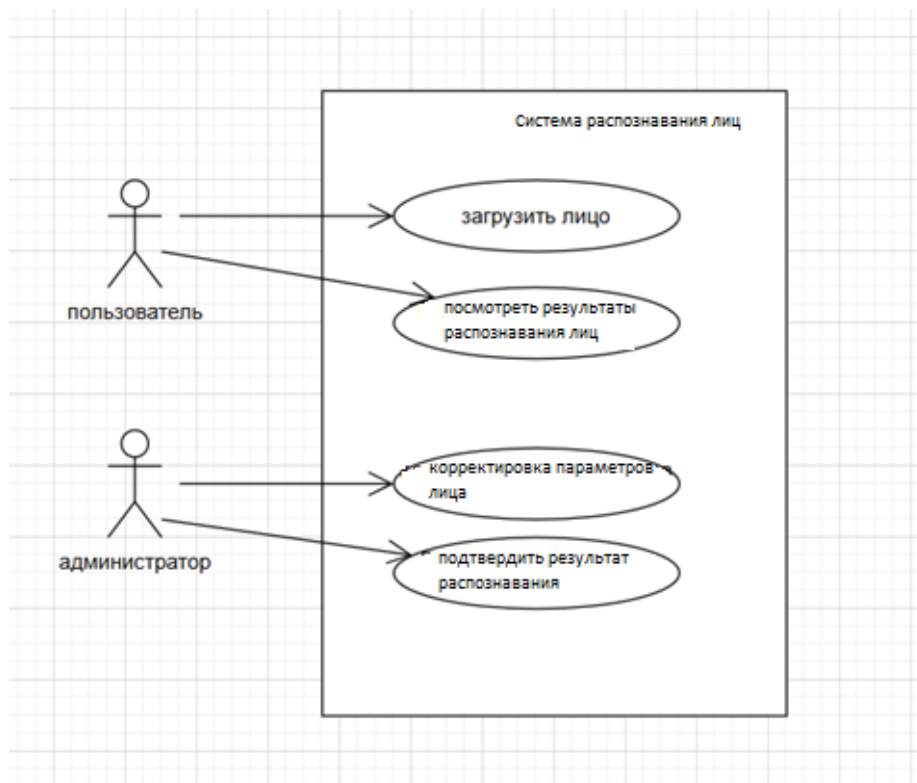


Рисунок 8 – Диаграмма вариантов использования системы распознавания лиц

Участники:

- пользователь, обычный пользователь системы, который взаимодействует с ней для выполнения базовых задач, таких как загрузка изображений и просмотр результатов распознавания;

- администратор, руководитель, который управляет системой, настраивает параметры и подтверждает результаты распознавания. Администратор имеет расширенные права доступа по сравнению с обычным пользователем.

Варианты использования:

- загрузить лицо, пользователь или администратор загружает изображение лица в систему для дальнейшего анализа и распознавания;

- посмотреть результаты распознавания лиц, после обработки изображения

система предоставляет результаты распознавания, которые могут быть просмотрены пользователем или администратором;

- корректировка параметров лица, администратор может корректировать параметры лица (например, уточнить ключевые точки или исправить ошибки распознавания);

- подтвердить результат распознавания, администратор подтверждает результаты распознавания, что может быть необходимо для фиксации данных в системе или для дальнейшего использования.

2.2.2 Требования к характеристикам

Требования к характеристикам описывают, как система должна выполнять свои функции, то есть её производительность, надежность, безопасность и другие аспекты.

Ниже приведены ключевые требования к характеристикам системы:

- обеспечение высокой точности распознавания лиц (не менее 95 % при стандартных условиях);
- поддержка работы в сложных условиях (например, при плохом освещении или частичных окклюзиях);
- обработка не менее 30 кадров в секунду (FPS) для работы в реальном времени;
- минимальная задержка при обработке изображений и видео;
- поддержка обработки большого количества лиц (тысячи и более) без потери производительности;
- возможность масштабирования для работы в распределенных системах;
- устойчивость к сбоям и ошибкам;
- поддержка механизмов восстановления после сбоев;
- обеспечение защиты данных, включая шифрование информации о лицах;
- поддержка аутентификации и авторизации для доступа к системе;
- интуитивно понятный пользовательский интерфейс;
- поддержка многоязычного интерфейса;
- поддержка работы с различными операционными системами (Windows,

Linux, macOS);

- поддержка различных форматов изображений и видео (JPEG, PNG, MP4 и др.);

- оптимизация для работы на устройствах с ограниченными ресурсами, таких как мобильные устройства или встраиваемые системы.

Для того, чтобы показать процесс работы системы распознавания лиц необходимо создать диаграмму последовательности.

Диаграмма последовательности работы системы распознавания лиц представлена на рисунке 9.

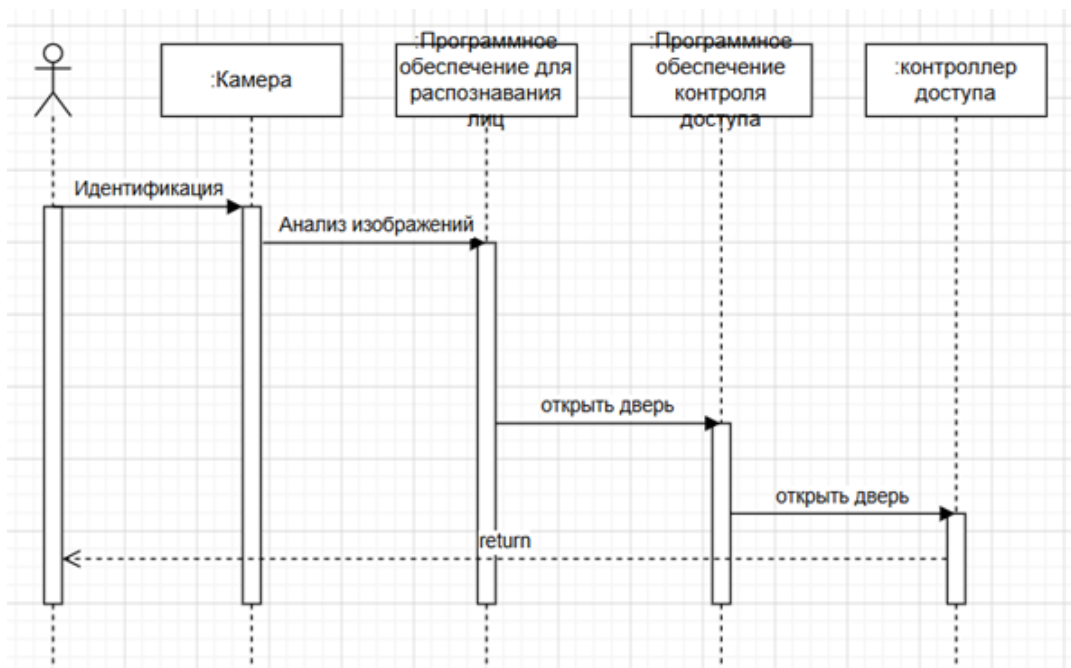


Рисунок 9 – Диаграмма последовательности системы распознавания лиц

Работа системы начинается с захвата изображения или видео с помощью камеры. Камера фиксирует изображение, на котором может быть лицо, и передает его в программное обеспечение для распознавания лиц. Это программное обеспечение анализирует изображение, выполняя несколько ключевых шагов: обнаружение лица, его идентификацию и сравнение с базой данных известных лиц.

После завершения анализа результат распознавания передается в программное обеспечение контроля доступа. Этот модуль проверяет, имеет ли распознанное лицо право доступа, основываясь на данных из базы данных или настроек системы. Если доступ разрешен, программное обеспечение контроля доступа отправляет команду контроллеру доступа, который физически управляет доступом, например, открывает дверь.

Контроллер доступа выполняет команду, открывая дверь, и возвращает результат выполнения операции в систему. Этот результат может быть использован для уведомления пользователя или администратора о успешном открытии двери или об отказе в доступе.

Таким образом, система проходит через несколько этапов: от захвата изображения и анализа до принятия решения о предоставлении доступа и выполнения соответствующего действия. Этот процесс демонстрирует, как различные компоненты системы взаимодействуют друг с другом для выполнения задачи распознавания лиц и управления доступом.

2.3 Обзор возможностей профильного программного обеспечения

Определение технологии распознавания лиц собирает многочисленные изображения лиц в базу данных для идентификации лиц в виде видеопотоков или информации об изображениях по сравнению с базами данных людей и информации, использующей веб-камеры или веб-камеры. Аутентичность распознавания лиц собирает данные о том, кто был подвергнут визуализирующему тесту. Полный процесс обнаружения включает в себя получение баз данных и функций, эффективный сбор информации об изображениях, предварительную обработку изображений, извлечение личных данных, идентификацию и проверку соответствия.

Конкретные функциональные требования каждого этапа заключаются в следующем:

- сбор и поиск данных в базе данных, предоставление выборки аналогичных характеристик для последующего процесса аудита и создание информационной базы данных для выявления альтернативных аудиторов.

Данные необходимо собирать в разных ситуациях с учетом местоположения, смайликов, позы, цвета кожи, очков или волос, выбора камеры видеонаблюдения, освещения и разрешения.

Канал, используемый для получения эффективной информации об изображении, может состоять из фото- или видеоданных в реальном времени, тем самым преобразуя информацию об объекте в систему распознавания.

Предварительная обработка изображения оптимизирует обработку входного изображения с учетом уровней серого, медиа фильтрации и существенного уменьшения перед извлечением собственных векторов и поиском. Целью данного этапа является удаление изображения, не имеющего отношения к людям, выделение полезной информации и повышение эффективности последующей обработки.

Особые обстоятельства, при которых физическое лицо может рассматриваться как опознаваемый объект. Алгоритмы распознавания лиц в основном изучают распознавание лиц. Это похоже на распознавание изображений, на сравнение изображений, хранящихся в блоках и базах данных. Любая база данных, изменяющая черты лица, аннулирует процесс координации. На уровне обнаружения существует множество алгоритмов на выбор. Если на изображении присутствует результат распознавания лица, то для последующего распознавания будет использоваться его калибровка.

Извлечение и идентификация персональных данных тесно связаны с точностью личности. Система должна извлекать значимые данные из изображений лиц, определять наиболее важные местоположения данных и игнорировать весь шум. Поэтому необходимо точно получить информацию о контуре лица, а затем сопоставить все данные. Этот процесс называется идентификацией. Основные принципы процесса распознавания называются методами классификации объектов.

2.3.1 Библиотеки и платформы для обработки изображений и машинного обучения

В данном разделе рассматриваются ключевые библиотеки и платформы,

которые широко используются в разработке систем распознавания лиц. Эти инструменты предоставляют функциональность для обработки изображений, обучения моделей машинного обучения и интеграции с другими компонентами системы.

OpenCV (Open Source Computer Vision Library) – это мощная библиотека с открытым исходным кодом, предназначенная для обработки изображений и компьютерного зрения.

Основные возможности:

- обнаружение и распознавание лиц;
- обработка и фильтрация изображений;
- работа с видео и камерами в реальном времени;
- интеграция с другими библиотеками, такими как TensorFlow и PyTorch.

OpenCV часто используется на начальных этапах обработки изображений, таких как обнаружение лиц и предварительная обработка данных.

TensorFlow – это фреймворк для машинного обучения с открытым исходным кодом, разработанный Google.

Основные возможности:

- создание и обучение нейронных сетей, включая сверточные нейронные сети (CNN) для задач распознавания лиц;
- поддержка распределенных вычислений и работы с большими объемами данных;
- интеграция с библиотеками обработки изображений, такими как OpenCV.

TensorFlow используется для разработки и обучения моделей, которые могут распознавать лица с высокой точностью.

Dlib – это библиотека на C++, которая также имеет интерфейсы для Python. Она предоставляет инструменты для машинного обучения и обработки изображений.

Основные возможности:

- обнаружение лиц и распознавание черт лица;
- реализация алгоритмов машинного обучения, таких как метод опорных

векторов (SVM);

- высокая производительность и точность.

Dlib часто используется для задач, связанных с анализом и сравнением лиц.

FaceNet – библиотека, разработанная Google, которая использует глубокие нейронные сети для создания векторных представлений лиц (эмбеддингов).

Основные возможности:

- преобразование изображений лиц в компактные векторные представления;
- сравнение лиц на основе расстояния между векторами;
- высокая точность распознавания даже при изменении освещения, угла поворота лица и других факторов.

FaceNet часто используется в системах, где требуется высокая точность и надежность.

PyTorch – фреймворк для машинного обучения с открытым исходным кодом, разработанный Facebook.

Основные возможности:

- гибкость в создании и обучении нейронных сетей;
- поддержка динамических вычислений, что упрощает отладку и разработку;
- широкие возможности для работы с изображениями и видео.

2.3.2 Технологии разработки

Microsoft Foundation Classes (MFC) – библиотека классов на C++, разработанная Microsoft, которая инкапсулирует наиболее часто используемые функции Windows API. MFC предоставляет тысячи классов и интерфейсов, что значительно упрощает разработку приложений для Windows.

В разрабатываемом проекте интегрированная среда разработки Visual Studio 2013 Premium, которая автоматически позволяет генерировать основную структуру проекта MFC. Разработчикам остается только добавить необходимый контент и адаптировать функции под конкретные задачи.

Внешний вид среды разработки Visual Studio 2013 Premium представлен на рисунке 10.

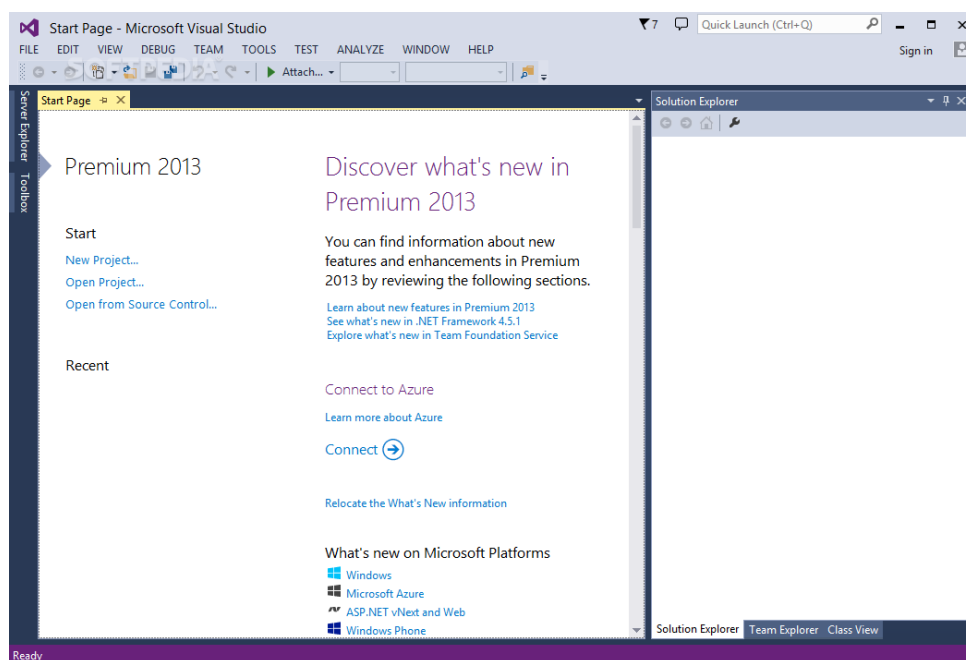


Рисунок 10 – Среда разработки Visual Studio 2013 Premium

Особенности пользовательского интерфейса разрабатываемого программного продукта:

- проект основан на структуре диалогового окна, что упрощает создание пользовательского интерфейса;
- инкапсулирует базовые операции рисования форм и элементов управления, позволяя разработчикам сосредоточиться на логике приложения MFC;
- для создания интерфейса используются визуальные инструменты Visual Studio, которые автоматически генерируют соответствующий код на C++;
- в диалоговое окно можно добавлять различные элементы управления (кнопки, текстовые поля и т. д.) путем перетаскивания;
- после редактирования интерфейса Visual Studio автоматически создает соответствующие функции-члены классов, которые можно адаптировать под нужды проекта.

Cython – оптимизирующий статический компилятор для Python, который позволяет преобразовывать Python-код в высокопроизводительный C-код, интегрировать внешние C-библиотеки, создавать Python-расширения с ускоренной работой.

На рисунке 11 показано превосходство Cython в оптимизации кода по сравнению с другими языками

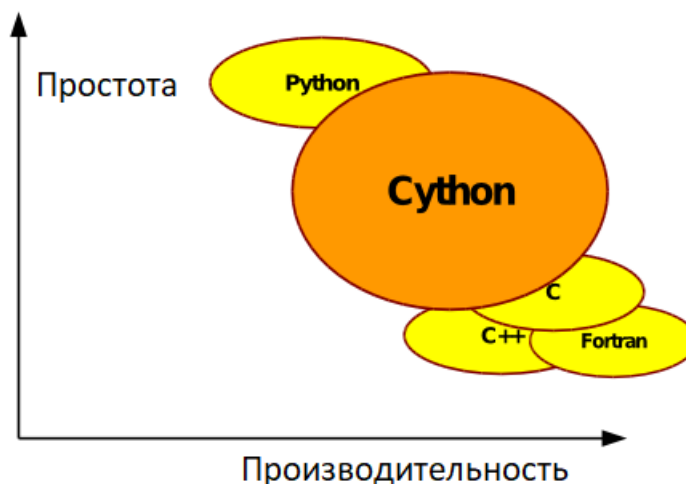


Рисунок 11 – Сравнение Cython с другими языками

Преимущества для разработки:

- сохранение удобства Python с производительностью C
- постепенная оптимизация (можно начать с чистого Python)
- широкая поддержка в научном стеке (NumPy, SciPy, OpenCV)

Для систем распознавания лиц Cython особенно полезен при обработке видео в реальном времени и оптимизации вычислительно сложных алгоритмов, позволяя достичь производительности, сравнимой с чистыми C-решениями, сохраняя при этом гибкость Python.

2.3.3 Типы данных и механизм сообщений

MFC определяет множество часто используемых типов данных, которые упрощают разработку приложений. В данном проекте наиболее часто используется тип CString, который представляет собой класс для работы со строками в MFC.

Основные особенности типа CString:

- все строки, отображаемые в элементах управления формы (например, текстовые поля, метки), имеют тип CString;
- данные, записываемые в базу данных или извлекаемые из неё, также

используют тип CString;

- MFC предоставляет множество функций и макросов для преобразования типов данных. Функции позволяют преобразовать тип CString в строковый тип (например, char*), а также выполнять базовые преобразования типов, такие как форматирование чисел в строки. Для более сложных преобразований типов данных может потребоваться написание собственных функций.

В Windows приложениях используется механизм сообщений, который позволяет системе и приложениям взаимодействовать друг с другом. Сообщения в Windows связаны с потоками, и каждое сообщение отправляется в соответствующий поток для обработки.

На рисунке 12 показана классификация сообщений на несколько категорий

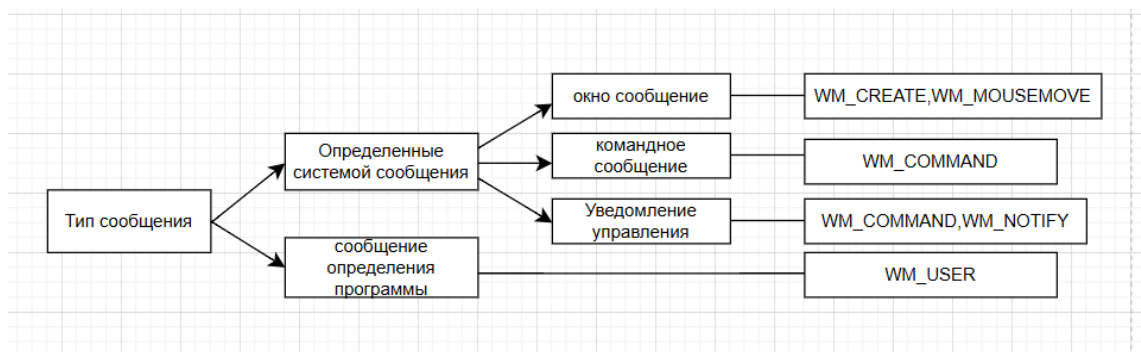


Рисунок 12 – Классификация сообщений

Тип сообщения делится на следующие подвиды:

- определенные системой сообщения, стандартные сообщения, которые генерируются операционной системой;
- сообщение, отправляемое при создании окна WM_CREATE;
- сообщение, отправляемое при движении мыши WM_MOUSEMOVE;
- сообщения определения программы, пользовательские сообщения, которые могут быть определены разработчиком для выполнения специфических задач;
- оконные сообщения, сообщения, связанные с окнами, такие как WM_COMMAND, которое отправляется при выборе пункта меню или нажатии кнопки;
- командные сообщения, сообщения, которые отправляются при

взаимодействии пользователя с элементами управления, такими как кнопки или меню;

- сообщение, отправляемое при выборе команды (например, нажатии кнопки) WM_COMMAND;
- сообщение, отправляемое при уведомлениях от элементов управления (например, изменение состояния элемента управления) WM_NOTIFY;
- пользовательские сообщения, сообщения, которые начинаются с WM_USER, могут быть определены разработчиком для реализации специфической логики приложения.

Обработка сообщений в MFC:

- инкапсулирует механизм обработки сообщений, предоставляя разработчику удобные методы для работы с сообщениями MFC;
- каждое сообщение может быть обработано в соответствующей функции-члене класса окна или диалогового окна;
- может быть обработано WM_COMMAND в функции OnCommand();
- WM_MOUSEMOVE в функции OnMouseMove().

В проекте типы данных, такие как CString, и механизм сообщений тесно связаны. Например:

- данные, введенные пользователем в текстовое поле (тип CString), могут быть обработаны при получении сообщения WM_COMMAND (например, при нажатии кнопки «ОК»);
- результаты обработки данных могут быть преобразованы с использованием функций CString и отображены в другом элементе управления.

Пример использования:

- пользователь вводит текст в текстовое поле (тип CString);
- при нажатии кнопки "ОК" генерируется сообщение WM_COMMAND;
- в функции обработки сообщения данные из текстового поля извлекаются и преобразуются с использованием функций CString;
- результат обработки данных отображается в другом элементе управления (например, в метке).

Такая структура позволит вам логично объединить описание типов данных и механизма сообщений, подчеркнув их важность в разработке приложений на основе MFC.

2.4 Характеристика выбранного программно-технического обеспечения

Платформа ROS (Robot Operating System) обеспечивает:

- управление аппаратными модулями (камеры, датчики);
- реализация распределенных вычислений;
- поддержка пакетов для компьютерного зрения (например, cv_bridge).

На рисунке 13 представлена схема взаимодействия компонентов алгоритма ROS.

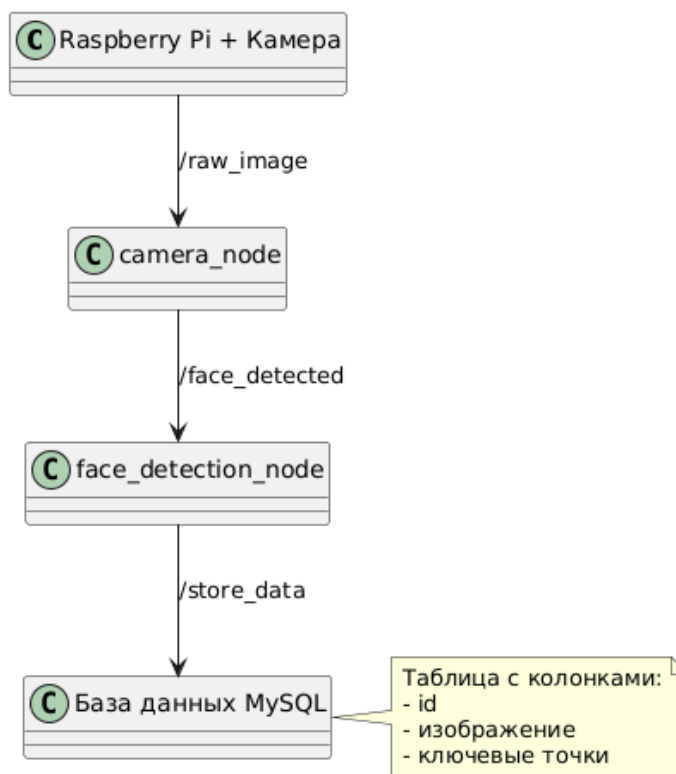


Рисунок 13 – Схема взаимодействия компонентов алгоритма ROS.

2.4.1 Аппаратная платформа: Raspberry Pi 3B+

Модель Raspberry Pi 3B+ оснащена четырехъядерным процессором ARM Cortex-A53 с тактовой частотой 1.4 ГГц и графическим процессором VideoCore IV. Она поддерживает 4 порта USB 2.0 и характеризуется энергопотреблением около 5 Вт. В отличие от нее, Raspberry Pi 4B (4 ГБ) использует более производительный процессор Cortex-A72 с частотой 1.5 ГГц и графический процессор

VideoCore VI. Эта модель также включает 2 порта USB 3.0 и 2 порта USB 2.0, что улучшает скорость передачи данных, однако энергопотребление возрастает до 7.5 Вт.

Основные компоненты аппаратной платформы:

- процессор Broadcom BCM2837 (4 ядра ARM Cortex-A53, 1.4 ГГц);
- память 1 ГБ LPDDR2;
- интерфейсы USB 2.0, HDMI, GPIO;

Преимущества:

- низкая стоимость;
- поддержка Linux.

Для повышения эффективности работы на Raspberry Pi были применены следующие меры:

- интеграция OpenVINO Toolkit, позволяющего ускорить выполнение нейронных сетей на совместимых с Intel чипах;
- настройка swar-файла объемом 2 ГБ для предотвращения перегрузки оперативной памяти при обработке больших наборов данных.

2.4.2 NVIDIA Jetson TX2

NVIDIA Jetson TX2 обладает графическим процессором с 256 ядрами Pascal, что обеспечивает поддержку CUDA и cuDNN для прямого обучения моделей на устройстве.

Платформа предлагает два режима работы:

- Max-N (максимальная производительность при 15 Вт);
- Min-N (энергосбережение при 7.5 Вт).

Интеграция с ROS:

- использование пакета `ros_deep_learning` для развертывания глубоких нейронных сетей в среде ROS;
- поддержка протокола ZMQ, обеспечивающего распределенные вычисления между узлами системы.

Производительность в сравнении с Raspberry Pi 4B:

- задача детекции лиц с использованием алгоритма MTCNN выполняется

на Jetson TX2 за 12 мс, тогда как на Raspberry Pi 4B аналогичная операция занимает 85 мс;

- сравнение эмбеддингов лиц на Jetson TX2 требует всего 3 мс, в то время как на Raspberry Pi 4B это занимает 22 мс.

2.4.3 Альтернативные платформы: Coral USB Accelerator

Coral USB Accelerator предоставляет ускорение для моделей TensorFlow Lite за счет встроенного TPU (Tensor Processing Unit), потребляя всего 1 Вт энергии. Однако его использование ограничено поддержкой только квантованных моделей, что требует дополнительной оптимизации исходных алгоритмов.

Сравнение с NVIDIA Jetson Nano:

- стоимость Coral USB Accelerator составляет около 75\$, это несколько ниже цены Jetson Nano (75\$, это в свою очередь несколько ниже цены Jetson Nano(99));

- по производительности Coral обеспечивает 4 TOPS (триллионов операций в секунду) для операций INT8, тогда как Jetson Nano достигает 472 GFLOPS (миллиардов операций с плавающей запятой);

- интеграция с ROS на Coral требует использования сторонних драйверов, в отличие от Jetson Nano, который поддерживает ROS нативно.

3 ПРОГРАММНАЯ РЕАЛИЗАЦИЯ ПРЕДЛАГАЕМОГО АЛГОРИТМА РЕШЕНИЯ ЗАДАЧИ РАСПОЗНАВАНИЯ ЛИЦ

3.1 Основные этапы практической разработки программного продукта

3.1.1 Проектирование архитектуры системы

Основная цель этого проекта – разработка системы распознавания лиц для мониторинга в реальном времени. Камера используется в ситуациях, когда требуется мониторинг. После инициализации системы она автоматически обнаруживает и идентифицирует лица, появляющиеся на экране. и может записывать идентификационные данные человека и время появления на экране, а неузнаваемые символы помечаются как Неизвестные и записывается временная метка. Система может идентифицировать только людей, которые заранее собрали информацию. На этапе сбора информации необходимо записать имя и аватар человека.

На рисунке 14 показана схема модуля мониторинга в реальном времени.



Рисунок 14 – Схема модуля мониторинга в реальном времени

Вся система разделена на модуль сбора информации и модуль мониторинга в реальном времени.

Часть обнаружения использует библиотеку обнаружения лиц, раскрытую

учителем Ю Шици. После настройки библиотеки dll импортируйте соответствующий файл заголовка в файл проекта, чтобы использовать интерфейс этой библиотеки. Эта библиотека предоставляет четыре разных приложения для разных приложений. в этом проекте тот, который имеет лучший эффект обнаружения, но самый трудоемкий

Список параметров функции включает в себя указатель на область, где хранятся результаты обнаружения, указатель на область, где хранятся данные входного изображения, длину и ширину входного изображения, и некоторые другие параметры. Функция возвращает указатель на. область, в которой хранятся результаты обнаружения. Ключевым моментом является извлечение данных, на которые указывает указатель, предполагая, что возвращаемый указатель это `pResult`, то первые целочисленные данные, на которые указывает `pResult`, представляют количество лиц, обнаруженных во входном изображении. Информация о лице каждого человека хранится в области непрерывной памяти, и каждый человек занимает 142 целочисленных данных. Первые четыре числа представляют собой верхнее число. левый угол и левый угол нижнего правого угла ограничивающей рамки лица, пятые данные представляют собой информацию, называемую соседом (ее конкретное значение в настоящее время неизвестно), а шестые данные представляют собой угол диапазона лица, равный $0 \sim 60$ градусов, тогда каждые два данных образуют координатную точку, обозначающую 68 характерных точек человеческого лица.

Эти характерные точки отмечают контур лица, положение и контур глаз, бровей, носа и рта, занимая в общей сложности. из 136 данных, как показано на рисунке 15.

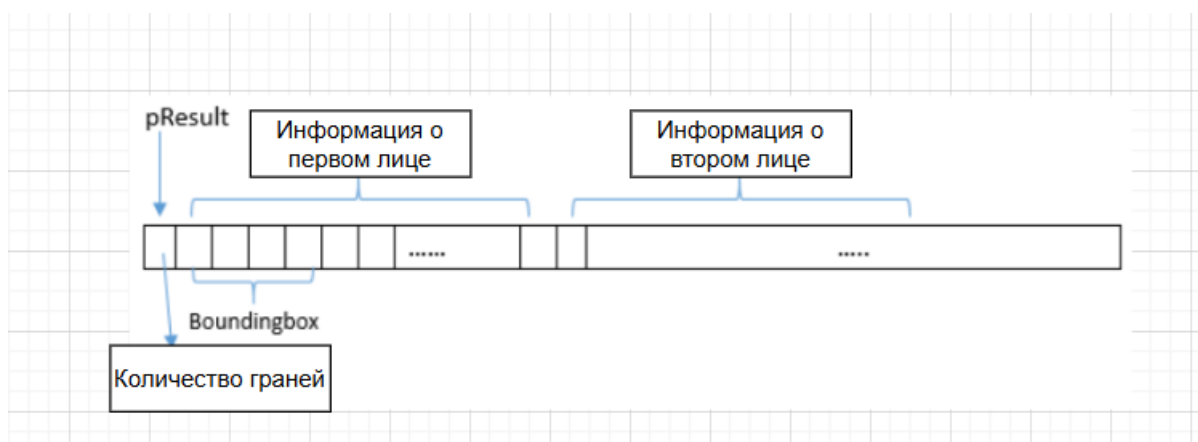


Рисунок 15 – Инфографика лица

На рисунке 16 показана результаты теста

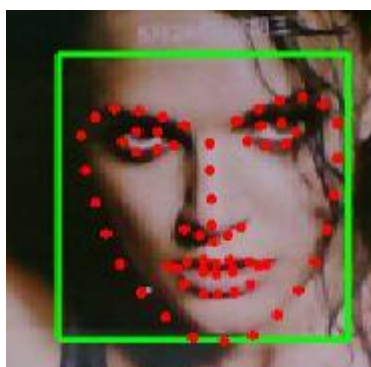


Рисунок 16 – Результаты теста

Обнаружению на основе мониторинга необходимо только вынимать изображение из камеры через определенные промежутки времени, отправлять изображение в функцию обнаружения, рисовать структуру обнаружения на изображении и выводить ее в пользовательский интерфейс.

Для извлечения признаков распознавания используется модель нейронной сети FaceNet, реализованная на основе TensorFlow. Как упоминалось ранее, C++ API TensorFlow не был успешно использован, поэтому использовался его Python API, а функции, которые непосредственно использовали Tensorflow, были написаны на Python и определены в Python через Cython. Преобразуйте функцию в функцию, определенную в форме C. Cython автоматически сгенерирует файл заголовка и исходный файл соответствующей функции. Импортируйте

сгенерированный файл в проект, чтобы использовать функцию, определенную в форме C. Все способы использования TensorFlow инкапсулированы в класс под названием TensorFlowInference. Этот класс инкапсулирует операции получения и обслуживания объектов сеанса, инициализацию среды Python, загрузку сетевых моделей, расчет функций сопоставления и все взаимодействия с данными C++ и Python. полностью скрывая весь контент, связанный с Python.

Стоит отметить, что вычисления в TensorFlow выполняются через объект сеанса. Объект сеанса хранит структурную информацию и информацию о весе графа потока данных. Модель сети, используемая в проекте, очень велика (141 МБ). Перегрузка модели с диска в память для каждого расчета серьезно увеличивает время работы программы (каждый раз загрузка модели занимает около двух минут), поэтому разумным решением должна быть загрузка модели в память при инициализации всей системы и получение ее объекта сеанса, чтобы сделать его резидентным в памяти. Каждое вычисление может завершиться вычислением путем вызова метода- члена объекта сеанса (. каждый расчет занимает около 0,3 ~ 0,4 секунды). В скрипте Python объект сеанса получается через интерфейс tf, предоставляемый Tensorflow. Session() получен , но перед получением объекта сеанса необходимо загрузить структурную информацию графа потока данных в граф по умолчанию , а затем восстановить веса в графе для объекта сеанса, чтобы последующий объект сеанса мог правильно выполнить форвардный расчет FaceNet. Полученный объект сеанса будет отображаться как тип PyObject в C++.

Взаимодействие данных между C++ и Python. Одной из трудностей в этой теме является взаимодействие данных между C++ и Python. Форма взаимодействия данных через формальные параметры и возвращаемые значения функций. В основном оно включает два направления. данные в части C++ передаются в Python, а во-вторых, результаты вычислений, выполненных Python, возвращаются в C++, согласно Питу. Официальный метод, предоставленный hon, преобразовать все данные типа C++ в объекты PyObject для взаимодействия. Однако, поскольку все объекты Python в C++ наследуются от типа PyObject, такой гибкий тип данных создает определенную сложность, и используются данные типа

Image. в этой теме, что в Python является числом `py.ndarray type`, `numpy` официально обеспечивает поддержку `ndarray` для версии C++ , но при фактическом использовании используемый интерфейс функции устарел. К счастью, Cython позволяет использовать структуры данных типа C в скриптах Python, что делает преобразование данных из C++ в Python гораздо более удобным. На рисунке 17 показана схема потока данных и процесса преобразования.

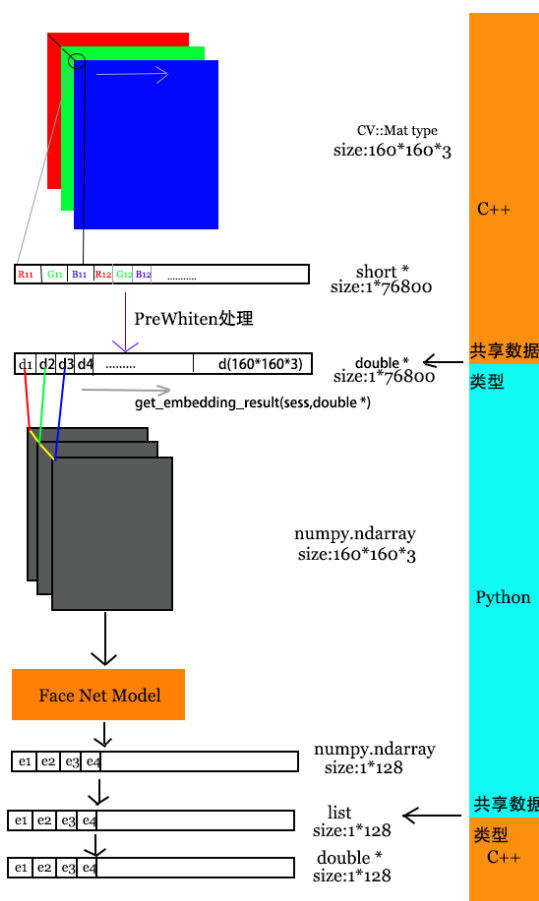


Рисунок 17 – Схема потока данных и процесса преобразования

Embedding: эта переменная представляет собой двойной указатель, указывающий на небольшую область данных, используемую для записи идентификационных характеристик персонажа. Она вычисляется на основе другой переменной-члена `PortraitInput`, что эквивалентно результату $f(\cdot)$.

На рисунке 18 представлена диаграмма, представляющая ключевые методы классов проекта в виде кодовых блоков с визуализацией зависимостей.

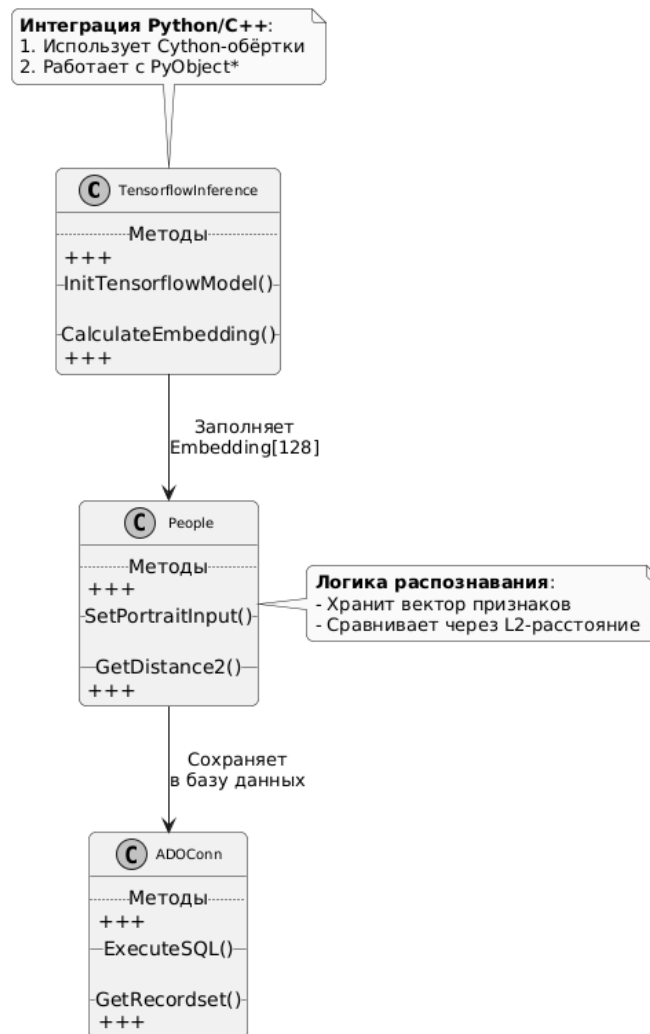


Рисунок 18 – Ключевые методы классов проекта

Особенности диаграммы:

- точные C++-объявления;
- основная логика в псевдокомментарии;
- TensorflowInference вычисляет эмбединги для People;
- People сохраняет данные через ADOConn;
- указание на Cython-обёртки;
- формат данных (128-D вектор);
- монохромный skinparam для научных работ;
- Courier-шрифт для кодовых блоков;
- чёткое разделение методов.

Полная цепочка вызовов представлена на диаграмме последовательности на рисунке 19.

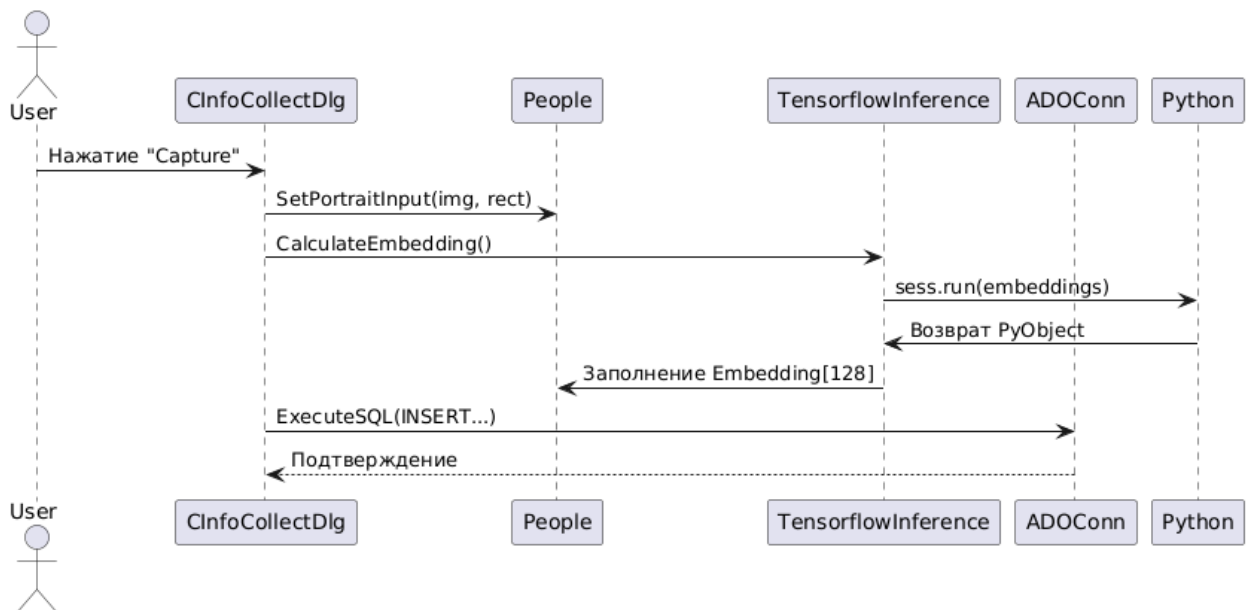


Рисунок 19 – Диаграмма последовательности цепочки вызовов

Обе диаграммы дополняют друг друга, показывая структуру (классы и методы), поток выполнения (последовательность вызовов).

Класс вывода TensorFlow используется в качестве инкапсуляции нескольких API-интерфейсов TensorFlow Python, используемых в этом разделе, всех операций взаимодействия с данными C++, Python и операций предварительной обработки изображений лиц. Этот класс предоставляет три чистых интерфейса в стиле C для поддержания важного объекта Session. В реализации он используется как переменная -член глобального класса CXXXApp. При инициализации всего проекта будет открыт отдельный рабочий поток для завершения очень трудоемкого FaceNet. Работа по загрузке модели. После инициализации TensorFlow можно использовать для расчета характеристик отображения лица, ссылаясь на объект TensorflowInference в глобальном объекте theApp в любом месте проекта.

Диаграмма, детализирующая основные классы проекта с визуализацией типов и связей представлена на рисунке 20.

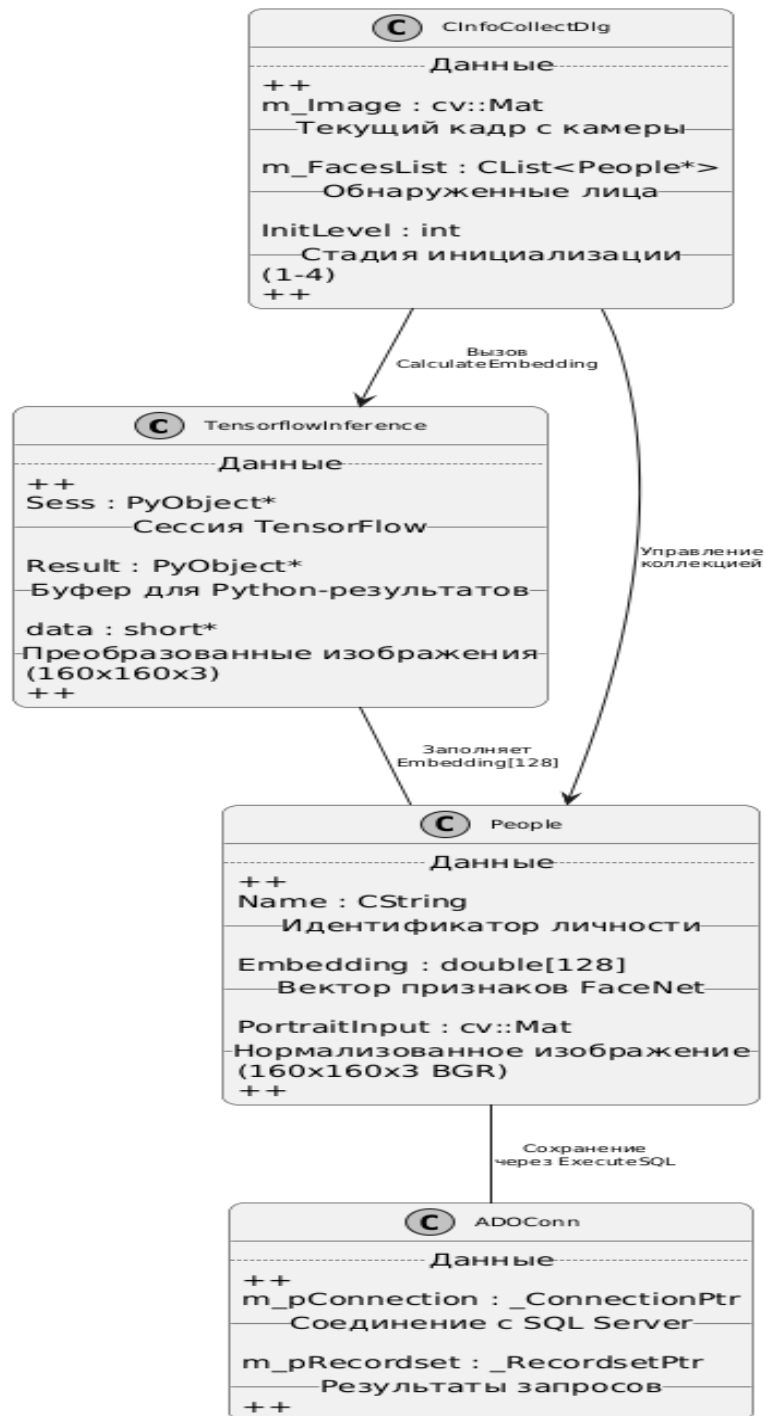


Рисунок 20 – UML-диаграмма основных классов проекта

Пояснения к элементам:

- указатель Sess на объект сеанса Python (PyObject);
- буфер Data для изображений в формате short (размер 160×160×3);
- массив Embedding из 128 чисел double (результат работы FaceNet);
- изображение PortraitInput в формате OpenCV (BGR, 160×160);

- указатели ADOConn для работы с SQL Server через ADO;
- графические связи, сплошные линии представляют прямое использование, а стрелки вызовы методов.

На рисунке 21 представлено дополнение для временных переменных.

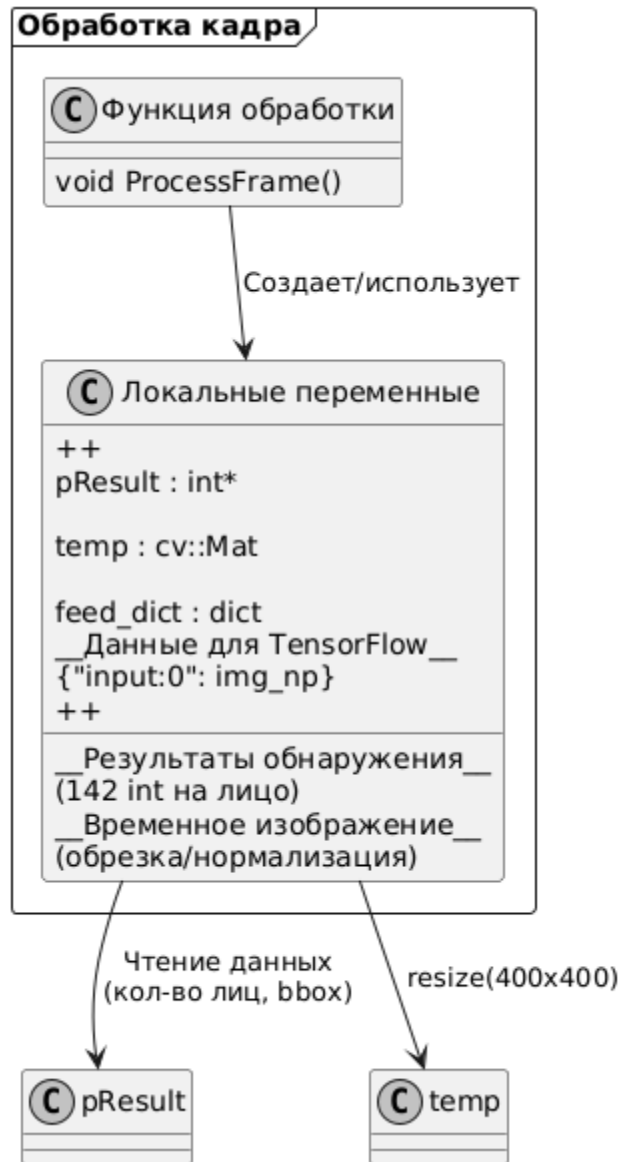


Рисунок 21 – Дополнение для временных переменных

Эта диаграмма помогает понять:

- типы данных (COM-указатели, PyObject*, OpenCV-матрицы);
- форматы хранения (размеры изображений, размерность векторов);
- критические зависимости между компонентами.

Поскольку полученный объект Session будет использоваться как формальный параметр функции, определенной в Python, C ++ не требуется анализировать

этот объект PyObject, а необходимо только поддерживать его существование в памяти.

Calculate Embedding – одна из основных функций этой темы. Ее функция заключается в вычислении характеристик отображения лица. Она принимает стандартный указатель типа 160, вычисленный результат будет сохранен в небольшой области памяти, на которую указывает. Диаграмма действий функции Calculate Embedding представлена на рисунке 22.

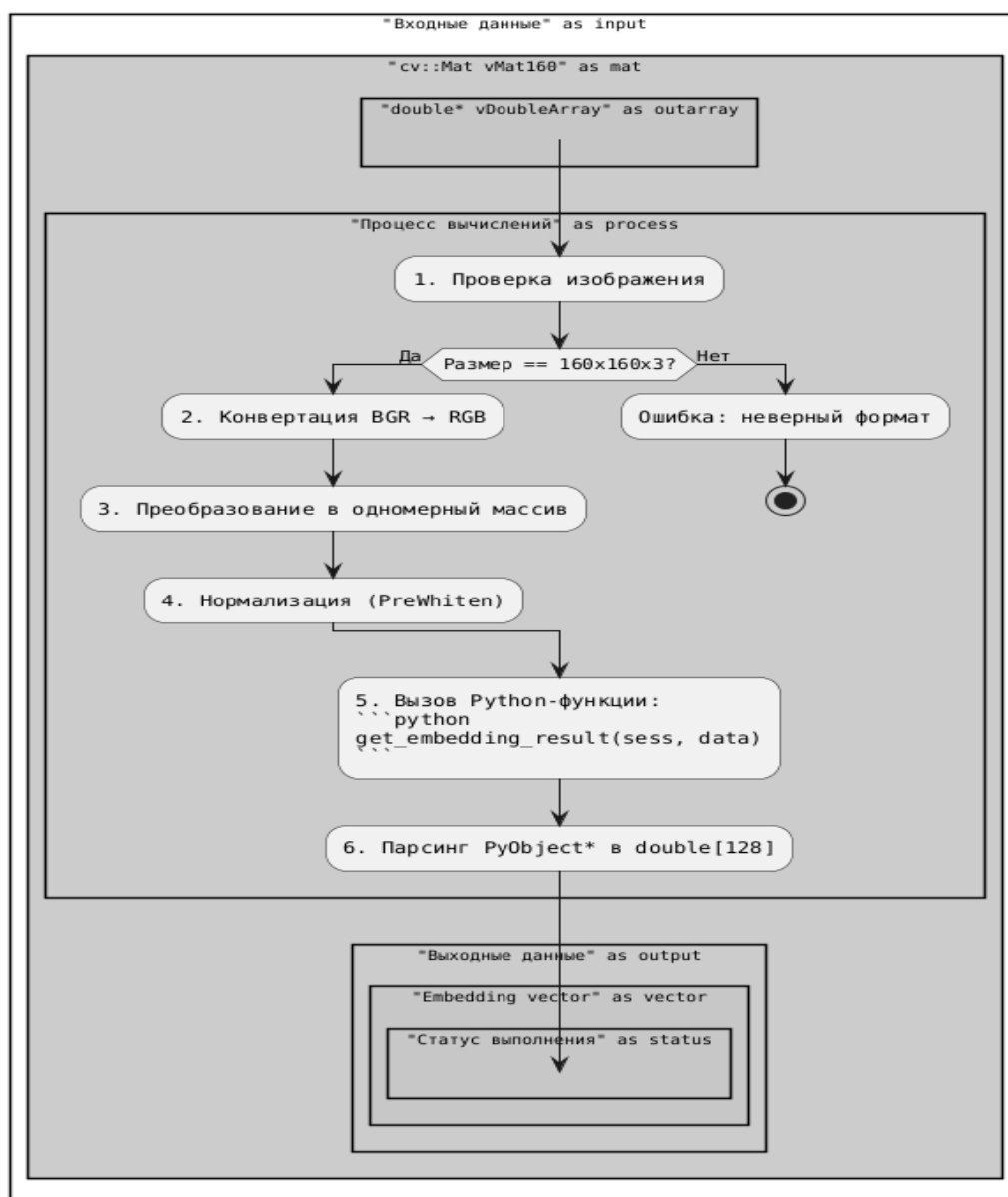


Рисунок 22 – Диаграмма действий функции Calculate Embedding

Вызывающий код (класс People) передает изображение лица $160 \times 160 \times 3$ в формате BGR (cv:Mat vMat160) и указатель на буфер для записи результата (double vDoubleArray).

Программа убеждается, что изображение имеет размер $160 \times 160 \times 3$, преобразует BGR (OpenCV) в RGB (требование FaceNet) и разворачивает 3D-матрицу в плоский массив `short` (76800 элементов). Последним этапом является нормализация (PreWhiten).

C++ передает `short* data` в Cython-обёртку. Cython преобразует массив в `NumPy.ndarray[1,160,160,3]` (формат, ожидаемый TensorFlow).

Создаются входные данные для TensorFlow (`feed_dict`) и запускается вычисление на графе TensorFlow. Далее возвращается результат в виде `NumPy.ndarray` размером `[1, 128]`.

Python-результат (`NumPy.ndarray`) конвертируется в `list[float]`. Cython распаковывает список в `double [128]` и заполняет переданный буфер.

В результате получается Выходной буфер `vDoubleArray`, содержащий 128-мерный вектор признаков.

Функция `get_embedding_result()`, как и функция `get_session()`, также определена в Python и преобразована Cython, поскольку NumPy не поддерживается в C++. Функция синтаксического анализа `ndarray` устарела, поэтому вывод Tensor типа `NumPy.ndarray` в модели необходимо преобразовать в тип `List`, который может быть успешно проанализирован C++.

Кроме того, согласно официальной информации, предоставленной Cython, в скриптах Python, написанных с синтаксисом Cython, использование типов данных C для определения переменных может ускорить выполнение программы. Это связано с тем, что нижний уровень самого Python является языком C, поэтому определение может быть определено. в C Все переменные определены в C.

В этом разделе делается попытка напрямую скомпилировать исходный код TensorFlow для использования его интерфейса C++, но официальным методом компиляции является использование Bazel, внутреннего инструмента компиляции, используемого Google. Этот инструмент очень нестабилен на платформе Windows. бета-версия. В процессе использования возникли различные проблемы, некоторые из которых остались нерешёнными и привели к сбою компиляции.

Для решения задачи рекомендуется использование интерфейса Python, так как большинство руководств также основаны на интерфейсе Python, поэтому в этом разделе используется гибридная технология C++ Python, инкапсулирующая интерфейс Python в форму языка C, чтобы можно было использовать TensorFlow косвенно в программе. Модель FaceNet, используемая в проекте, предварительно обучена, а структура ее графа потока данных закреплена и сохранена в формате. В метафайле веса на графике закрепляются и сохраняются в skpt- файл.

Платформа MFC инкапсулирует большинство операций обработки сообщений. В сгенерированный класс формы можно добавить функции ответа на сообщения. При обнаружении поведения пользователя или системы, которое генерирует определённое сообщение, система автоматически вызывает соответствующую функцию для выполнения операции. Основные сообщения, используемые в проекте:

- сообщение рисования окна ON_WM_PAINT;
- сообщение системного времени ON_WM_TIMER;
- сообщение о нажатии кнопки ON_BN_CLICKED.

TensorFlow – среда численных вычислений с открытым исходным кодом, использующая графы потоков данных. Она широко используется для реализации различных нейронных сетей. Узлы графа потока данных представляют собой математические операции, а линии на графике представляют собой массивы данных (также называемые тензорами).

TensorFlow изначально был разработан исследователями и инженерами Google Brain Group для исследований в области машинного обучения и глубоких нейронных сетей, но поскольку платформа универсальна, ее также можно использовать в других областях вычислений. Его гибкая архитектура позволяет развертывать узлы на одном или нескольких процессорах или графических процессорах для параллельных вычислений.

Схема, визуализирующая ключевые Python-интерфейсы, используемые в проекте для работы с TensorFlow представлена на рисунке 23.

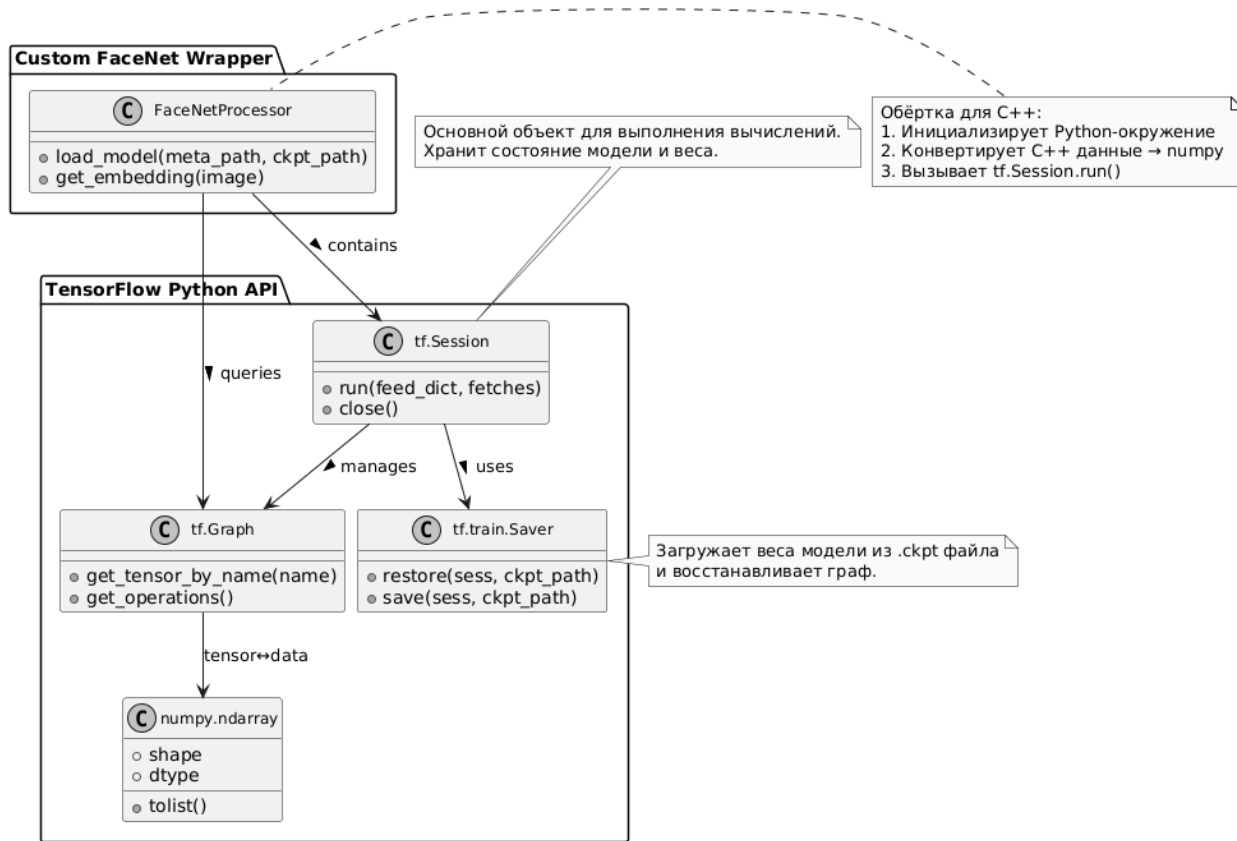


Рисунок 23 – Используемые Python интерфейсы

Ядро TensorFlow состоит из центрального объекта для выполнения вычислений `tf.Session`, загрузка/сохранение весов модели `tf.train.Saver`, доступ к узлам модели (входы/выходы) `tf.Graph`.

Класс `FaceNetProcessor` инкапсулирует преобразование изображений из OpenCV (`cv:Mat`) в NumPy `ndarray`, вызов `sess.run()` для получения числовых векторов и обратную конвертацию результатов в C++-формат.

3.1.2 Проектирование и разработка базы данных

ActiveX – компонент, используемый для доступа к источникам данных. Он обеспечивает средний уровень для программирования и унифицированные методы доступа к данным OLE DB, позволяя разработчикам писать код для доступа к данным, независимый от реализации базы данных. Необходимо только оставить ссылку на базу данных.

Диаграмма потока данных – вычислительная модель, управляемая потоком данных. Она использует ориентированный граф узлов и ребер для описания

математических вычислений. Узлы обычно используются для представления математических операций, применяемых к данным, но их также можно выразить в начальной или конечной точке вывода данных.

В этом разделе операции с базой данных инкапсулируются в классе ADOConn. Инкапсулированный класс предоставляет интерфейсы для операций с базой данных, таких как инициализация соединения с базой данных, непосредственное выполнение операторов SQL, получение указателя результирующего набора оператора запроса и отключение.

Класс ADOConn инкапсулирует все операции с базой данных и находится на один уровень ниже уровня абстракции класса People. Класс People выполняет операции с базой данных, ссылаясь на объект класса ADOConn. На рисунке 24 показана UML-диаграмма действий в классе ADOConn.

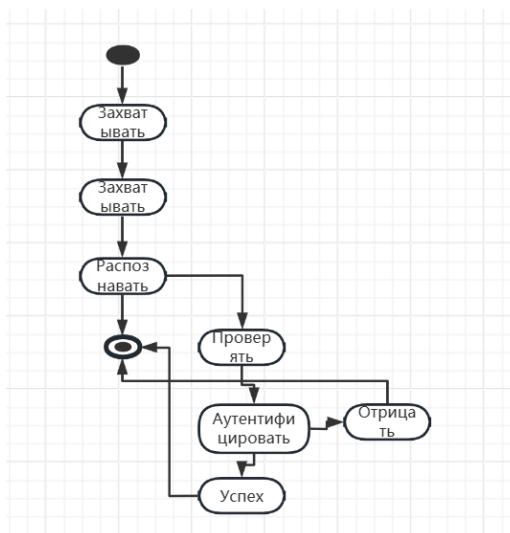


Рисунок 24 – UML-диаграмма действий в классе ADOConn

Уровень абстракции этого класса относительно низок, а технология относительно отстала. Большую часть времени он используется в качестве библиотеки, а часть базы данных в этой теме не рассматривается как основное внимание, поэтому она представлена лишь вкратце.

формальным параметром которой является `_bstr_t`, может напрямую выполнять оператор SQL, написанный на C++. Результат, возвращаемый базой данных, сохраняется в области, на которую указывает `_RecordsetPtr`. С помощью определенного анализа мы можем получить нужные данные, которые включают

в себя некоторые типы данных. Что касается проблем преобразования, функции для операций с базой данных в классе People инкапсулировали эти операции, включая преобразование CString в двойной массив, преобразование двойного массива в строку CString и т. д., которые здесь не будут подробно описаны.

Определение сущностей и атрибутов необходимо для создания логической модели базы данных, которая позволяет структурировать данные и определить их взаимосвязи. Это помогает избежать ошибок и недоразумений на более поздних этапах разработки.

В таблице 2 показано описание атрибутов сущности «Люди». Данные в этой таблице хранят информацию о зарегистрированных пользователях.

Таблица 2 – сущность «Люди»

Поле	Тип данных	Описание	Ограничения
PersonID	INT	Уникальный идентификатор	PRIMARY KEY, AUTO_INCREMENT
Name	VARCHAR(50)	Имя человека	NOT NULL, UNIQUE
Embedding	BLOB	128-мерный вектор признаков	NOT NULL
PortraitPath	VARCHAR(255)	Путь к изображению лица	NOT NULL
CreatedAt	DATETIME	Дата регистрации	DEFAULT CURRENT_TIMESTAMP

В таблице 3 показано описание атрибутов сущности «Лог обнаружений». Данные в этой таблице необходимы для фиксации всех случаев распознавания.

Таблица 3 – сущность «Лог обнаружений»

Поле	Тип данных	Описание	Ограничения
1	2	3	4
LogID	INT	Уникальный ID записи	PRIMARY KEY
PersonID	INT	Ссылка на People.PersonID	FOREIGN KEY
DetectionTime	DATETIME	Время обнаружения	NOT NULL

1	2	3	4
Confidence	FLOAT	Уверенность распознавания (0..1)	DEFAULT 0.0
CameraID	VARCHAR(20)	Идентификатор камеры	NULL

В таблице 4 показано описание атрибутов сущности «Неизвестные лица». Данные в этой таблице необходимы для записи неузнаваемых лиц.

Таблица 4 – сущность «Неизвестные лица»

Поле	Тип данных	Описание
FaceID	INT	Уникальный ID
ImagePath	VARCHAR(255)	Путь к сохраненному изображению
DetectionTime	DATETIME	Время обнаружения

На рисунке 25 представлена ER-диаграмма отношения сущностей.

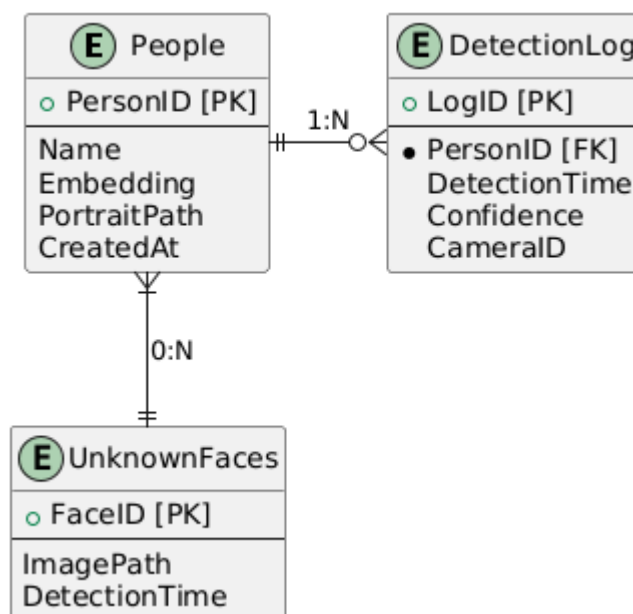


Рисунок 25 – ER-диаграмма отношения сущностей

Один пользователь может иметь множество записей в журнале распознаваний.

Неопознанные лица могут быть позже ассоциированы с пользователем.

3.1.3 Реализация многопоточной обработки

В MFC есть поток пользовательского интерфейса и рабочий поток. Важное различие между ними заключается в том, что поток пользовательского интерфейса имеет собственную очередь сообщений и цикл сообщений, а в рабочем потоке такой функции не предусмотрено.

Задача потоков обычно заключается в обработке интерфейса, реагировании на пользовательские операции, события и сообщения, генерируемые системой. Большинство функций-членов в классе окна, автоматически создаваемых `vs`, выполняются как поток пользовательского интерфейса.

Неподходящий метод проектирования заключается в использовании всего кода в потоке пользовательского интерфейса. Это действительно просто и удобно, но это принесет некоторые серьезные последствия. Проблемы с эффективностью, такие как некоторые вычисления, которые занимают много памяти и занимают много времени, если их поместить в поток пользовательского интерфейса, приведут к тому, что интерфейс перестанет отвечать на запросы в течение длительного времени.

Поэтому обработка большого количества вычислений должна быть выполнена. выполняться отдельно в отдельном рабочем потоке, синхронизироваться с потоком пользовательского интерфейса, используя передачу сообщений или общую память.

Основная схема функций MFC представлена на рисунке 26.

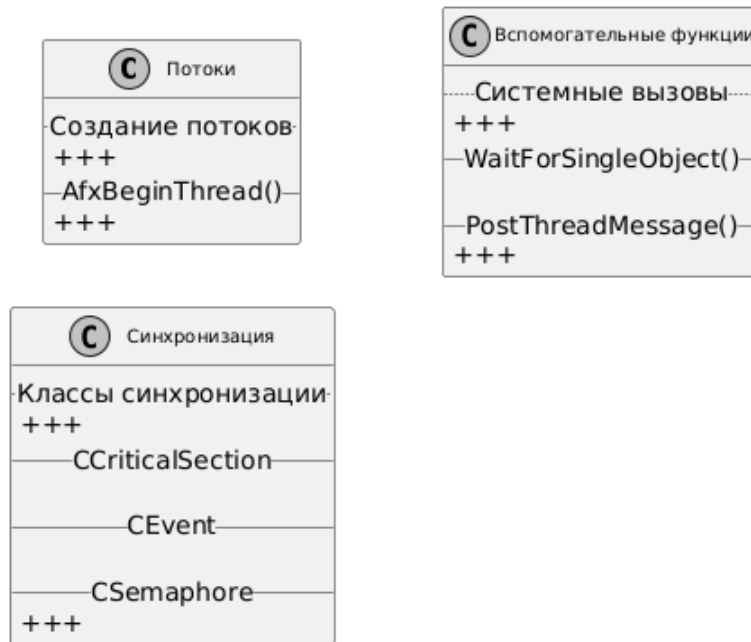


Рисунок 26 – Основная схема функции MFC

Центральным элементом блока создания потоков (Threads) является функция AfxBeginThread()

Основные параметры:

- указатель на функцию потока (AFX_THREADPROC);
- параметр для передачи данных (LPVOID);
- приоритет и атрибуты создания.

Тремя основными классами для блока синхронизации (Sync) являются:

- для защиты критических секций CCriticalSection;
- для сигнализации между потоками CEvent;
- для ограничения доступа к ресурсам CSemaphore.

Вспомогательные функции (Utils) являются системными вызовами Windows API и делятся на подфункции:

- ожидание объектов ядра WaitForSingleObject;
- отправка сообщений между потоками PostThreadMessage.

Связи:

- потоки используют механизмы синхронизации для координации работы;
- потоки могут вызывать системные функции для расширенного

управления.

Особенности реализации:

- акцент на thread-safe операции в многопоточной среде;
- использование паттернов синхронизации (критические секции, события).

Схема примеров потоков представлена на рисунке 27.

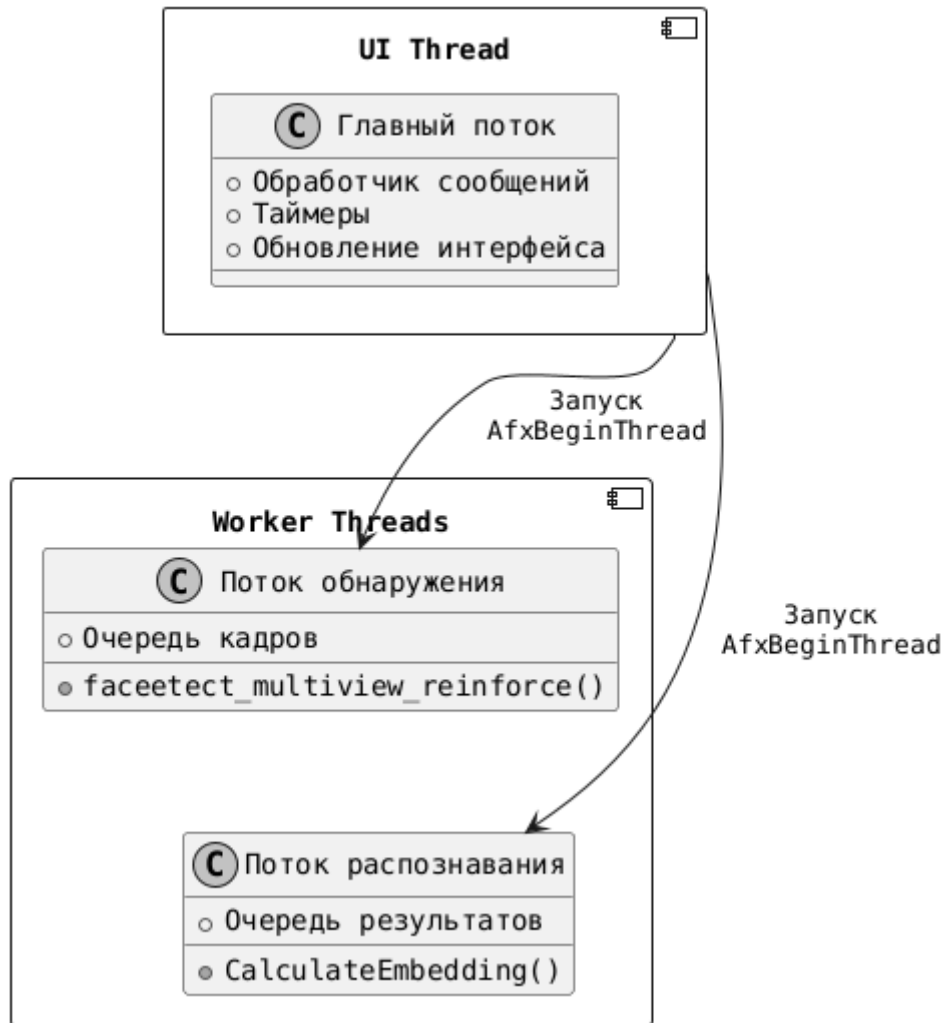


Рисунок 27 – Схема примеров потоков

Главный (UI) поток:

- обрабатывает сообщения окон;
- управляет таймерами;
- обновляет пользовательский интерфейс;
- запускает рабочие потоки.

Поток обнаружения:

- выполняет функцию `faceetect_multiview_reinforce()`;
- работает с очередью кадров;

- синхронизация через CCriticalSection.

Поток распознавания:

- вычисляет эмбединги CalculateEmbedding();
- использует очередь результатов;
- взаимодействует с TensorflowInference.

Ключевые моменты:

- четкое разделение UI и фоновых операций;
- использование очередей для передачи данных;
- схема отражает паттерн Producer-Consumer.

На рисунке 28 представлена схема, детализирующая интеграцию TensorFlow в проект на C++ через Python API.

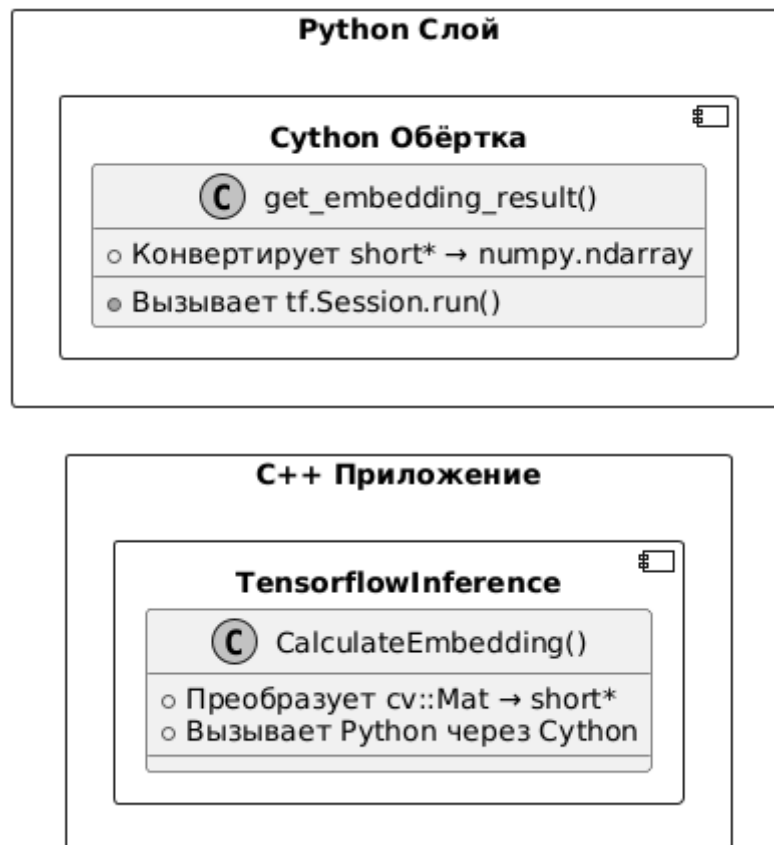


Рисунок 28 – Схема, детализирующая интеграцию TensorFlow

Класс CalculateEmbedding() является компонентом TensorflowInference и содержит:

- вход cv:Mat (160x160x3 BGR);
- выход double (128-мерный вектор);

- преобразования (конвертация BGR в RGB, нормализация (PreWhiten), упаковка в short).

Функция `get_embedding_result()` является компонентом Cython Обёртки и выполняет следующие функции:

- конвертирует short в `NumPy.ndarray[1,160,160,3]`;
- готовит `feed_dict` для TF;
- вызывает `sess.run()`.

Модель FaceNet является компонентом TensorFlow и содержит:

- входной узел `input:0` (формат `[None,160,160,3]`);
- выходной узел `embeddings:0` (формат `[None,128]`);
- вес 141MB (.ckpt файл).

Диаграмма последовательности вызовов представлена на рисунке 29.

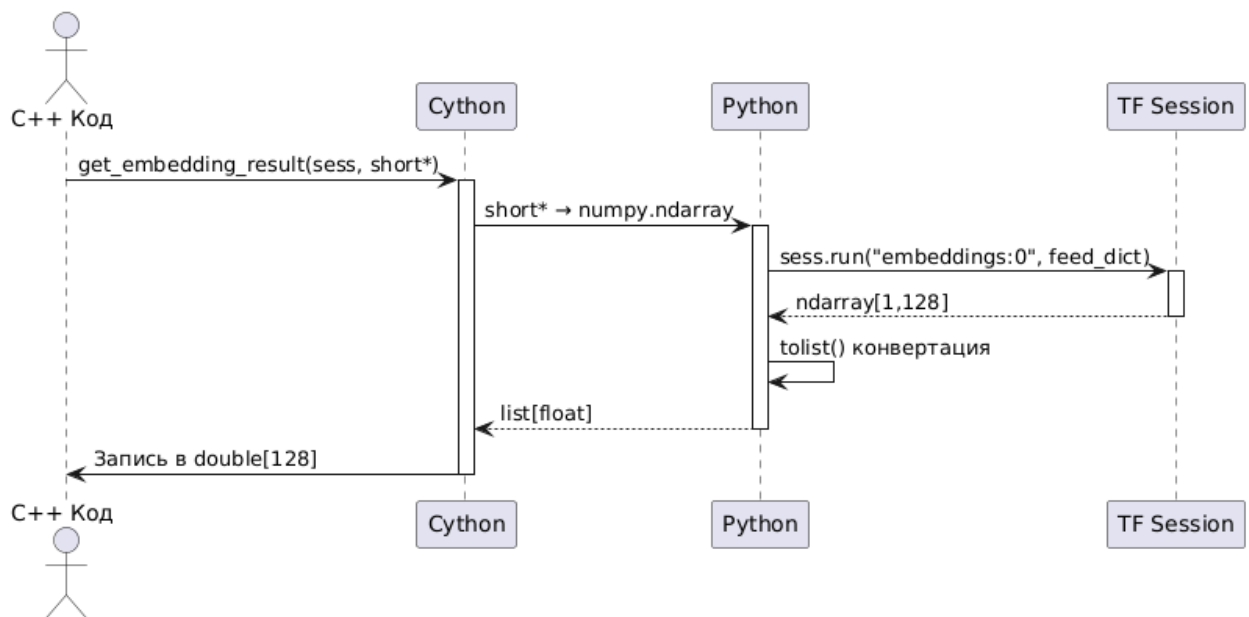


Рисунок 29 – Диаграмма последовательности вызовов

Шаги выполнения:

- C++ вызывает Cython-функцию;
- преобразование данных;
- вычисление в TensorFlow;
- обратное преобразование;
- запись результата в C++.

Спецификация данных в виде форматов передачи представлена в таблице 5.

Таблица 5 – Спецификация данных в виде форматов передачи

Этап	Тип данных	Размер	Описание
Вход C++	cv::Mat	160x160x3	BGR (OpenCV)
В Python	numpy.ndarray	[1,160,160,3]	RGB нормализованный
Выход	double [128]	1024 байт	Вектор признаков

Особенности реализации оптимизации:

- единоразовая загрузка модели;
- кэширование сессии.

Обработка ошибок:

- проверка входов;
- захват Python-исключений.

Эта архитектура обеспечивает:

- сохранение производительности C++;
- гибкость Python-экосистемы;
- безопасность типов через Cython;
- простое обновление моделей TF.

3.2 Пример фактического тестирования программного продукта

3.2.1 Функции интерфейса

Основная функция интерфейса – инициализация операционной среды Python системы, загрузка модели FaceNet и отображение входа в модуль сбора информации и модуль мониторинга в реальном времени.

Функция `OnInitDialog` используется для выполнения простой, трудоемкой работы по инициализации. Если в этой функции выполняется трудоемкая работа по инициализации, окно зависает перед вызовом функции рисования. Компромиссный вариант перенести трудоемкую работу по инициализации. Работаем над `OnPaint ()` трудоемкая работа по инициализации выполняется после отрисовки всех окон. В этой теме, хотя трудоемкая работа по инициализации выполняется в отдельном потоке, после тестирования было обнаружено, что если поток запускается в функцию `OnInitDialog`, это также может привести к сбою окна. Тело не может быть правильно нарисовано.

Функция `OnPaint`: запустите дополнительный поток инициализации в конце этой функции, чтобы гарантировать выполнение трудоемкой работы по инициализации после отрисовки формы. Следует отметить, если пользователь перетаскивает окно или меняет размер окна, это происходит. функция также будет вызвана, чтобы избежать повторного выполнения инициализации системы. В системе переменная типа `BOOL` используется для обозначения того, завершена ли инициализация. Только если она не завершена, будут запущены дополнительные потоки инициализации. После завершения инициализации переменная типа `BOOL` будет немедленно инвертирована. избегать последующих запусков дополнительных потоков инициализации во время рисования.

Ниже на рисунке 30 приведен дополнительный код для инициализации функции рабочего потока.

```
21  UINT InitPT(LPVOID lpParam) //Вспомогательная функция
22  {
23      //1.Проверьте, существует ли папка Portraits, если она не существует, создайте ее
24      если(!PathIsDirectory(_T("Портреты")))
25      {
26          Начальный уровень = 1;
27          Sleep(InitPT_DECAY);
28          если (!CreateDirectory(_T("Портреты"),NULL))
29          {
30              AfxMessageBox(_T("ОШИБКА: не удалось создать папку Portraits!"));
31              InitLevel = -1;
32              вернуть 0;
33          }
34      }
35      //2.Проверьте, не существует ли файл models, если он не существует, сообщите об ошибке и выйдите из
36      программы //Примечание: Здесь предполагается, что раз папка models существует, все файлы внутри нее также существуют
37      нормально. //Поскольку неясно, есть ли зависимости от файлов внутри папки price, а файлов много, это неудобно.
38      //Проверить!Таким образом, файлы внутри папки здесь не проверяются
39      если(!PathIsDirectory(_T("модели")))
40      {
41          AfxMessageBox(_T("ОШИБКА: папка models не существует!"));
42          InitLevel = -1;
43          вернуть 0;
44      }
45      //3.Инициализируйте среду Python
46      InitLevel = 2;
47      TheApp.TF.InitPythonEnv();
48      //4. Загрузить модель FaceNet Tensorflow
49      Начальный уровень = 3;
50      часы с таймером();
51      TheApp.TF.LoadTensorflowModel();
52      timer = clock()-таймер;
53      Начальный уровень = 4;
54      вернуть 0;
55  }
```

Рисунок 30 – Код для инициализации функции рабочего потока

Функция OnTimer: система запускается с SetTimer программы и вызывает эту функцию каждый короткий период времени. Ее важной функцией является постоянный запрос значения InitLevel и отображение прогресса инициализации пользователю на основе этого значения.

В модуль функции сбора информации, нажав кнопку сбора информации в главном интерфейсе. Его функция заключается в сборе информации о символах и записи ее в базу данных. Блок-схема алгоритма запуска представлена на рисунке 31.

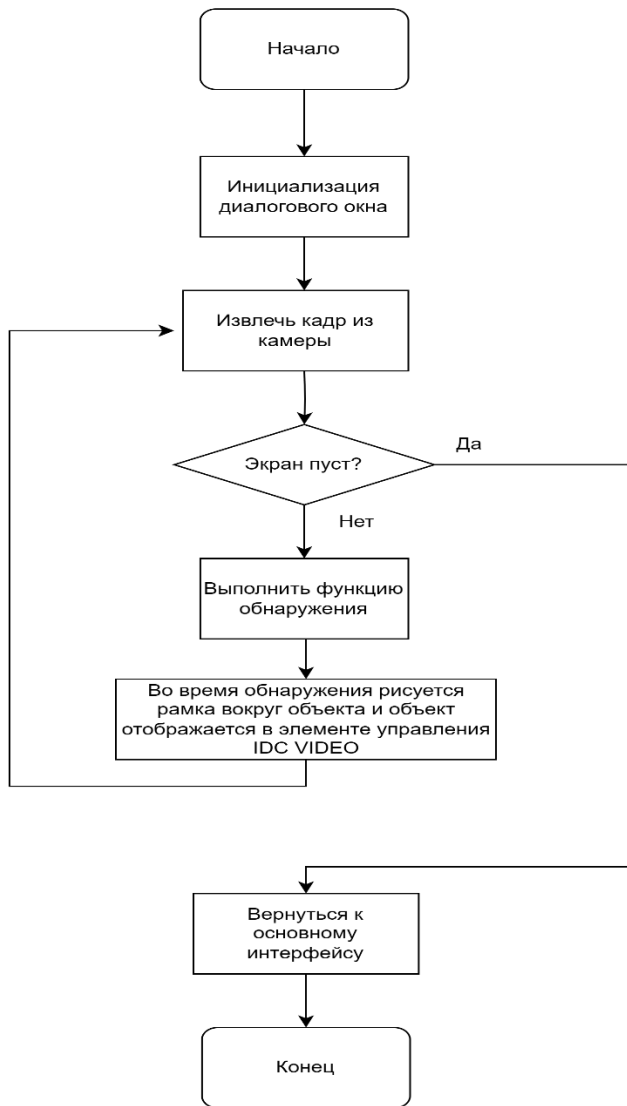


Рисунок 31 – Блок-схема алгоритма запуска

Модуль сбора информации реализован с использованием диалогового класса, сгенерированного MFC, а функции ответа на сообщение реализованы как функции-члены этого класса.

Блок-схема обработки функции ответа на сообщение кнопки захвата CInfoCollectDlg:OnBnClickedButtonCaptrue() показана на рисунке 32.

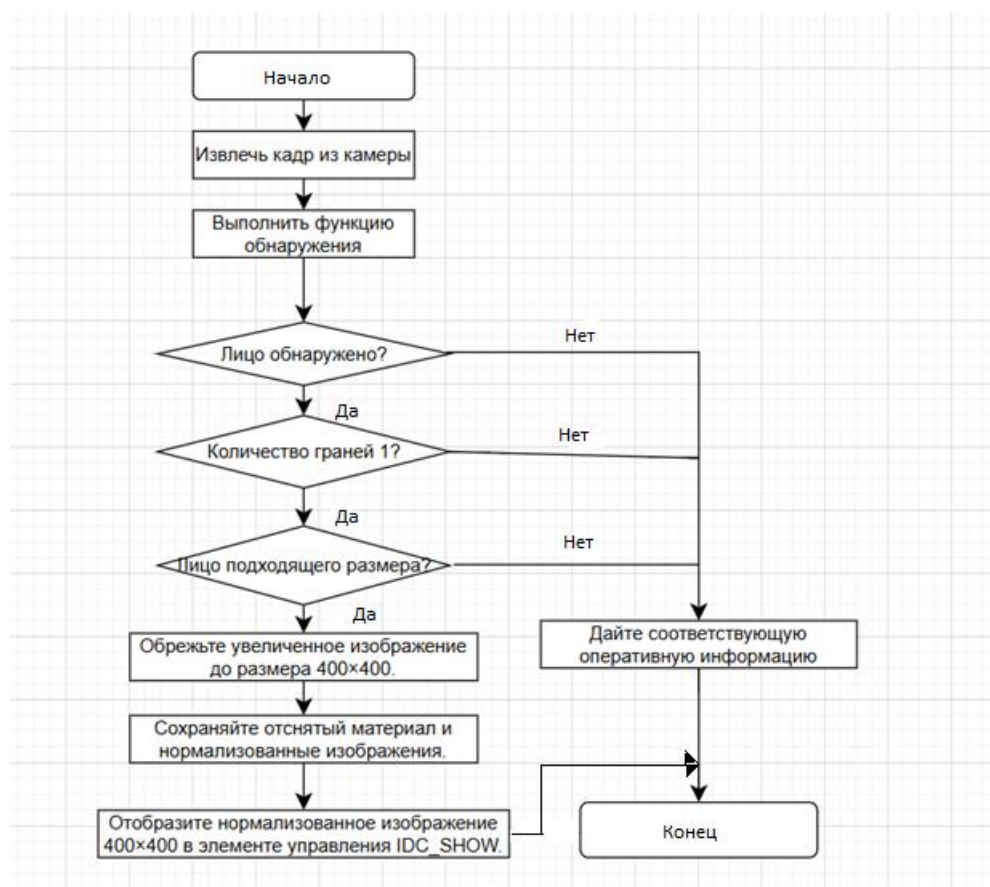


Рисунок 32 – Блок-схема функции ответа на сообщение
Обработка функции ответа на сообщение срабатывает при нажатии кнопки CInfoCollectDlg:OnBnClickedButtonComfier().

Блок-схема обработки функции ответа на сообщение кнопки показана на рисунке 33.

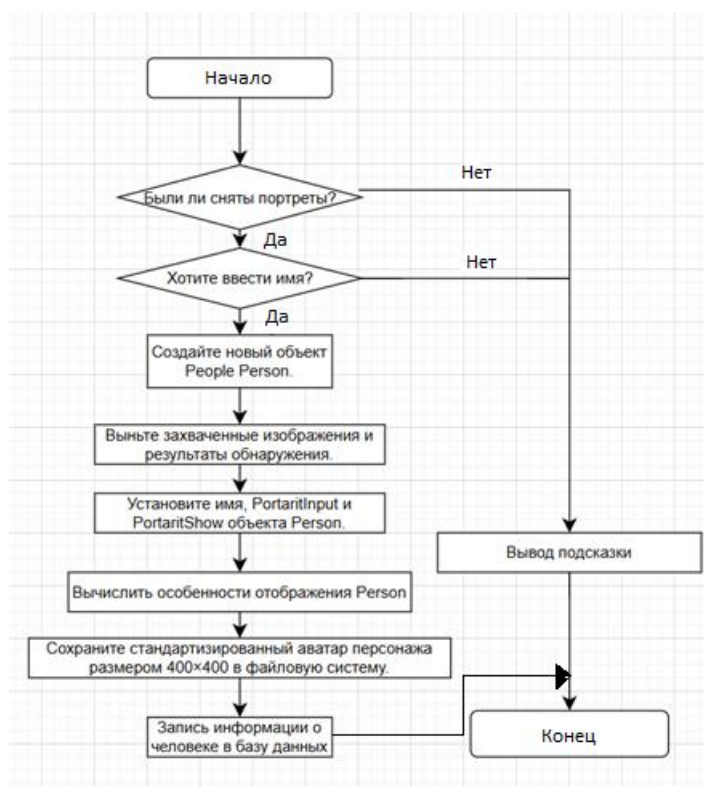


Рисунок 33 – Блок-схема обработки функции ответа на сообщение кнопки

Основной функцией модуля мониторинга в реальном времени является идентификация и запись идентификационной информации людей, появляющихся на экране камеры. Этот модуль имеет огромные проблемы, и его трудно удовлетворить требованиям реального времени. Основная причина заключается в том, что отдел распознавания использует модель глубокой нейронной сети. Преимуществами этого решения являются высокая точность распознавания и высокая устойчивость к освещению и положению. Однако его недостатком является то, что объем вычислений относительно велик, и каждое распознавание занимает от 0,3 до 0,5 секунды. Такая скорость в основном может соответствовать производительности в реальном времени, но ее нельзя обеспечить плавностью, видимой невооруженным глазом (15 кадров в секунду или 15 кадров в секунду). Поэтому при разработке проекта не всегда в реальном времени отображается процесс распознавания, а только процесс обнаружения.

Мониторинг одного потока пользовательского интерфейса. Этот метод

помещает все процессы отображения, обнаружения и распознавания в поток пользовательского интерфейса. Преимущество этого метода заключается в том, что результаты обнаружения и распознавания могут отображаться на экране в режиме реального времени. Это приведет к тому, что экран будет сильно тормозить и может достигать только 2–5 кадров в секунду, и это при том, что размер соответствующей базы данных очень мал.

Один поток пользовательского интерфейса, один рабочий поток, этот метод помещает отображение мониторинга в поток пользовательского интерфейса, помещает обработку обнаружения и идентификации в отдельный рабочий поток и сохраняет результаты идентификации в общей области памяти типа списка, в которой хранятся информация об именах идентифицированных людей. Поток пользовательского интерфейса извлекает информацию об имени и отображает ее в форме. Этот метод может сделать отображение мониторинга более плавным, но на самом деле он не повышает эффективность распознавания. Работа общей области памяти типа List уже является минималистской версией рабочей модели производителя.

Один поток пользовательского интерфейса, два рабочих потока, этот метод помещает отображение мониторинга в поток пользовательского интерфейса, а обработку обнаружения и распознавания помещает в два рабочих потока соответственно. Это связано с проблемой синхронизации трех потоков. Этот метод реализует две модели производитель-потребитель. В первой модели поток обнаружения (также называемый выборкой) является производителем, а поток идентификации потребителем. Во второй модели поток идентификации это поток пользовательского интерфейса. Следует отметить, что этот метод на самом деле не ускоряет одиночное распознавание, но время можно полностью использовать, используя объект List <People>, общий для частей обнаружения и распознавания в качестве буфера. Общая эффективность распознавания составляет. именно исходя из того, что в реальных ситуациях картинка не Лица появляются не всегда. Между появлением лиц проходит определенный интервал времени. Это решение эффективно использует время, когда лица не появляются на изображении, для

повышения эффективности. Буфер можно сохранить на изображении, когда их большое количество. лиц или при отборе пробы. Когда интервал мал, с идентифицированной частью невозможно справиться вовремя. После обработки информации о лицах, как только лица на изображении не появятся, поток обнаружения не будет добавлять людей в буфер, а поток распознавания завершит работу, которая не могла быть завершена вовремя до этого разрыва, поэтому ее можно сохранить для максимальный размер. Для актуальной информации на сайте система использует это решение по умолчанию.

3.2.2 Структура базы данных и пользовательский интерфейс

Чтобы упростить структуру базы данных, предполагая, что все символы не имеют одинаковых имен, структура таблицы «Люди», как показано на рисунке 34.

列名	数据类型	允许 Null 值
Name	varchar(50)	☐
Embedding	varchar(MAX)	☒
PortraitPath	varchar(50)	☒

Рисунок 34 – Структурная диаграмма таблицы «Люди»

Основное окно интерфейса в процессе инициализации показано на рисунке 35.

Пока не произошла инициализация всех необходимых компонентов доступна только кнопка «Инициализация»

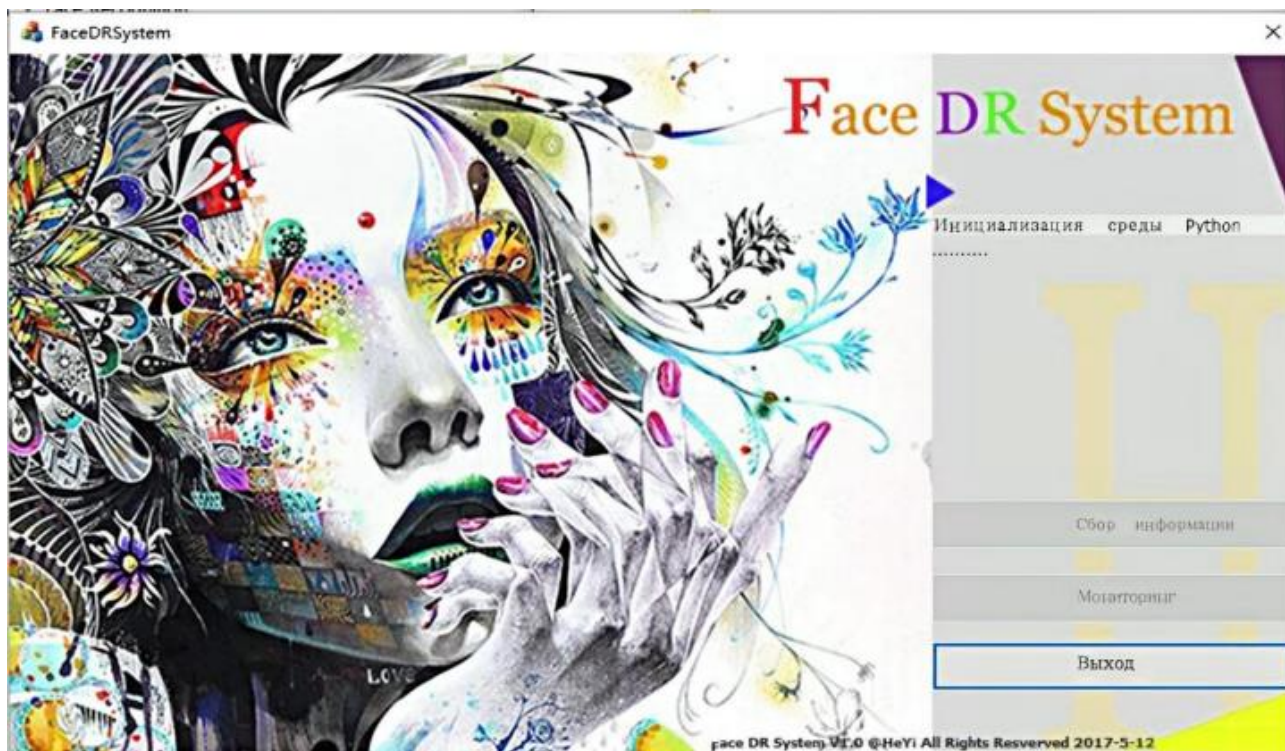


Рисунок 35 – Основное окно интерфейса инициализируется

После инициализации становятся доступны кнопки «Сбор информации» и «Мониторинг в реальном времени».

Основное окно после инициализации представлено на рисунке 36.



Рисунок 36 – Основное окно интерфейса после инициализации

На рисунке 37 показано изображение после нажатия кнопки захвата.

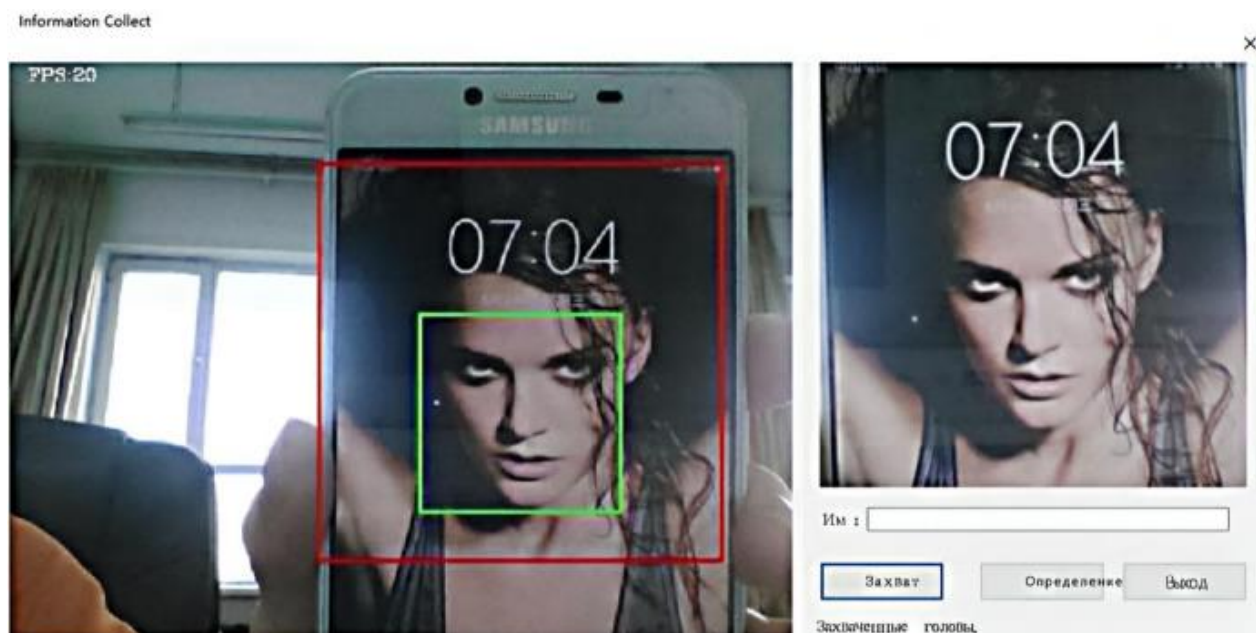


Рисунок 37 – Изображение после нажатия кнопки захвата

После нажатия кнопки ОК (персонаж на картинке известная певица ToveLo).

Информация добавлена в базу данных, как показано в таблице 6.

Таблица 6 – Информация, добавленная в базу данных

Name	Embedding	PortraitPath
ToveLo	-0.047241397202014923,- 0.077077955...	Portraits//ToveLo.jpg
HeYi	- 0.11537960171699524,0.0241924785...	Portraits//heyijpg
Ван Цзяньюй	-0.066199786961078644,- 0.136705607.	Portraits//Ван Цзяньюй

При идентификации ToveLo появляется запись в ListBox. Окно интерфейса при идентификации показано на рисунке 38.

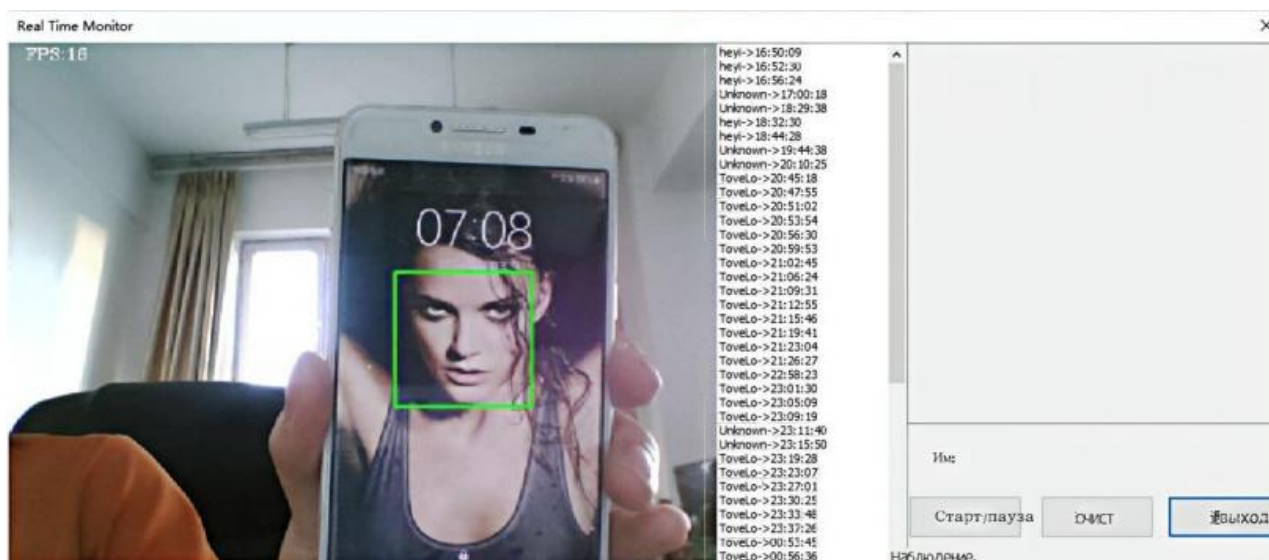


Рисунок 38 – Окно интерфейса при идентификации

После двойного щелчка по имени человека в ListBox элемент управления справа отобразит портретное изображение, собранное в ходе сбора информации.

Все собранные данные об изображении показаны на рисунке 39.

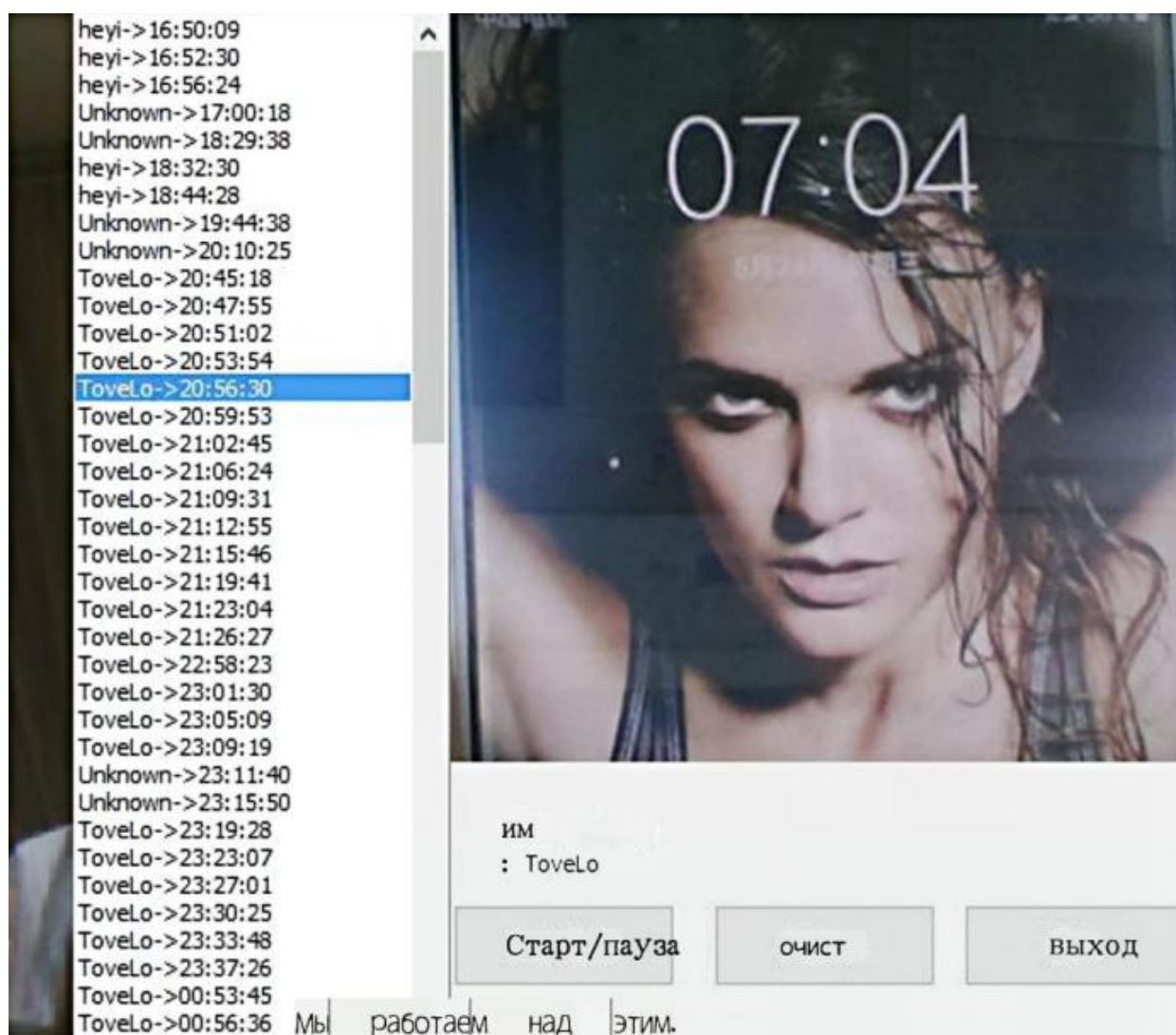


Рисунок 39 – Информация, собранная в ходе сбора информации

Таким образом, приложение успешно решает поставленные задачи распознавания и учета лиц, предоставляя надежную основу для дальнейшего развития системы.

ЗАКЛЮЧЕНИЕ

В ходе проведения исследования выполнен анализ существующих технологий сбора и распознавания лиц, выполнены: разработка аппаратной архитектуры и системы распознавания лиц с высокой точностью распознавания при минимальных затратах вычислительных ресурсов.

Использование гибридного подхода на основе языков C++ и Python в сочетании с библиотекой TensorFlow и предобученной моделью FaceNet позволило обеспечить высокую точность распознавания при одновременной оптимизации вычислительных ресурсов. Реализация многопоточной архитектуры и грамотное распределение задач между пользовательским интерфейсом и рабочими потоками способствовали стабильной и эффективной работе системы в реальных условиях.

По результатам работы опубликовано 2 статьи в научном журнале «Флагман науки» (№11, №12, 2024).

Проведенное тестирование позволило оптимизировать систему для работы в реальных условиях.

Разработанная система обеспечивает автоматическое определение личности по изображению лица, фиксацию времени появления объекта в кадре.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

- 1 Абрамов, А. А. Введение в компьютерное зрение: основы и практическое применение / А. А. Абрамов. – М.: ДМК Пресс, 2021. – 304 с.
- 2 Астахов, А. Н. Основы проектирования программного обеспечения: Учебное пособие / А. Н. Астахов. – М.: Форум, 2020. – 368 с.
- 3 Бондаренко, В. А. Технологии программирования: Учебник / В. А. Бондаренко. – СПб.: Питер, 2022. – 432 с.
- 4 Ван Лэй. Обеспечение безопасности биометрических данных при распознавании лиц // Информационная безопасность. – 2021. – №10.
- 5 Ван Чжэн. Методы повышения точности распознавания лиц с использованием нейросетей // Искусственный интеллект. – 2023. – №4.
- 6 Гао Чэн. Интеграция системы распознавания лиц в пропускные системы предприятий // Автоматизация процессов. – 2023. – №2.
- 7 Го Фэн. Интеграция системы видеонаблюдения с функцией распознавания лиц // Информационная безопасность. – 2023. – №5.
- 8 Горшков, И. В. Искусственный интеллект и машинное обучение: от теории к практике / И. В. Горшков. – М.: КноРус, 2023. – 352 с.
- 9 Ковалёв, С. В. Методы и алгоритмы распознавания образов: Учебное пособие / С. В. Ковалёв. – М.: Инфра-М, 2019. – 278 с.
- 10 Кузнецов, С. О. Алгоритмы компьютерного зрения: учебник и практикум / С. О. Кузнецов. – М.: Юрайт, 2021. – 390 с.
- 11 Лебедев, В. П. Информационные технологии в биометрических системах / В. П. Лебедев. – М.: Радио и связь, 2020. – 296 с.
- 12 Ли Жуй. Методы оптимизации нейросетевых моделей под устройства с низкой вычислительной мощностью // Встраиваемые системы. – 2022. – №9.
- 13 Ли Тянь. Разработка логики сравнения лицевых шаблонов в реальном времени // Цифровые системы. – 2022. – №3.
- 14 Минаев, Ю. В. Архитектура программных систем: учебник / Ю. В. Минаев. – СПб.: Питер, 2021. – 408 с.

- 15 Морозов, А. С. Технологии разработки программного обеспечения: Учебное пособие / А. С. Морозов. – М.: Юрайт, 2022. – 318 с.
- 16 Новиков, Д. А. Биометрические технологии: теория и практика / Д. А. Новиков. – М.: Горячая линия – Телеком, 2019. – 256 с.
- 17 Павлов, А. Н. Программная инженерия: теория и практика проектирования ПО / А. Н. Павлов. – М.: Академия, 2020. – 336 с.
- 18 Поляков, И. В. Машинное обучение и анализ данных: основы и практика / И. В. Поляков. – М.: ДМК Пресс, 2021. – 420 с.
- 19 Рудаков, В. В. Проектирование информационных систем: Учебное пособие / В. В. Рудаков. – М.: КноРус, 2022. – 296 с.
- 20 Сергеева, Н. И. Современные технологии программирования: Учебник / Н. И. Сергеева. – СПб.: БХВ-Петербург, 2020. – 368 с.
- 21 Смирнов, В. Г. Системы компьютерного зрения и их применение / В. Г. Смирнов. – СПб.: Питер, 2023. – 310 с.
- 22 Соловьев, А. А. Основы биометрической идентификации / А. А. Соловьев. – М.: Горячая линия – Телеком, 2021. – 244 с.
- 23 Сунь Хуа. Особенности предобученных моделей в задачах распознавания лиц // Нейрокомпьютерные системы. – 2022. – №9.
- 24 Сюань Лун. Особенности работы с потоками видеоданных в OpenCV // Видеотехнологии. – 2022. – №7.
- 25 Тан Юн. Использование SQLite в проектах на Python // Системное программирование. – 2022. – №4.
- 26 Тарасов, И. Н. Системы искусственного интеллекта: учебное пособие / И. Н. Тарасов. – М.: Форум, 2019. – 280 с.
- 27 Фан Юн. Сравнительный анализ моделей распознавания лиц: FaceNet, DeepFace, Dlib // Технологии машинного обучения. – 2023. – №3.
- 28 Федоров, В. П. Интеллектуальные системы: распознавание и принятие решений / В. П. Федоров. – М.: Академия, 2022. – 336 с.
- 29 Хе Юн. Практическое использование фреймворка TensorFlow для обучения моделей распознавания лиц // Нейронные сети. – 2021. – №5.

- 30 Хромов, С. Н. Методы и средства программной инженерии: учебник / С. Н. Хромов. – М.: Юрайт, 2020. – 392 с.
- 31 Ху Мин. Практика использования Face Recognition API в задачах идентификации // Прикладные аспекты ИИ. – 2023. – №8.
- 32 Хуэй Лэй. Обеспечение отказоустойчивости в системах с непрерывным видеопотоком // Информационные системы. – 2022. – №8.
- 33 Цай Чжун. Принципы работы с OpenCV: фильтрация, обнаружение, распознавание // Компьютерное зрение. – 2023. – №5.
- 34 Цзинь Вэй. Применение алгоритма PCA в задаче уменьшения размерности признаков лица // Математические методы в ИТ. – 2021. – №8.
- 35 Чернышев, Д. А. Практика применения нейросетей в задачах компьютерного зрения / Д. А. Чернышев. – СПб.: Питер, 2023. – 274 с.
- 36 Чжан Юн. Сравнительный анализ моделей распознавания лиц: FaceNet, DeepFace, Dlib // Технологии машинного обучения. – 2023. – №3.
- 37 Чжоу Мэй. Внедрение многопоточности в системы реального времени на C++ // Программная инженерия. – 2022. – №7.
- 38 Чэнь Цзюнь. Применение REST API в ASP.Net для e-commerce решений // Электронная коммерция. – 2021. – №4.
- 39 Шевченко, Р. Л. Архитектура информационных систем: Учебник / Р. Л. Шевченко. – М.: КноРус, 2021. – 312 с.
- 40 Щукин, А. В. Базы данных: проектирование и реализация в современных СУБД / А. В. Щукин. – М.: ДМК Пресс, 2022. – 384 с.
- 41 Ян Шуан. Построение REST API для интеграции системы распознавания лиц // Веб-программирование. – 2022. – №11.
- 42 Амурский государственный университет [Электронный ресурс]. – Режим доступа: <https://amursu.ru/>. – 03.03.2025.
- 43 CodinGame [Электронный ресурс]. – Режим доступа: <https://codingame.com/>. – 15.04.2025.
- 44 Metanit [Электронный ресурс]. – Режим доступа: <https://metanit.com/>. – 22.02.2025.