МИНИСТЕРСТВО ПО РАЗВИТИЮ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И КОММУНИКАЦИЙ РЕСПУБЛИКИ УЗБЕКИСТАН

ТАШКЕНТСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ имени МУХАММАДА АЛ-ХОРАЗМИ

ФАКУЛЬТЕТ ТЕЛЕВИЗИОННЫЕ ТЕХНОЛОГИИ

КАФЕДРА «АУДИОВИЗУАЛЬНЫЕ ТЕХНОЛОГИИ»

МЕТОДИЧЕСКОЕ ПОСОБИЕ ПО ВЫПОЛНЕНИЮ КУРСОВОЙ РАБОТЫ ПО ПРЕДМЕТУ «ОБРАБОТКА ЦИФРОВОЙ ИНФОРМАЦИИ»

Ташкент 2019

Автор: Касымова Ш.Т

«Методическое пособие по выполнению курсовой работы по предмету Обработка цифровой информации»/ ТУИТ. 82 с. Ташкент, 2019.

Методическое пособие предназначено для проведения курсовых работ в первом семестре 4-го курса.

В первом семестре четвертого курса студенты должны изучить методы обработки цифровой информации. С учетом того, что возможна обработка числовой, текстовой, графической, аудио-видео информации, темами курсовой работы могут быть рассмотрены снятие и монтаж видеороликов, создание и визуализация 3D моделей, создание и обработка информационной системы в среде C++ Builder, корректировка базы данных в этой среде, разработка компьютерной игры, разработка веб сайта.

В представляемом методическом пособии примерная работа посвящена определённой теме, снабжена методическими указаниями и примерами выполнения, что поможет, самостоятельно усвоить прорабатываемый материал. Кроме того, в методическом пособии содержатся примерные темы заданий курсовой работы.

Методическое пособие было рассмотрено на заседании учебно-методического Совета ТУИТ имени Мухаммада ал-Хорезми №4(116) от 23.10.2018 года, одобрено и рекомендовано к тиражированию.

ВВЕДЕНИЕ

Курсовая работа является одной из форм учебной деятельности, которая выполняется студентом самостоятельно под руководством преподавателя. Курсовая работа представляет собой учебно- исследовательскую деятельность, требующую от студентов освоения элементов научного исследования. Выполнение курсовой работы направлено на формирование у студентов способности самостоятельно мыслить, анализировать и сопоставлять факты, обобщать и логически излагать материал. В результате выполнения курсовой работы у студентов формируется субъективно новое знание по одной из частных проблем.

В ходе работы у студента развивается научная наблюдательность, студент учится не только находить необходимую информацию, но и корректно ее использовать в своем исследовании, грамотно демонстрировать, как и откуда были получены те или иные сведения, и каково их значение для данного исследования.

Курсовая работа способствует формированию у студентов опыта самостоятельного научного творчества, повышению уровня теоретической и профессиональной подготовки, лучшему усвоению учебного материала.

При написании работы студент должен показывать практические навыки работы с персональным компьютером, анализировать литературные данные, делать обоснованные выводы и предложения.

Курсовые работы способствуют закреплению, углублению, обобщению и прикладному применению знаний и умений, формируемых студентами при изучении курса «Обработка цифровой информации».

Во время подготовки курсовой работы перед студентом не стоит задача открыть новые научные положения в области обработки цифровой информации. В процессе изложения темы, студенту необходимо показать способность научно использовать литературу, понимать методологию

изложения материала, уметь систематизировать данные, обрабатывать фактический материал, делать обобщения и выводы, увязывать теорию с практикой и современной действительностью.

1. РЕКОМЕНДАЦИИ ПО ПОДГОТОВКЕ КУРСОВОЙ РАБОТЫ

Темы курсовых работ предлагаются преподавателем. По согласованию с руководителем студент может уточнить формулировку предлагаемой темы или предложить собственную тему, обосновав целесообразность исследования.

После утверждения темы курсовой работы и изучения литературы, рекомендованной преподавателем, определяется направление исследований, его цель и задачи. Затем студент самостоятельно подбирает дополнительные источники информации (книги, периодические издания, электронные ресурсы), которые планируется использовать при выполнении исследования, разрабатывает структуру содержания курсовой работы. Составленный список литературы и план курсовой необходимо согласовать с преподавателем.

Выполнение курсовой работы включает в себя изучение теоретического материала, рассмотрение и оценку возможных решений, подбор методов исследования, сбор, анализ и обобщение собственного материала, разработку программного продукта (обеспечения), написание текста, тестирование и отладку программы, анализ результатов, формулировку комментариев и выводов.

2. ТРЕБОВАНИЯ К СОДЕРЖАНИЮ КУРСОВОЙ РАБОТЫ

Результат учебно-исследовательской деятельности во многом зависит от понимания студентом основных характеристик научного исследования и их формулировок. К основным характеристикам исследования относятся:

актуальность, проблема, объект, предмет, основная цель, частные задачи и методы исследования.

Большинство тем курсовых работ являются своевременными и актуальными. Если тематика курсовой работы актуальна, то изложение следует начинать с описания *актуальности,* которая определяется необходимостью проведения исследования в современных условиях. В содержании курсовой работы обязательно указывается *проблема исследования*, характеризующая то, что надо изучить из того, что ранее не было изучено.

С проблемой исследования тесно связаны объект и предмет исследования. Их формулировки также обязательно приводятся в содержании курсовой работы.

Под *объектом исследования* понимают часть объективной реальности, которая изучается в процессе теоретической и практической деятельности. *Предметом исследования* считают свойства, отношения объекта, исследуемые в процессе практической деятельности с определенной целью в данных условиях и обстоятельствах. Поэтому объект и предмет исследования как категории научного познания соотносятся между собой как общее и частное. В объекте выделяется та его часть, которая служит предметом исследования. Необходимо четко представлять границы исследования и предполагаемые результаты.

Цель исследования состоит в том, чтобы разрешить поставленную проблему, достичь определенный результат. При формулировке цели исследования обычно используются следующие термины: анализ, выявление, внедрение, изучение, развитие, разработка и т.д.

В зависимости от цели курсовой работы необходимо сформулировать две-три конкретные *задачи исследования*, которые необходимо решить для достижения цели. Это обычно делается в форме перечисления: *изучить* ..., *описать* ..., *установить* ..., *выявить* ..., *вывести* ..., *разработать* ... и т.п. Формулировку задач необходимо выполнить тщательно, так как описание хода и результатов их решения составит основное содержание курсовой работы.

3. СТРУКТУРА И СОДЕРЖАНИЕ КУРСОВОЙ РАБОТЫ

Титульный лист

Аннотация

Задание

Оглавление

Введение

1. Теоретическая часть.

    1.1. Анализ и исследование задачи, построение модели.

    1.2. Алгоритмы, обеспечивающие функциональность приложения.

2. Основная часть

    2.1. Цели и задачи курсовой работы.

    2.2. Руководство пользователю.

    2.3. Руководство программисту.

3. Тестирование программы

    3.1. Оценка результатов проведения тестирования

Заключение.

Список литературы.

Приложения.

4. СОДЕРЖАТЕЛЬНОЕ НАПОЛНЕНИЕ РАЗДЕЛОВ ПОЯСНИТЕЛЬНОЙ ЗАПИСКИ

Аннотация

Содержит очень краткое содержание работы, число страниц пояснительной записки, число рисунков, таблиц, приложений.

Задание

Основным документом, в соответствии с которым выполняется разработка некоторого проекта в любой отрасли, включая проекты по разработке программного обеспечения (ПО), является задание.

При разработке ПО задание – документ (спецификация), оговаривающий перечень требований к системе и утверждённый как заказчиком/пользователем, так и исполнителем/производителем системы.

Пример задания приведен в приложении.

Введение

Введение должно содержать общие сведения по теме курсовой работы. Так, если в основе работы лежат информационные системы, то требуется дать информацию о информационных системах (что это, зачем, особенности и т.д.). Если речь идет о компиляторах, то сведения о компиляторах и т.д.

Во введении также необходимо отразить:

- актуальность выбранной темы (например, сказать, что в современном обществе или в деятельности любого современного предприятия информация является одним из важнейших ресурсов, выделяясь в самостоятельный фактор для принятия решений, и от средств ее обработки во многом зависит эффективность принятия решений);
- используемые модели программирования (например, императивное программирование, модульное программирование, структурное программирование, объектно-ориентированное программирование, основанное на классах,…)
- практическую значимость полученных результатов (где можно использовать);
- какого рода ресурсы необходимы для реализации (ПК, программное обеспечение…уточнить какое);
- перспективы совершенствования изготавливаемого программного продукта.

Цели и задачи курсовой работы

Составляющие элементы этого раздела могут быть следующие:

- формулировка условия задачи;
- сбор информации о задаче и выделение физического объекта;

- определение конечных целей решения задачи;
- определение задач, т.е форм выдачи результатов;
- описание данных (их типов, диапазонов величин, структуры и т.п.).

Формулировка условия задачи. В качестве примера для объяснения хода и логической основы некоторых элементов выполнения работы предлагается к рассмотрению следующая постановка:

Электронный учебник *— это специальное устройство либо программное обеспечение, используемое в образовательном процессе и заменяющее собой традиционный бумажный учебник. В настоящее время трактовка словосочетания «электронный учебник» очень широка: в некоторых случаях под ним подразумевается электронная версия бумажного учебника, в некоторых — сложный комплекс программ на электронных устройствах, позволяющий демонстрировать ученикам, помимо текста, обучающий мультимедийный материал, содержащий в себе также интерактивные блоки проверки знаний, обновляющийся из централизованного источника и так далее.*

В электронном учебном пособии размещаются тексты лекций, задания лабораторных работ, описание лабораторных работ, тесты. Необходимо иметь ввиду, что в соответствии с требованиями пользователя разрабатывается приложение (информационная система) позволяющее изучение необходимой темы, ввод, корректировку и отображение информации.

3D-моделирование описывает порядок создания 3D модели (или каркасной модели ввиде трехмерного объекта). Трехмерная модель создается с помощью множества точек связанных с линиями и непрямыми поверхностями. В настоящее время расширяются области применяемые трехмерное моделирование:игры-моделирование реалистических персонажей, медицина-модели органов человеческого тела, инженерия-транспортные средства, киноматография-создание виртуальных персонажей и различных специальных эффектов. Также. 3D моделирование успешно используется в рекламных продуктах.

Этапы создания видеороликов можно условно разделить на три группы: пре-продакшн, продакшн и пост-продакшн. В зависимости от вида, цели и сложности видеоролика, некоторые этапы можно пропускать или упрощать. Пре-продакшн включает в себя исследование рынка, разработка идеологии, написание сценария, подготовка к съемкам. Этап продакшн включает в себя подготовку к видеосъемкам на месте и непосредственно съемку. Постпродакшн-собственно создание ролика. На этом этапе осуществляется монтаж, озвучивание, наложение спецэффектов и другие необходимые действия.

Сбор информации о задаче. Следует определиться, с какими данными разработчик проекта имеет дело. Так как в задаче речь идет о товаре, размещенном на складе, нужно указать, какой товар (перечислить его: кирпич, цемент, гвозди...), что связано с поставкой/не поставкой (документы на заказ и на доставку). Выделить физический объект (множество товаров) и определить его свойства. В том числе, количество элементов—единиц товара. А т.к. в данном случае физический объект определен как множество, то следует выделить свойства отдельного представителя из этого множества. При этом необходимо выявить самые существенные свойства, необходимые для решения задачи. Выделив наиболее важные факторы, можно пренебречь менее существенными. Параметрами отдельного представителя (единицы товара), например, могут быть: наименование, цена, фирма-изготовитель, поставщик, количество, ...

Определения конечных целей решения задачи. В соответствии с условием данной задачи разрабатываемая программа должна предоставлять пользователю хранить информацию о товарах и формировать из нее нужную информацию по его требованию. Такими требованиями могут быть: получить список товаров заданного наименования с его характеристиками (какими?), список поставщиков, ...

Определение формы выдачи результатов. Например, представить список поставщиков в виде таблицы, в которой можно было бы увидеть наименование поставляемого им товара …

Описание данных (их типов, диапазонов величин, структуры и т.п.). Например, наименование товара представляется строкой символов, количество символов не превышает 20. Цена указывается в сумах, может быть представлена вещественным числом, диапазон ценовых колебаний в промежутке от 10000 до 500. ….

Анализ и исследование задачи, построение модели. Составляющими данного параграфа могут быть следующие элементы:

- Выделение математического объекта;
- Анализ существующих аналогов;
- Анализ технических и программных средств;
- Разработка математической модели;
- Разработка требований к приложению.

Выделение математического объекта. Сказать, например, что для того, чтобы иметь возможность хранить и обрабатывать данные о физическом объекте, эти данные нужно представить в виде, приспособленном для обработки математическими методами. Для этого нужно перейти от физического объекта к объекту математическому. В нашем случае нужно перейти от физического объекта — множество товаров, где каждый товар имеет свои физические характеристики, к математическому объекту, описывающему это множество. Причем, каждый элемент математического множества должен быть представлен эквивалентными свойствами элемента физического множества (имеет структуру из свойств).

Далее можно сказать о том, что для описания математического множества можно воспользоваться таким математическим объектом, как «массив». В данном случае это будет массив структур.

Анализ существующих аналогов. Сказать, что хранить и обрабатывать массив структур описанных данных можно разными способами, например, средствами базы данных. Привести примеры известных баз, например, Access (особенности…), в Excel (особенности…).

Анализ технических и программных средств. ПК – техническое средство. Объяснить, почему выбрана среда C++ Builder, а не Excel или др. (.. возможности ООП и визуального….).

Разработка математической модели. Под математической моделью понимают систему математических соотношений — формул, уравнений, неравенств и т.д., отражающих существенные свойства объекта. Метод математического моделирования сводит исследование поведения объекта или его свойств к математическим задачам. Следует выписать формулы, например с использованием знаков суммирования…

При этом можно пользоваться и такой схемой действий:

1. выделить предположения, на которых будет основываться математическая модель. Например, предположить, что информация о товаре будет сосредоточена в структурах данных (или содержатся в файле, или в таблице, или…);

2. определить, что считать исходными данными и результатами;

3. записать математические соотношения, связывающие результаты с исходными данными. Какие методы обработки (с описанием подхода к решению, например, «…. решение сводится к задаче накопления суммы элементов…формула….») данных потребуются для решения задачи?

Выведенные зависимости и формулы в общем случае могут быть представлены уравнениями, системами уравнений – линейных, интегральных, матричных, дифференциальных и др.. На этом этапе для нахождения решения также строится неформальный алгоритм (производная от длины и решение уравнения: производная равна 0, …).

При построении математических моделей далеко не всегда удается найти формулы, явно выражающие искомые величины через данные. В таких случаях

используются математические методы, позволяющие дать ответы той или иной степени точности.

Разработка требований к приложению. Например, сказать о том, что пользователь должен иметь возможность для задания исходных данных вручную, возможность для сохранения данных в файле, возможность формировать файл данных с целью повторного использования; извлекать данные из файла данных; выбирать вид отображаемой информации (например, общее количество товара, суммарная стоимость всего товара, номенклатура товара и т.д. в соответствии с требованиями); отображать нужную информацию в соответствии с выбранным видом..., а также обеспечение соединений этих данных в соответствии с пользовательской логикой (все исходя из поставленной задачи).

В первой главе-обычно теоретической-дается анализ научной и методической литературы. Здесь предлагается провести подробное исследование теоретической части курсовой работы.

Необходимо последовательно и логично рассмотреть сущность и основное содержание проблемы, изучаемых вопросов и понятий; изложить мнения различных авторов и свои умозаключения. Не следует забывать о необходимости делать ссылки на литературные источники, материал которых использовался при написании работы.

Первая глава демонстрирует общий научно-методический уровень подготовки студента, его умение подбирать и изучать литературу, систематизировать знания, делать обобщения и выявлять возможные направления решения проблемы. Содержание параграфа должно быть посвящено отдельному аспекту исследования.

Во второй главе – обычно практической – следует описать и обосновать конкретный подход к решению поставленной проблемы. В качестве проблемы в задании на курсовую студенту предлагается выполнить практическое задание – составить программу.

В разделе «Руководство пользователю» (Этапы выполнения работы) второй главы приводится полное описание работы продукта (программы) с использованием соответствующих скриншотов.

Описание раздела «Руководство программисту» второй главы включает в себя описание модулей и информационных объектов.

Тестирование программы

Тестирование созданного программного продукта следует начинать с разработки плана тестирования. При этом следует помнить, что весь процесс тестирования можно разделить на три этапа:

- Проверка в нормальных условиях. Предполагает тестирование на основе данных, которые характерны для реальных условий функционирования программы.
- Проверка в экстремальных условиях. Тестовые данные включают граничные значения области изменения входных переменных, которые должны восприниматься программой как правильные данные. Типичными примерами таких значений являются очень маленькие или очень большие числа и отсутствие данных. Еще один тип экстремальных условий — это граничные объемы данных, когда массивы состоят из слишком малого или слишком большого числа элементов.
- Проверка в исключительных ситуациях. Проводится с использованием данных, значения которых лежат за пределами допустимой области изменений.

Известно, что все программы разрабатываются в расчете на обработку какого-то ограниченного набора данных. Поэтому важно получить ответ на следующие вопросы:

- Что произойдет, если в программе, не рассчитанной на обработку отрицательных и нулевых значений переменных, в результате какой-либо ошибки придется иметь дело как раз с такими данными?
- Как будет вести себя программа, работающая с массивами, если количество их элементов превысит величину, указанную в объявлении массива?

- Что произойдет, если числа будут слишком малыми или слишком большими?

Наихудшая ситуация складывается тогда, когда *программа воспринимает неверные данные как правильные и выдает неверный, но правдоподобный результат.* Программа должна сама отвергать любые данные, которые она не в состоянии обрабатывать правильно.

Следует попытаться представить возможные типы ошибок, которые пользователь способен допустить при работе с Вашей программой и которые могут иметь неприятные для нее последствия. Нужно не забывать, что способ мышления пользователя отличается от способа мышления программиста. Если помнить об этом, то нужно предусмотреть обработку ошибок типа: отсутствие нужного файла, неправильные форматы данных и т.д. Список действий, которые могут привести к неправильному функционированию программы, довольно длинный и зависит от того, что делает приложение в данный момент времени.

Основной смысл этого этапа состоит в проверке того, насколько программный продукт в том виде, в котором он получился, соответствует требованиям, установленным в процессе согласования спецификации. Каждая функция или метод класса должны соответствовать требованиям, определенным для них на этапе спецификации.

Разработка плана тестирования - состоит из следующих шагов:

1) определение последовательности действий, которые позволяют проверить как работу отдельных методов, так и их совокупности;

2) подготовка входных данных, для которых известны результаты тестирования;

3) определение места расположения тестовых данных.

Разработка алгоритма процедуры тестирования состоит в разработке процедуры в соответствии с составленным выше планом тестирования. Эту процедуру можно представить блок-схемой с соответствующими комментариями.

Оценка результатов тестирования – содержит сравнение выходных данных работы отдельных процедур с контрольными значениями, которые получены в результате выполнения процедуры тестирования.

В заключении подводятся итоги проделанной работы, на основе теоретических выводов и данных практической главы делаются общие выводы по теме исследования. Необходимо показать, как решены задачи, привести основные результаты работы, сделать свои умозаключения о целесообразности и эффективности использования результатов исследования на практике. Выводы должны соответствовать содержанию работы, быть краткими, ясно, четко и логично сформулированными. В заключении также намечаются дальнейшие перспективы и пути исследования, возможность использования результатов проведенной учебно- исследовательской работы.

Библиография содержит перечень названий книг, статей, документов и электронных ресурсов, которые были использованы при подготовке курсовой работы и включает в себя всю литературу, на которую имеются ссылки и сноски в тексте.

В приложения помещают вспомогательные или дополнительные материалы, изложение которых необходимо для полноценного описания, проведенного исследования, но которые могут затруднить восприятие основного текста курсовой работы, сделать его трудночитаемым. В приложения следует вынести нормативные акты, требования к программным средствам, коды разработанных компьютерных программ, рисунки, фотографии, демонстрационные материалы и т.п.

5. ПРИМЕРНЫЙ ПЕРЕЧЕНЬ ТЕМ КУРСОВОЙ РАБОТЫ

Данный перечень тем носит рекомендательный характер. Студент может предложить собственную тему для исследования, но обязательно согласовать ее с преподавателем.

1. Создание видеокурса по обучению работы в 3D Max Studio.

2. Editing TV films using Adobe After Effect

3. Making virtual studio by using Green Screen

4. Создание видеокурса по обучению работы в MS Access

5. Создание видеокурса по обучению работы в Power Paint.

6. Создание видеокурса по обучению работы в Adobe Flash.

6. Передача и прием ТВ сигналов в стандарте DVB-H

7. Создание видеокурса по обучению работы в Pinacle Studio.

8. Моделирование трехмерной пространственной виртуальной студии соответствии с требованиями

9. Creating audiobook using Adobe Audition base Uzbek popular fairies tales

10. Состояние и перспектива цифровых видео камер.

11. Анализ современных методов сжатия видеоизображения.

12. Разаботка алгоритма обработки видеоинформации с помощью Xaara bazis.

13. Анилиз по технологии Blue ray при записи изображения.

14. Создание видеокурса по обучению работы в инженерной програм AutoCAD.

15. Создание программы по нелинейной видеомонтажной систе предназначенной для дистанционного обучения.

16. Создание видеокурса по обучению работы в Adobe Photoshop.

17. Produce systems of sending digital television broadcashing programs through o fiber.

18. Создание Morfing а с помощью программы Adobe After Effect.

19. Создание заставки для телевизионных программ.

20. Создание видеоролика О РЕСПУБЛИКЕ УЗБЕКИСТАН.

21. Распределительная сеть телевизионного сигнала с предоставлен дополнительной услуги "видео по запросу"

22. Создание видеокурса по "Основам спецосвещения" в программе Adobe Flash.

23. Создание детской обучающей программы на базе Андроид.

24. Особенности использования сервисной информации в стандартных системах цифрового ТВ – вещания.

25. Автоматизированная информационная система: вузы Узбекистана.

26.WEB сайт о видных ученых Узбекистана.

Тема теоретической части работы должна соответствовать методам и инструментам, используемым при разработке программы в практической части.

6. ТРЕБОВАНИЯ К СОДЕРЖАНИЮ ПРАКТИЧЕСКОЙ ЧАСТИ КУРСОВОЙ РАБОТЫ

Цель курсовой работы по дисциплине «Обработка цифровой информации» состоит в закреплении и углублении знаний и навыков, полученных при изучении дисциплины. Курсовая работа предполагает выполнение задания повышенной сложности по проектированию, разработке и тестированию программного обеспечения, а также оформлению сопутствующей документации.

В качестве среды разработки следует использовать MS Visual C++ 2010/Borland C++ Builder, Adobe Premiere Pro, Adobe After Effects, 3D MAX Studio, что позволит получить первичные навыки работы в современной и широко используемой на практике среде разработки приложений. Следует создать Win32 приложение с использованием средств быстрой разработки (RAD-средств) пользовательского интерфейса, обработчиков и т.п., что, в свою очередь, способствует более глубокому пониманию основ создания современного ПО с графическим интерфейсом.

7. ПРИМЕРНОЕ ПРАКТИЧЕСКОЕ ЗАДАНИЕ

Необходимо разработать небольшое Windows-приложение в среде визуального программирования MS Visual C++ с использованием элементов WindowsForms - программу ввода заказа в вашем любимом кафе. Для примера

вам предлагается взять кафе «Паркуша» или «Пиноккио», информация по меню этих кафе может быть взята с их официальных сайтов. Но вы можете выбрать для задания любое ваше любимое кафе.

Программа должна позволять просматривать меню кафе, с картинками блюд и подробной информацией о каждом блюде – цене, составе, весу и пищевой и энергетической ценности.

В таблице меню должна быть реализована возможность выбрать блюда и оформить заказ. После оформления на экран выводится форма заказа с общей суммой и информацией по общей пищевой и энергетической ценности заказа. Также реализовать вывод отчета с информацией о сделанных заказах за произвольный период времени (с возможностью выбора дат). Реализовать вывод полученного заказа и отчета в файл и на печать.

8. СОДЕРЖАНИЕ ПРАКТИЧЕСКОЙ ЧАСТИ

КОДИРОВАНИЕ

При составлении исходного текста программ стоит придерживаться определенных стандартов и правил.

Стандарт оформления кода обычно принимается и используется некоторой группой разработчиков программного обеспечения с целью единообразного оформления совместно используемого кода. Такой стандарт сильно зависит от используемого языка программирования. Например, стандарт оформления кода для языка C/C++ будет серьёзно отличаться от стандарта для языка.

Стиль именования переменных, констант и функций. Соглашение об именах делает программы более понятными и упрощает их чтение. Также соглашение может дать информацию о функции, выполняемой тем или иным идентификатором. Например, является ли запись константой, пакетом или классом, что может быть полезным для понимания кода.

Переменные. Имена переменных, по возможности, должны быть короткими, но при этом отражать возложенные на переменную функции. Выбор имени переменной должен быть мнемонический, т.е. указывающий случайному читателю назначение той или иной переменной. Односимвольные имена являются исключением для временных «одноразовых» переменных.

Обычно таким переменные дают следующие имена:

- i, j, k, m, n для целых типов;
- c, d, s для символьных типов.

Константы. Имена переменных, объявленных константами внутри класса, и ANSI констант должны содержать символы только в верхнем регистре со словами разделяемыми символом подчеркивания («_»). ANSI констант следует избегать для упрощения отладки.

Пример:

static final int MIN_WIDTH = 4; static final int MAX_WIDTH = 999; static final int GET_THE_CPU = 1;

Количество операторов в строке. Для улучшения читаемости исходного текста программы рекомендуется писать не более одного оператора в строке, что вызвано особенностями человеческого восприятия текста. Кроме того, это облегчает пошаговую отладку в символьных отладчиках.

Пример:

Строки

```
int *ptr; ptr = new int [100];
ptr[0] = 0;
```

Следует оформить следующим образом:

```
int *ptr;
ptr = new int [100];
ptr[0] = 0;
```

Стиль отступов – правила форматирования исходного кода, в соответствии с которыми отступы проставляются в удобочитаемой манере. Редакторы текста, входящие в состав большинства популярных сред

разработки, часто предоставляют средства для поддержки используемого стиля отступов, например, автоматическую вставку пробелов/табуляции при вводе скобок, обозначающих начало/конец логического блока. Желательно при программировании придерживаться стиля отступов, так как это во многом упрощает дальнейшую работу с исходным кодом и помогает избежать ошибок в структуре кода при его составлении.

Пустые строки. Использование пустых строк является важным средством для выделения участков программы. При этом имеет смысл отделять:

1) определения переменных:

```
charstr[80];
intcounter = 0;
fgets(str, 79, infile);
counter++;
```

2) последовательности однотипных инструкций или директив:

```
#include
#include
#include
#define NAME_SIZE 256
#define MAX_LEN 3000
```

3) функции:

```
int main()
{
...
}
char *get_name(FILE *f)
{
...
}
```

4) любые логически завершенные блоки кода:

```
printf( "Enter size and delta: " ); // Блок ввода данных
scanf( "%d", &size );
scanf( "%f", &delta );
for( i=0; i <size; i++ ) //Блок использования данных
{
a[i] -= delta;
b[i] += delta;
}
```

Комментарии. Большинство специалистов сходятся во мнении, что комментарии должны объяснять намерения программиста, а не код; то, что можно выразить на языке программирования, не должно выноситься в комментарии – в частности, надо использовать говорящие названия переменных, функций, классов, методов и пр., разбивать программу на лёгкие для понимания части, стремиться к тому, чтобы структура классов и структура баз данных были максимально понятными и прозрачными и т. д.

Концепция грамотного программирования настаивает на включение в текст программы настолько подробных и продуманных комментариев, чтобы она стала исходным текстом не только для исполняемого кода, но и для сопроводительной документации.

Время, потраченное на написание комментариев, многократно окупится при любых модификациях программы. Однако комментировать все подряд включая самоочевидные действия тоже не стоит.

Комментировать следует:

- заголовок файла, описывая содержимое данного файла;
- заголовок функции, поясняя назначение ее аргументов и смысл самой функции;
- вводимые переменные и структуры данных;
- основные этапы и особенности реализуемых алгоритмов;

• любые места, которые трудны для быстрого понимания, в особенности использование различных программных "трюков" и нестандартных приемов.

Некоторые комментарии программисты используют в ходе своей работы. Подобные комментарии особенно полезны, когда над одним кодом работает несколько разработчиков. Так, комментарием TODO обычно помечают участок кода, который программист оставляет незавершённым, чтобы вернуться к нему позже. Комментарий FIXME помечает обнаруженную ошибку, которую решают исправить позже. Комментарий XXX или ZZZ обозначает найденную критическую ошибку, без исправления которой нельзя продолжать дальнейшую работу.

9. ТРЕБОВАНИЯ К ОФОРМЛЕНИЮ ПОЯСНИТЕЛЬНОЙ ЗАПИСКИ

При оформлении пояснительной записки стоит обращать внимание на количество используемых стилей – в списке стилей должны быть только необходимые (остальные следует скрыть или удалить), это поможет привести части документа к единообразию.

1. Используются поля следующих размеров: слева – 30 мм; справа – 10 мм; сверху – 20 мм; снизу – 20 мм.

2. Нумерация страниц – сквозная, номер проставляется в правом нижнем углу страницы. На титульном листе номер не ставится.

3. Заголовки разделов и подразделов должны быть сформулированы кратко, на первом месте должно стоять имя существительное.

4. Все заголовки иерархически нумеруются. В конце заголовка точка не ставится. Такие разделы как «СОДЕРЖАНИЕ», «ВВЕДЕНИЕ», «ЗАКЛЮЧЕНИЕ», «СПИСОК ЛИТЕРАТУРЫ», «ПРИЛОЖЕНИЕ» не нумеруются.

5. Каждый раздел (заголовок 1-го уровня) следует начинать с новой страницы. Заголовок 1-го уровня следует располагать в середине строки и

набирать прописными буквами. Заголовки 2-го уровня и ниже следует начинать с абзацного отступа и печатать с прописной буквы.

6. Заголовки следует отделять от окружающего текста промежутком размером не менее чем в 15 мм снизу и 30 мм сверху. После любого заголовка должен следовать текст, а не рисунок, формула или таблица.

7. Все рисунки, таблицы и формулы нумеруются. Нумерация может быть либо сквозной по всему тексту, например «таблица 7», либо по разделам, например «Рисунок – 2.5», что означает рисунок 5 в разделе 2. Ссылаться на рисунок следует как «рис. 2.5», на таблицу – «табл. 7». Номер формулы располагается справа от нее в круглых скобках. Нумеруются только те формулы, на которые есть ссылки в тексте.

8. Каждый рисунок должен иметь название. Название для таблиц не обязательно. Точка после названия рисунка и таблицы не ставится.

9. Название рисунка располагается под рисунком по центру. Название таблицы располагается над таблицей справа.

10. На каждый рисунок, таблицу и приложение в тексте должна быть ссылка в основной части пояснительной записки.

11. Для каждого вида текста (обычный, заголовки различных уровней, подписи к рисункам и таблицам) рекомендуется создать соответствующий стиль в текстовом редакторе и использовать его.

12. В разделе «СПИСОК ИСТОЧНИКОВ» помещаются только те источники, которые использовались при написании текста.

13. Ссылки на литературные источники в тексте приводятся в квадратных скобках, например, [1, 2] или [1, 3–7].

14. Приложения идентифицируются номерами или буквами, например «ПРИЛОЖЕНИЕ 1»или «ПРИЛОЖЕНИЕ А». На следующей строке, при необходимости, помещается название приложения, например «ЛИСТИНГ ПРОГРАММЫ», которое оформляется как заголовок 1- го уровня без нумерации.

10. ТРЕБОВАНИЕ К РУКОВОДСТВУ ПОЛЬЗОВАТЕЛЯ

Одной из важнейших составляющих любой законченной программы является руководство пользователя. От того, насколько понятно и доступно написано руководство, зависит успех программы, ее распространенность и популярность.

Существует 3 основных типа руководств:

1. Описание с пошаговой инструкцией. Это наиболее распространённый тип, в котором даётся руководство по выполнению различных действий в программе с точки зрения целей пользователя: каждое действие разбивается на несколько простых операций и пользователю предъявляется информация о том, каким образом и в какой последовательности он может их выполнять (как правило, описание сопровождается большим количеством изображений кнопок, панелей, форм и т.д.).

2. Другим типом является описание, ориентированное на перечисление и описание операций, допустимых в программе. В этом случае структура руководства будет совпадать с иерархией программных операций. В этом типе руководства не описывается, каким образом можно решить какую-либо довольно крупную задачу, предоставляя пользователю самому размышлять над этим вопросом.

3. И, наконец, последним типом является руководство для начинающих, которое помогает новичкам ознакомиться с программным продуктом, узнать, как работает программа и как управлять интерфейсом.

Несмотря на довольно существенные различия в подходах к написанию руководства пользователя, можно рекомендовать следующие обязательные разделы:

1. «О программе» – описание программы, назначение и основные возможности.

2. «Системные требования» – список аппаратных и программных средств и их характеристик, необходимых для запуска и успешного функционирования программы.

3. «Интерфейс» – описание интерфейса программы, основных элементов управления и горячих клавиш. При наличии графического интерфейса необходимо привести скриншоты, иллюстрирующие интерфейс пользователя.

4. «Запуск программы» – описание действий, необходимых для запуска программы.

5. «Работа с программой» – пошаговое описание основных действий (в соответствии с выбранным подходом к написанию руководства), которые доступны в программе, с пояснениями и скриншотами-примерами.

6. «Приложение». Необязательный раздел, добавляется при необходимости и может включать любые сведения, не вошедшие в вышеперечисленные разделы, например, глоссарий.

Необходимо учитывать, что руководство пользователя пишется для людей, которые, вполне возможно, плохо умеют работать с компьютером и не знакомы со многими понятиями информационных технологий. Поэтому руководство пользователя следует писать понятным языком, по возможности употребляя как можно меньше специфической терминологии и аббревиатур.

Также не допускается использование жаргонной лексики. Структура предложений должна быть как можно более простой, не перегруженной сложными речевыми оборотами.

11. ТРЕБОВАНИЕ К РУКОВОДСТВУ ПРОГРАММИСТА

Описание можно начать словами: "Функционирование программ приложения формируется на базе следующих модулей:

Project.bpr – главный модуль, процедурами которого являются следующие модули: Unit1.cpp, MyUnit. cpp, ...

Unit1.dfm, Unit1cpp – модули, соответствующий главной форме Form1, с помощью которой осуществляется презентация данного программного продукта, а также вызов модулей: MyUnit.cpp, ...

MyUnit.cpp – модуль, содержащий классы для решения задачи (реализующий класс (имя класса) и его подкласс (наследник) (имя)).

Между модульной структурой и структурами данных существует связь, которая может быть представлена в виде следующей схемы (рис.1).

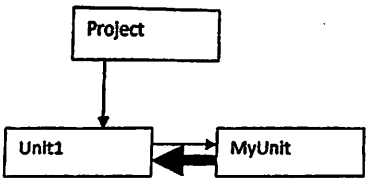

Рис.1. Схема связности модулей

В интерфейсе приложения были определены следующие информационные объекты:

1. Файл, содержащий исходные данные, который имеет следующую структуру: (описание структуры файла).

2. Компонент Edit1, позволяющий создать однострочное текстовое поле, предназначенное для задания числа строк в матрице.

3. Компонент Edit2, позволяющий создать однострочное текстовое поле, предназначенное для задания числа столбцов в матрице.

4. Компонент StringGrid1, позволяющий задать элементы матрицы.

5. Компонент Edit3, позволяющий создать однострочное текстовое поле, предназначенное для отображения на форме результата

ЛИТЕРАТУРА

1.1. Основная литература

1. Мирзиёв Ш.М. "Ўзбекистон Республикасини янада ривожлантириш бўйича ҳаракатлар стратегияси тўғрисида" Ўзбекистон Республикаси Президентининг фармони. 2017 йил 7 феврал.

2. Burger W., Burge M.J. Digital Image Processing: An Algorithmic Introduction using Java. –New York: Springer, 2008. –564 p.

3. Гонзалес Р., Вуде Р. Цифровая обработка изображений. Издание 3-е, испр. и доп. – М.: Техносфера, 2012. – 1104 с.

4. Гашников М. В., Глумов Н. И., Ильясова Н. Ю. и др. Методы компьютерной обработки изображений. Изд. 2. Под ред. В.А. Сойфера. - М.: Физматлит, 2003. -784 с.

1.2. Дополнительная литература

5. Mirziyoyev Sh.M. Buyuk kelajagimizni mard va olijanob xalqimiz bilan birga quramiz. 2017.

6. Mirziyoyev Sh.M. Qonun ustuvorligi va inson manfaatlarini ta'minlash – yurt taraqqiyoti va xalq farovonligining garovi. 2017.

7. Mirziyoyev Sh.M. Erkin va farovon, demokratik O'zbekiston davlatini birgalikda barpo etamiz. 2017

8. Мирзиёв Ш.М. Танқидий таҳлил, қатъий тартиб-интизом ва шахсий жавобгарлик – ҳар бир раҳбар фаолиятининг кундалик қоидаси бўлиши керак. Ўзбекистон Республикаси Вазирлар Маҳкамасининг 2016 йил якунлари ва 2017 йил истиқболларига бағишланган мажлисидаги Ўзбекистон Республикаси Президентининг нутқи. // Халқ сўзи газетаси. 2017 йил 16 январь, №11.

9. Абламейко С.В., Лагуновский Д.М. Обработка изображений: технология, методы, применение. - Мн.: Амалфея, 2000.

10. Яне Б. Цифровая обработка изображений. - М.: Техносфера, 2007. – 584 с.

11. Грузман И.С., Киричук В.С., Косых В.П. и др. Цифровая обработка изображений в информационных системах. – Новосибирск: НГТУ, 2002. – 352 с.

12. Визильтер Ю.В. Обработка и анализ изображений в задачах машинного зрения. – М.: Физматкнига, 2010. – 672 с.

13. Pratt W.K. Digital Image Processing. – New York: Wiley, 2001. – 792 p

14. Jahne B. Digital Image Processing. - New York: Springer, 2005. – 585 p.

15. Chi Zh., Yan H., Pham T. Fuzzy algorithms : with applications to image processing and pattern recognition. New Jersey: World Scientific, 1996. – 230 p.

16. Gonzalez R., Woods R. Digital Image Processing. 3rd Edition - New Jersey: Prentice-Hall, 2008. – 954 p

1.3. Интернет-ресурсы

17. http://www.seas.harvard.edu/courses/cs283/

18. http://infolab.stanford.edu/~ullman/mining/mining.html

19. http://www.machinelearning.ru

Приложение 1

График выполнения курсовой работы

№ №	Содержание блока	Число %	Число баллов/(накопленное)	Срок выполнения (номер недели/дата)	Реальные сроки
1	Получение задания. Анализ задачи. Описание объекта и анализ требований.	10%	10/(10)	2-3 (21 сентября)	
2	Разработка интерфейса и алгоритма реализации.	10%	10/(20)	4 (28 сентября)	
3	Проектирование интерфейса программного приложения.	5%	5/(25)	5 (5 октября)	
	Формализация расчетов.	15%	15/(40)	6 (10 октября)	
4	Разработка кода программы	15%	15/(55)	7(20 октября)	
5	Отладка программы	10%	10/(65)	8 (30 октября)	
6	Тестирование	10%	10/(75)	9 (7 ноября)	
7	Оформление отчета	10%	10/(85)	10 (14 ноября)	
8	Защита работы	15%	15/(100)	11-12 (30 ноября)	
	Итого:	100%	100		

Приложение 2

МИНИСТЕРСТВО ПО РАЗВИТИЮ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И КОММУНИКАЦИЙ РЕСПУБЛИКИ УЗБЕКИСТАН

ТАШКЕНТСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ имени Мухаммада ал-Хорезми

ФАКУЛЬТЕТ ТЕЛЕВИЗИОННЫХ ТЕХНОЛОГИЙ

КАФЕДРА «АУДИОВИЗУАЛЬНЫЕ ТЕХНОЛОГИИ»

Разработка приложений средствами C++Builder

Пояснительная записка к курсовой работе

по курсу «Обработка цифровой информации»

Выполнил: Студент гр.823-15

И.О.Фамилия

Проверил: доцент кафедры

И.О.Фамилия

Ташкент 2018

Приложение 3

МИНИСТЕРСТВО ПО РАЗВИТИЮ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И КОММУНИКАЦИЙ РЕСПУБЛИКИ УЗБЕКИСТАН

ТАШКЕНТСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ имени Мухаммада ал-Хоразми

ФАКУЛЬТЕТ ТЕЛЕВИЗИОННЫЕ ТЕХНОЛОГИИ

КАФЕДРА «АУДИОВИЗУАЛЬНЫЕ ТЕХНОЛОГИИ»

УТВЕРЖДАЮ:

Зав. кафедрой_____

«_____» _____

ЗАДАНИЕ

на выполнение курсовой работы

Студенту_____

Тема курсовой работы

_____

Срок сдачи студентом готовой работы

_____

1. Исходные данные к работе

_____

_____

_____

2. Содержание расчетно-пояснительной записки (перечень подлежащих разработке вопросов)

_____

_____

_____

_____Перечень графического материала (с точным указанием обязательных чертежей)

_____

_____

Дата выдачи задания_____

Руководитель_____(подпись)_____

Задание принял к исполнению_____(подпись)_____

Приложение 4

МИНИСТЕРСТВО ПО РАЗВИТИЮ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И КОММУНИКАЦИЙ РЕСПУБЛИКИ УЗБЕКИСТАН

ТАШКЕНТСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ имени Мухаммада ал-Хорезми

ФАКУЛЬТЕТ ТЕЛЕВИЗИОННЫЕ ТЕХНОЛОГИИ

КАФЕДРА «АУДИОВИЗУАЛЬНЫЕ ТЕХНОЛОГИИ»

ИДЕНТИФИКАЦИЯ ЛИЧНОСТИ ПО ФОТОГРАФИИ ЛИЦА

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА К КУРСОВОЙ РАБОТЕ ПО КУРСУ «ОБРАБОТКА ЦИФРОВОЙ ИНФОРМАЦИИ»

Выполнил:

Студент гр.823-15 И.О.Иванов

Проверил:

доцент кафедры Л.Т.Марышева

Ташкент 2018

Курсовая работа на тему «Идентификация личности по фотографии лица» посвящена изучению технологий компьютерного зрения, биометрических систем идентификации личности и разработке программного обеспечения идентификации личности по фотографии лица.

Курс иши "Юз тасвири бўйича шахсни идентификация қилиш" компьютер куриш технологияларни, шахсни идентификация қилиш биометрик тизимларини ўрганиш ва юз тасвири бўйича шахсни идентификация қилиш дастурий таъминотини яратишга бағишланган.

This course work on the subject of "Identification of the personality according to the photo of the person" is devoted for studying computer vision technology, biometrical identification system and developing software for identification of the personality according to the photo of the person.

МИНИСТЕРСТВО ПО РАЗВИТИЮ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И КОММУНИКАЦИЙ РЕСПУБЛИКИ УЗБЕКИСТАН

ТАШКЕНТСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

имени Мухаммада ал-Хорезми

ФАКУЛЬТЕТ ТЕЛЕВИЗИОННЫЕ ТЕХНОЛОГИИ

КАФЕДРА «АУДИОВИЗУАЛЬНЫЕ ТЕХНОЛОГИИ»

УТВЕРЖДАЮ:

Зав. Кафедрой "АВТ"

А. Мухамадиев_____

"_____"_____

ЗАДАНИЕ

на выполнение курсовой работы

Студенту_И.О.Иванову_____

1. Тема курсовой работы: Идентификация личности по фотографии лица
2. Срок сдачи студентом готовой работы 15.12.2018г._____
3. Исходные данные к работе _*научно-техническая литература в соответствии темы, сведения о Internet, Microsoft Visual Studio 2016*
4. Содержание расчетно-пояснительной записки (перечень подлежащих разработке вопросов)

_____Введение, Теоретическая часть, Основная часть, Заключение

Перечень графичедениеского материала (с точным указанием обязательных чертежей) _*ПрезентацияMicrosoft Office PowerPoint 2007*_____

Дата выдачи задания _7.09.2018_____

Руководитель_____(подпись)_____

Задание принял к исполнению_____(подпись)_____

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В Узбекистане проведена большая работа по внедрению биометрических паспортов и приняты следующие постановления и указы Президента Республики Узбекистан:

- Указ Президента Республики Узбекистан от 23 июня 2009 года № УП-4117 «О мерах по дальнейшему совершенствованию паспортной системы в Республике Узбекистан»
- Постановление Кабинета Министров Республики Узбекистан от 23.06.2010 г. № 122 «О дальнейших мерах по созданию биометрической паспортной системы».

Паспорта с биометрическими данными в Узбекистане введены в целях обеспечения безопасности международной гражданской аэронавигации, повышения степени защищенности удостоверяющих личность документов от возможной подделки, совершенствования механизмов оперативной и точной идентификации личности при пересечении Государственной границы, повышения качества и эффективности международных авиаперевозок.

Сегодня в процессе бурного развития информационно-коммуникационных технологий в Узбекистане, одним из актуальных вопросов является развитие технологий компьютерного зрения.

Одним из важных направлений, активно использующих технологию компьютерного зрения, являются биометрические системы идентификации личности. Биометрия — это раздел биологии, занимающийся количественными биологическими экспериментами с привлечением методов математической статистики — появившийся лишь в конце XIX века, хотя сама наука имеет древнее происхождение. В середине 70-х годов XX века интерес к биометрии вновь возрос в связи с созданием биометрических систем безопасности, основанных на измерении уникальных биологических, физиологических и поведенческих характеристик, индивидуальных для каждого человека. Также их называют биологическим кодом человека.

Значимость биометрических систем идентификации, по сравнению с PIN-кодовыми системами или системами доступа по паролю, состоит в том, что идентифицируется не внешний предмет, относящийся человеку, а собственно сам человек. Рассматриваемые характеристики неразрывно связаны с человеком, их нельзя потерять, передать, забыть и крайне сложно подделать. Данные характеристики не требуют замены или восстановления и практически не подвержены износу.

На первых порах такие измерительные системы были весьма дорогостоящими из-за высокой стоимости регистрирующей и вычислительной техники. Улучшение технологии производства регистрирующей оборудования и конечные достижения в сфере микропроцессоров значительно удешевили и увеличили рынок биометрических систем. Вследствие этого уже сейчас многие организации в Узбекистане и за рубежом полностью или частично перешли на биометрические системы для обеспечения физической и информационной безопасности, также в нашей Республике во многих организациях в системах контроля управления доступом применяются системы биометрической идентификации личности.

Одним из бурно развивающихся в наше время технологических направлений в биометрии является распознавание по геометрии лица. Действительно, вряд ли кто оспорит тот факт, что лицо является одним из самых доступных и естественных для идентификации объектов. Человек всегда использует этот биометрический метод, позволяющий ему без труда отыскивать среди толпы знакомое лицо. Основной задачей которую ставит научные круги является осознание биометрической логики человека и построение систем на схожих принципах.

К преимуществам таких систем можно отнести: дистанционное функционирование (не требуется физический контакт пользователя с системой), естественность предъявления лица для большинства людей, гигиеничность (например, по сравнению со снятием отпечатков пальцев), скрытность (при необходимости) и т.п. Следует отметить и доступность данных

для подобной системы идентификации. Так, для системы, обеспечивающей безопасность вокзала, аэропорта или банка, возможно, достаточно будет включить в ее базу данных одну фотографию лица разыскиваемого человека или даже его фоторобот. При таких же условиях обеспечить безопасность на базе других биометрических систем невозможно в принципе. К примеру, процесс сбора отпечатков пальцев либо сканирование радужной оболочки глаза в условиях вокзала или аэропорта потребует целого штата сотрудников с высокотехнологичным оборудованием, а кроме того отнимет много времени у проверяемых пассажиров.

Целью курсовой работы является изучение технологий компьютерного зрения, биометрических систем идентификации личности и выбор алгоритма, постановка задачи разработка программного обеспечения идентификации личности по фотографии лица с использованием интегрированной среды разработки Delphi XE6 компании Embarcadero Technologies и стандартных средств разработки Luxand FaceSDK.

Теоретико-методологическую базу данного исследования составляют учебная и методическая литература, ресурсы в сети интернет, статьи в научных журналах и лекции преподавателей Ташкентского Университета Информационных Технологий имени Мухаммада Ал-Хоразмий.

Задачи курсовой работы:

- Изучение мирового опыта по использованию технологий компьютерного зрения, биометрических систем идентификации личности.
- Изучение мирового опыта по разработке программного обеспечения идентификации личности по фотографии лица.
- Изучение возможностей интегрированной среды разработки Delphi XE6 и стандартных средств разработки Luxand FaceSDK для идентификации личности по фотографии лица.
- Выбор алгоритма по идентификации личности по фотографии лица и постановка задач для разработки программного обеспечения.

- Создание программного обеспечения идентификации личности по фотографии лица и его использование.

Структура курсовой работы состоит из:

Теоритических основ компьютерного зрения и методы технологии; Постановки задачи; Алгоритма разработки и проектирования программного обеспечения по идентификации личности по фотографии лица; Заключения.

1-ГЛАВА. ТЕОРИТИЧЕСКИЕ ОСНОВЫ КОМПЬЮТЕРНОГО ЗРЕНИЯ И МЕТОДЫ ТЕХНОЛОГИИ

1.1.Технология компьютерного зрения

Компьютерное зрение — теория и технология создания машин, которые могут производить обнаружение, слежение и классификацию объектов.

Как научная дисциплина, компьютерное зрение относится к теории и технологии создания искусственных систем, которые принимают информацию из изображений и фотографий. Видеоданные могут быть представлены видеопоследовательностью, изображениями с различных камер либо трехмерными данными.

Одним из важнейших направлений в области информационных технологий является автоматическая обработка визуальной информации. Внимание к вопросам компьютерной обработки обусловливается расширением возможностей компьютерных систем и разработкой новых технологий обработки, анализа, идентификации различных видов изображений. При этом для формирования эффективных технологий разрабатываемые методы и алгоритмы должны удовлетворять ряду требований по быстродействию и точности.

Разработка компьютерных систем обнаружения и распознавания объектов на изображениях является довольно актуальной проблемой. Актуальность проявляется вследствие необходимости повышения качества и уменьшения рутинной работы человека, а также возможности усовершенствования алгоритмов обработки изображений в связи с развитием компьютерной техники. Формирование признаков – это первый этап в любой системе распознавания образов. Качество всей системы оказывается жестко зависимо от того, насколько хорошо подобраны признаки для описания изображения.

Распознавание образов и их идентификация – сложная задача, как с научной точки зрения, так и в приборном исполнении. Сейчас идентификация изображений применяется во многих сферах деятельности людей. Она используется для подтверждения подлинности документов и денежных знаков, для распознавания личности, для повышения достоверности и оперативности контроля продукции на промышленных предприятиях и т.д.

Помимо указанных подходов к проблеме компьютерного зрения, многие из исследуемых вопросов могут быть изучены с чисто математической точки зрения. Например, многие методы опираются на статистике, методах оптимизации или геометрии. Также большие работы ведутся в области практического применения компьютерного зрения — того, как существующие методы могут быть реализованы программно и аппаратно, или как они могут быть изменены с тем, чтобы достичь высокой скорости работы без существенного увеличения потребляемых ресурсов.

Обработка изображений или анализ изображений, в основном сосредоточены на работе с двухмерными изображениями, то есть как преобразовать одно изображение в другое. Например, пиксельные операции увеличения контрастности, операции по выделению краёв, устранению шумов или геометрические преобразования. Данные операции предполагают, что обработка изображения действуют независимо от содержания самих изображений.

Также существует область, названная Визуализация, которая первоначально была связана с процессом создания изображений, но иногда имела дело с обработкой и анализом. Например, рентгенография работает с анализом видеоданных медицинского применения.

Наконец, распознавание образов является областью, которая использует различные методы для получения информации из видеоданных, в основном, основанные на статистическом подходе. Значительная часть этой области посвящена практическому применению этих методов.

1.2. Основные этапы развития технологий компьютерного зрения

Компьютерное зрение оформилось как самостоятельная дисциплина к концу 60-х годов. Это направление возникло в рамках искусственного интеллекта в тот его период, когда еще были горячи споры о возможности создания мыслящей машины. Оно выделилось из работ по распознаванию образов.

В истории развития машинного зрения можно выделить следующие этапы:

- 1955 г. – профессор Массачусетского технологического института (МТИ) Оливер Селфридж опубликовал статью «Глаза и уши для компьютера». В ней автор выдвинул теоретическую идею оснащения компьютера средствами распознавания звука и изображения.

- 1958 г. – психолог Фрэнк Розенблатт из Корнеллского университета создал компьютерную реализацию персептрона (от perception - восприятие) – устройства, моделирующего схему распознавания образов человеческим мозгом. Персептрон был впервые смоделирован в 1958 году, причем его обучение требовало около получаса машинного времени на ЭВМ IBM-704. Аппаратный вариант – Mark I Perceptron – был построен в 1960 г. и предназначался для распознавания зрительных образов.

Однако рассмотрение задач машинного зрения носило скорее умозрительный характер, так как ни техники, ни математического обеспечения для решения таких сложных задач еще не было.

- 1960-е гг. – появление первых программных систем обработки изображений (в основном для удаления помех с фотоснимков, сделанных с самолетов и спутников); стали развиваться прикладные исследования в области распознавания печатных символов. Однако все еще существовали ограничения в развитии данной области науки, такие как отсутствие дешевых оптических систем ввода данных, ограниченность и довольно узкая специализация вычислительных систем. Бурное развитие систем компьютерного зрения на

протяжении 60-х годов можно объяснить расширением использования вычислительных машин и очевидной потребностью в более быстрой и эффективной связи человека с ЭВМ. К началу 60-х годов задачи компьютерного зрения в основном охватывали область космических исследований, требовавших обработки большого количества цифровой информации.

- 1970-е гг. – Лавренсе Робертс, аспирант МТИ, выдвинул концепцию машинного построения трехмерных образов объектов на основе анализа их двумерных изображений. На данном этапе стал проводиться более глубокий анализ данных. Начали развиваться различные подходы к распознаванию объектов на изображении, например, структурные, признаковые и текстурные.

- В конце 1980-х годов были созданы роботы, способные более-менее удовлетворительно оценивать окружающий мир и самостоятельно выполнять действия в естественной среде.

- 80-е и 90-е годы ознаменовались появлением нового поколения датчиков двухмерных цифровых информационных полей различной физической природы. Развитие новых измерительных систем и методов регистрации двухмерных цифровых информационных полей в реальном масштабе времени позволило получать для анализа устойчивые во времени изображения, генерируемые этими датчиками. Совершенствование же технологий производства этих датчиков позволило существенным образом снизить их стоимость, а значит, значительно расширить область их применения.

- С начала 90-х годов в алгоритмическом аспекте последовательность действий по обработке изображения принято рассматривать в согласии с так называемой модульной парадигмой. Эта парадигма, предложенная Д. Марром на основе длительного изучения механизмов зрительного восприятия человека, утверждает, что обработка изображений должна опираться на несколько последовательных уровней восходящей информационной линии: от «иконического» представления объектов (растровое изображение,

неструктурированная информация) к их символическому представлению (векторные и атрибутивные данные в структурированной форме, реляционные структуры и т. п.).

- В середине 90-х годов появились первые коммерческие системы автоматической навигации автомобилей. Эффективные средства компьютерного анализа движений удалось разработать в конце XX века.

- 2003 г. – на рынок были выпущены первые достаточно надежные корпоративные системы распознавания лиц.

2-ГЛАВА. РАЗРАБОТКИ И ПРОЕКТИРОВАНИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ПО ИДЕНТИФИКАЦИИ ЛИЧНОСТИ ПО ФОТОГРАФИИ ЛИЦА

2.1. Постановка задачи

Даны изображения лица, хранящиеся в растровых графических файлах формата jpeg.

Ставится задача классификации, распознавания и идентификации личности по изображению лица. Сперва должно загружаться несколько изображений человека (галерея изображений) и потом загружаться изображение сравниваемого человека (называемое «тестовым изображением»). Требуется определить, присутствует ли человек с тестового изображения в галерее, и если да – то определить его идентичность/схожесть в процентном соотношении, мимику и пол (мужской или женский), также в процентном соотношении.

Точность идентификации зависит от многих факторов: разрешения изображения (точнее размера лица на изображении), ракурса съёмки (фронтальный или под углом), освещения, мимики, четкости изображения.

2.2. Алгоритм для разработки программного обеспечения

Алгоритм работы разрабатываемой программы должен быть следующим:

1. Загрузка изображений (галерея изображений) в формате jpeg в программное обеспечение;

2. Определение лица на загружаемых изображениях и обрезание лица человека на фотографии;

3. Определение точек черт лица (включая глаза, брови, рот, нос и контуры лица) в загружаемых изображениях с применением алгоритма активной модели внешнего вида;

4. Загрузка изображения (тестовое изображение) для сравнения с загруженными изображениями в пункте 1;

5. Определение лица на тестовом изображении и обрезание лица человека на фотографии;

6. Определение точек черт лица (включая глаза, брови, рот, нос и контуры лица) в тестовом изображении с применением алгоритма активной модели внешнего вида;

7. Сравнение тестового изображения с галерей изображений с применением алгоритма активной модели внешнего вида;

8. Вывод в удобном виде результатов сравнения идентичности/идентичность, мимики и пола.

Блок схема алгоритма работы программы представлена на рис. 2.1.

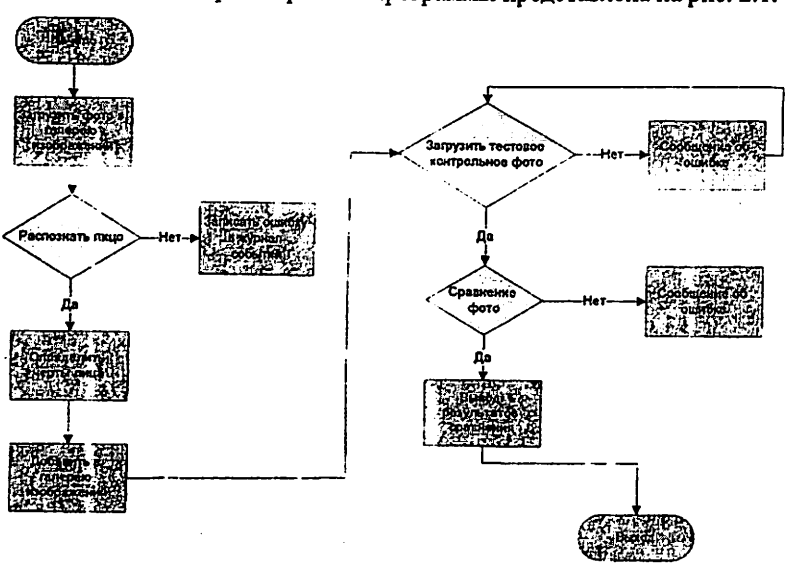

Рис.2.1. Блок схема работы программы

Биометрическое распознавание лица не должно требовать специальной технической аппаратуры. Изображение лица человека можно получить с обычной WEB-камеры, фотоаппарата или планшета и загрузить на компьютер, ноутбук. Следовательно, реализация и внедрение данной программы не должно быть сопряжено с рядом технических или финансовых трудностей.

2.3. Разработка и проектирование программного обеспечения. Выбор среды разработки

В ходе выполнения курсовой работы были изучены различные подходы к распознаванию лиц (идентификации) по фотографиям лица и произведено сопоставление результатов работы различных методов и программных продуктов, предоставляющих разработчику встраивать возможности распознавания лиц в собственное ПО. Также были изучены требования стандартов в области биометрии и ограничения существующих алгоритмов распознавания лиц.

Для свободно распространяемых средств, таких, например, как OpenCV, исследования показали их недостаточную точность при работе с выборкой лиц и сложными условиями регистрации лица.

Для реализации поставленных задач мною была выбрана стандартная библиотека Luxand FaceSDK 6.1 для распознавания лиц и их идентификации.

Библиотека Luxand FaceSDK

Luxand FaceSDK кросс – платформенная библиотека определения и распознавания лиц, которая может быть интегрирована в клиентское приложение. FaceSDK предлагает интерфейс прикладного программирования API – набор функций, предоставляемый для использования в прикладных программах, для обнаружения и отслеживания лиц и контуров лица, определения пола, распознавания лица на неподвижных изображениях и видео. Luxand FaceSDK обеспечивает определение координат (x;y) 66 точек черт лица (включая глаза, брови, рот, нос и контуры лица), которые хранятся в базе

данных (рисунок 3.2). Каждой из этих точек присваивается идентификационное название, которое впоследствии задается соответствующей функцией. По данным точкам осуществляется построение шаблона активной модели внешнего вида с лицевыми точками, сохраняющимися по своим идентификационным названиям в массиве библиотеки FSDK_Features.

Шаблоны лиц при биометрическом распознавании Luxand FaceSDK строятся с помощью применения описанного выше алгоритма активных моделей внешнего вида. Активные модели формы – это модели формы объектов, которые могут многократно деформироваться, для подгонки к объекту, присутствующему на новом изображении. Деформация ограничена моделью распределения точек статистической формы модели, чтобы модель могла меняться только в пределах размеченных примеров из обучающей выборки. Форму объекта представляет множество точек, контролируемое формой модели. Цель алгоритма активной модели формы состоит в сопоставлении модели с новым изображением. Алгоритм состоит из двух чередуемых действий:

- поиск на изображении вокруг каждой точки лучшей позиции для данной точки;
- обновление параметров модели путем определения наилучшего соответствия с вновь найденными позициями(рис 2.2).

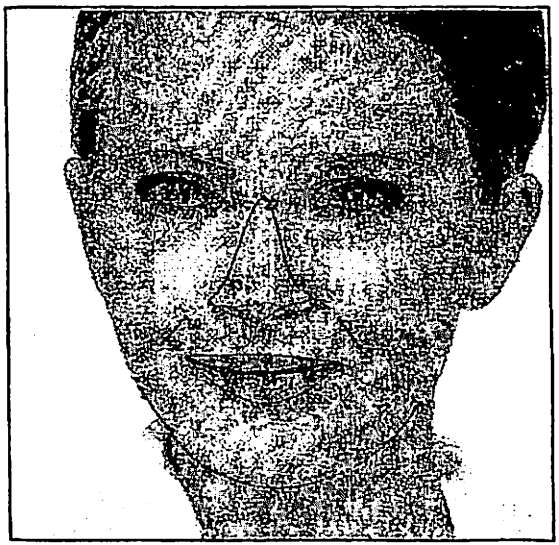

Рис.2.2. Шаблон лица, заданный 66 лицевыми точками, хранящийся в библиотеке распознавания Luxand FaceSDK

Для нахождения лучшей позиции для каждой точки можно либо отыскать четкие края, либо совместить статистическую модель с тем, что ожидается для данной точки.

Программное обеспечение Luxand FaceSDK – это инструмент обнаружения и распознавания человеческих лиц. Luxand FaceSDK позволяет разработчикам Microsoft Visual C++, C#, VB, Java и Borland Delphi создавать приложения для Windows, Linux, Mac и Интернета с функциями распознавания лиц и биометрической идентификации. Luxand FaceSDK применяется в сотнях программ аутентификации пользователей с помощью web-камер: приложение проверяет соответствие лиц фотографиям, автоматически обнаруживает лица в реальной жизни, графических редакторах, статичных изображениях и анимированных видео. Luxand FaceSDK представляет собой высокопроизводительное и кроссплатформенное решение, совместимое с 32- и

64-разрядной архитектурой и легко интегрируемое в новые и существующие проекты.

Характеристики Luxand FaceSDK:

- Быстрое и точное распознавание и идентификация лиц.
- Стабильное распознавание, не зависящее от условий освещения.
- Распознавание черт и выражений лиц.
- Поддержка любых web-камер.
- Поддержка небольших и крупных многопиксельных изображений.

Luxand FaceSDK может использоваться для разработки следующих приложений:

- Систем биометрической аутентификации в режиме реального времени – пользователи могут пройти идентификацию, просто посмотрев в web-камеру.
- Автоматического удаления эффекта красных глаз.
- Программ обработки изображений и графических редакторов.
- Создания эффектов анимации лиц для индустрии развлечений.
- Систем автоматизации рабочих процессов с графикой.
- Инструментов просмотра и организации изображений с функцией поиска графики по лицам.
- Приложений для цифровых камер, сканеров и web-камер.

Embarcadero RAD STUDIO

Embarcadero RADSTUDIO— среда быстрой разработки приложений (RAPIDAPPLICATIONDEVELOPER) фирмы Embarcadero Technologies, работающая под Microsoft Windows.

Embarcadero RAD Studio представляет собой набор средств разработки приложений, который позволяет создавать приложения с графическим пользовательским интерфейсом для *Windows*, *Mac OS X*, *.NET*, *PHP* и веб-решений. В её состав входят:

- *Embarcadero Delphi* дает возможность создавать полнофункциональные приложения для *Windows* и *Mac OS X*.

RAD Studio включает в себя широкий набор дополнительных программ:

- *InstallAware Express* предоставляет средства, позволяющие пользователям, не имеющим навыков программирования и разработки сценариев, создавать сложные установочные пакеты.
- *Rave Reports* компании Nevrona — набор решений для создания отчётов.
- *FastReport.*
- *FireMonkey.*
- *AppWave.*
- *TeeChart Standard* компании Steema — компоненты для создания диаграмм.
- *VCL* для веб-решений (IntraWeb) компании Atozed Software — платформа веб-приложений RAD.
- *FinalBuilder Embarcadero Edition* служит для автоматизации процесса сборки.
- *CodeSite Express* — средства ведения журнала для сборки приложений.
- *AQTime Standard* компании SmartBear — создание профилей производительности.
- *Beyond Compare Text Compare* — сравнение файлов исходного кода.
- *RemObjects Internet Tools* и *Oxfuscator* — дополнительная функциональность для веб-разработки и обфускации кода в Delphi Prism.

Delphi XE6

Delphi XE6 – это интегрированная среда разработки программного обеспечения для Mac OS, Microsoft Windows, Android и iOS на языке Delphi (раньше носил наименование Object Pascal). Была создана первоначально

компанией Borland и сейчас принадлежит и разрабатывается Embarcadero Technologies. Embarcadero Delphi – это часть пакета Embarcadero RAD Studio и выпускается в пяти редакциях: Ultimate и Architect, Starter, Professional, Enterprise. Среда RAD предназначена для быстрой разработки прикладного ПО для операционных систем Windows, Mac OS X, а также IOS и Android. RAD позволяет непосредственно взаимодействовать с операционными системами и с библиотеками, которые написаны на C/C++, при помощи уникальной генерации машинного кода и совокупности простоты языка.

Программы уже созданные не имеют зависимости от стороннего ПО, как-то Microsoft .NET Framework, или Java Virtual Machine. В основном пользовательским кодом контролируется выделение и освобождение памяти, что, во-первых, дает возможность создания сложных приложений, с высокими требованиями к отзывчивости (создание в реальном времени), а во-вторых – ужесточает требования к качеству кода. Для мобильных платформ в кросс-компиляторах предусмотрен автоматический подсчёт ссылок на объекты, который облегчает задачу управления их временем жизни.

Проектирование и разработка программного обеспечения

Разработка интерфейса

На рисунке 3.3 показана главная форма и название компонентов.

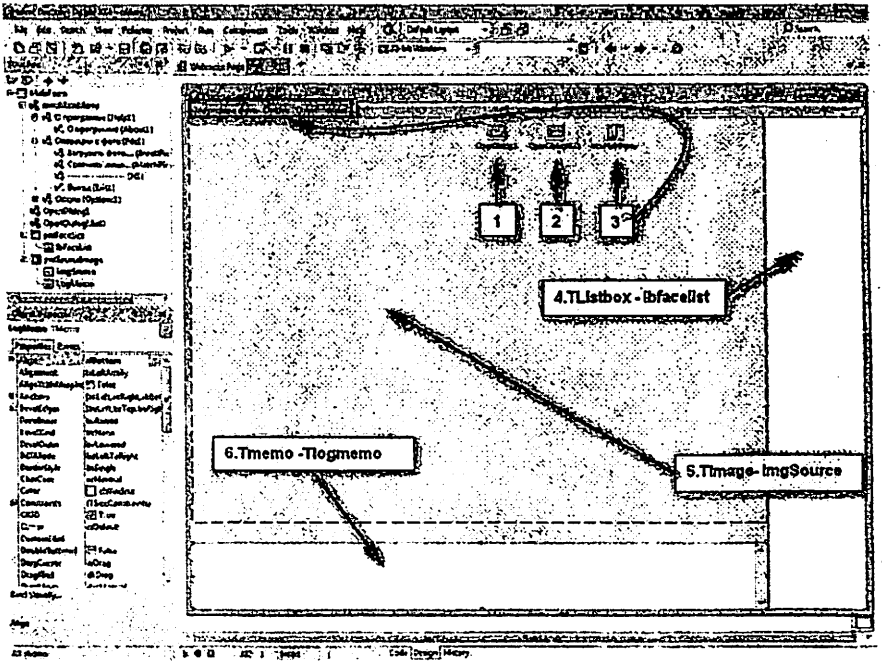

Рис.3.3. Главная форма в среде разработки Delphi XE6

Сперва была создана главная форма - **MainForm.**

На главной форме были установлены следующие компоненты:

1. Компонент TOpenDialog(Наименование OpenDialog1) - диалоговая форма для загрузки изображений.

2. Компонент TOpenDialog(Наименование OpenDialogMulti) - диалоговая форма для загрузки изображений.

3. TMainMenu(Наименование mnuMainMenu) - основное меню главной формы приложения.

4. Компонент TListBox (Наименование lbFaceList) - для хранения списка фотографий.

5. Компонент TImage (Наименование imgSource) - для работы с графическими файлами.

6. Компонент Tmemo (Наименование TLogMemo) - для ведения журнала программы.

Вид формы настройки программы frmOptions показан на рисунке 2.4.

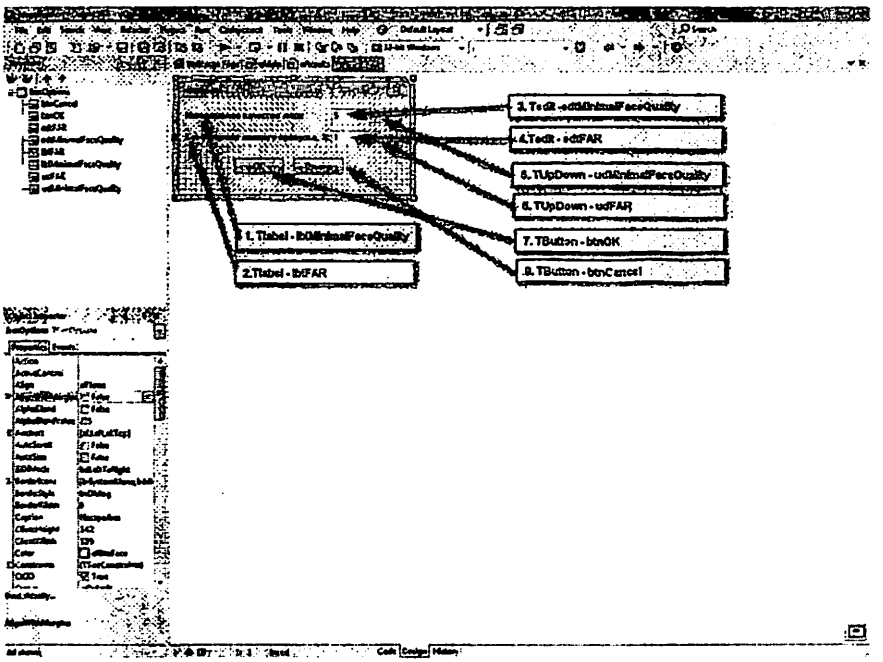

Рис.2.4. Форма настройки программы в среде разработки Delphi XE6

На форме настроек программы frmOptions были установлены следующие компоненты:

1. Компонент TLabel (Наименование lblMinimalFaceQuality) - текстовая надпись на форме.

2. Компонент TLabel (Наименование lblFAR) - текстовая надпись на форме.

3. TEdit (Наименование edtMinimalFaceQuality) - для ввода и редактирования данных о минимальном качестве лица; представляет собой однострочное поле.

4. TEdit (Наименование edtFAR) - для ввода и редактирования данных о коэффициенте ложного пропуска; представляет собой однострочное поле.

5. Компонент TUpDown(Наименование udMinimalFaceQuality) - для изменения числовых значений (в текстовом поле edtMinimalFaceQuality) с помощью кнопок со стрелками и курсорных клавиш.

6. Компонент TUpDown (Наименование udFAR) - для изменения числовых значений (в текстовом поле edtFAR) с помощью кнопок со стрелками и курсорных клавиш.

7. Компонент TButton (Наименование btnOK) -кнопка для сохранения изменений в настройках программы.

8. Компонент TButton (Наименование btnCancel) -кнопка для отмены внесенных изменений в настройки программы.

Вид **формы результаты сравнения frmSearchResults** показан на рисунке 2.5.

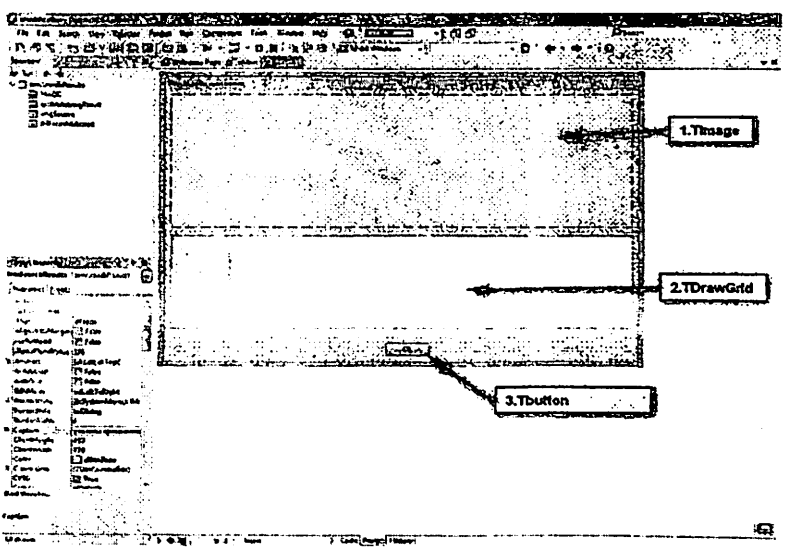

Рис.2.5. Форма результаты сравнения в среде разработки Delphi XE6

На форме результаты сравнения frmSearchResults были установлены следующие компоненты:

1. Компонент TImage (Наименование imgSource) - для работы с графическими файлами.

2. Компонент TDrawGrid(Наименование grdMatchingResults)- для создания в приложении таблицы, которая может содержать графические изображения.

3. Компонент TButton (Наименование btnOK) -кнопка для закрытия данной формы.

Процедуры и функции, используемые в программном обеспечении

Разработанное программное обеспечение включает в себя несколько pas файлов (исходных файлов программы). Полный листинг исходного кода программы приведен в Приложении А.

Umain.pas, принадлежащий главной форме программы, состоит из следующих процедур и функций:

1. procedure FormCreate(Sender:TObject);
2. procedure About1Click(Sender:TObject);
3. procedure Exit1Click(Sender:TObject);
4. procedure Options2Click(Sender:TObject);
5. procedure lbFaceListDrawItem(Control: TWinControl; Index: Integer; Rect: TRect; State: TOwnerDrawState);
6. procedure lbFaceListClick(Sender:TObject);
7. procedure RedrawCurrentImage;
8. procedure EnrollPictures1Click(Sender:TObject);
9. procedure EnrollPicture(Sender:TObject);
10. procedure ClearDatabase1Click(Sender:TObject);
11. procedure MatchPictures1Click(Sender:TObject);

1. Процедура FormCreate загружает и инициализирует библиотеку устанавливает параметры переменным и константам программы.

2. Процедура About1Click показывает диалоговое окно о разработчике программы.

3. Процедура Exit1Click закрывает полностью программу и состоит из одной команды close.

4. Процедура Options2Click открывает форму настроек программы.

5. Процедура lbFaceListDrawItem обрезает рисунки для вставки в список фотографий в компонент TListBox.

6. Процедура lbFaceListClickпри нажатии на фотографию запускает процедуру RedrawCurrentImage. Процедура RedrawCurrentImage описана ниже.

7. Процедура RedrawCurrentImage предназначена для начертания точек краев лица и точек черт лица(включая глаза, брови, рот, нос и контуры лица) на изображениях.

8. ПроцедураEnrollPictures1Clickзапускаетпроцедуру EnrollPictureдлязагрузкифотографийвсписок/галереюизображенийдлясравнения. Процедура EnrollPicture описана ниже.

9. Процедура EnrollPicture является основной процедурой для определения лица на фотографии и точек черт лица (включая глаза, брови, рот, нос и контуры лица) на изображениях для сравнения и нанесения точек на фотографии и добавления в список фотографий(галерея изображений).

10. Процедура ClearDatabase1Click очищает список изображений и память программы.

11. Процедура MatchPictures1Click является основной процедурой для определения точек черт лица (включая глаза, брови, рот, нос и контуры лица) на тестовом/контрольном изображении и нанесения точек на фотографии и запуска формы результатов сравнения frmSearchResults.

uResults.pas принадлежащий форме результатов сравнения, состоит из следующих процедур и функций:

1. procedure FormActivate(Sender:TObject);

2. procedure grdMatchingResultDrawCell(Sender: TObject; ACol, ARow: Integer; Rect: TRect; State: TGridDrawState);

3. procedure SortList(var List: array of integer; var Scores: array of Single; l, r: integer);

4. Процедура FormActivate предназначена для начертания черт лица (глаза, брови, рот, нос и контуры лица) на изображениях и проведения сравнений изображений в списке (галерея изображений) с контрольным/тестовым изображением.

5. Процедура grdMatchingResultDrawCell предназначена для оформления и вывода результатов сравнения изображений в гриде(в табличном виде).

6. Процедура SortList предназначена для сортировки изображений в таблицы в убывающем порядке.

2.4. Результаты работы программного обеспечения по идентификации личности по фотографии лица

Для начала работы с программным обеспечением, предназначенным для идентификации личности по фотографии лица, необходимо загрузить исполняемый файл identification.exe – загрузится главное окно приложения, представленное на рис. 3.6.

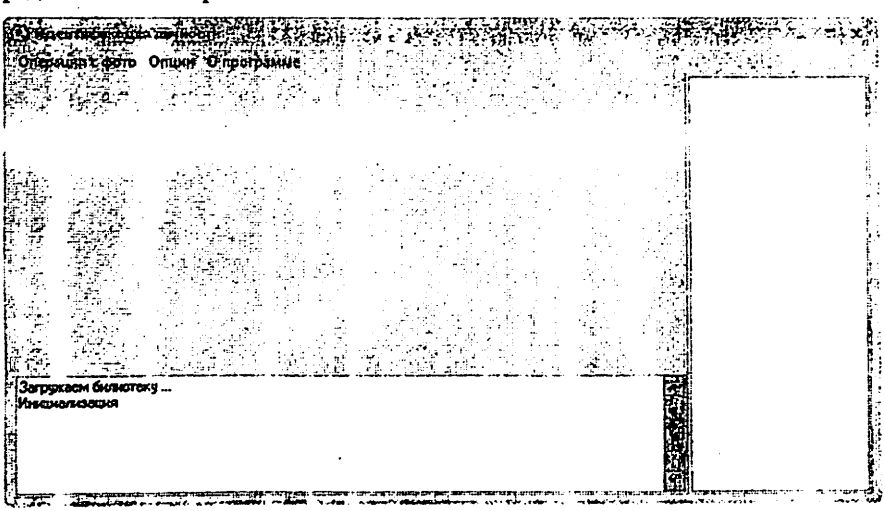

Рис. 3.6. Главное окно приложения identification.exe

Для идентификации личности по фото в библиотеку программного обеспечения необходимо загрузить одно/несколько фото с персонального компьютера/ноутбука пользователя. Для этого следует выбрать пункт главного меню «Операции с фото» → «Загрузить фото» – откроется стандартное окно «Открыть», в котором необходимо выбрать требуемые фото (рис. 2.7) и нажать кнопку «Открыть». Для выбора нескольких фотографий, удерживайте кнопку Ctrl на клавиатуре. Выбранные фотографии автоматически будут загружены в галерею изображений.

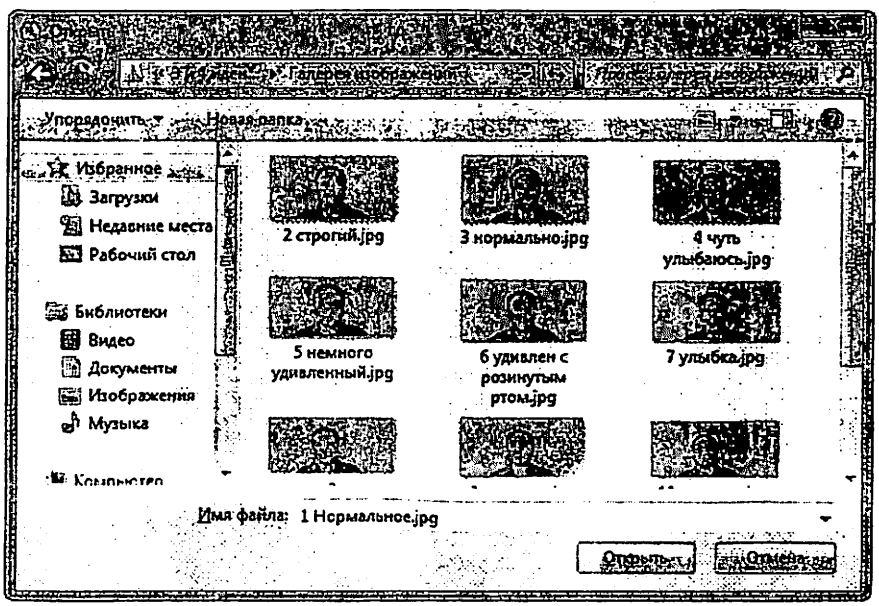

Рис. 2.7. Стандартное окно «Открыть»

С применением алгоритма активной модели внешнего вида разработанное программное обеспечение определит 66 точек черт лица (рис. 3.8), по которым далее будет идентифицироваться личность человека, изображенного на выбранных фотографиях, с тестовым фото.

На панели справа (рис. 2.8[1]) отображаются все фото, загруженные в галерею изображений. Для просмотра построенных точек черт лица на каждой фотографии, достаточно выбрать фото, щелкнув по нему левой кнопкой мыши.

В журнале событий (рис. 2.8[2]) отображаются все выполненные пользователем действия (например, перечень загруженных фото).

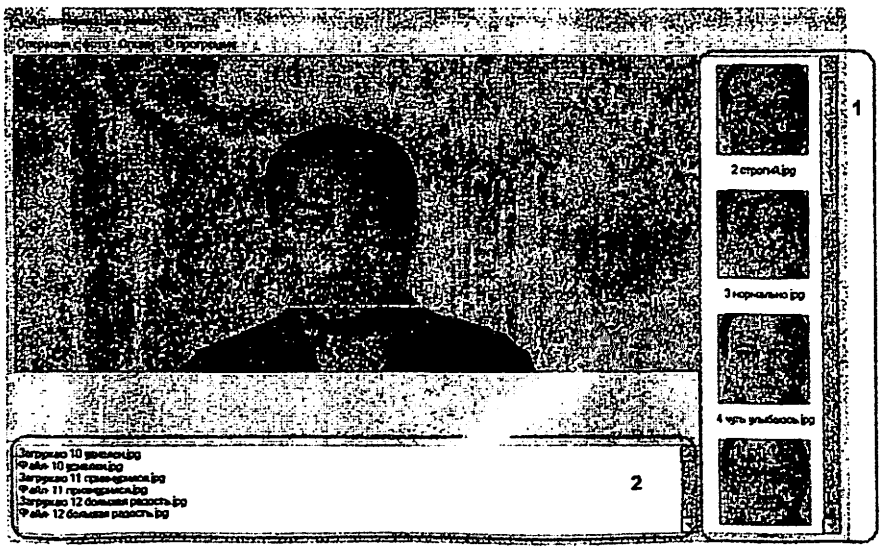

Рис. 2.8. Определение точек черт лица на загруженных фото

После того, как Вы загрузили фото в библиотеку программного обеспечения, необходимо выбрать тестовое фото, которое и будет сравниваться с уже имеющимися изображениями. Для этого выберите пункт главного меню «Операции с фото» → «Сравнить лицо». В открывшемся окне (рис. 3.7) выберите нужную фотографию и нажмите кнопку «Открыть».

Программное обеспечение автоматически выстроить точки черт лица и сравнит «тестовое» изображение с фотографиями, сохраненными в галерее ПО.

Пункт главного меню «Опции» предназначен для:

- настройки параметров «Минимальное качество лица» и «Коэффициент ложного пропуска, %»

- очистки базы программного обеспечения.

Рассмотрим результаты идентификации на конкретном примере.

Тестирование № 1.

Загрузим в галерею изображений фото 11 разных людей (рис. 2.9)

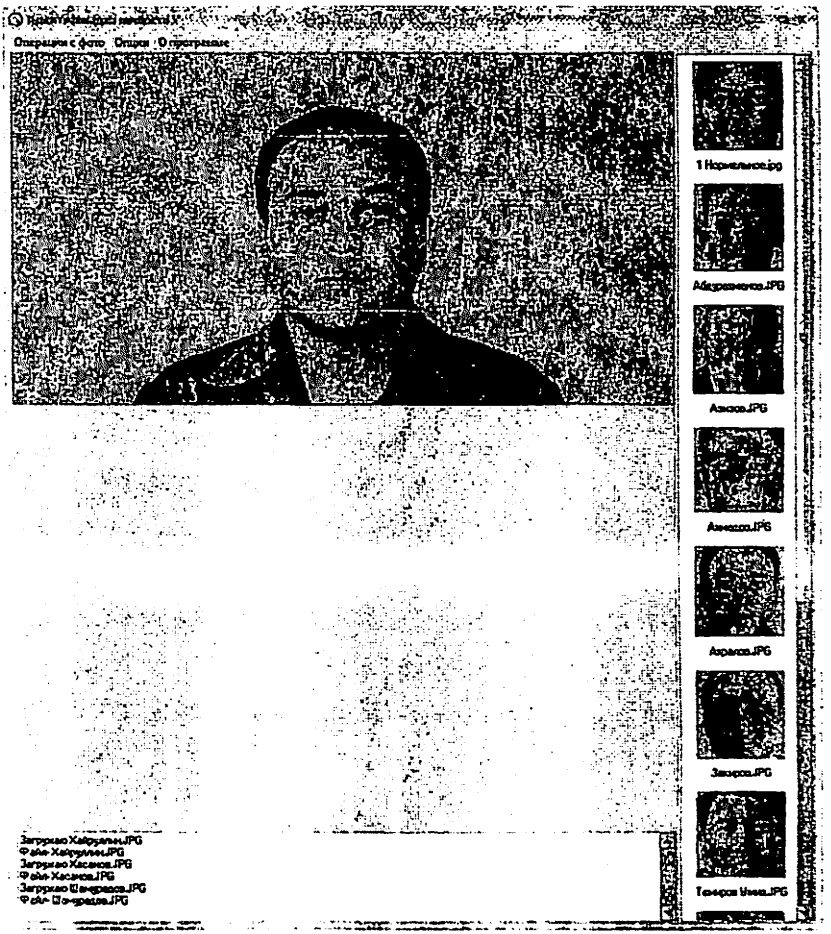

Рис. 2.9. Фотографии 11 разных людей, загруженные в галерею изображений

В качестве «тестового» изображения выберем одно из загруженных фото. Результат идентификации представлен на рис. 2.10.

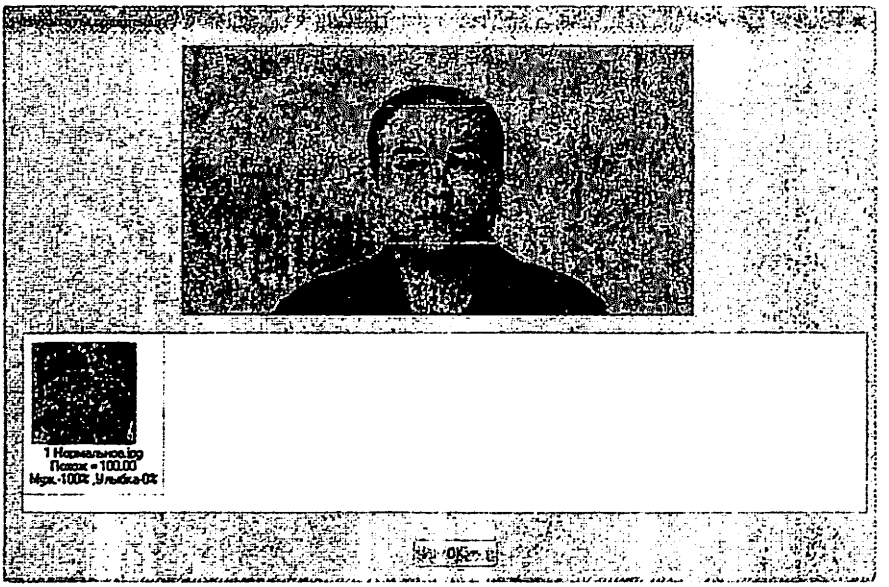

Рис. 2.10. Результат тестирования № 1

Как видно из результатов, программное обеспечение, проведя сравнительный анализ по построенным точкам черт лица, идентифицировало человека, изображенного на «тестовом» фото и на одном из фото из галереи изображений со следующими параметрами: идентичность – 100 %, пол – мужской (100 %), улыбка – 0 %. Остальные фото, на которых были изображены другие люди, не прошли идентификацию.

ЗАКЛЮЧЕНИЕ

В представленной работе рассмотрены технологии компьютерного зрения, этапы его развития, примеры их использования и будущее данной технологии.

В выполненной курсовой работе проведено исследование алгоритмов идентификации личности по фотографии лица. Подробно рассмотрен и изучен алгоритм «Активные модели внешнего вида» для идентификации личности по фотографии лица.

В результате выполнения курсовой работы было разработано программное обеспечение по идентификации личности. Используемая в программном обеспечении методика распознавания и идентификация лиц показала себя надежной и качественной технологией. Простота и гибкость интерфейса программного обеспечения позволят овладеть им при наличии минимальных компьютерных навыков.

Проведенные вовремя работы различные тесты по идентификации личности на разработанном программном обеспечении показали высокие результаты идентификации личности.

Разработанная программа показала себя устойчивым к поворотам лица ±30°, мимике, использованию очков и изменениям освещения.

Интеллектуальное видеонаблюдение с биометрической идентификацией лиц заменит традиционные способы контроля и управления доступом, поскольку оно удобно и эффективно там, где есть возможность проникновения на объект по подложным пропускам, или с использованием чужого пропуска. Система распознавания лиц позволит отслеживать, имеет ли сотрудник доступ на завод и сколько он там находился. Кроме того, с помощью биометрии лиц осуществляется автоматический, полный и достоверный учет рабочего времени персонала. Подобная система становится удобной и многофункциональной альтернативой ID-картам, которые сегодня довольно распространены.

Использование системы распознавания лиц предоставляет следующие возможности:

– контролировать время прибытия сотрудников на рабочее место за счет фиксации точного времени прохождения их на различные участки территории предприятия;

– мотивировать сотрудников прибывать на рабочее место без опозданий;

– предупреждать и предотвращать хищения производственной продукции;

– предотвращать проникновение посторонних лиц на территорию предприятия;

– создать психологический барьер для потенциальных воров;

– контролировать передвижение сотрудников на территории предприятия за счет фиксации их прохождения на определенные участки;

– получить отчеты об опозданиях сотрудников, попытках хищений продукции, проникновений на территорию посторонних лиц и местонахождении сотрудников в определенный момент времени.

Разработанную программу по идентификации личности с минимальными доработками и интеграцией в существующие системы видеонаблюдения можно применить не только в целях безопасности, но и в других сферах деятельности таких как:

1. Контроль посещения учениками и студентами школ, колледжей и институтов;
2. Фейс-контроль в сегменте общепита и развлечений, поиск подозрительных и потенциально опасных посетителей;
3. Верификация банковских карт;
4. Онлайн-платежи;
5. Контекстная реклама, цифровой маркетинг;
6. Криминалистика;
7. Телеконференции;
8. Мобильные приложения;
9. Поиск фото в больших базах фотоснимков;

Приложение. Листинг исходного кода программы

UMAIN.PAS

```
////////////////////////////////////////////////
// Исходный код программы
// Идентификация личности по фотографии лица
////////////////////////////////////////////////

unit uMain;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, LuxandFaceSDK, StdCtrls, Buttons, ExtCtrls, jpeg, Menus,
  ComCtrls;
type
  TMainForm = class(TForm)
    LogMemo: TMemo;
    OpenDialog1: TOpenDialog;
mnuMainMenu: TMainMenu;
    File1: TMenuItem;
    Exit1: TMenuItem;
    Help1: TMenuItem;
    About1: TMenuItem;
    EnrollPictures1: TMenuItem;
    MatchPictures1: TMenuItem;
    N1: TMenuItem;
    Options1: TMenuItem;
    Options2: TMenuItem;
    N2: TMenuItem;
    ClearDatabase1: TMenuItem;
pnlSourceImage: TPanel;
pnlFaceList: TPanel;
imgSource: TImage;
lbFaceList: TListBox;
```

```
    OpenDialogMulti: TOpenDialog;
  procedure FormCreate(Sender: TObject);
  procedure About1Click(Sender: TObject);
  procedure Exit1Click(Sender: TObject);
  procedure Options2Click(Sender: TObject);
  procedure lbFaceListDrawItem(Control: TWinControl; Index: Integer;
      Rect: TRect; State: TOwnerDrawState);
  procedure lbFaceListClick(Sender: TObject);
  procedure EnrollPictures1Click(Sender: TObject);
  procedure ClearDatabase1Click(Sender: TObject);
  procedure MatchPictures1Click(Sender: TObject);
  private
  { Private declarations }
  public
  { Public declarations }
    FaceDetectionThreshold: integer;
    FARValue: integer;
  procedure EnrollPicture(FileName: string);
  procedure RedrawCurrentImage;
  end;

  type
   TFaceRecord = record
    Template: FSDK_FaceTemplate; //Шаблонлица;
    FacePosition: TFacePosition; // Позиции, размерылица
    FacialFeatures: FSDK_Features; //Чертылица;
    ImageFileName: string;
    ImageHandle: HImage;
    ImageBmp: TBitMap;
    FaceImageHandle: HImage;
    FaceImageBmp: TBitMap;
  end;
  const
   FacePreviewWidth = 96;
  var
```

```
 MainForm: TMainForm;
 FaceList: array of TFaceRecord;
implementation
uses uOptions, math, uResults;
{$R *.dfm}
procedure TMainForm.FormCreate(Sender: TObject);
begin
LogMemo.Lines.Add('Загружаембилиотеку ...');
//Активируем и загружаем библиотеку Luxand FaceSDK
 if
FSDK_ActivateLibrary(PAnsiChar(AnsiString('bSB3NdbTnv/0eW/uhypSe6hDMtjZ76Sisw5Nwc
N+0sfahxOtoUW22el54e/M6cSG5/xsdVIorPgugbTIfolIn7ltyw1QMSleNebVx/Xe8aRA8bP+aVDy
bjoWdW/0rDP9Pv7yqBzNXyuwjgsVhPB53VGP8oTirTSUP7PTzSwOEe0='))<>FSDKE_OK
then
 begin
Application.MessageBox('Незарегистрированная программа. Обратитесь Давлатову Баходир
Равшановичу','Ошибка нелицинзионная программа');
  exit;
 end;
  //Инициализируем библиотеку Luxand FaceSDK
FSDK_Initialize('');
LogMemo.Lines.Add('Инициализация');
lbFaceList.ItemHeight := FacePreviewWidth + 8 + 28;
 // Устанавливаем параметр FaceDetectionThreshold для распознования даже нечетких лиц.
// если увеличим параметр он будет распознавать только четкие лица,
// если уменьшим то будет распознавать даже размытые лица.
FaceDetectionThreshold := 3;
FARValue := 10;
FaceList := nil;
end;

procedure TMainForm.About1Click(Sender: TObject);
begin
  MessageBox(Handle, 'Программа "Идентификация личности по фотографии лица"
разработана Телевизионные технологии', 'Информация', mb_OK);
```

```
end;

procedure TMainForm.Exit1Click(Sender: TObject);
begin
  Close;
end;

procedure TMainForm.Options2Click(Sender: TObject);
begin
frmOptions.udFAR.Position := FARValue;
frmOptions.udMinimalFaceQuality.Position := FaceDetectionThreshold;
  frmOptions.ShowModal;
FARValue := frmOptions.udFAR.Position;
FaceDetectionThreshold := frmOptions.udMinimalFaceQuality.Position;
end;

procedure TMainForm.lbFaceListDrawItem(Control: TWinControl; Index: Integer;
 Rect: TRect; State: TOwnerDrawState);
var
 BrushColor, PenColor: integer;
begin
BrushColor := lbFaceList.Canvas.Brush.Color;
PenColor := lbFaceList.Canvas.Pen.Color;

lbFaceList.Canvas.Brush.Style := bsSolid;
lbFaceList.Canvas.Brush.Color := clWhite;
lbFaceList.Canvas.Pen.Color := clWhite;
lbFaceList.Canvas.Rectangle(Rect);

if FaceList[Index].FaceImageBmp <> nil then
lbFaceList.Canvas.Draw(Rect.Left + 16, Rect.Top + 8, FaceList[Index].FaceImageBmp);

lbFaceList.Canvas.TextOut(Rect.Left + 16 + FacePreviewWidth div 2 -
lbFaceList.Canvas.TextWidth(lbFaceList.Items[Index]) div 2,
  Rect.Top + FacePreviewWidth + 8 + 8, lbFaceList.Items[Index]);
```

```
lbFaceList.Canvas.Brush.Color := BrushColor;
lbFaceList.Canvas.Pen.Color := PenColor;
end;

procedure TMainForm.lbFaceListClick(Sender: TObject);
begin
  RedrawCurrentImage;
end;

procedure TMainForm.RedrawCurrentImage;
var
i,y: integer;
left, top, right, bottom: integer;
begin
if (lbFaceList.ItemIndex >= 0) and (lbFaceList.ItemIndex < length(FaceList)) then
begin
imgSource.Picture.Assign(FaceList[lbFaceList.ItemIndex].ImageBmp);

imgSource.Canvas.Brush.Style := bsClear;
imgSource.Canvas.Pen.Color := clLime;

   // Рисуем позиции лица
with FaceList[lbFaceList.ItemIndex].FacePosition do
if w <> 0 then // Если обнаружили
    begin
left := xc - w div 2;
top := yc - w div 2;
right := xc + w div 2;
bottom := yc + w div 2;

imgSource.Canvas.Rectangle(left, top, right, bottom);
end;

   // Рисуемпозициилица
```

```
with FaceList[lbFaceList.ItemIndex] do
if (FacialFeatures[0].x <> 0) and (FacialFeatures[1].x <> 0) then // Еслиобнаружено
for i := 0 to 1 do
begin
imgSource.Canvas.Pen.Color := clBlue; // Глазавыделяемсинимцветом

      FSDK_DetectFacialFeaturesinRegion(ImageHandle, @FacePosition, @FacialFeatures);
for y := 2 to FSDK_FACIAL_FEATURE_COUNT - 1 do
imgSource.Canvas.Ellipse(FacialFeatures[y].x - 2, FacialFeatures[y].y - 2, FacialFeatures[y].x + 2,
FacialFeatures[y].y + 2);


imgSource.Canvas.Ellipse(FacialFeatures[i].x - 2, FacialFeatures[i].y - 2,
      FacialFeatures[i].x + 2, FacialFeatures[i].y + 2);
end;
end
else
imgSource.Picture.Assign(nil);

end;

procedure TMainForm.EnrollPicture(FileName: string);
var
hbitmapHandle: integer;
imageHandle: integer;
r: integer;
k: integer;
w: integer;
imageWidth, imageHeight: integer;
ratio: double;
begin
k := length(FaceList);
setlength(FaceList, k+1);
FillChar(FaceList[k], SizeOf(FaceList[k]), 0);
```

```
r := FSDK_LoadImageFromFile(@ImageHandle, PAnsiChar(AnsiString(FileName)));
if r <> FSDKE_OK then
Application.MessageBox('Ошибкапризагрузкифайла', 'Ошибка')
else
begin
LogMemo.Lines.Add('Загружаю ' +  ExtractFileName(FileName));
   FSDK_GetImageWidth(imageHandle, @imageWidth);
   FSDK_GetImageHeight(imageHandle, @imageHeight);
ratio := Min((imgSource.Width + 0.4) / imageWidth, (imgSource.Height + 0.4) / imageHeight);
   FSDK_CreateEmptyImage(@FaceList[k].imageHandle);
   FSDK_ResizeImage(imageHandle, ratio, FaceList[k].ImageHandle);
   FSDK_FreeImage(imageHandle);
   FSDK_SaveImageToHBitmap(FaceList[k].ImageHandle, @hbitmapHandle);

   FaceList[k].ImageBmp := TBitMap.Create;
   FaceList[k].ImageBmp.Handle := hbitmapHandle;
   FaceList[k].ImageFileName := FileName;

lbFaceList.Items.Add(ExtractFileName(FileName));
lbFaceList.ItemIndex := k;

   RedrawCurrentImage;
   Application.processmessages;
   FSDK_SetFaceDetectionParameters(false, true, 384);
   FSDK_SetFaceDetectionThreshold(FaceDetectionThreshold);
r := FSDK_DetectFace(FaceList[k].ImageHandle, @FaceList[k].FacePosition);

if r <> FSDKE_OK then
begin
if OpenDialogMulti.Files.Count <= 1 then
Application.MessageBox('Необнаруженылицанафото. Попытайтесь уменьшить минимальное
количество лица на фото.', 'Ошибка при загрузки')
    else
LogMemo.Lines.Add(ExtractFileName(FileName) + ': Необнаружены лица на фото
Попытайтесь уменьшить минимальное количество лица на фото..')
```

```
  end
  else
  begin
RedrawCurrentImage;
    Application.processmessages;

    // вырезаннелицанафотографии
    FSDK_CreateEmptyImage(@ImageHandle);
with FaceList[k].FacePosition do
     FSDK_CopyRect(FaceList[k].ImageHandle, xc - round(w*0.5), yc - round(w*0.5), xc +
round(w*0.5), yc + round(w*0.5), ImageHandle);
    FSDK_GetImageWidth(ImageHandle, @w);
    FSDK_CreateEmptyImage(@FaceList[k].FaceImageHandle);
    FSDK_ResizeImage(ImageHandle, FacePreviewWidth/w, FaceList[k].FaceImageHandle);
    FSDK_SaveImageToHBitmap(FaceList[k].FaceImageHandle, @hbitmapHandle);
    FSDK_FreeImage(ImageHandle);

    FSDK_FreeImage(FaceList[k].FaceImageHandle);

    FaceList[k].FaceImageBmp := TBitMap.Create;
    FaceList[k].FaceImageBmp.Handle := hbitmapHandle;
    // redraw list
    lbFaceList.Repaint;

r := FSDK_DetectEyesInRegion(FaceList[k].ImageHandle, @FaceList[k].FacePosition
@FaceList[k].FacialFeatures);
if r <> FSDKE_OK then
Application.MessageBox('Ошибкаприопределениичертылица.', 'Ошибка' )
else
begin
    RedrawCurrentImage;
    Application.processmessages;

r := FSDK_GetFaceTemplateInRegion(FaceList[k].ImageHandle, @FaceList[k].FacePosition,
@FaceList[k].Template);
```

```
if r <> FSDKE_OK then
Application.MessageBox('Ошибкаприполученииишаблоналица.', 'Ошибка' );
end;
end;
end;

if r <> FSDKE_OK then
begin
  // Удалитьзагруженноефото
if FaceList[k].ImageBmp <> nil then
begin
   FSDK_FreeImage(FaceList[k].ImageHandle);
   FaceList[k].ImageBmp.Free;
end;
if FaceList[k].FaceImageBmp <> nil then
begin
   FaceList[k].FaceImageBmp.Free;
   FSDK_FreeImage(FaceList[k].FaceImageHandle);
end;

lbFaceList.Items.Delete(lbFaceList.ItemIndex); // Удалитьэлементылиста
SetLength(FaceList, k); // Удалить последний элемент
lbFaceList.ItemIndex := k-1;

  RedrawCurrentImage;
exit;
end;

LogMemo.Lines.Add('Файл- ' +  ExtractFileName(FileName) );
end;

procedure TMainForm.EnrollPictures1Click(Sender: TObject);
var
i: integer;
begin
```

```
try
if OpenDialogMulti.Execute then
for i := 0 to OpenDialogMulti.Files.Count - 1 do
begin
Enabled := false;
EnrollPicture(OpenDialogMulti.Files[i]);
    Application.processmessages;
Enabled := true;
end;
finally
end;
end;

procedure TMainForm.ClearDatabase1Click(Sender: TObject);
var
k: integer;
begin
for k := 0 to length(FaceList) - 1 do
begin
  FSDK_FreeImage(FaceList[k].ImageHandle);
  FaceList[k].ImageBmp.Free;

if FaceList[k].FaceImageBmp <> nil then
begin
   FaceList[k].FaceImageBmp.Free;
   FSDK_FreeImage(FaceList[k].FaceImageHandle);
end;
end;

 lbFaceList.Items.Clear;
SetLength(FaceList, 0);

 RedrawCurrentImage;
end;
```

```
procedure TMainForm.MatchPictures1Click(Sender: TObject);
var
hbitmapHandle: integer;
 image2Handle: integer;
imageWidth, imageHeight: integer;
ratio: double;
begin
if length(FaceList) = 0 then
begin
Application.MessageBox('Пожалуйстазагрузитефотослицами', 'Ошибка');
exit;
end;

if OpenDialog1.Execute then
begin
FillChar(frmSearchResults.Face, SizeOf(frmSearchResults.Face), 0);
with frmSearchResults.Face do
begin
if FSDK_LoadImageFromFile(@ImageHandle, PAnsiChar(AnsiString(OpenDialog1.FileName)))
<> FSDKE_OK then
begin
Application.MessageBox('Ошибкапризагрузкефото', 'Ошибка');
exit;
end
else
begin
      FSDK_GetImageWidth(imageHandle, @imageWidth);
      FSDK_GetImageHeight(imageHandle, @imageHeight);
ratio := Min((frmSearchResults.imgSource.Width + 0.4) / imageWidth,
(frmSearchResults.imgSource.Height + 0.4) / imageHeight);
      FSDK_CreateEmptyImage(@image2Handle);
      FSDK_ResizeImage(imageHandle, ratio, image2Handle);
      FSDK_CopyImage(image2Handle, imageHandle);
      FSDK_FreeImage(image2Handle);
      FSDK_SaveImageToHBitmap(ImageHandle, @hbitmapHandle);
```

```
ImageBmp := TBitMap.Create;
ImageBmp.Handle := hbitmapHandle;

    FSDK_SetFaceDetectionParameters(false, true, 384);
    FSDK_SetFaceDetectionThreshold(MainForm.FaceDetectionThreshold);
if FSDK_DetectFace(ImageHandle, @FacePosition) <> FSDKE_OK then
Application.MessageBox('Необнаружены лица на фото. Попытайтесь уменьшить минимальное
количество лица на фото.', 'Ошибка при загрузке' )
else
if FSDK_DetectEyesInRegion(ImageHandle, @FacePosition, @FacialFeatures) <> FSDKE_OK
then
Application.MessageBox('Ошибка при определении черт лица.', 'Ошибка' )
else
if FSDK_GetFaceTemplateInRegion(ImageHandle, @FacePosition, @Template) <> FSDKE_OK
then
Application.MessageBox('Ошибка при получении шаблона лица.', 'Ошибка' )
else
        frmSearchResults.ShowModal;
end;
end;
  FSDK_FreeImage(frmSearchResults.Face.ImageHandle);
  frmSearchResults.Face.ImageBmp.Free;
end;
end;
end.
```

СПИСОК ОСНОВНОЙ ЛИТЕРАТУРЫ

1. Указ Президента Республики Узбекистан от 7 февраля 2017 года №УП-4947 «О стратегии действий по дальнейшему развитию Республики Узбекистан».
2. Форсайт Д., Понс Ж. Компьютерное зрение. Современный подход. Вильямс, 2004
3. Гонсалес Р., Вудс Р. Цифровая обработка изображений. М.: Техносфера, 2005.
4. Красильников Н. Цифровая обработка 2D- и 3D- изображений. СПб: БХВ-Петербург, 2011.
5. Viola P., Jones M.J. Robust real-time face detection. // Intern. J. of Computer Vision. 2004. Vol. 57, iss. 2. P 137 – 154

СПИСОК ДОПОЛНИТЕЛЬНОЙ ЛИТЕРАТУРЫ

6. Mirziyoyev Sh.M. Buyuk kelajagimizni mard va olijanob xalqimiz bilan birga quramiz. 2017.
7. Mirziyoyev Sh.M. Qonun ustuvorligi va inson manfaatlarini ta'minlash – yurt taraqqiyoti va xalq farovonligining garovi. 2017.
8. Mirziyoyev Sh.M. Erkin va farovon, demokratik O'zbekiston davlatini birgalikda barpo etamiz. 2017
9. Мирзиёв Ш.М. Танқидий таҳлил, қатъий тартиб-интизом ва шахсий жавобгарлик – ҳар бир раҳбар фаолиятининг кундалик қоидаси бўлиши керак. Ўзбекистон Республикаси Вазирлар Маҳкамасининг 2016 йил якунлари ва 2017 йил истикболларига бағишланган мажлисидаги Ўзбекистон Республикаси Президентининг нутқи.// Халқ сўзи газетаси. 2017 йил 16 январь, №11.
10. Татаренков Д. А. Анализ методов обнаружения лиц на изображении // Молодой ученый. 2015. № 4. С. 270 – 276.

11.Matthews Iain, Baker Simon. Active Appearance Models Revisited, International Journal of Computer Vision, Vol. 60, No. 2, November, 2004, pp. 135-164.

12.Фозилов Ш.Х., Мирзаев Н.М., Акромов А.А., Тўхтасинов М.Т., Юзнинг геометрик белгилари асосида шахсни идентификация қилиш алгоритмлари, Информатика институти, 2005й.

13.Абрамов Н. С., Хачумов В. М. Распознавание на основе инвариантных моментов // Вестник РУДН. Серия: Математика. Информатика. Физика. — 2014. — № 2. — С.142-149.

14.Хрулев А. Системы распознавания лиц. Состояние рынка. Перспективы развития. Журнал «Системы безопасности». 2012. – № 1.

15.Фазылов Ш., Тухтасинов М., Старовойтов В., Самаль Д., Ригол Г. Локализация фрагментов лица на цветных фотопортретах// Доклады 4-й Международной научной конференции "Обработка информации и управление в чрезвычайных и экстремальных ситуациях". Минск, 2004.С. 218-223.

16. Активные модели внешнего вида - https://habrahabr.ru/post/155759/

17. Анализ существующих подходов к распознаванию лиц URL.:https://habrahabr.ru/company/synesis/blog/238129/

18. Современные биометрические методы идентификации. URL: URL.:https://habrahabr.ru/post/126144/

19. Культин Н. Основы программирования в Delphi XE — С.: «БХВ-Петербург», 2011

20. Мартин Р. Ньюкирк В., Косс Р. Быстрая разработка программ. Принципы, примеры, практика/ Мартин Р.К.,.; М.: Вильямс, 2004.

21. Введенский Б. Современные системы охраны периметров // URL.: http://www.r-kontrol.ru/articles/system-video/article-11.

22. Щербина В. Особенности охраны периметра // Сетевой журнал «Бизнес.Онлайн». URL: http://sec.bl.by/articles/detail178418.

23. Hughes P., Ferrett E. Introduction to Health and Safety at Work. The Boulevard, Langford Lane, Kidlington, Oxford OX5 1GB, UK. ISBN: 978-0-08-097070-7. 2011/ p-636/

24. Goldfein M, Ivanov A, Kozhevnikov N, Kozhevnikov V, Fundamentals of General Ecology, Life Safety and Environment Protection.. Nova Science Publishers, Inc. p-96. (April 25, 2013).

25. Кривошеин Д., Муравей Л., Роева Н. Экология и безопасность жизнедеятельности: Учеб.пособие для вузов/; Под ред. Муравья Л. – М.: ЮНИТИ-ДАНА, 2000.

СОДЕРЖАНИЕ

МЕТОДИЧЕСКОЕ ПОСОБИЕ ПО ВЫПОЛНЕНИЮ КУРСОВОЙ РАБОТЫ ПО ПРЕДМЕТУ «ОБРАБОТКА ЦИФРОВОЙ ИНФОРМАЦИИ»

5350200 - Televizion texnologiyalar (Audiovizual texnologiyalar) mutaxassisligi talabalari uchun uslubiy qo'llanma

AVT kafedrasining 2018 yil 25 sentyabr (5-sonli bayonnoma) majlisida ko'rib chiqildi va chop etishga tavsiyalandi

TT fakultetining ilmiy-uslubiy Kengashida ko'rib chiqildi va chop etishga tavsiyalandi
2018 yil 25-sentyabr 2-sonli bayonnoma

TATU ilmiy-uslubiy Kengashida ko'rib chiqildi va chop etishga tavsiyalandi
2018 yil _23-oktyabrdagi 4(116) –sonli bayonnoma

Muallif: Sh.T.Kasimova,

Taqrizchilar: Q. Raxmanov
E. Nazirova

Mas'ul muharrir: S.Beknazarova
Musahhih: N.X.Raximova

Формат 60x84 1/16. Печ.лист 5,25.
Заказ №293. Тираж 50.
Отпечатано в «Редакционно издательском»
отделе при ТУИТ.
Ташкент ул. Амир Темур, 108.