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

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

ДОПУСТИТЬ К ЗАЩИТЕ
Зав. кафедрой
_____ А.В. Бушманов
«_____»_____2019 г.

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

на тему: Разработка системы управления графическим интерфейсом компьютера на основе положения зрачка глаза человека с использованием многопоточных вычислений.

Исполнитель
студент группы 757об _____ А.В. Федцов
(подпись, дата)

Руководитель
доцент, канд. техн. наук _____ Н.П. Семичевская
(подпись, дата)

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

Руководитель
доцент, канд. техн. наук _____ Н.П. Семичевская
(подпись, дата)

Рецензент
_____ _____ _____
(подпись, дата)

Благовещенск 2019

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

Факультет математики и информатики
Кафедра информационных и управляющих систем

УТВЕРЖДАЮ
Зав.кафедрой
_____А.В.Бушманов
«_____»_____2017 г.

З А Д А Н И Е

К магистерской работе студента Федцова Алексея Владимировича.
Тема магистерской работы: Разработка системы управления графическим интерфейсом компьютера на основе положения зрачка глаза человека с использованием многопоточных вычислений.
(утверждено приказом от 26.05.2019 № 950-уч)

1. Срок сдачи студентом законченной работы 21.06.2019 г.
2. Исходные данные к выпускной квалификационной работе: отчет по преддипломной практике.
3. Содержание магистерской работы: анализ предметной области, проектирование информационной системы, разработка программного обеспечения, рассмотрение аспектов безопасности жизнедеятельности.
4. Перечень материалов приложения: блок-схемы, изображения экранных форм.

Руководитель магистерской работы Семичевская Наталья Петровна, доцент, канд. техн. наук.
Задание принял к исполнению (дата):_____А.В. Федцов

РЕФЕРАТ

Магистерская работа содержит 79 с., 37 рисунков, 2 приложения, 9 источников.

ПРОЕКТИРОВАНИЕ, РАЗРАБОТКА, КОМПЬЮТЕНОЕ ЗРЕНИЕ, ПРИЛОЖЕНИЕ, ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ, СИСТЕМА, АНАЛИЗ, ВЕБ-КАМЕРА, ВИДЕОПОТОК, БЕЗОПАСНОСТЬ ЖИЗНЕДЕЯТЕЛЬНОСТИ

Для данной выпускной квалификационной работы объектом исследования была выбрана технологическая дисциплина «компьютерное зрение».

Целью работы является проектирование и разработка программного приложения для «системы управления графическим интерфейсом компьютера на основе положения зрачка глаза человека».

Работа выполнялась последовательно в соответствие со следующими этапами: анализ предметной области, проектирование информационной системы, разработка приложения, рассмотрение аспектов безопасности жизнедеятельности.

Разработанная система и программное обеспечение к ней позволят управлять процессами и работать с приложениями на персональном компьютере с помощью движений зрачка глаза.

Система разработана в первую очередь для людей с ограниченными возможностями, но и обычные люди так же смогут использовать систему для взаимодействия с компьютером.

СОДЕРЖАНИЕ

НОРМАТИВНЫЕ ССЫЛКИ

В настоящей магистерской работе использованы ссылки на следующие стандарты и нормативные документы:

ГОСТ 2.104-68 ЕСКД Основные надписи

ГОСТ 2.105-95 ЕСКД Общие требования к текстовым документам

ГОСТ 2.106-96 ЕСКД Текстовые документы

ГОСТ 2.111-68 ЕСКД Нормоконтроль

ГОСТ 2.306-68 ЕСКД Обозначение графических материалов и правил нанесения их на чертежах

ГОСТ 2.316-68 ЕСКД Правила нанесения на чертежах надписей, технических требований и таблиц

ГОСТ 2.701-84 ЕСКД Схемы. Виды и типы. Общие требования к выполнению

ГОСТ 2.721-74 ЕСКД Обозначения условно-графические в схемах. Обозначения общего применения

ГОСТ 3.1103-83 ЕСКД Основные надписи

ГОСТ 3.1105-84 ЕСКД Правила оформления документов общего назначения

ГОСТ 3.1130-93 ЕСКД Основные требования к формам и бланкам документов

ГОСТ 34.601-90 Информационная технология. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Стадии создания

ГОСТ 34.602-89 Техническое задание на создание автоматизированной системы

ГОСТ 30772-2001 Ресурсосбережение. Обращение с отходами. Термины и определения

ГОСТ Р ИСО 6385-2007 Эргономика. Применение эргономических принципов при проектировании производственных систем

СП 12.13130.2009 Определение категорий помещений, зданий и наружных установок по взрывопожарной и пожарной опасности

СанПиН 2.2.2/2.4.1340-03 Гигиенические требования к персональным электронно-вычислительным машинам и организации работы

Федеральный закон "Об отходах производства и потребления" от 24.06.1998 N 89-ФЗ (ред. от 28.12.2016)

ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

АС – автоматизированная система;

БД – база данных;

БЖД – безопасность жизнедеятельности;

ВКР – выпускная квалификационная работа;

ИС – информационная система;

НПА – нормативно-правовой акт;

НФ – нормальная форма;

ОС – операционная система;

ПК – персональный компьютер;

ПО – программное обеспечение;

РД – руководящий документ;

РФ – Российская Федерация;

СВТ – средство вычислительной техники;

СУБД – система управления базами данных;

СУ – система управления;

ТЗ – техническое задание;

ЧС – чрезвычайная ситуация;

ВВЕДЕНИЕ

Компьютерное зрение — набор алгоритмов, знаний и способов разработки приложений, которые могут производить обнаружение, отслеживание и классификацию визуальных объектов. Это очень перспективное и современное направление, а так же одна из составляющих искусственного интеллекта. Компьютерное зрение позволяет компьютерным системам видеть - то есть воспринимать и обрабатывать информацию в с изображений.

В современном мире компьютерное зрение активно используется в различных отраслях, таких как медицина, производство, контроль процессов и объектов, безопасность. Использование компьютерного зрения для создания системы управления позволит компьютеру видеть действия пользователя и совершать взаимодействие с ним без прямого физического контакта. Эта возможность особенно актуальна для людей с ограниченными возможностями, но и обычные люди так же смогут использовать систему для взаимодействия с компьютером.

Для создания ИС были определены следующие задачи:

– проанализировать предметную область компьютерного зрения;

– обосновать необходимость создания системы управления с использованием компьютерного зрения;

– поставить задачу разработки системы;

– определить компоненты будущей системы;

– проектирование ИС;

– разработка ИС;

– рассмотреть аспекты БЖД пользователей системы;

– разработать руководство для пользователей системы.

1 АНАЛИЗ ВОЗМОЖНОСТЕЙ ИСПОЛЬЗОВАНИЯ КОМПЬЮТЕРНОГО ЗРЕНИЯ

Этап анализа предметной области является общим при проектировании и разработке СУ. Он позволяет собрать сведения о предметной области и на их основе формировать видение будущей системы, выбирать технологии создания, а так же определять набор требований к системе.

1.1 Область применения технологий компьютерного зрения.

Компьютерное является научной дисциплиной и относится к теориями и технологиям создания искусственных систем, которые получают информацию из изображений. На вход таких система могут поступать достаточно большое количество видов данных, таких как видеопоследовательность, изображения с различных камер или трехмерными данными, например с устройства Kinect или медицинского сканера.

Компьютерное зрение обширно применяется на практике.

Примеры применения систем компьютерного зрения:

– системы управления процессами (промышленные роботы, автономные транспортные средства. Например, системы проверки качества изготавливаемой на заводе продукции;

– системы видеонаблюдения. В этих системах с помощью компьютерного зрения происходит обнаружение движения и отслеживание перемещения объектов;

– системы организации информации (например, для индексации баз данных изображений). В таких системах компьютерное зрение используется так же для распознавания текстов и штрих-кодов;

– системы моделирования объектов или окружающей среды (анализ медицинских изображений, топографическое моделирование);

– системы взаимодействия (например, устройства ввода для системы человеко-машинного взаимодействия);

– системы дополненной реальности;

– вычислительная фотография, например для мобильных устройств с камерами.

Сфера применения компьютерного зрения стремительно расширяется, а рост вычислительных мощностей компьютерной техники открывает новые возможности для развития этого направления.

1.2 Обоснование разработки системы управления с использованием компьютерного зрения.

Одним из популярных направлений использования компьютерного зрения является создание систем человеко-машинного взаимодействия.

В своей работе я создам систему, которая позволит осуществлять взаимодействие пользователя с компьютером с помощью движений глаз.

Система позволит управлять процессами и работать с приложениями без физического контакта с компьютером. Для работы системы потребуется обычная веб-камера, которую уже имеют или могут позволить приобрести большинство пользователей персональных компьютеров.

Для использования системы пользователю персонального компьютера нужно быть в поле зрения в веб-камеры и выполнять движения зрачком глаза. Движения зрачка будет определяться системой и из этих движений будут формироваться управляющие команды.

Использовать такою систему сможет владелец персонального компьютера, имеющий веб-камеру.

Особенно актуальным станет использование этой системы для людей с ограниченными возможностями, так как для работы с системой необходимы только движения зрачком глаза.

С помощью этой системы можно будет управлять приложениями, которые выполняются на персональном компьютере. Для интеграции системы с приложением будет создан программный интерфейс, который позволит разработчикам встраивать мою систему в их собственные приложения.

Возможность интеграции откроет широкие возможности использования системы. Например, с помощью движений зрачка можно будет переклю-

чать каналы телевизора, изменять громкость звука, вызвать экстренную помощь, и даже и просто поиграть в игру.

Это так же будет в первую очередь актуально для людей с ограниченными возможностями, но и обычные люди так же смогут использовать систему для взаимодействия с компьютером.

1.3 Анализ существующих методов поиска объектов в области компьютерного зрения.

Для большинства задач, решаемых с помощью компьютерного зрения уже разработаны методы решения. Одной из задач, которую потребуется решить в процессе разработки системы является задача поиска положения лица человека на изображении. Метод, используемый для решения задачи должен иметь высокое быстродействие, так как его работа будет производится в реальном времени, а так же не зависеть от размеров лица относительно изображения. Для выбора оптимального способа решения этой задачи произведем анализ и сравнение всех имеющихся методов поиска объектов на изображении:

а) методы поиска по шаблону. Детектирование объектов на основании некоторого шаблона предполагает, что имеется изображение объекта с выделенными признаками – шаблон, и тестовое изображение, которое сопоставляется этому шаблону. В примитивном варианте происходит сравнения шаблона со всеми участками изображения, по размеру совпадающие с шаблоном и находится участок который наиболее соответствует шаблону. Преимуществом метода является простота создания шаблона искомого объекта. Минусом метода является невозможность поиска нескольких объектов, зависимость от размеров объекта, а так же низкая производительность;

б) методы каскада перепада яркостей. Метод заключается в создании каскада примитивов перепадов яркости искомого объекта и поиском участка изображения, соответствующего каскаду примитивов. Метод состоит из двух алгоритмов: алгоритм обучения и алгоритм распознавания. Алгоритм обучения позволяет создать каскад примитивов перепадов яркости для иско-

мого объекта. Алгоритм распознавания позволяет найти искомый объект на изображении. Преимуществом метода является быстрая скорость работы алгоритма распознавания, независимость от размера искомого объекта на изображении, а так же возможность обнаружения нескольких объектов. Недостатками метода является длительное время работы алгоритма обучения, в процессе которого алгоритму необходимо проанализировать большое количество тестовых изображений. Этот метод назван «Методом Хаара и Виолы–Джонса» в честь его авторов. Он так же является самым популярным для поиска лиц на изображении;

в) методы ключевых точек. Метод заключается в выделении ключевых точек искомого объекта и их признаков и дальнейшем поиске объекта по ключевым точкам на изображении. Признаки (Дескрипторы) строятся на основании информации об интенсивности, цвете и текстуре особой точки. Особые точки могут представляться углами, ребрами, контуром и другими сложными свойствами объекта. Преимуществом метода является независимость размера и наклона искомого объекта на изображении. Недостатком метода является сложность используемого математического аппарата для нахождения совокупности ключевых точек, способной с требуемой точностью описать искомый объект, а так же условия использования эффективных реализаций метода. Дело в том что этот метод запатентован и использование большинства математических описаний и программных реализаций алгоритмов этого метода доступно только на коммерческой основе;

г) методы поиска объектов по контуру. Под контуром понимается кривая, которая описывает границу объекта на изображении. В этих методах контур искомого объекта описывается в виде формул геометрических примитивов. Использование данного подхода предполагает, что контур содержит достаточно информации о форме объекта, при этом внутренние точки не учитываются. В процессе нахождения объекта происходит нахождение контуров на изображении и поиск наиболее соответствующего контура. Преимуществом метода является высокая скорость работы и независимость ис-

комого объекта от размера и наклона на изображении. Недостатком метода является возможность искажения контура искомого объекта в зависимости от фона и шумов на изображении.

В итоге, для эффективного поиска лица на изображении был выбран Метод каскада перепада яркостей «Хаара и Виолы–Джонса», так как этот метод отвечает требованиям быстродействия и условиям возможного размещения искомого объекта в поставленной задачи.

2 ПРОЕКТИРОВАНИЕ СИСТЕМЫ

Этап проектирования информационной системы является основополагающим, поскольку именно от приложенных усилий и уровня проработки данного аспекта зависит эффективность работы системы.

2.1 Постановка цели проектирования и анализ требований к системе.

Цель проектирования - создать систему, которая сможет определять положения зрачка человека и на основе положения зрачка формировать команды управления компьютером (далее, система).

С точки зрения теории управления, для управления объектом система должна работать следующим образом:

Оператор взаимодействует с компьютером с через систему. На компьютере выполняется приложение, которое позволяет управлять каким-либо объектом. Приложение для управления объектом взаимодействует с системой через программный интерфейс.

Оператор совершает движение зрачком глаза. Система распознает положение зрачка, формирует команду управления и через программный интерфейс передает ее приложению, которое управляет объектом. Приложение принимает команду управления от системы, и формирует управление для объекта. Приложение получает обратную связь от объекта и передает ее системе через программный интерфейс. Система передает сообщение оператору.

Объектом управления может быть объект, которым можно управлять с помощью приложения для компьютера. Объект не обязательно должен быть физическим. Условием возможности управления так же является достаточность списка команд управления, поступающих от системы через программный интерфейс в приложение для полноценного управления объектом. Например, с помощью системы можно управлять активным пунктом меню

компьютера, громкостью звука музыкального плеера, яркостью экрана монитора, итд.

Но нельзя управлять сразу всеми системами самолета, так как для этого не хватит управляющих команд, которые можно сформировать исходя из движений зрачка.

Схема интеграции системы с приложением для управления объектом представлена ниже. Программный интерфейс обозначен как API.

Рисунок 1 – Схема интеграции системы в систему управления объектом

Схема взаимодействия аппаратного обеспечения следующая. Для работы системы будет использоваться веб-камера и персональный компьютер на базе ОС Windows. Веб-камера используется для захвата изображения зрачка глаза оператора и подключается к компьютеру через последовательный интерфейс для подключения периферийных устройств USB. Отображения информации оператору используется монитор.

Монитор подключается к компьютеру через видеоинтерфейсы VGA или HDMI.

Ниже представлена схема взаимодействия оператора и аппаратного обеспечения системы.

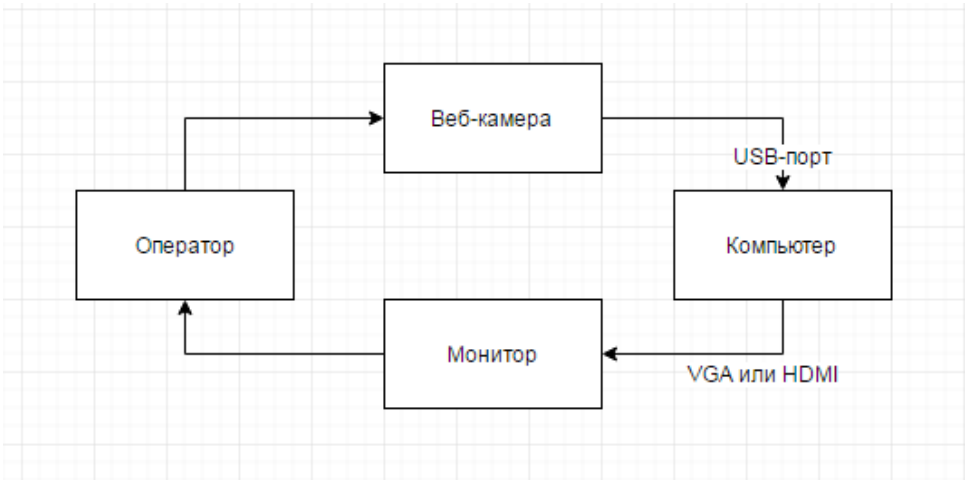


Рисунок 2 – Схема взаимодействия оператора и аппаратного обеспечения системы

Минимальное разрешение изображения, на котором возможно стабильно определить положение глаза и зрачка это 640x480 пикселей. Следовательно, матрица веб-камеры должна иметь разрешение не менее чем 640x480 пикселей. При меньшем разрешении информации о положении глаза на изображении будет недостаточно, следовательно определить положение зрачка будет невозможно.

Пример интеграции системы с приложением: Оператор управляет приложением "Просмотр фотографий" на персональном компьютере. Объект управления - фотография из галереи, которая показывается оператору. Приложение интегрировано с системой. Приложение направляет запрос оператору "Открыть следующее изображение в галерее?". Система обрабатывает запрос и показывает оператору диалоговое окно с вопросом. Оператор с помощью движения зрачка отвечает на вопрос. Во время чтения вопроса оператор смотрит по центру. Если оператор посмотрит влево, он ответит "Да", если посмотрит вправо, то ответит "Нет". Ответ распознается системой, и передается через программный интерфейс в приложение "Просмотр фотографий". Если оператор ответил "Да", то программа покажет следующее изображение. Если "Нет", то останется текущее изображение, и через некоторое время программа повторит вопрос оператору. Таким образом, происходит взаимодействие оператора и программы с помощью системы.

Для разрабатываемой системы предъявляются следующие требования:

– система должна обеспечить интерфейс управления приложениями на персональном компьютере на основе движения зрачка глаза оператора;

– система должна иметь возможность интеграции с приложениями сторонних разработчиков через программный интерфейс;

– минимальное разрешение матрицы веб-камеры, которое поддерживает система: 640x480 пикселей;

– система должно иметь интуитивно-понятный графический интерфейс;

– система должна учитывать тот факт, что в процессе взаимодействия с графическим интерфейсом системы и чтения текстовой информации пользователь так же совершает движения зрачком и может вызвать ложные срабатывания.

2.2 Описание функциональных подсистемы.

В данном подразделе более подробно будут рассмотрены функциональные подсистемы с целью боле полного понимания решаемых ими задач и, тем самым, выбора наиболее точного способа их дальнейшей реализации.

Общая схема работы системы представлена в приложении А рисунок 1.

На вход системы подаются запросы от интегрированного через программный интерфейс приложения и действия оператора. На выходе системы создаются команды управления приложением через программный интерфейс и сообщения оператору.

Схема соединения функциональных подсистем представлена в приложении А рисунок 2.

2.2.1 Функция получения видеопотока с Веб - Камеры.

Получить данные с веб-камеры можно в виде видеопотока.

Видеопоток - это последовательность кадров, получаемая с веб-камеры в режиме реального времени. Кадр представляет собой обычное изображение и кодируется в виде матрицы.

Основные характеристики видеопотока - скорость кадров в секунду и разрешение получаемых изображений.

Для взаимодействия веб-камеры и операционной системы используется драйвер. В большинстве ОС, например в ОС Windows драйвер поставляется вместе с операционной системой.

Получение видеопотока с веб-камеры происходит с помощью драйвера. Драйвер имеет программный интерфейс и предоставляет видеопоток в виде стандартных объектов операционной системы. Функция с помощью драйвера подключается к камере и получает от нее видеопоток. Полученный видеопоток будет использоваться в функциях обработки и распознавания.

Рисунок 3 – Схема функции захвата видеопотока

2.2.2 Функция преобразования видеопотока в изображения.

Обработка видеопотока сводится к обработке каждого кадра как отдельного изображения. Перед тем как работать с изображениями, их нужно извлечь из видеопотока. Эта функция примет на вход видеопоток и вернет изображение кадра в виде матрицы. Так как видеопоток это последовательность кадров - функция создаст цикл и будет в начале цикла создавать новое

изображение для каждого кадра. Дальнейшая обработка изображений видеопотока будет происходить именно в этом цикле.

2.2.3 Функция обработки и поиска объекта на изображении по методу Хаара и Вилоы-Джонса.

Функция произведет поиск лица и глаза на изображении. Метод Хаара позволяет найти объект на изображении по размещению перепадов яркости.

Для поиска объекта используется каскад примитивов яркости, соответствующий этому объекту.

Функция должна выполнить преобразование изображения по методу Хаара для нахождения положения искомого объекта. Преобразование выполняется для участков изображения с помощью сканирующего окна. Сканирующее окно перемещается над всеми возможными участкам изображения. Для каждого участка изображения происходит расчет перепадов яркости и сравнение с имеющимся каскадом. В результате преобразования происходит вычисление участка изображения, максимально соответствующего каскаду примитив яркости искомого объекта, в итоге находятся координаты области изображения, содержащей объект.

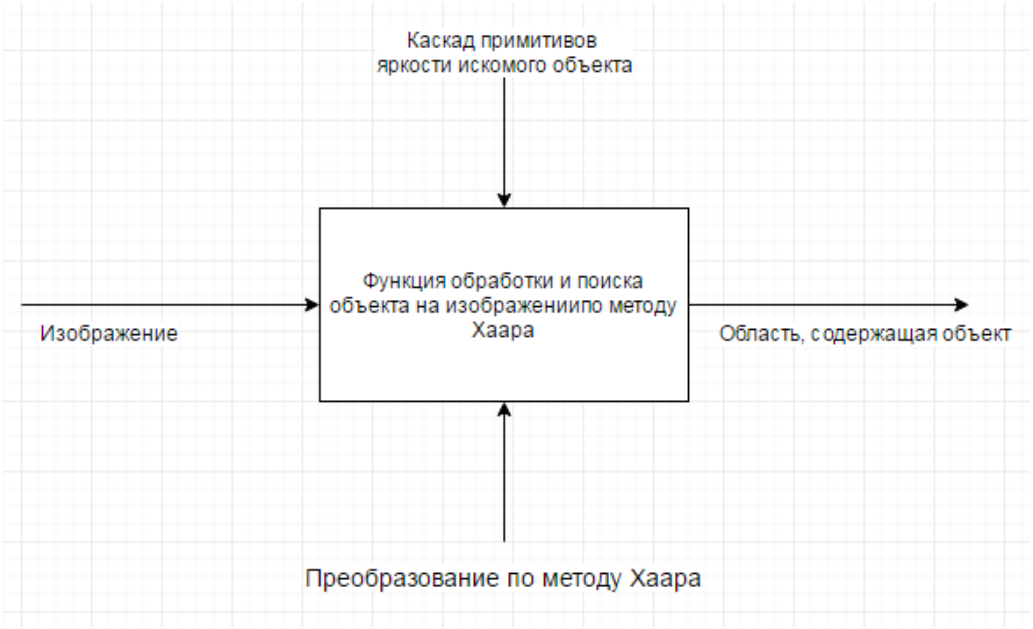


Рисунок 4 – Схема функции обработки и поиска объекта на изображении по методу Хаара

В системе функция будет находит положение лица и глаза оператора на изображении, которое получено из видеопотока.

2.2.4 Функция поиска положения и границ глаза на изображении лица человека.

Функция произведет поиск положения и границ глаза с помощью сверточной нейронной сети. Нейронная сеть будет использовать метод лицевой маски (англ. landmark или лицевой ориентир), который позволит определить границы участка изображения, содержащего глаз с учетом искажений, вызванных движениями мимики и поворотом головы.

2.2.5 Функция обработки и поиска положения зрачка на изображении глаза.

Функция получает изображение глаза на вход и преобразование изображения таким образом, чтобы отсеять на нем информацию, не связанную со зрачком. Затем информация связанная со зрачком преобразуется в координату положения зрачка относительно глаза, и из этой координаты вычисляется положение зрачка. В системе функция будет определять три положения зрачка: зрачок смотрит влево, зрачок смотрит по-центру, зрачок смотрит вправо.

Функция состоит из следующих подфункций:

- подфункция замены диапазонов цветов на изображении.

Эта подфункция должна произвести преобразование изображения таким образом, чтобы заменить участки изображения определенного диапазона цвета на требуемый цвет. В системе будет производиться замена цвета кожи (оттенки розового) на цвет глазного яблока (белый) для удаления информации о коже вокруг глаза с изображения;

- подфункция преобразования цветовой модели изображения.

Изображение получаемое с камеры кодируется в цветовой палитре RGB.

Для преобразований, связанных с заменой цвета на изображении в системе используется цветовая модель HSV;

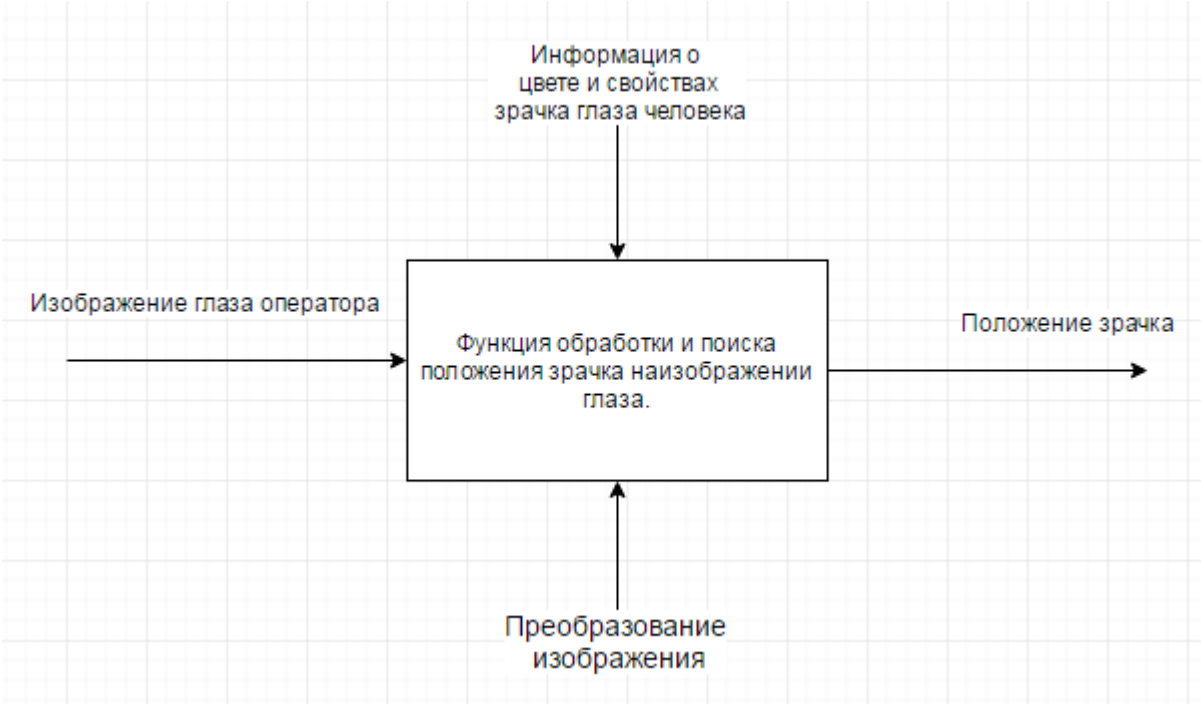


Рисунок 7 – Схема функции обработки и поиска положения зрачка на изображении глаза

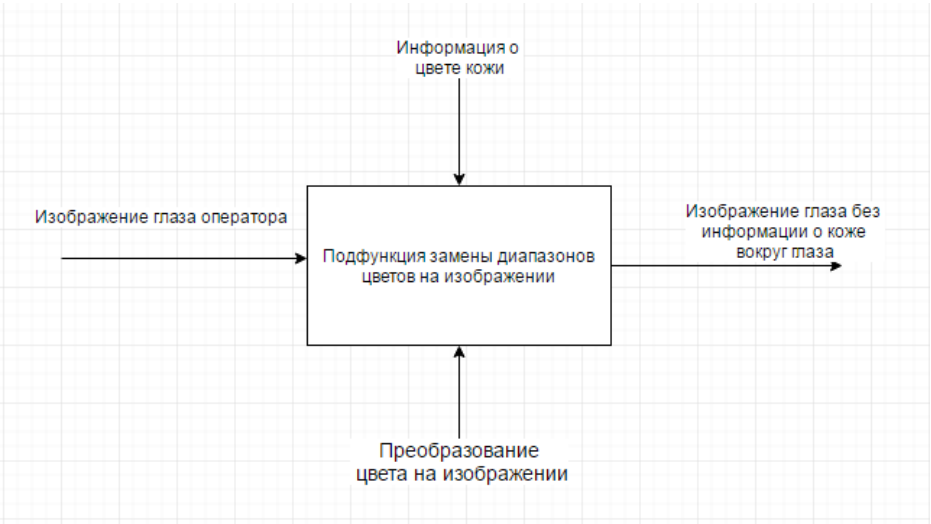


Рисунок 8 – Подфункция замены диапазонов цветов на изображении

– подфункция преобразования изображения в оттенки серого. Функция оставит только один канал цвета, содержащий грации серого.

В системе функция нужна для удаления цветовой составляющей изображения;

- подфункция бинаризации изображения. Эта функция нужна для разделения изображения в грациях серого на темные и светлые области по пороговому значению яркости, оставив только черные и белые значения. В системе эта функция оставит информацию о зрачке, так как он темнее и удалит всю информацию о глазном яблоке (так как оно светлее) В результате на изображении глаза останется только информация о зрачке;
- подфункция вычисления положения зрачка. Подфункция выполнит преобразования бинарного изображения таким образом, чтобы найти положение зрачка. Положением зрачка на изображении будет область, содержащая максимальный по площади черный участок. В результате работы функции найдется центр области изображения, содержащей зрачок;
- подфункция определения положения зрачка относительно глаза.

Подфункция произведет сравнение координаты центра зрачка относительно размеров глаза и определит положение зрачка относительно глаза.

2.2.6 Функция управления многопоточными вычислениями.

Функция должна определить конфигурацию текущего оборудования для использования многопоточных вычислений. Так же, функция должна запустить обработку изображений в многопоточном режиме, выбрав оптимальное количество потоков для используемого оборудования. Функция так же должна восстановить исходный порядок параллельно обработанных изображений, чтобы избежать ситуации, когда последующий кадр видеопотока был обработан раньше, чем предыдущий.

2.2.7 Функция взаимодействия с приложениями через программный интерфейс.

Функция должна обеспечить взаимодействие системы с интегрируемыми приложениями сторонних разработчиков.

Функция должна получить запрос от интегрированного приложения, отобразить графическую форму оператору, получить ответ от оператора и вернуть его в приложение.

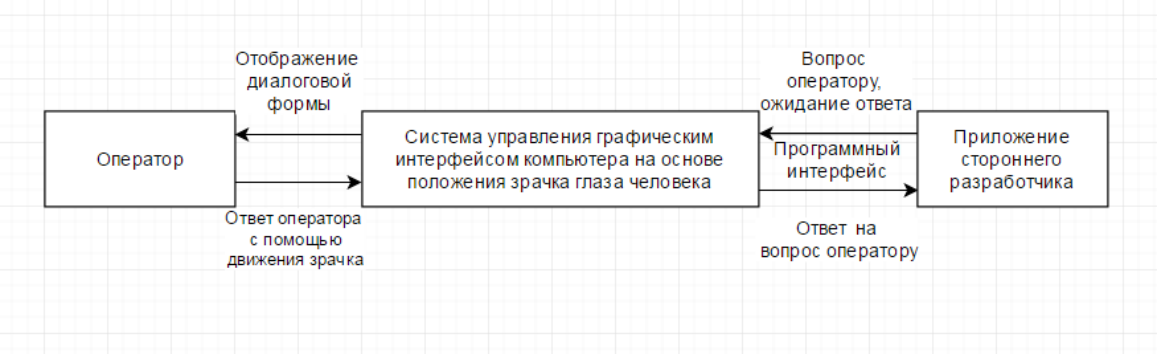


Рисунок 9 – Процесс взаимодействия оператора, системы и стороннего приложения с использованием программного интерфейса

При возникновении вопроса для оператора в стороннем приложении, оно отправляет запрос в систему через программный интерфейс. Система получает запрос и отображает его оператору в специальной графической форме, адаптированной под управление с помощью движения зрачка. Оператор производит ответ на вопрос с помощью движения зрачка, этот ответ интерпретируется системой и передается через программный интерфейс в стороннее приложение.

Программный интерфейс должен предоставить интегрированному приложению 3 вида взаимодействия с оператором:

- диалог с вопросом оператору и возможностью ответить "Да" или "Нет";
- меню с возможностью выбора одного объекта из списка;
- регулятор линейной величины в требуемом диапазоне (например, яркость, громкость, итд.

Функция должна содержать следующие подфункции:

- подфункция приема запроса по программному интерфейсу и вызова диалога взаимодействия с оператором;

Подфункция реагирует на входящий запрос от приложения по программному интерфейсу, обрабатывает его и показывает оператору диалог согласно требуемому способу взаимодействия. В диалог включается текст запроса, например текст вопроса или список пунктов меню;

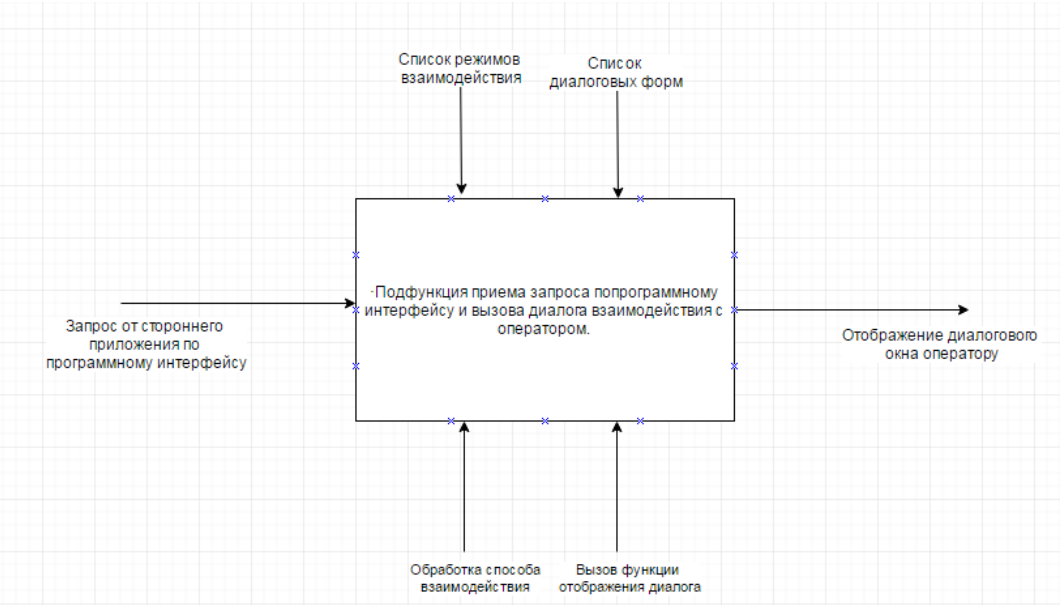


Рисунок 10 – Подфункция приема запроса по программному интерфейсу и вызова диалога взаимодействия с оператором

– подфункция запуска функций обработки и распознавания зрачка на изображения;

Подфункция запускает функции обработки и распознавания изображения и запускает процесс распознавания действия пользователя;

– подфункция интерпретации ответа оператора в соответствии с режимом взаимодействия с оператором.

Подфункция определяет ответ оператора на основе положения зрачка оператора и режима взаимодействия. На вход функции подается положение глаза. На выходе функция определит ответ оператора;

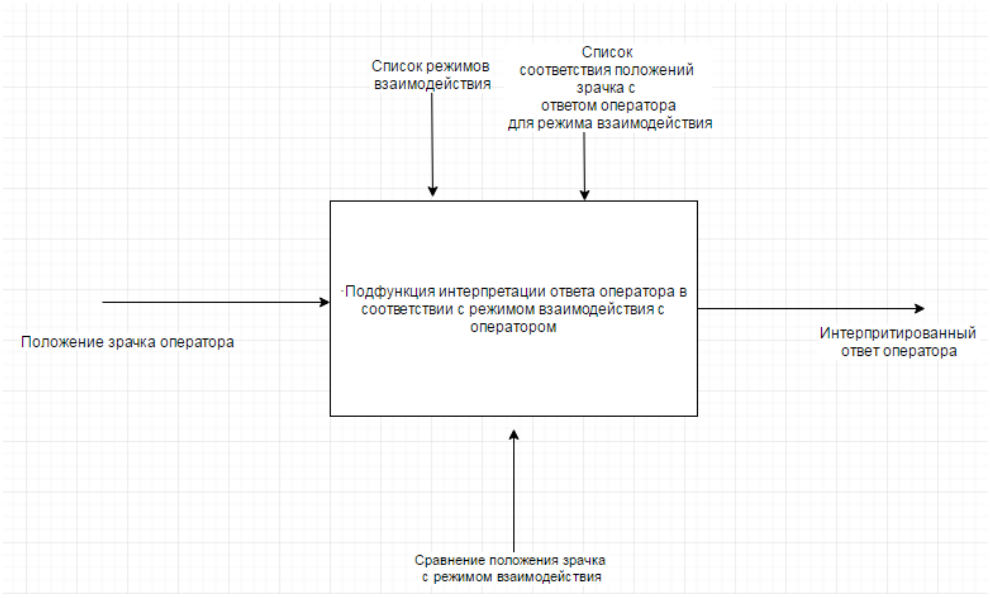


Рисунок 11 – Подфункция интерпретации ответа оператора в соответствии с режимом взаимодействия с оператором

– подфункция передачи ответа оператора в программный интерфейс

Эта функция возвращает ответ оператора в интегрированное приложение через программный интерфейс;

– подфункция остановки функций распознавания и обработки изображений.

Подфункция останавливает работу функций обработки и распознавания изображений после завершения взаимодействия с оператором. Это необходимо чтобы не производить обработку изображения в то время когда она не требуется. Таким образом, экономятся ресурсы ЭВМ.

2.2.8 Функция отображения графического интерфейса.

Функция должна отобразить графические формы системы на мониторе пользователя и обеспечить обратную связь с оператором. Функция должна показать текущее состояние системы, то есть положения глаза оператора, а так же вопрос, который система получила через программный интерфейс от интегрированного приложения.

Функция имеет следующие подфункции:

– подфункция отображения графической формы диалога.

Подфункция отображает окно с графическим интерфейсом для пользователя. Окно отображается на мониторе. Окно формируется используются стандартные функции управления окнами операционной системы. Графическая форма отображает текст вопроса и кнопки "Да" и "Нет";

- подфункция отображения графической формы меню. Подфункция отображает окно с графическим интерфейсом для пользователя. Окно отображается на мониторе. Окно формируется используются стандартные функции управления окнами операционной системы. Графическая форма отображает список пунктов меню и кнопки переключения активного пункта меню. Активный пункт меню выделяется чтобы быть отличным от остальных пунктов;
- подфункция отображения графической формы регулятора.

Подфункция отображает окно с графическим интерфейсом для пользователя. Окно отображается на мониторе. Окно формируется используются стандартные функции управления окнами операционной системы. Графическая форма отображает название регулируемой величины, значение регулируемой величины, кнопки увеличения и уменьшения регулируемой величины, а так же регулятор в виде ползунка. Ползунок визуально показывает текущее состояние системы;

- Подфункция отображения положения зрачка оператора.

Функция наглядно отображает текущее состояние зрачка оператора на диалоговой форме. Отображение состояния происходит на кнопках, которые задают варианты ответа пользователя на вопрос. Если оператора совершает движение зрачком, соответствующая кнопка подсвечивается. Эта функция создает обратную связь для действия оператора.

2.3 Описание алгоритмического обеспечения системы.

Этот раздел содержит описание алгоритмов, используемых в системе.

2.3.1 Алгоритм захвата видеопотока с Веб - Камеры.

Начало.

1. После подачи команды запуска происходит обращение к драйверу веб-камер операционной системы.

Драйвер возвращает список всех камер, подключенных с компьютеру. Если камер нет - то происходит выход из программы.

2. Если обнаружены одна и более камер - то используется первая камера из списка доступных камер и происходит инициализация этой камеры.

3. С помощью драйвера у камеры считываются характеристики - рекомендуемая ширина и высота кадра, а так же максимальная скорость получения кадров в секунду.

4. Запускается процесс получения видеопотока с камеры.

5. После подачи команды завершения процесс захвата прерывается.

Конец.

2.3.2 Алгоритмы работы метода каскада Хаара и Виолы-Джонса.

В основе метода лежит расчет призрака Хаара для участка изображения. Признак представляет собой разницу между средними значениями яркости двух соседних областей изображения и хранится виде примитивов перехода яркости:

Рисунок 12 – Примитивы перехода яркости в алгоритме Хаара

Как работает метод:

Производится расчет средних значения яркости для участков изображения.

Перед расчетом средних значения яркости проводится вычисление интегрального изображения.

Интегральное изображение – это матрица, одинаковая по размерам с исходным изображением. В каждом элементе матрицы хранится сумма интенсивностей всех пикселей, находящихся левее и выше данного элемента - правого нижнего угла прямоугольной области (0,0) до (x,y). Элементы матрицы L можно рассчитать по формуле:

$$L(x,y) = \sum_{i=0,j=0}^{i\leq x,\ j\leq y} I(i,j)$$

Формула расчета интегрального изображения в алгоритме Хаара.

где I(i,j) - элемент матрицы исходного изображения, а $\sum_{i=0,j=0}^{i\leq x,\ j\leq y} I(i,j)$ - сумма интенсивностей всех пикселей, находящихся левее и выше данного элемента.

Расчет интегрального изображения позволяет мгновенно вычислить среднюю яркость для любой области изображения и оптимизировать затраты на вычислительные операции при работе с алгоритмом.

Затем на исходном изображении создается сканирующее окно, которое перемещается относительно всей площади изображения

Над каждым участком изображения, где проходит окно, рассчитываются призрак Хаара.

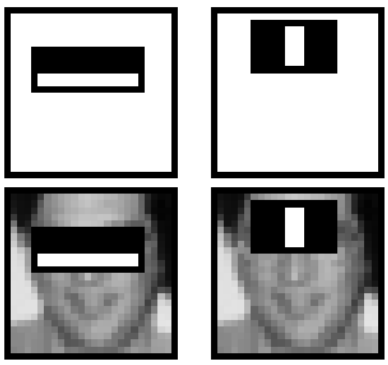


Рисунок 13 – Пример расчета признака Хаара для лица человека

Признаки Хаара сравниваются с заранее подготовленной и обученной базой данных для конкретного объекта. При сравнении вычисляется положение окна, которое имеет наибольшее совпадение признаков. Это положение и будет результатом поиска.

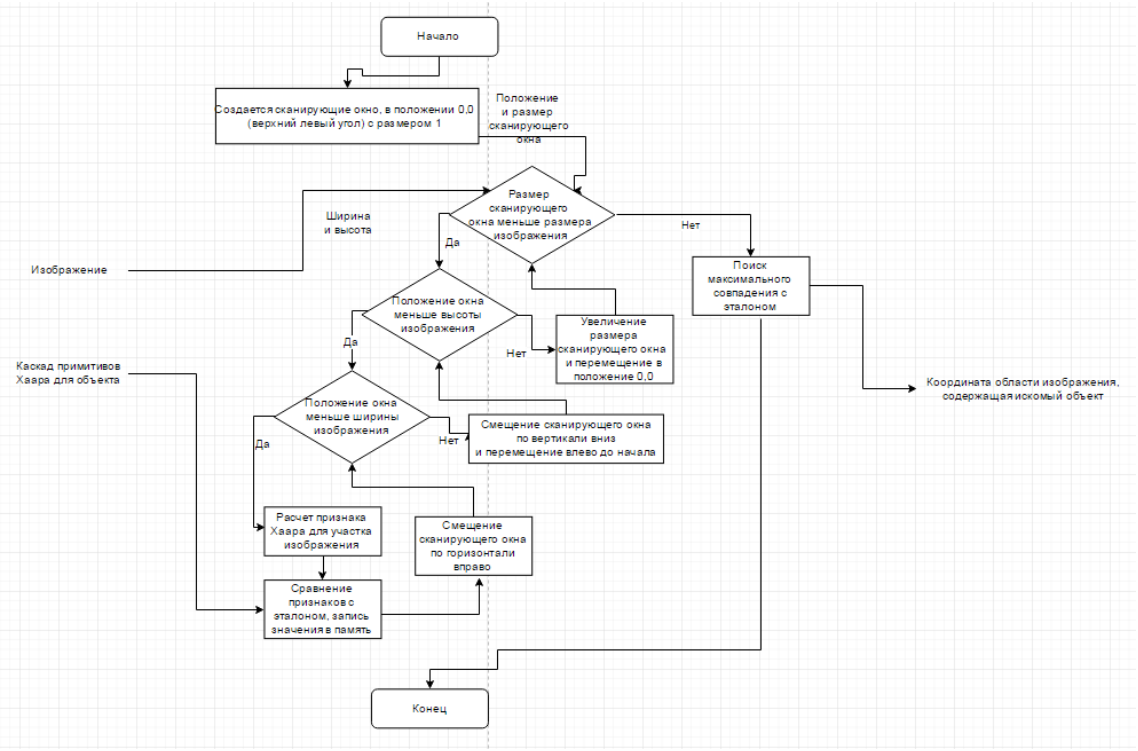


Рисунок 14 – Блок-схема алгоритма каскада Хаара

В алгоритме Хаара используются только вертикальные и горизонтальные примитивы и при расчете используются только 2 области для каждого изображения. Значение яркости в одной области вычитается из значения яркости в другой (см. рис N)

Таких признаков недостаточно для точного распознавания объекта, поэтому работа этого алгоритма в чистом виде мало эффективна.

Однако, Полом Виолой и Майклом Джонсом этот алгоритм был доработан.

В методе Виолы-Джонса были дополнены признаки Хаара и возможность сравнивать более 2х областей в одном признаке, что позволило повысить качество работы алгоритма.

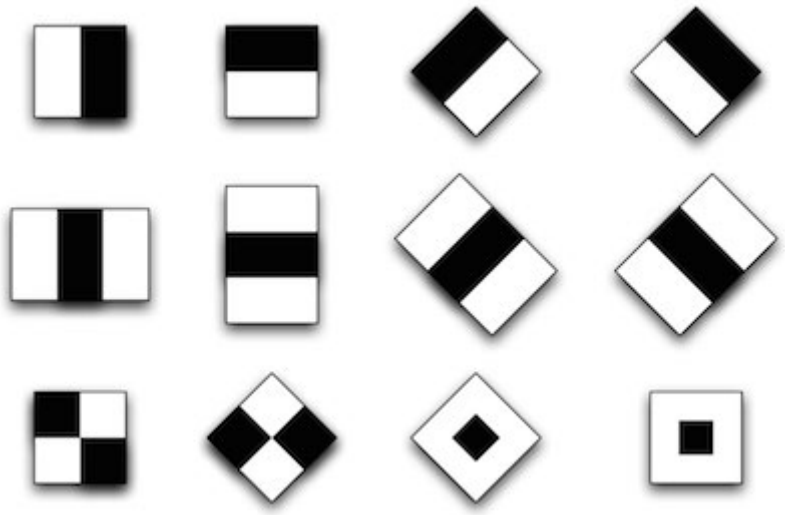


Рисунок 15 – Дополненные примитивы в методе Виолы-Джонса

Для работы алгоритма необходимо создать каскад графических примитивов Хаара для конкретного объекта. Этот процесс называется обучением классификатора.

Процесс обучения достаточно долгий и представляет собой прохождение заранее подготовленных изображений с положительным и отрицательным результатом через классификатор.

Процесс обучения классификатора:

1. Создается база изображений, содержащих нужный объект. Для успешного обучения необходимо иметь 3000-4000 изображений. Эти изображения называются изображения с положительным результатом.

2. Создается база изображений с отрицательным результатом. Это изображения, на которых отсутствует искомый объект. Изображения могут быть абсолютно любые. Для успешного обучения необходимо так же иметь 3000-4000 изображений.

3. Производится обучение классификатора. Выполняется расчет признаков Хаара для все положительных и отрицательных изображений.

4. Находятся совпадающие признаки и вероятности совпадения, эти данные сохраняются.

В итоге происходит автоматический поиск признаков Хаара, которые присущи только требуемому объекту.

Алгоритмы обучения классификатора Хаара и метода Виолы-Джонса разработаны и запрограммированы в большинстве математических пакетах (MatLab) и прикладных библиотеках компьютерного зрения (OpenCV).

Эти алгоритмы используют до 162 тысячи признаков Хаара при обучении, очень сложны и разрабатывались командами ученых, поэтому смысла их подробно описывать здесь нет.

Обучение классификатора можно производить в автоматическом режиме, подав на вход приложению базу из положительных и отрицательных изображений.

Кроме того, в сети интернет достаточное количество уже обученных классификаторов для популярных объектов. К ним относятся лицо и глаза, которые требуется найти в разрабатываемой системе.

Поэтому, при разработка алгоритма я использовал готовый обученный классификатор для поиска лица и глаза, поставляемый вместе с библиотекой компьютерного зрения OpenCV. Подробнее о библиотеке OpenCV и программировании приложения смотрите в разделе 4.

Чтобы найти положение глаза, сначала на исходном кадре находится положение лица, затем на уже на лице находится положение глаза.

В результате работы алгоритма система определит положение глаза на исходном изображении.

2.3.3 Алгоритмы нахождения положения и границ глаза на изображении лица человека.

Для поиска положения и границ глаза на изображении лица был выбран метод лицевой маски. Данный метод заключается в нахождении на изображении лица ключевых точек, такие как границы ресницы, бровей, глаз, носа, рта и подбородка, и отличается высокой надежностью и точностью локализации, не зависимо от положения лица, освещенности, разрешения изображения и частичного перекрытия. В основе метода лежит сверточная нейронная сеть, которая предварительно обучается на большом объеме заранее размеченных изображений лиц.

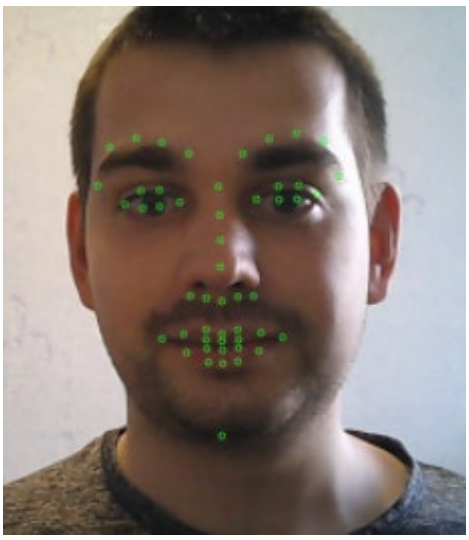


Рисунок 16 – Ключевые точки лицевой маски, отмеченные на изображении лица.

Искусственная нейронная сеть — математическая модель, а также её программное или аппаратное воплощение, построенная по принципу организации и функционирования биологических нейронных сетей — сетей нервных клеток живого организма. Это понятие возникло при изучении процессов, протекающих в мозге, и при попытке смоделировать эти процессы.

ИНС представляет собой систему соединённых и взаимодействующих между собой простых процессоров (искусственных нейронов). Такие процессоры обычно довольно просты (особенно в сравнении с процессорами, используемыми в персональных компьютерах). Каждый процессор подобной сети имеет дело только с сигналами, которые он периодически получает, и сигналами, которые он периодически посылает другим процессорам. И, тем не менее, будучи соединёнными в достаточно большую сеть с управляемым взаимодействием, такие по отдельности простые процессоры вместе способны выполнять довольно сложные задачи.

Нейронная сеть получает на вход массив дробных чисел, как правило, в диапазоне от 0 до 1 заранее определенной размерности. На выходе нейронная сеть так же возвращает массив чисел определенной размерности.

Размерности входного и выходного массива определяются задачей, которую будет решать нейронная сеть. Например, для простой классификации изображения (кошка или собака), входной массив будет предоставлять матрицу, равной по размерности разрешению входящего изображения, а выходной – вектор, состоящий из двух чисел, которое будут показывать вероятность отнесения входного изображения к первому или второму классу. Например, результат [0.3, 0.85] будет означать, что изображение отнесено к ко второму классу, а результат [0.913, 0.15] к первому.

Свёрточная нейронная сеть — специальная архитектура искусственных нейронных сетей, нацеленная на эффективное распознавание образов и входит в состав технологий глубокого обучения. Использует некоторые особенности зрительной коры, в которой были открыты так называемые простые клетки, реагирующие на прямые линии под разными углами, и сложные клетки, реакция которых связана с активацией определённого набора простых клеток. Таким образом, идея свёрточных нейронных сетей заключается в чередовании свёрточных слоёв и слоёв активации.

Слой свёртки — это основной блок свёрточной нейронной сети. Слой свёртки включает в себя для каждого канала свой фильтр, ядро свёртки которого обрабатывает предыдущий слой по фрагментам (суммируя результаты матричного произведения для каждого фрагмента). Значения матрицы свёртки настраиваются в процессе обучения и являются весами слоя нейронной сети.

Свертка — основная операция, проводимая в сверточных нейронных сетях. Этот математический термин применяется для движущегося окна или фильтра по исследуемому изображению. Перемещающееся окно применяется к определенному участку узлов, как показано ниже. Где примененный фильтр — (0.5 * значение в узле):

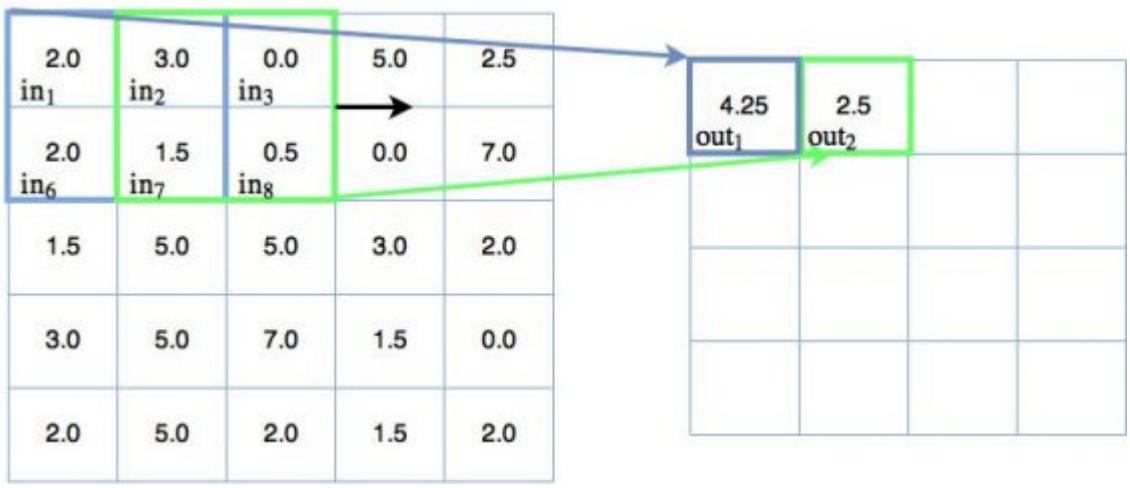


Рисунок 17 – Операция сверстки на матрице изображения.

На диаграмме (Рисунок 17) показаны только два выходных значения, каждое из которых отображает входной квадрат размера 2×2. Вес отображения для каждого входного квадрата, как ранее упоминалось, равен 0.5 для всех четырех входов (inputs). Поэтому выход может быть посчитан так:

$$
\begin{aligned}
out_1 &= 0.5in_1 + 0.5in_2 + 0.5in_6 + 0.5in_7 \\
&= 0.5 \times 2.0 + 0.5 \times 3.0 + 0.5 \times 2.0 + 0.5 \times 1.5 \\
&= 4.25 \\
out_2 &= 0.5in_2 + 0.5in_3 + 0.5in_7 + 0.5in_8 \\
&= 0.5 \times 3.0 + 0.5 \times 0.0 + 0.5 \times 1.5 + 0.5 \times 0.5 \\
&= 2.5
\end{aligned}
$$

Рисунок 18 – Формулы сверстки на матрице изображения.

Каждый новый элемент матрицы (пиксель) вычисляется на основе окружающих его пикселей изображения. Коэффициенты сверстки определяются в процессе обучения нейронной сети.

Слой активации получает на вход скалярный результат каждой свёртки, и преобразует его по формуле гиперболического тангенса (f(x) = tanh(x), f(x) = |tanh(x)|). Перед подготовкой к обучению настраивается количество слоев сверстки и активации, подбирая параметры опытным путем. После оценки результатов обучения оставляются наиболее точные параметры.

Процесс обучения сверточной нейронной сети представляет собой

передачу большего объема заранее размеченных данных. Для обучения используются метод обратного распространения ошибки. Метод обратного распространения ошибки можно разделить на 4 отдельных блока: прямое распространение, функцию потери, обратное распространение и обновление веса. Во время прямого распространения, берётся тренировочное изображение и пропускается через всю сеть. В начале обучения все веса или значения фильтра инициализированы случайным образом и выходным значением будет массив случайных чисел, например [.1 .1 .1 .1 .1]. Однако, в процессе обучения мы имеем правильный результат, который должна выдавать сеть, и он не совпадает с текущим, соответственно, происходит переход к функции потери.

С помощью функции потери для каждого слоя определяются какие из весов слоя нейронной сети привели к ошибке. Функция потери может быть выражена по-разному, но часто используется СКО (среднеквадратическая ошибка): $E_{total} = \sum \frac{1}{2}(target - output)^2$, где target – требуемое показание, а output – текущее показание нейронной сети. Дальнейшая задача это отрегулировать веса слоя нейронной сети так, чтобы снизить ошибку. На этом этапе происходит обновление весов, затем обучение повторяется для каждого слоя, до получения требуемой точности. Процесс обучения повторяется для каждого нового изображения из обучающего набора. Затем происходит тестирование – сопоставление заранее размеченных данных с результатами работы нейронной сети. В случае получения неудовлетворительных результатов изменяются параметры нейронной сети: количество слоев, размерности входных и выходных матриц, формулы слоя активации и определения потерь в процессе обучения.

В данной работе использовалась готовая обученная модель нейронной сети из библиотеки компьютерного зрения OpenCV 3. Характеристики используемой модели: обучение на 50 000 размеченных изображений, размерности входных и выходных матриц: 400x400, количество

используемых слоев: 68.

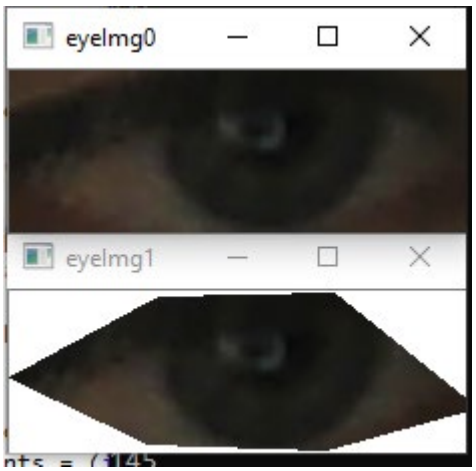
В связи с большими вычислительными затратами на обучение данной нейронной сети, которое возможно осуществить за премлимое время только в условиях дата-центра или высокопроизводительного компьютерного кластера, в данной работе было принято решение использовать именно готовую модель.

Результат работы данной сети представлен на рисунке 17 – это матрица, содержащая светлые области в местах обнаружения ключевых точек лицевой маски.

Рисунок 19 – Результат работы нейронной сети.

Определение координат точек происходит простым по-пиксельным проходом полученного изображения. Каждая точка будет иметь одинаковый порядковый номер на любом полученном изображении лица (с учетом ориентации), что позволяет определить к какой области лица она относится и границу какой области отображает.

В данной работе нас интересует граница левого глаза. В результате работы алгоритма, область изображения, содержащая глаз вырезается для дальнейшей обработки. Так же, с изображения удалятся информация, находящаяся за пределами найденных границ. Результат работы алгоритма представлен на рисунке N.



Изображение 20 – результат работы алгоритма нахождения положения и границ глаза на изображении лица человека.

2.3.4 Алгоритм параллельных вычислений.

Первоначально, алгоритм распознавания в системе имеет следующую последовательность действий:

- захват изображения с веб - камеры;
- нахождение расположения лица человека на изображении;
- нахождение расположения глаза на лице человека;
- стабилизация положения глаза на изображении, по сравнению с другими кадрами видеопотока;
- нахождение положения зрачка относительно глаза;
- наиболее ресурсоемкими являются шаги:
- нахождение расположения лица человека на изображении;
- нахождение расположения глаза на лице человека;

Однако, цель данной работы подразумевает использование технологий параллельных вычислений.

Для использования параллельных вычислений, предлагается следующее изменение работы алгоритма:

Наиболее затратные операции будут выполняться в разных потоках, что позволит ускорить работу алгоритма.

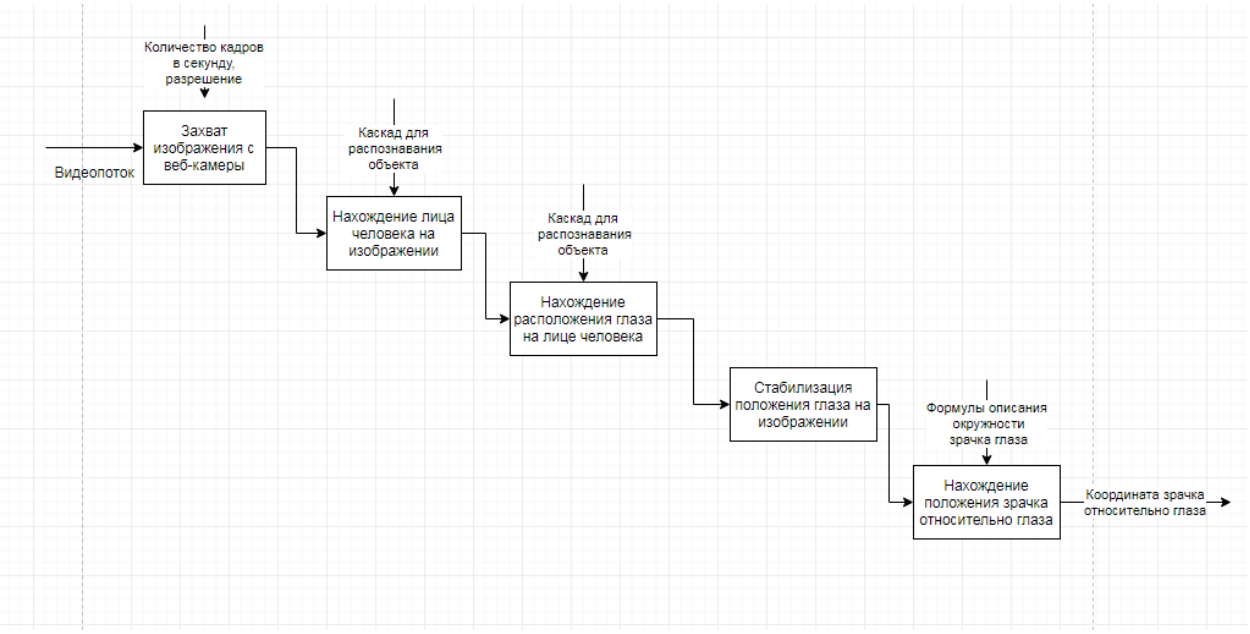


Рисунок 21 - Диаграмма алгоритма при последовательной обработке данных

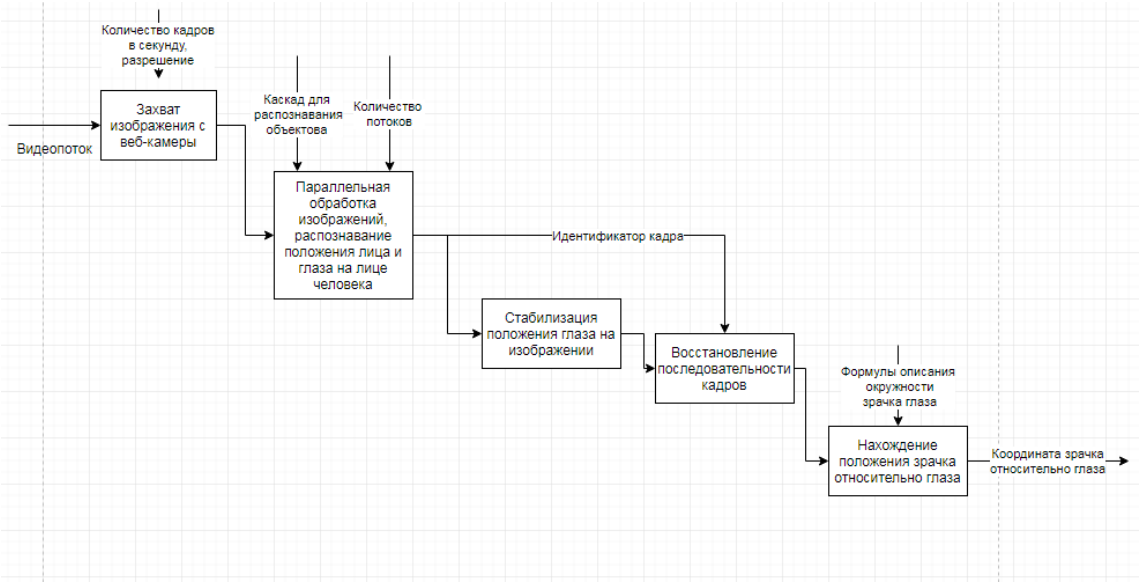


Рисунок 22 - Диаграмма алгоритма при параллельной обработке данных

Пояснение к новому алгоритму:

Захват изображения с веб - камеры будет производиться последовательно, при этом каждый новый кадр будет передаваться свободному потоку для выполнения ресурсоемких операций. Нахождение расположения лица человека на изображении и нахождение расположения глаза на лице человека

будет выполняться в параллельных потоках. Количество потоков = n, и может быть произвольным. По завершению обработки кадра любым потоком, кадр передается в функцию стабилизации, которая, на основе имеющихся данных о предыдущих кадрах, выполняет стабилизацию.

Затем, стабилизированный кадр, содержащий изображение глаза, попадает в функцию восстановления последовательности кадров. Эта функция необходима на случай, если какой либо кадр в потоке обработается быстрее, чем предыдущий. Если не производить восстановление последовательности кадров, то возможна ситуация, в которой кадры будут попадать в распознавание на по порядку, что вызовет некорректную работу всего алгоритма.

Для восстановления последовательности кадров, каждому захваченному с веб-камеры кадру будет присваиваться порядковый номер т.е. идентификатор. Функция восстановления кадров на основе идентификатора не будет пропускать на дальнейшую обработку кадр, если он обработался раньше предыдущий.

2.3.5 Алгоритмы поиска положения зрачка.

Рисунок 23 – Блок-схема алгоритма поимка положения зрачка

3 РАЗРАБОТКА ПРИЛОЖЕНИЯ

В этом разделе описана разработки спроектированной системы.

3.1 Выбор среды разработки и языка программирования. Язык программирования C++.

Для разработки приложения выбран язык разработки C++.

C++ — компилируемый, статически типизированный язык программирования общего назначения.

Поддерживает такие парадигмы программирования, как процедурное программирование, объектно-ориентированное программирование, обобщённое программирование. Язык имеет богатую стандартную библиотеку, которая включает в себя распространённые контейнеры и алгоритмы, ввод-вывод, регулярные выражения, поддержку многопоточности и другие возможности.

Так же для этого языка написано огромное количество сторонних библиотек, в том числе библиотек обеспечения алгоритмов компьютерного зрения.

Плюс этот позволяет максимально быстро выполнять высоконагруженные операции с данными.

Для реализации алгоритмов обработки изображений и компьютерного зрения выбрана библиотека OpenCV версии 3.0

Библиотека предоставляет достаточно большой набор реализаций методов и алгоритмов обработки изображений и содержит достаточное количество методов для программирования алгоритмов в данной работе.

Приложение будет разработано для платформы Windows32 бита. Среда разработки: Microsoft Visual Studio 2013

3.2 Общие требования к оборудованию и ПО ЭВМ для запуска и работы программы.

Для запуска программы необходимо использовать персональный компьютер на базе ОС Windows 7 и выше.

Разрядность операционной системы: 32 или 64 бита.

Минимальные допустимые требования для работы программы:

- двухъядерный процессор с частотой 1 ГГц и выше;
- 2 ГБ оперативной памяти;
- 200МБ свободного места на жестком диске;
- видеокарта с видеопамятью от 256 мегабайт;
- разрешение экрана от 1024x600;
- наличие веб-камеры с разрешением от 640x480 и больше.

Требования к системе взяты с учетом минимальных требований к библиотеке OpenCV и протестированы на практике на аналогичном ПК.

3.3 Разработка дизайн графический форм.

Пример макета экранной формы диалога:

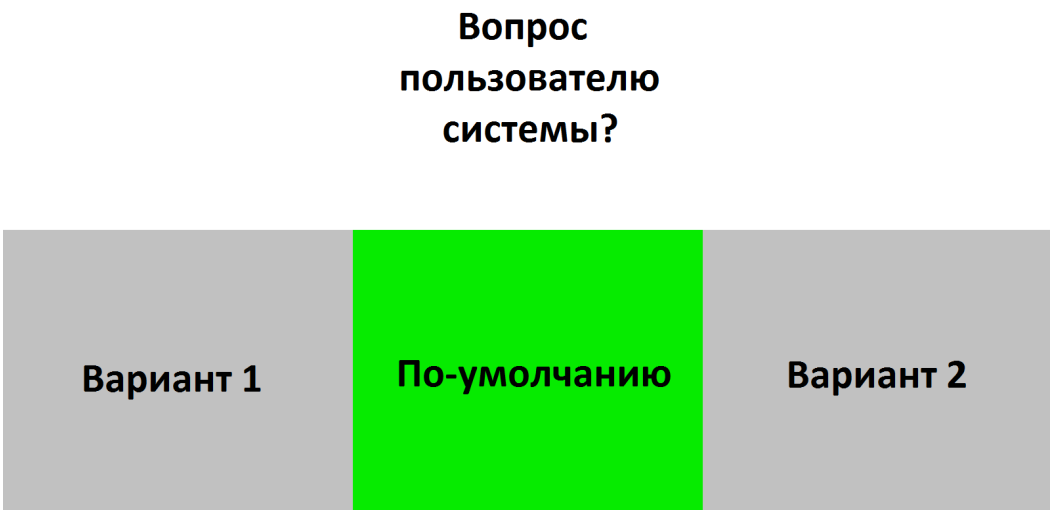


Рисунок 24 – Пример графической формы диалога

Если пользователь посмотрел влево, это отобразится в нижней части формы:

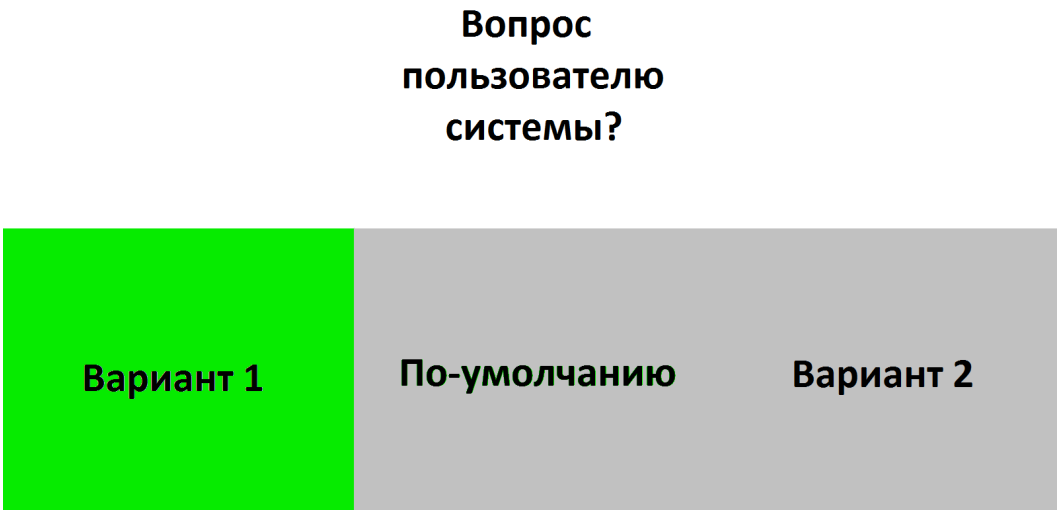


Рисунок 25 – Пример реакции графической формы диалога на действие пользователя

Аналогично для состояния вправо.

Вопрос пользователю должен отображаться именно по центру формы, чтобы исключить ложные срабатывания.

Если текст будет располагаться так же в левой или правой части формы, то пользователь в процессе чтения вопроса будет вынужден посмотреть влево или вправо а это вызовет срабатывание системы.

Для обеспечения функции выбор объекта из списка разработана следующая форма:

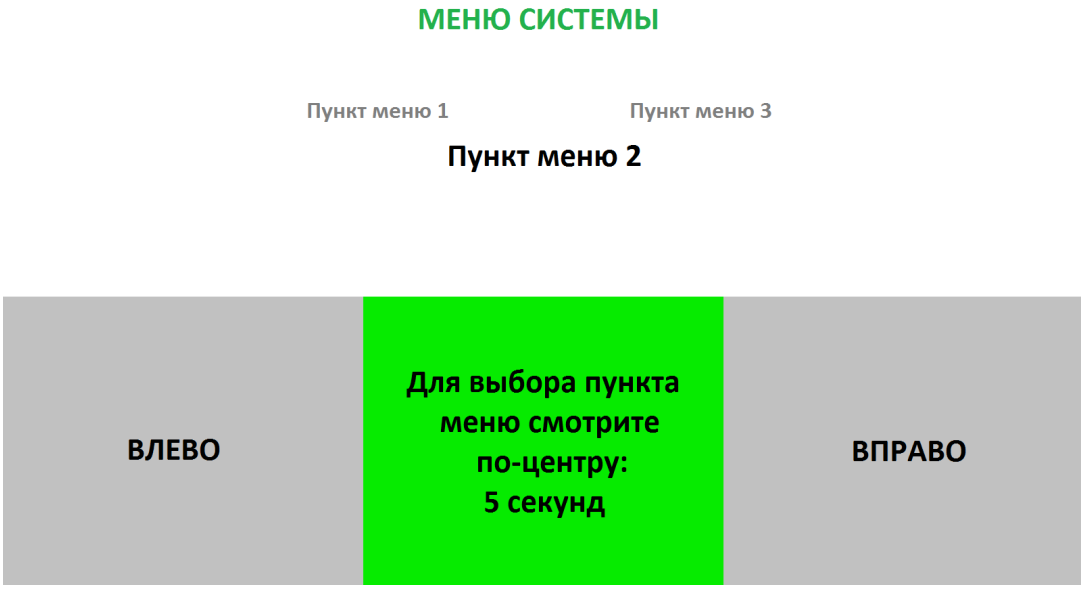


Рисунок 26 – Пример графической формы выбор объекта из списка

Данная форма представляет собой интуитивно понятное меню, процесс взаимодействия с которым понятен большинству пользователей компьютерной техники. Для смены активного пункта меню пользователь должен посмотреть влево или вправо. Для выбора пункта меню пользователь должен смотреть по центру в течение 5 секунд. Если пользователь меняет активный пункт меню то таймер до выбора пункта сбрасывается.

Для обеспечения функции изменения регулируемой величины в заданном диапазоне, разработана следующая форма:

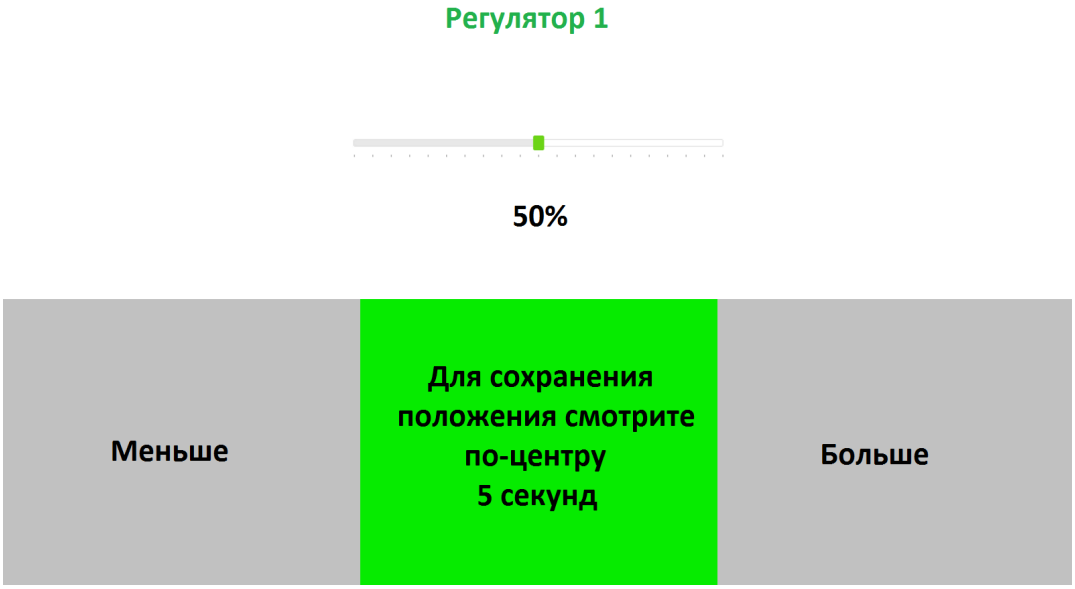


Рисунок 27 – Пример графической формы изменения регулируемой величины в заданном диапазоне

Для увеличения или уменьшения положения регулятора пользователь должен посмотреть влево или вправо. Для сохранения положения регулятора пользователь должен смотреть по центру в течение 5 секунд. Если пользователь меняет положения регулятора то таймер до сохранения положения регулятора сбрасывается.

После каждого изменения значения пользователем будет вызвана функция обратного вызова (в случае если она используется в программном интерфейсе), чтобы обеспечить изменение величины в режиме реального времени.

3.4 Разработка приложения.

Приложение разработано согласно структуре функциональных подсистем. Для функциональных подсистем созданы соответствующие методы и классы. Для реализации алгоритмов, необходимых для обработки изображений использовалась библиотека OpenCV. Часть функциональных подсистем уже имеет реализацию в библиотеке.

Все методы обернуты в пространство имен EyeControlSystemAPI, чтобы исключить конфликты с методами интегрируемого приложения.

Функция захвата видеопоток с веб-камеры:

В библиотеке OpenCV для подключения к камере используется класс VideoCapture. Инициализация камеры происходит с помощью создания объекта класса VideoCapture. После создания объекта начитается захват видеопотока с камеры.

Функция извлечения кадров из видеопотока:

Для извлечения кадров создан метод StartCapture, содержащий цикл видеозахвата. В начале цикла происходит извлечение нового карда из видеопотока. Для хранения матриц изображения используется тип данных cv::Mat. Получить кадр из видеопотока используется оператор вывода >> из объекта VideoCapture в объект матрицы изображения cv::Mat.

Функция обработки и поиска объекта на изображении по методу Хаара и Вилоы-Джонса:

Для реализации алгоритма используется библиотека OpenCV. Она содержит методы работы по алгоритму Хаара и Вилоы-Джонса. Для использования метода создается объект классификатора cv::CascadeClassifier, в него загружается каскад примитивов яркостей искомого объекта. Обученный каскад примитивов для лица человека и глаза человека поставляется с библиотекой в виде файлов xml. Загрузка каскада происходит с помощью метода load, который вызывается из созданного объекта классификатора cv::CascadeClassifier. В качестве параметра классификатору передается путь к xml файлу, содержащему каскад.

Для реализации функциональной подсистемы создан метод findeye, который получает на вход изображение кадра cv::Mat. В метода последовательно создаются объекты cv::CascadeClassifier, загружают файлы классификаторов и производится поиск сначала положения лица, а затем глаза. Метод возвращает массив, содержащий координату и размеры лица относительно изображения и координату и размеры глаза относительно лица.

Функция проверки корректности положения глаза:

Для работы функции создан метод eyefilter, который получает на вход положение лица и глаза. Метод производит сравнение координат и возвращает true, если глаз может находится в области лица, которая передана методу и false, если положение глаза не прошло проверку.

Функция стабилизации положения кадра в видеопотоке.

Для работы функции создан метод stabilize, который выполняет стабилизацию изображения. На вход метода поступает координата положения лица, глаза и изображение для текущего и эталонного кадра.

Метод производит по-пиксельное вычитание областей матриц эталонного и текущего изображений, содержащих лицо и находит положение с минимальным отклонением от эталона. Для по-пиксельного обхода изображений используется вложенный цикл по строкам и столбцам матрицы изображений. Для каждого пикселя эталонного и текущего изображения происхо-

дит вычитание цвета пикселей и находится разница. Чем меньше разница, тем больше соответствие текущего положения изображения эталонному.

Функция обработки и поиска положения зрачка на изображении глаза:

Для работы используются несколько методов. Каждый метод описан в соответствии со структурой подфункций этой функциональной подсистемы.

Для описание метода создано пространство имен eyeposition, внутри которого содержатся методы всех требуемых подфункций. В пространстве имен так же содержится метод process, который выполняет последовательный вызов всех методов подсистемы, согласно алгоритму. Метод получает на вход часть изображения кадра, содержащую глаз и возвращает позицию зрачка относительно центра глаза.

Функция отображения графического интерфейса:

Этот метод пространства имен GUI. Содержит методы showDialog, showMenu и showSlider, которые отображают графические формы диалога, меню и регулятора соответственно. Так же пространство имен содержит метод removeALL, который закрывает все графические окна этой системы.

Функция взаимодействия с приложениями через программный интерфейс:

Этот метод представлен в виде класса EyeControlSystem. Класс содержит методы init, stop, question, menu и slider.

Метод init инициализируем систему и цикл видеозахвата.

Метод stop останавливает систему, закрывает диалоговые окна, останавливает процесс захвата и очищает память.

Метод question задает вопрос пользователю. Метод получает текст вопроса, значение для кнопки "Да" по-умолчанию и значение для кнопки "Нет"

по-умолчанию. Метод возвращает true если пользователь ответил "Да" и false, если пользователь ответил "Нет"

Метод menu отображает пользователю меню. Метод получает список пунктов меню, состоящий из индекса и названия и заголовок меню. Метод возвращает индекс выбранного пользователем пункта меню.

Метод slider отображает пользователю регулятор. Метод получает заголовок регулятора, минимальное значение регулятора, максимальное значение регулятора, значение регулятора по-умолчанию и шаг изменения значения. Метод возвращает выбранное пользователем значение регулятора.

Исходный код класса EyeControlSystem:

```
class EyeControlSystem
{
private:
	VideoCapture V;
	/*
		Функция инициализации системы. Запускает процесс захвата с веб - камеры.
		@ Fedtsov A.V 19.05.2017
	*/
	bool init(){
		V = VideoCapture(0);
		if (!V.isOpened()) return false;
		EyeControlSystemAPI::StartCapture(V);
	}
	/*
		Функция остановки системы. Закрывает окна, останавливает процесс захвата и очищает память.
```

```
            @ Fedtsov A.V 19.05.2017
        */
        bool stop(){
            EyeControlSystemAPI::GUI::removeALL();
            EyeControlSystemAPI::stopCapture(V);
            delete V;
        }
    public:
        /*
            Функция вопопроса пользователю.
            text - Текст вопроса,
            YesValue - Значение для кнопки "Да" по-умолчанию
            NoValue - Значение для кнопки "Нет" по-умолчанию
            @ Fedtsov A.V 19.05.2017
        */
        bool question(string text, string YesValue, string NoValue) {
            // Инициализируем систему и цикл видеозахвата
            if (!init()) std::runtime_error("Ошибка подключения к веб-камере. Убедитесь, что камера подключена к компьютеру!");
            // По окончании инициализации отображаем диалог вопроса
            EyeControlSystemAPI::GUI::showDialog(text, YesValue, NoValue);
            // Считываем позицию зрачка из подсистемы поиска зрачка
            int pos = EyeControlSystemAPI::eyeposition();
            // Ожидаем, пока пользователь смотрит по-ценру.
            // Время зарержки равно времени до появления следующего кадра с учетом скорости получения кадров в секунду от камеры.
```

```
		while (pos == 0) { pos = EyeControlSystemAPI::eyeposition(); Sleep(1000 / V.getFPS()); }

		// Останавливаем работу системы

		stop();

		// Возвращаем true, если пользователь посмотрел влево. Иначе возвращаем false.

		return pos == -1;

	}

	/*

		Функция отображения меню пользователю.

		std::vector<int, std::string> item - Список пунктов меню, состоящий из индекса и названия,

		titie - Заголовок меню

		@ Fedtsov A.V 19.05.2017

	*/

	int menu(std::vector<int, std::string> items,string titie) {

		// Инициализируем систему и цикл видеозахвата

		if (!init()) std::runtime_error("Ошибка подключения к веб-камере. Убедитесь, что камера подключена к компьютеру!");

		int itemindex = 0; // Активный индекс меню

		// По окончании инициализации отображаем диалог меню

		EyeControlSystemAPI::GUI::showMenu(items, titie, itemindex);

		int delay = 5000; // зарержка до сохранения 5 секунд

		// Считываем позицию зрачка из подсистемы поиска зрачка

		int pos = EyeControlSystemAPI::eyeposition();

		// Ожидаем, пока время выбора активного пункта меню не выйдет

		while (delay > 0) {

			pos = EyeControlSystemAPI::eyeposition();
```

```
                    if (pos != 0) {
                        // Если пользователь смотрит влево или вправо, меняем активный индекс меню.
                        if (pos == -1) itemindex--;
                        else itemindex++;
                        if (itemindex < 0) itemindex = items.size() - 1;
                        if (itemindex >= items.size()) itemindex = 0;
                        // Показываем активный пункт меню.
                        EyeControlSystemAPI::GUI::showMenu(items, titie, itemindex);
                        //Сбрасываем таймер
                        delay = 5000;
                    }
                    // Ожидаем появления следующего кадра
                    Sleep(1000 / V.getFPS());
                    // Убавляем таймер на время, соответствующее задержке появления нового кадра.
                    delay -= 1000 / V.getFPS();
                }
                // Останавливаем работу системы
                stop();
                // Возвращаем выбранный индекс меню
                return itemindex;
            }
            /*
            Функция отображения меню пользователю.
                titie - Заголовок регулятора
                minvalue - Минимальное значение регулятора
                maxvalue - Максимальное значение регулятора
```

```
        defvaue - Значение по-умолчанию
        step - шаг изменения значения
        @ Fedtsov A.V 19.05.2017
    */
    int slider(string titie, int minvalue, int maxvalue, int defvaue, int step)
{
        // Инициализируем систему и цикл видеозахвата
        if (!init()) std::runtime_error("Ошибка подключения к веб-камере. Убедитесь, что камера подключена к компьютеру!");
        // По окончании инициализации отображаем диалог регулятора
        EyeControlSystemAPI::GUI::showSlider(titie, minvalue, maxvalue, defvaue);
        int delay = 5000; // зарержка до сохранения 5 секунд
        // Считываем позицию зрачка из подсистемы поиска зрачка
        int pos = EyeControlSystemAPI::eyeposition();
        // Ожидаем, пока время выбора положения регулятора не закончится
        while (delay > 0) {
            pos = EyeControlSystemAPI::eyeposition();
            if (pos != 0) {
                // Если пользователь смотрит влево или вправо, меняем положение регулятора.
                if (pos == -1) defvaue -= step;
                else defvaue += step;
                if (defvaue < minvalue) defvaue = minvalue;
                if (defvaue > maxvalue) defvaue = maxvalue;
                // Показываем текущее положение регулятора.
                EyeControlSystemAPI::GUI::showSlider(titie, minvalue, maxvalue, defvaue);
```

```
                //Сбрасываем таймер
                delay = 5000;
            }
            // Ожидаем появления следующего кадра
            Sleep(1000 / V.getFPS());
            // Убавляем таймер на время, соответствующее задержке появления нового кадра.
            delay -= 1000 / V.getFPS();
        }
        // Останавливаем работу системы
        stop();
        // Возвращаем значение регулятора
        return defvaue;
    }
};
```

Доступ к программному интерфейсу системы осуществляется с помощью объекта этого класса. Подробнее о процессе интеграции программного интерфейса в приложение смотрите раздел 4.2.

4 РУКОВОДСТВО ПО ЭКСПЛОАТАЦИИ И ИНТЕГРАЦИИ СИСТЕМЫ

Этот раздел содержит инструкцию по использованию системы и руководство по интеграции через программный интерфейс для разработчика.

4.1 Инструкция для пользователя.

Данная система управления используется совместно с приложением, которое было интегрировано с системой и используется на персональном компьютере. Для использования системы оператор должен уметь пользоваться приложением и предварительно ознакомиться с его инструкцией по использованию.

Данная система предоставляет следующие возможности управления приложением с помощью движений зрачка глаза:

– диалог с возможностью положительного или отрицательного ответа на вопрос;

– меню, с возможностью выбора одного элемента из списка;

– регулятор с возможностью установки значения характеристики объекта.

В качестве основных действий, совершаемых зрачком используются следующие действия:

– повернуть зрачок влево;

– смотреть по-центру;

– повернуть зрачок вправо.

На основе этих трех действий происходит взаимодействие с формами управления.

При возникновении необходимости взаимодействия с пользователем, приложение вызовет нужную форму и она отобразится пользователю.

Форма диалога.

Форма имеет следующий интерфейс:

Рисунок 28 – Форма диалога

Для взаимодействия с формой диалога пользователю нужно:

1. Прочитать сообщение с формы.

2. Совершить движение зрачком глаза влево для ответа "Да".

3. Совершить движение зрачком глаза вправо для ответа "Нет".

После совершение действия ответ распознается системой и передастся в приложение, которое задало этот вопрос.

Форма меню.

Форма имеет следующий интерфейс:

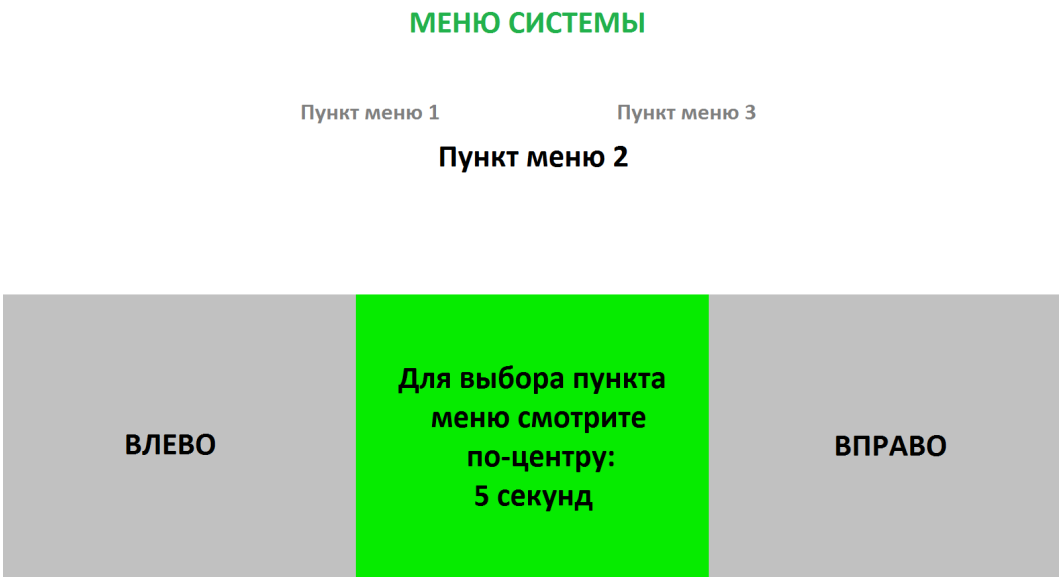


Рисунок 29 – Форма меню

Для взаимодействия с формой меню пользователю нужно:

1. Прочитать активный пункт меню. Активный пункт меню находится по центру формы. Он имеет более жирный стиль и размер текста. Слева и справа находятся дополнительные пункты меню. Они имеют меньший размер текста.

2. Если требуется выбрать активный пункт меню, пользователь смотрит по центру в течение 5 секунд. За это время истечет таймер выбора и активный пункт сохранится.

3. Если требуется изменить активный пункт меню, совершить движение зрачком глаза влево или вправо. При движении влево активным станет пункт меню, который расположен слева. При движении вправо - тот который расположен справа от текущего активного пункта меню.

При изменении активного пункта меню таймер выбора сбросится.

После истечения таймера активный пункт меню передастся в приложение, которое вызвало это меню.

Форма регулятора.

Форма имеет следующий интерфейс:

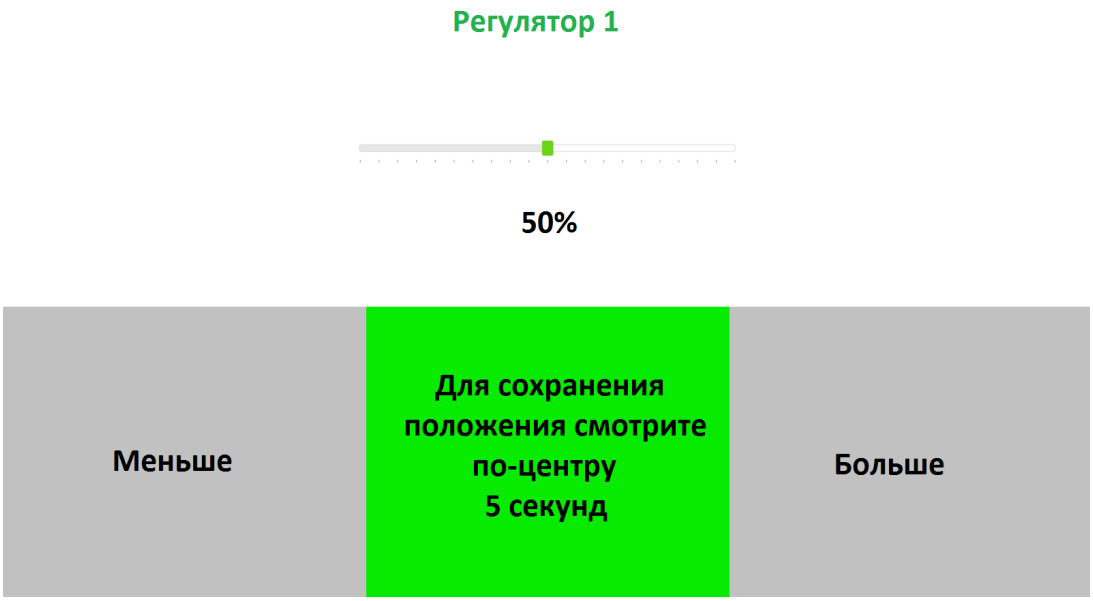


Рисунок 30 – Форма регулятора

Для взаимодействия с формой регулятора пользователю нужно:

1. Прочитать название и значение регулятора.

2. Если требуется сохранить текущее положение регулятора, пользователь смотрит по центру в течение 5 секунд. За это время истечет таймер выбора и положение регулятора сохранится.

3. Если требуется изменить положение регулятора, совершить движение зрачком глаза влево или вправо. При движении влево положение регулятора изменится в меньшую сторону вправо - в большую.

При изменении положения регулятора таймер выбора сбросится.

После истечения таймера текущее значение регулятора передастся в приложение, которое вызвало эту форму регулятора.

4.2 Руководство для разработчика.

Для интеграции приложения с системой используется программный интерфейс. Программный интерфейс поддерживает язык программирования C++ и предоставляется в виде подключаемого файла к исходному коду приложения. Для работы системы так же потребуется подключение библиотеки OpenCV.

Для интеграции системы в приложение необходимо подключить заголовочные файлы и библиотеки к проекту вашего приложения.

Руководство по подключению составлено на примере среды разработки Visual Studio:

1. Открываем Visual Studio. Создаем новый проект приложения. Если проект приложения уже создан, переходите к пункту 2.

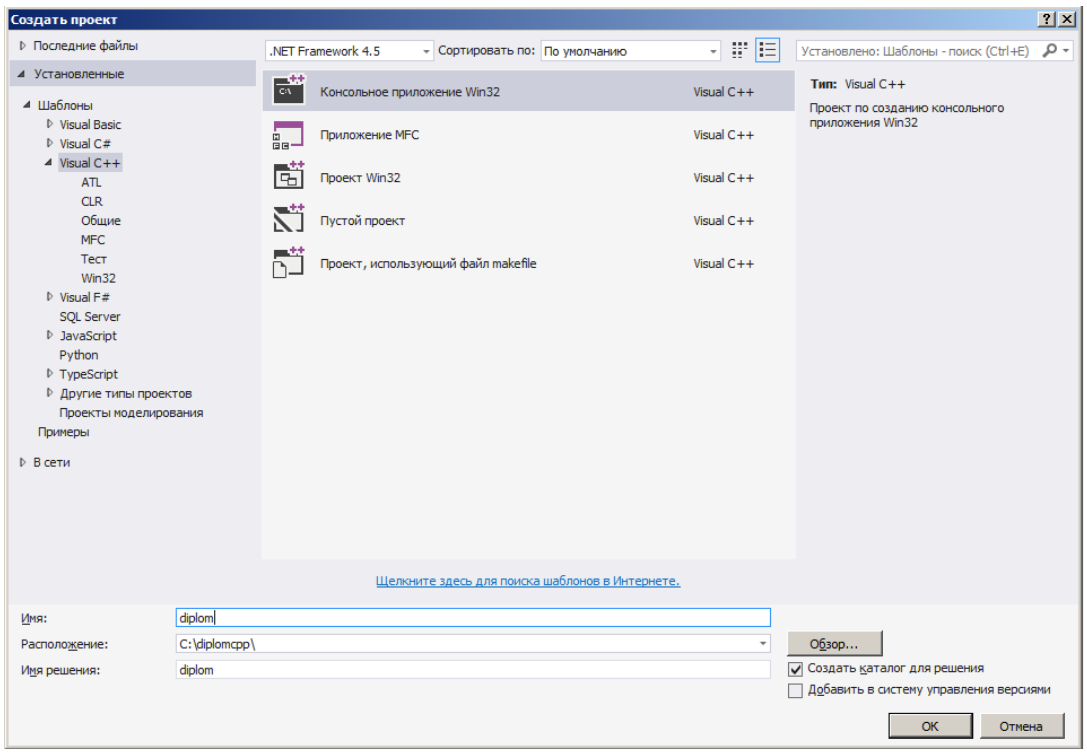


Рисунок 31 – Создание проекта в Visual Studio. Шаг 1

Проект будет иметь следующие настройки:

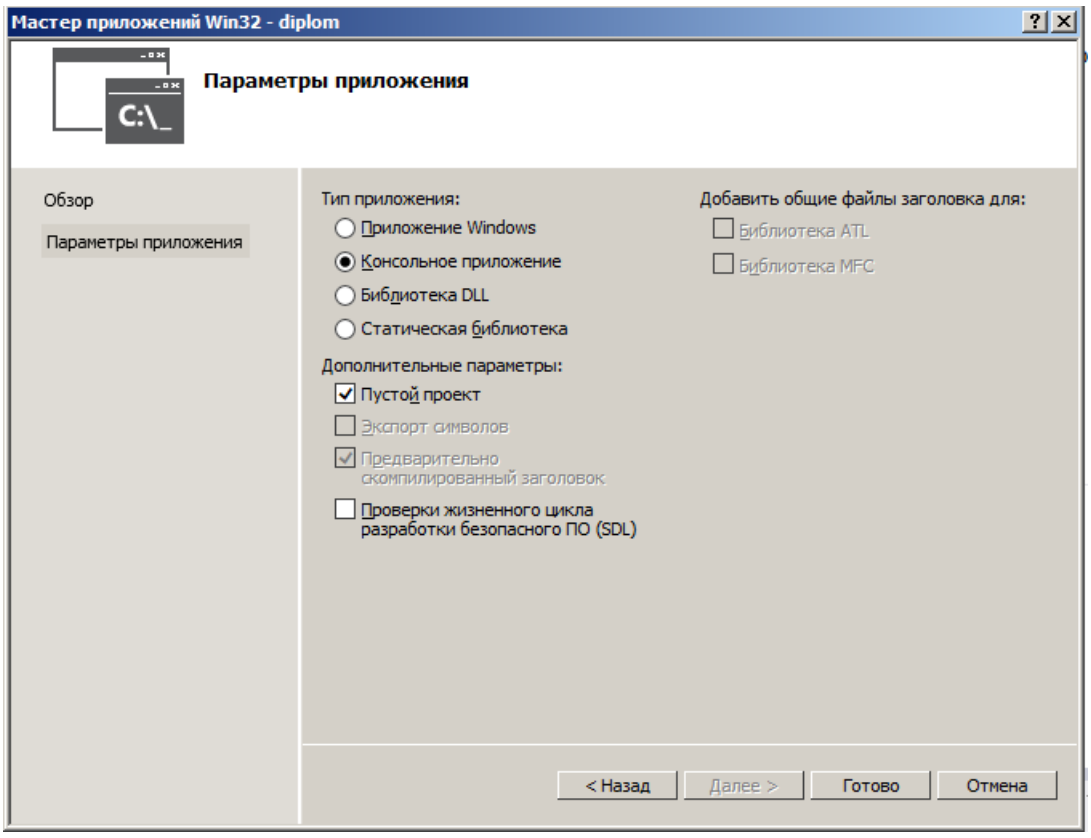


Рисунок 32 – Создание проекта в Visual Studio. Шаг 2

2. Подключаем библиотеку OpenCV и систему управления к проекту:

Перед подключением библиотеку нужно предварительно скачать с сайта разработчика, если вы не получили ее вместе с системой. Загрузить библиотеку можно по адресу: https://sourceforge.net/projects/opencvlibrary/. Для загрузки следуйте инструкциям на экране вашего интернет-обозревателя.

В данном примере скачанная библиотека размещена в папке C:\diplomcpp\opencv. Система размещена в папке C:\diplomcpp\include. Вы можете выбрать свой путь, тогда вам нужно будет указать этот путь при подключении.

Заходим в свойства проекта, вкладка c++ ->Общее

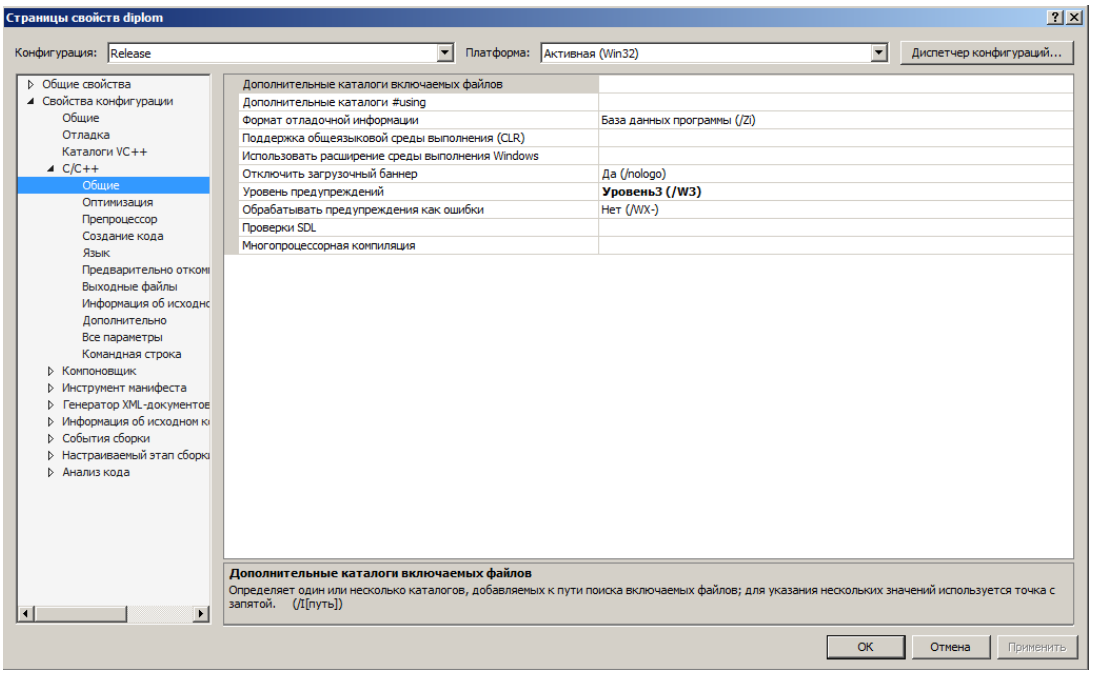


Рисунок 33 – Свойства проекта

Указываем дополнительные каталоги подключаемых файлов.

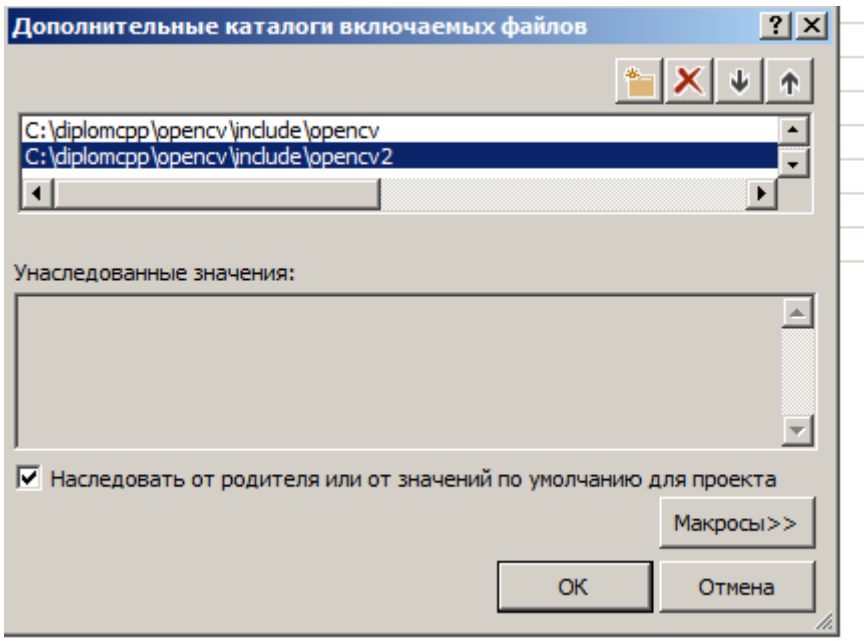


Рисунок 34 – Дополнительные каталоги подключаемых файлов

В форме дополнительные каталоги включаемых файлов укажите следующие папки:

- C:\diplomcpp\opencv\include
- C:\diplomcpp\opencv\include\opencv
- C:\diplomcpp\opencv\include\opencv2
- C:\diplomcpp\include

В свойствах проекта переходим в Компоновщик ->Общее и указываем дополнительные каталоги библиотек:

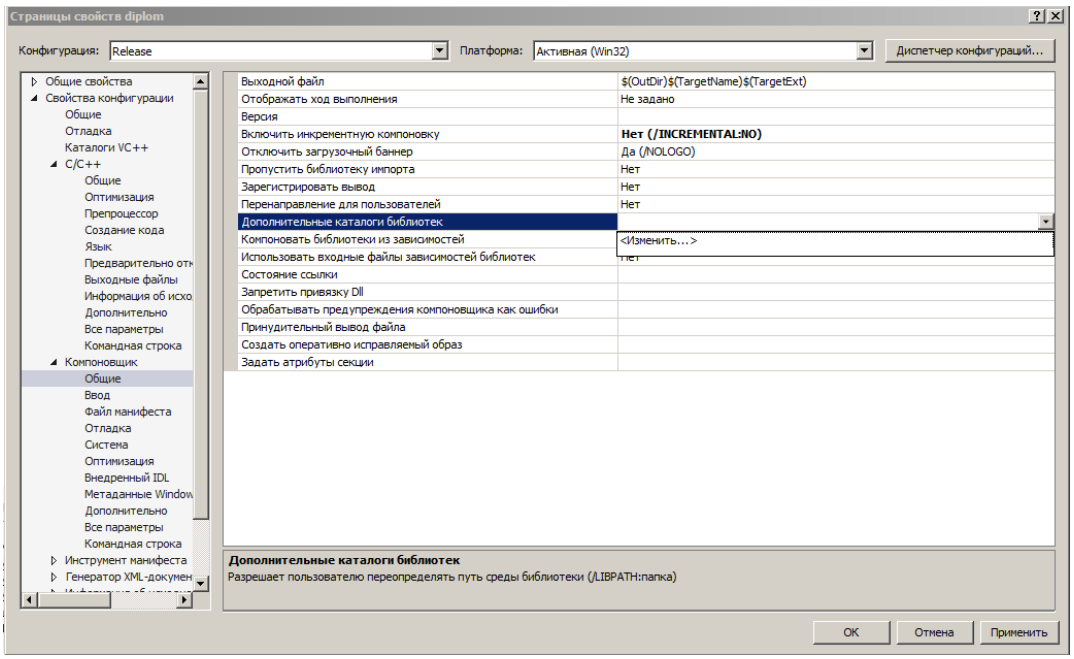


Рисунок 35 – Дополнительные каталоги библиотек

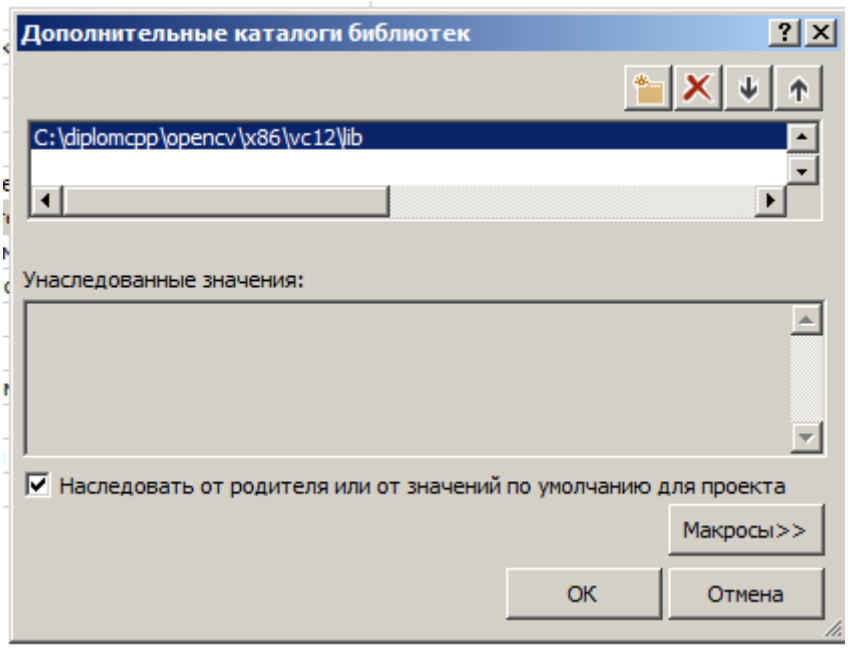


Рисунок 33 – Установка дополнительных каталогов библиотек

Далее, в разделе Компоновщик - Ввод указываем дополнительные зависимости для библиотеки OpenCV.

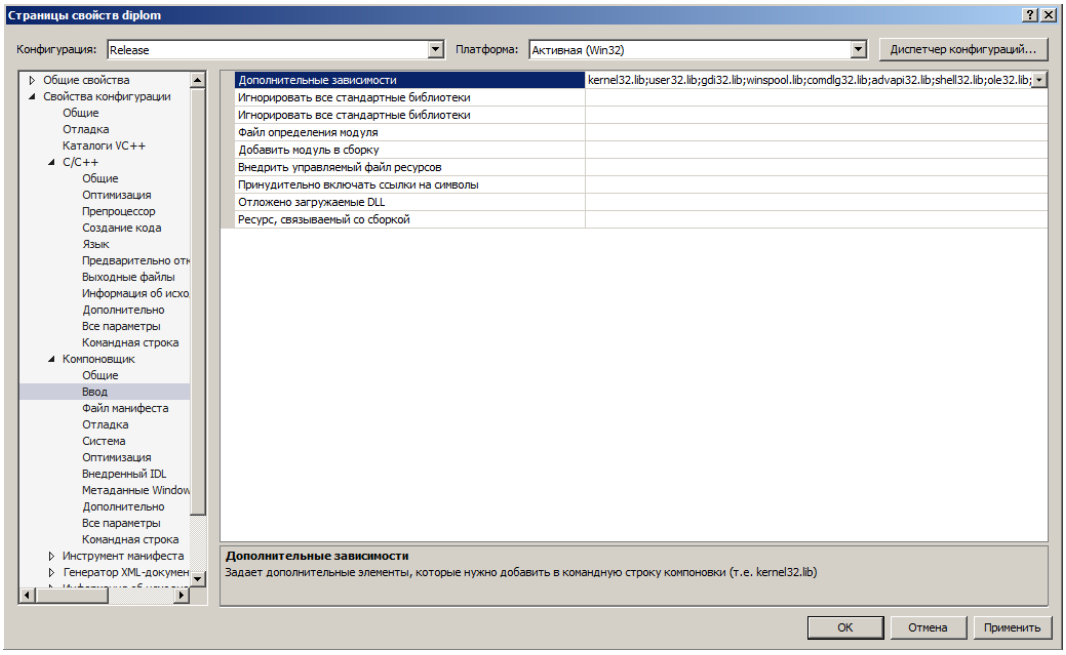


Рисунок 36 – Установка дополнительных зависимостей библиотек

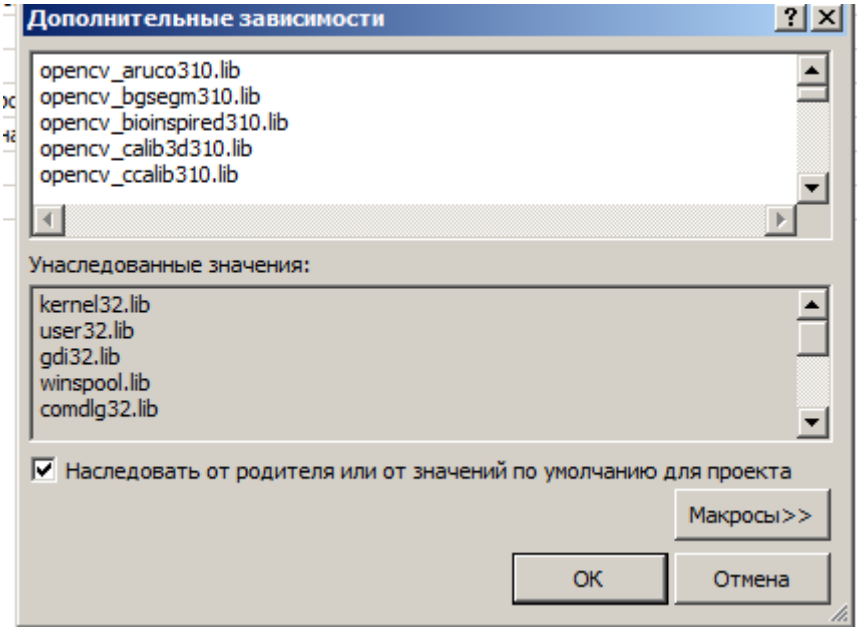


Рисунок 37 – Установка дополнительных зависимостей библиотек

В поле дополнительные зависимости вставьте следующие названия библиотек:

- opencv_aruco310.lib
- opencv_bgsegm310.lib
- opencv_bioinspired310.lib
- opencv_calib3d310.lib

- opencv_ccalib310.lib
- opencv_core310.lib
- opencv_datasets310.lib
- opencv_dnn310.lib
- opencv_dpm310.lib
- opencv_face310.lib
- opencv_features2d310.lib
- opencv_flann310.lib
- opencv_fuzzy310.lib
- opencv_highgui310.lib
- opencv_imgcodecs310.lib
- opencv_imgproc310.lib
- opencv_line_descriptor310.lib
- opencv_ml310.lib
- opencv_objdetect310.lib
- opencv_optflow310.lib
- opencv_photo310.lib
- opencv_plot310.lib
- opencv_reg310.lib
- opencv_rgbd310.lib
- opencv_saliency310.lib
- opencv_shape310.lib
- opencv_stereo310.lib
- opencv_stitching310.lib
- opencv_structured_light310.lib
- opencv_superres310.lib
- opencv_surface_matching310.lib
- opencv_text310.lib

- opencv_tracking310.lib
- opencv_ts310.lib
- opencv_video310.lib
- opencv_videoio310.lib
- opencv_videostab310.lib
- opencv_xfeatures2d310.lib
- opencv_ximgproc310.lib
- opencv_xobjdetect310.lib
- opencv_xphoto310.lib

Далее, добавьте в файл main.cpp вашего проекта заголовочный файл:

#include "EyeControlSystem.h"

Подключение библиотеки OpenCV и системы управления к проекту завершено.

3. Использование программного интерфейса.

Программный интерфейс позволяет использовать следующие режимы взаимодействия с пользователем:

– Диалог с возможностью положительного или отрицательного ответа на вопрос;

– Меню, с возможностью выбора одного элемента из списка;

– Регулятор с возможностью установки значения характеристики объекта.

Для обращения к системе используется класс EyeControlSystem.

Для начала использования системы в вашем приложении создайте объект класса EyeControlSystem.

Вызов диалога вопроса происходит с помощью метода question, который вызывается из объекта класса EyeControlSystem.

Метод question принимает следующие параметры:

text - Текст вопроса, тип данных std::string

YesValue - Значение для кнопки "Да" по-умолчанию, тип данных std::string

NoValue - Значение для кнопки "Нет" по-умолчанию, тип данных std::string

Метод имеет тип данных bool и возвращает ответ пользователя.

Вызов диалога меню происходит с помощью метода menu, который вызывается из объекта класса EyeControlSystem.

Метод menu принимает следующие параметры:

item - Список пунктов меню, состоящий из идентификатор и названия, тип данных std::vector, состоящий из пар элементов int - std::string (идентификатор пункта меню - название)

titie - Заголовок меню , тип данных std::string

Метод имеет тип данных int и возвращает идентификатор меню.

Вызов диалога регулятора происходит с помощью метода slider, который вызывается из объекта класса EyeControlSystem.

Метод slider принимает следующие параметры:

titie - Заголовок регулятора, тип данных std::string

minvalue - Минимальное значение регулятора, тип данных int

maxvalue - Максимальное значение регулятора, тип данных int

defvaue - Значение регулятора по-умолчанию, тип данных int

step - шаг изменения значения, тип данных int

Метод имеет тип данных int и возвращает значение регулятора, которое установил пользователь.

Примеры использования методов программного интерфейса:

```
// Диалог:
if (EyeControlSystem::question("Вам больше 18 лет?","Да","Нет"))
std::cout<<  "Пользователь ответил Да"<< std::endl;
// Меню:
std::vector<int,std::string> menu;
menu.push_back(std::make_pair(0,"Пункт 0"));
menu.push_back(std::make_pair(1,"Пункт 1"));
menu.push_back(std::make_pair(2,"Пункт 2"));
menu.push_back(std::make_pair(3,"Пункт 3"));
menu.push_back(std::make_pair(4,"Пункт 4"));
int index = EyeControlSystem::menu(menu,"Выберите пункт меню"));
std::cout<<  "Выбран пункт меню номер  "<< index << std::endl;
// Регулятор:
int k = EyeControlSystem::slider("Громкость",0,100,40,5);
std::cout<<  "Новая громкость звука: "<< k << std::endl;
```

5 БЕЗОПАСНОСТЬ И ЭКОЛОГИЧНОСТЬ

Данный раздел посвящен вопросам безопасности жизнедеятельности пользователей приложения. Необходимо определить на основе санитарно-эпидемиологических норм правила работы за ПК, способы безопасной утилизации носителей информации и компонентов ИС, а также меры, позволяющие предотвратить чрезвычайные ситуации, форс-мажоры и их нежелательные последствия. Исследоваться будут 3 положения: безопасность, экологичность и защита от ЧС.

Основным сборником нормативов, на основе которого будет проведен анализ аспектов БЖД, является СанПиН 2.2.2/2.4.1340-03.

5.1 Безопасность.

В данном подразделе рассмотрим правильные, с точки зрения БЖД, способы построения графических интерфейсов. Выделим 3 аспекта: зонирование или позиционирование элементов, цветовое оформление и типографические представления.

Построение экранных форм начинается с выбора структуры отображения ее элементов. Основное требование в данном вопросе – это интуитивно понимание функционально назначения всех компонентов. Для графических окон разработанной системы произведено разделение следующие блоки:

– верхняя часть, содержащая текст вопроса для пользователя;

– нижняя часть, содержащая три блока навигации, которые соответствуют трем используемым в системе положениям зрачка: влево, по центру и вправо.

Такой способ размещения блоков интуитивно понятен для пользователя и удобен в использовании. В частности стоит отметить, информационный блок занимает большую часть формы и принимает на себя фокус внимания пользователя. Для соответствия концепции адаптивной верстки содержимое блоков центрируется по горизонтали.

Согласно ГОСТ Р ИСО 6385-2007 учет эргономических аспектов при построении графических интерфейсов позволит добиться оптимизации производственной нагрузки, исключить эффекты расслабления, а также снизить вероятность появления производственного стресса. Потому рассмотрим некоторые правила проектирования эргономических интерфейсов.

Компоненты навигации являются наиболее используемыми в системе, поэтому должны быть в зоне досягаемости пользователя и легкодоступны. Для соответствия этому требованию блоки навигации занимают в совокупности половину площади экранной формы и легко доступы для взаимодействия.

Информационные сообщения и сигналы должны акцентировать внимание пользователя, чтобы избежать вероятности ошибки при работе с системой. В разработанной системе предусмотрена реакция формы на действие пользователя. В случае смещения зрачка влево или вправо, соответствующие блок в форме будет подсвечен.

Соблюдение подобных правил позволит субъекту системы сконцентрироваться на непосредственной работе, чем повысит ее эффективность.

От цветового оформления проекта зачастую зависит визуальный опыт работы с приложением. Работа за ПК, характеризуется тем, что монитор является источником света. Отсюда вытекают некоторые особенности.

Во-первых, важен правильный подбор яркости и контрастности элементов дизайна. Так тусклые цвета заставляют пользователя напрягать зрение, а яркие «режут» глаз.

Во-вторых, содержимое монитора периодически изменяется. Частая и резкая смена цветов мешает восприятию, развеивает внимание и негативно влияет на работу мозга.

В-третьих, работая за монитором, пользователь сильно концентрируется на содержимом экрана, что приводит к большим нагрузкам на зрительную систему пользователя.

Так же стоит отметить, что в теории цвета существует разделение на теплые (красный, желтый), холодные (синий, голубой) и нейтральные (серый) цвета.

В соответствие с этими принципами один из способов построения цветовой палитры интерфейса выглядит следующим образом. Для фоновых элементов и элементов, занимающих наибольшую площадь интерфейса, следует использовать нейтральные или неяркие холодные цвета. Для функциональных и управляющих элементов, с которыми взаимодействует пользователь, стоит использовать яркие холодные или теплые цвета. Для выделения важных компонентов или определения их текущего состояния, возможно использование ярких теплых оттенков, но количество таких элементов должно быть минимально.

Тогда сформируется концепция, позволяющая пользователю не напрягать свое зрение, работая с основным содержимым, функциональные компоненты всегда будут выделены, а при необходимости внимание будет акцентировано на важном элементе.

Данной концепции придерживается разрабатываемое приложение. В основе лежит белый оттенок, для отображения текстовой информации используется черный цвет, а блоки, отображающие реакцию на действия пользователя, выделяются ярким зеленым цветом.

Принципы, используемые в типографии, основаны на многовековом опыте верстки печатных изданий. Для компьютерных интерфейсов характерно заимствование этих принципов. Слишком маленький шрифт перенапрягает зрение пользователя, поэтому от него лучше отказаться. Так, используемый по умолчанию размер шрифта достаточно крупный для того, чтобы можно было комфортно его прочитать.

Еще один аспект - это выбор шрифтов, так шрифты можно разделить на шрифты без засечек и с засечками. Первые характерны для вывода на экран, а вторые для печатных изданий. Шрифты без засечек стоит использовать, когда:

– монитор низкого разрешения;

– читатели – дети;

– используется большое количество неконтрастных цветов;

– текст маленький или узкий.

Это позволит избежать резких пиксельных граней, сливания текста с фоном или слипания букв.

5.2 Экологичность.

Данный аспект рассмотрим с точки зрения сбора и утилизации отходов в виде ЭВМ, их составных частей, вспомогательного оборудования и оргтехники. ФЗ № 89 от 24.06.1998 г. (ред. от 28.12.2016) является основным в вопросах регулирования обращения с отходами производства и потребления с целью предотвращения вредного воздействия отходов на здоровье человека и окружающую среду.

В данном НПА определяется разделение отходов на классы опасности. Всего определенно 5 классов опасности. В Федеральном классификационном каталоге отходов выделены отдельные технические средства и их комплектующие. Для многих из них неопределенны классы и они устанавливаются в частном порядке. Например, системный блок компьютера определяется, как изделие из нескольких материалов, и имеет класс опасности – IV (малоопасные отходы). Аккумуляторы ноутбуков имеют класс опасности – II (высокоопасные отходы) и т.д.

В целом утилизация ЭВМ комплексный и сложный процесс, поэтому его стоит рассмотреть с разных сторон.

Во-первых, наиболее простым этот процесс представляется для физических лиц. Им необходимо обратиться в организацию, занимающуюся непосредственно утилизацией отходов. Такие организации должны пройти обязательное лицензирование своей деятельности.

Для юридических лиц этот процесс намного сложнее. Он также основан на передаче вычислительных средств сторонней организации, но этому

должен предшествовать этап списания аппаратного оборудования. Списание оборудования сопровождается оценкой их экологических свойств штатным или приглашенным экспертом, который составляет паспорт отходов оргтехники и компьютеров (вычислительной техники). Соответственно, организации выгоднее накапливать единицы непригодной в работе техники, а затем утилизировать ее в больших объемах. После этого она передается специализированной организации по утилизации.

Отдельно стоит отметить утилизацию информации на носителях и компонентах ЭВМ. Данным вопросом может заниматься как сторонняя организация, так и владелец техники. Способы и требования по уничтожению информации с носителей описываются в ГОСТ Р 50739-95, а также в РД от 30.03.1992 1 и 2, защита от НСД к информации. Согласно этим нормативам уничтожение может производиться, как с помощью блокирования доступа к информации на носителях, ее затиранию, а также дополнительным включением маскирующей информации.

Особое внимание стоит уделить утилизации компонентов, содержащих ртуть, в частности ртутных ламп. Ртуть – это чрезвычайно опасное вещество I класса опасности по ГОСТ 17.4.1.02-83. В соответствии с «Санитарно-эпидемиологическими требованиями к атмосферному воздуху» предельно допустимая концентрация в атмосферном воздухе ртути (ПДК) — 0,0003 мг/м3. Даже небольшое количество ртути может быть опасным, поэтому лампу, содержащую любое, даже самое малое количество ртути нельзя выбрасывать в мусорное ведро с бытовым мусором. С такими лампами следует обращаться как с опасными отходами, и их необходимо хранить таким образом, чтобы не допустить повреждения и после использования сдавать на переработку.

Наличие в компонентах ЭВМ технического золота или других драгоценных металлов накладывает на организацию дополнительную ответственность. Эти аспекты регулируются законодательством в соответствие с ФЗ №

41. Несоблюдение данных требований может повлечь административную ответственность.

5.3 Чрезвычайные ситуации.

Помещения, в которых происходит работа с ЭВМ, относят к категории В – пожароопасные помещения, согласно СП 12.13130.2009. Проблема обеспечения противопожарной безопасности в них является одной из основополагающих при рассмотрении аспектов БЖД.

Специфика эксплуатации ЭВМ подразумевает наличие большого количества электрических приборов, токопроводящих кабелей и высоких нагрузок на электросеть. Поэтому их установка, эксплуатация, техническое обслуживание, проверка, замена и утилизация должны соответствовать принятым законодательным нормам и стандартам.

При расположении ЭВМ необходимо учитывать не только их расположение внутри помещения, но взаимодействия друг с другом, а также расположение смежных помещений. Так, например, площадь одного рабочего места с ПК для взрослого должна составлять не менее 6 $м^2$, а объем не менее 20 $м^3$. Для хранения носителей информации, расходных и комплектующих частей ЭВМ или оргтехники, необходимо оборудовать соответствующее помещение, оборудованных негорящими стеллажами и шкафами.

Хранение технических средств должно осуществляться в закрытых контейнерах для предотвращения накопления пыли в их составных частях.

При эксплуатации ЭВМ и оргтехники необходимо проверять целостность токопроводящих кабелей, вилки и розетки, отсутствие повреждений аппаратуры.

Компоненты ЭВМ должны иметь функцию самоотключения при повышении температуры входе неисправности систем охлаждения и кондиционирования. Для предотвращения перегрева.

При работе электроприбором возможно образование статических зарядом на корпусах ЭВМ, периферии и оргтехники. Такие разряды могут приве-

сти к выводу из техники строя. Для их предотвращения необходимо использовать антистатическое покрытие полов, увлажнители воздухов и т.д.

Так же в помещениях, оборудованных ЭВМ, необходима установка средств пожаротушения. К таким средствам относятся огнетушители следующих конструкций: порошковые (ПСБ, ПФ, ОП). Так же распространение получили установки водяного, пенного и газового пожаротушения.

Для оповещения посетителей и работников помещения при возникновении пожар следует устанавливать средства пожарной сигнализации.

Технические средства должны проходить проверки и техническое обслуживание. Так необходимо проверять работоспособность, целостность и другие рабочие характеристики. Необходимо проводить уборку и очистку этих устройств. Так для удаления пыли и пятен должны применяться негорючие жидкости и материалы.

Таким образом, в данном разделе рассмотрены основные вопросы, связанные обеспечением БЖД при использовании ЭВМ. Подробно рассмотрены аспекты эргономичного проектирования интерфейсов взаимодействия с пользователем, проблемы утилизации ЭВМ, ее компонентов и вспомогательной техники, а также вопросы обеспечения пожарной безопасности.

ЗАКЛЮЧЕНИЕ

Произведенные работы в ходе выполнения выпускной квалификационной работы позволили создать систему управление графическим интерфейсом компьютера на основе положения зрачка глаза человека и разработать программное обеспечение к ней в виде приложения для ПК.

Этапы по анализу предметной области, проектированию информационной системы, разработке программного приложения, исследования вопросов информационной безопасности и рассмотрении аспектов безопасности жизнедеятельности определили следующие результаты:

– исследована предметная область;

– обоснована цель создания приложения;

– проанализированы, дополнены и оформлены требования к будущему приложению;

– выявлены функциональные и обеспечивающие подсистемы;

– согласно проектному описанию разработано приложение на языке программирования C++;

– Созданы руководства по эксплуатации для пользователя и руководство для интеграции для программиста;

– рассмотрены вопросы проектирования эргономичного графического интерфейса пользователя, утилизации ПК, их компонентов, оргтехники и комплектующих, а также вопросы противопожарной безопасности при работе с ЭВМ.

В совокупности были выполнены все поставленные для данной выпускной квалификационной работы задачи.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1 Дэвид А. Форсайт, Жан Понс. Компьютерное зрение. Современный подход.

2 Гонсалес Р. Вудс Р. Цифровая обработка изображений.

3 Лекториум [Электронный ресурс]/ Виктор Ерухимов - Компьютерное зрение и библиотека OpenCV. Режим доступа: https://www.lektorium.tv/course/22800. – «Лекториум»

4 IBM developerWorks© [Электронный ресурс] : офиц. сайт. – Режим доступа : https://www.ibm.com/developerworks/ru/. – 04.06.2017.

5 Маглинец, Ю. А. Анализ требований к автоматизированным информационным системам [Электронный ресурс]/ Ю.А. Маглинец – Электрон. текстовые данные. – М.: Интернет-Университет Информационных Технологий (ИНТУИТ), 2018. – 191 с. – Режим доступа: http://www.iprbookshop.ru/52184.html. – ЭБС «IPRbooks»

6 Yandex LLC - Компьютерное зрение. Лекция для Малого ШАДа Яндекса. [Электронный ресурс]. Режим доступа: https://habrahabr.ru/company/yandex/blog/203136/. – Информационный ресурс «habrahabr.ru»

7 Академия Intel. Введение в разработку мультимедийных приложений с использованием библиотек OpenCV и IPP. [Электронный ресурс]. Режим доступа: http://www.intuit.ru/studies/courses/10621/1105/lecture/17983?page=4 – Информационный ресурс «ИНТУИТ»

8 ItSeez LTD. Документация к библиотеке OpenCV. [Электронный ресурс] : офиц. сайт. – Режим доступа : http://docs.opencv.org/3.2.0/ – 04.06.2017.

9 Adrian Kaehler, Gary Bradski. Learning OpenCV 3. Computer Vision in C++ with the OpenCV Library.

ПРИЛОЖЕНИЕ А

Схема взаимодействия функциональных подсистем

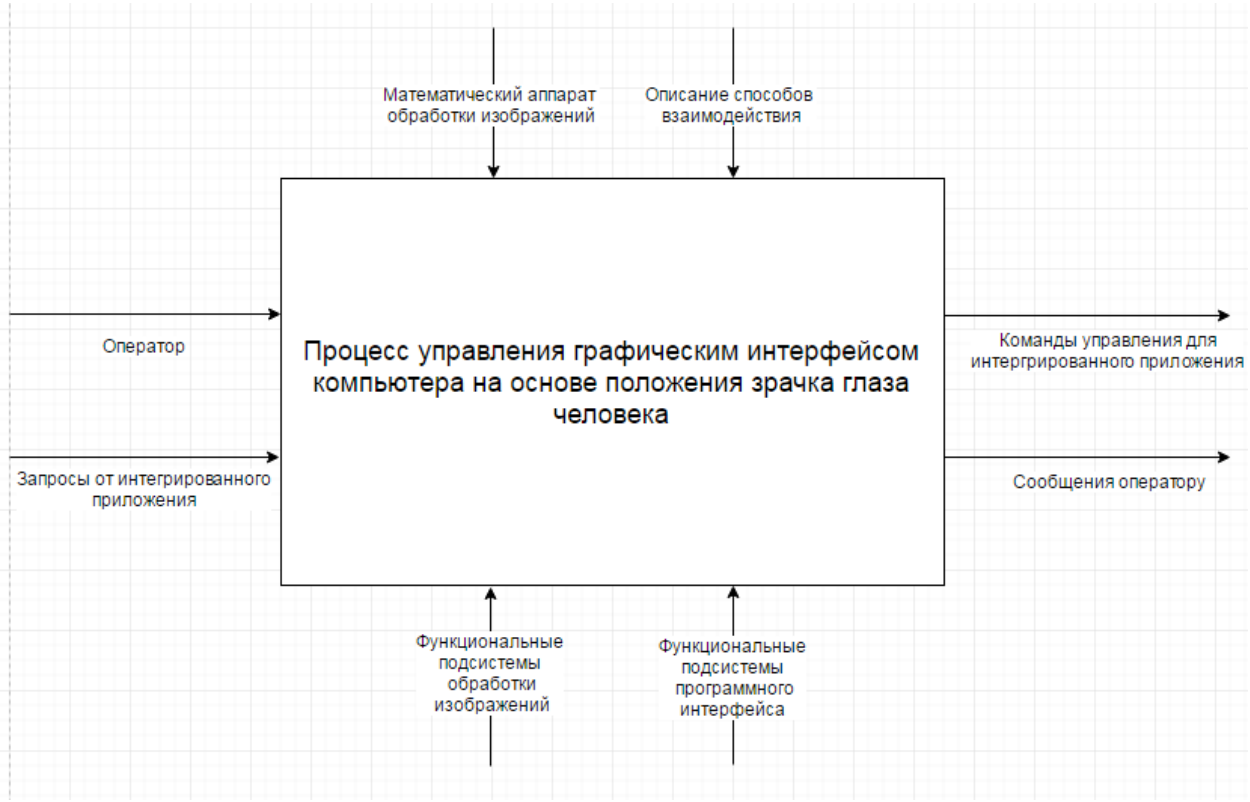


Рисунок А.1 – Схема процесса работы системы

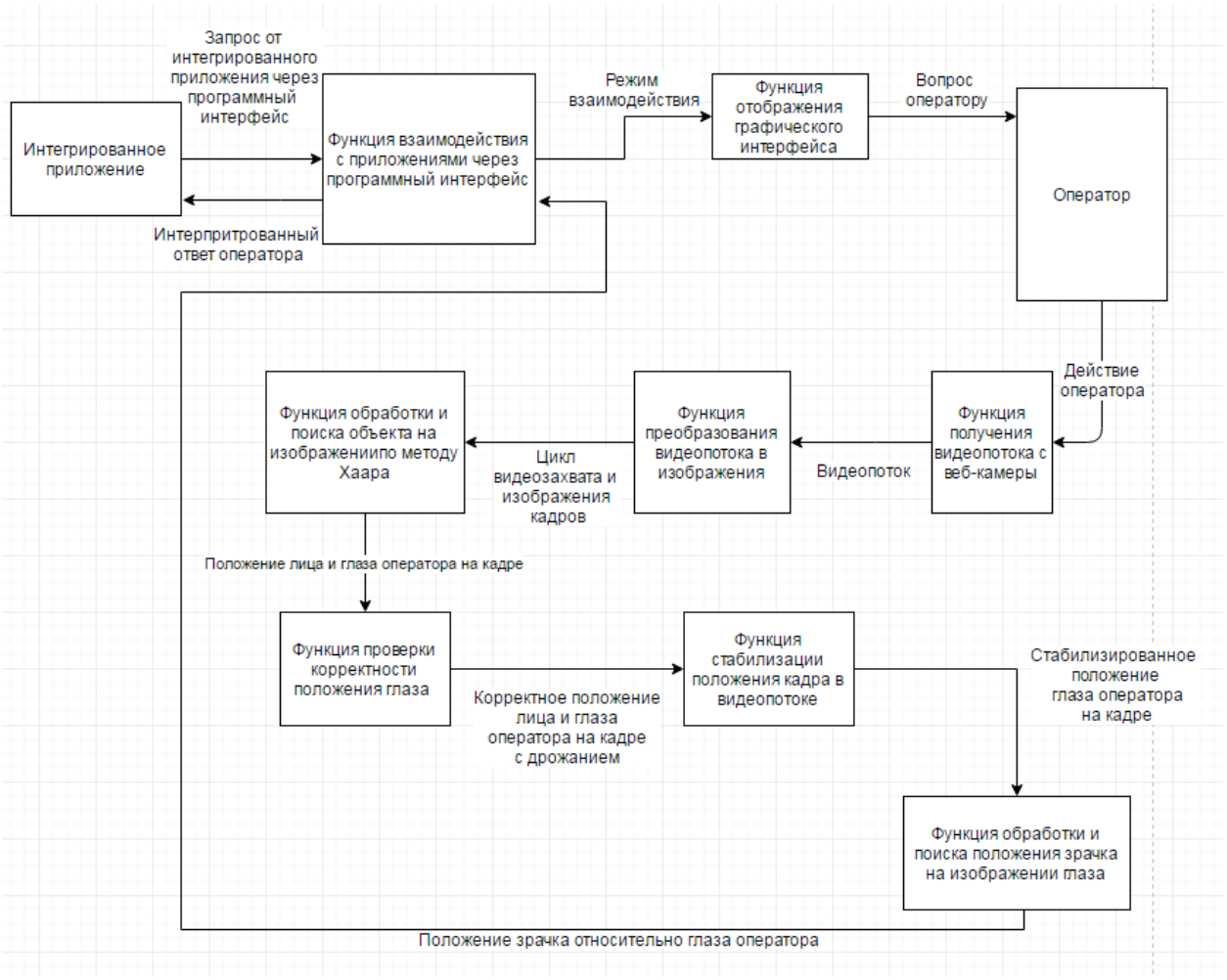


Рисунок А.2 – Схема взаимодействия функциональных подсистем

ПРИЛОЖЕНИЕ В

Изображения экранных форм приложения

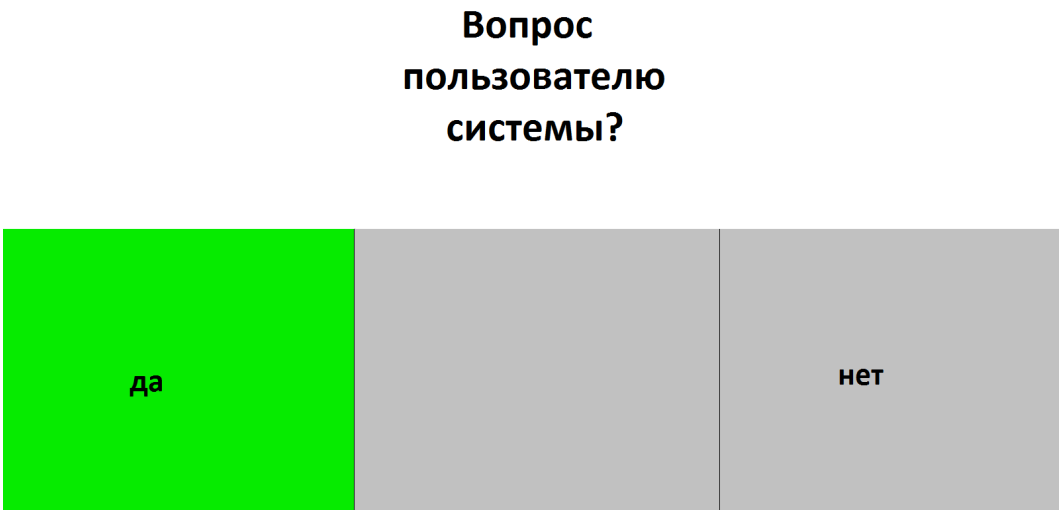


Рисунок Б.1 – Экранная форма диалога

Рисунок Б.2 – Экранная форма меню

Рисунок Б.3 – Экранная форма регулятора