МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«МИРЭА – Российский технологический университет»
РТУ МИРЭА

Институт искусственного интеллекта
(наименование института, филиала)

Кафедра автоматических систем
(наименование кафедры)

КУРСОВОЙ ПРОЕКТ (РАБОТА)

по дисциплине «Программирование микропроцессорных систем управления»

Тема курсовой работы «Цифровая система передачи сигналов в мобильном комплексе»

Студент группы КВБО-05-21 Обухова А. М.
(учебная группа, фамилия, имя, отчество студента) *(подпись студента)*
« обмено » 26.12.24

Руководитель курсового проекта (работы) преподаватель, к.т.н. Новоженин М.Б.
должность, звание, ученая степень *(подпись руководителя)*
Верхир А.С.

Работа представлена к защите «26» 12 2024 г.

Допущен к защите «26» 12 2024 г.

1. Согласно теме ВКР, определить структуру объекта управления, составить его математическую модель, рассмотреть задачи, решаемые микропроцессорной системой управления, границы использования системы управления.

   Цифровая система передачи сигналов м/у мобильным роботом и модулем кибербезопасности

2. Построить структурную схему связи системы управления и объекта управления и структурную схему управления, которая включает как минимум, два контура управления.

   Система контроля передаваемого сигнала

3. Обосновать выбор микропроцессора, представить схему подключения, и необходимую периферию для корректной работы, техническую документацию в необходимом объёме.

   Raspberry Pi, Arduino

4. Представить алгоритм управления с применением микропроцессорной системы.

   Алгоритм кодирования/декодирования сигнала, алгоритм проверки расшифровки шифрования

5. Разработать программу на языках программирования микроконтроллеров для данной структурной схемы управления. В качестве языков программирования должны использоваться языки Assembler или C (со вставками низкоуровневого кода, в тех случаях, где оно обоснованно). При программировании МП STM32 допускается использовать C++. Программирование ПЛК необходимо выполнить на языках стандарта МЭК 61131-3.

   Python, C++

6. Составить отчет, приложив в качестве подтверждения результаты испытания программы в эмуляторе или на натурном объекте.

   Программная реализация

Задание на курсовой проект (работу) выдал _____ Новоженин М.Б.
Подпись руководителя
«23» 09 2024 г.

Задание на курсовой проект (работу) получил _____ Обухов А.М.
Подпись обучающегося

МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«МИРЭА – Российский технологический университет»

РТУ МИРЭА

Институт искусственного интеллекта

(наименование Института)

Кафедра автоматических систем

(наименование кафедры)

Утверждаю

Заведующий кафедрой АС

_____ Лютов А.Г.

Подпись *ФИО*

«___» _____ 2024 г.

ЗАДАНИЕ

на выполнение курсового проекта (работы) по дисциплине

«Программирование микропроцессорных систем управления»

Студент Обухова Анастасия Михайловна Группа КВБО-05-21

Тема Цифровая система передачи сигналов в мобильном комплексе

Исходные данные: согласно варианту задания

Перечень вопросов, подлежащих обработке, и обязательного графического материала:

Математическая модель объекта управления, структурная схема управления; перечень измеряемых, контролируемых, управляемых и регулируемых параметров; блок-схема алгоритма работы программы; листинг программы для работы МП; результаты тестирования программы.

Срок представления к защите курсового проекта (работы): до «27» 12 2024 г.

Задание на курсовой проект (работу) выдал _____ **Новоженин М.Б.**

Подпись руководителя

«23» 09 2024 г.

Задание на курсовой проект (работу) получил _____ Обухова А.М.

Подпись обучающегося

МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«МИРЭА – Российский технологический университет»

РТУ МИРЭА

Институт искусственного интеллекта

(наименование института, филиала)

Кафедра автоматических систем

(наименование кафедры)

КУРСОВОЙ ПРОЕКТ (РАБОТА)

по дисциплине «Программирование микропроцессорных систем управления»

Тема курсовой работы «Цифровая система передачи сигналов в мобильном комплексе»

Студент группы КВБО-05-21 Обухова А. М. 

(учебная группа, фамилия, имя, отчество студента) *(подпись студента)*

Руководитель курсового проекта (работы) преподаватель, к.т.н. Новоженин М.Б.

должность, звание, ученая степень *(подпись руководителя)*

Работа представлена к защите «___» _____ 2024 г.

Допущен к защите «___» _____ 2024 г.

СОДЕРЖАНИЕ

Цель

Цель работы заключается в разработке и тестировании эффективной системы передачи сигналов между компьютерным зрением, реализованным с помощью языка программирования Python, и платформой Arduino, которая в дальнейшем будет использоваться для взаимодействия с мобильным комплексом. Эта система должна обеспечить надежную и быструю передачу данных, что является важным аспектом для создания современных автоматизированных решений.

Актуальность работы

Актуальность работы обуславливается растущим интересом к интеграции программного обеспечения и аппаратного обеспечения, что открывает новые горизонты для разработки мобильных и автоматизированных систем. В последние годы наблюдается активное развитие технологий, позволяющих объединять различные языки программирования с аппаратными платформами. Это создает возможности для создания более сложных и функциональных систем, которые могут эффективно взаимодействовать с окружающим миром. В данной курсовой работе будет подробно рассмотрен процесс настройки последовательного порта, который является ключевым для передачи данных между Python и Arduino. Также будет представлен алгоритм автоматизации процесса взаимодействия между Python и Arduino. Этот алгоритм позволит разработать гибкие и эффективные интерфейсы для управления различными устройствами, что значительно упростит процесс разработки и интеграции. Использование Python, как высокоуровневого языка программирования, позволяет быстро и удобно реализовывать сложные алгоритмы, что делает его идеальным выбором для задач, связанных с обработкой данных и взаимодействием с аппаратным обеспечением. В рамках работы будет проведен анализ существующих решений и примеров их реализации. Это поможет обосновать актуальность и значимость

данного исследования, а также выявить недостатки и ограничения уже существующих систем. Например, многие современные решения не учитывают возможность управления устройствами с помощью жестов, что становится все более актуальным в условиях стремительного развития технологий. Управление с помощью жестов предоставляет пользователям интуитивно понятные и естественные методы взаимодействия с роботами и другими автоматизированными системами. Этот подход использует технологии компьютерного зрения и распознавания жестов, которые делают робототехнические системы более доступными и эффективными в различных областях применения, таких как медицина, образование, промышленность и развлечения. Таким образом, данная курсовая работа не только изучает возможности интеграции Python и Arduino, но и рассматривает перспективы применения современных технологий для улучшения взаимодействия между человеком и машиной, что является важным шагом в развитии автоматизированных систем и робототехники.

Задачи

1. Изучить теоретические основы передачи данных через последовательный порт.
2. Разработать Python-скрипт для взаимодействия с Arduino.
3. Реализовать обработчики команд на Arduino.
4. Оценить эффективность предложенного решения.

Введение

Цифровые системы передачи данных представляют собой основу для обмена информацией между различными компонентами, такими как компьютеры, микроконтроллеры, мобильные устройства и т.д.. В данной работе мы сосредоточимся на разработке цифровой системы передачи сигналов между языком программирования Python и мобильным комплексом на базе Arduino. Arduino представляет собой популярную

платформу для создания прототипов и разработки различных электронных устройств, а Python является мощным инструментом для обработки данных и создания приложений. Совместное использование этих технологий открывает новые горизонты для реализации различных проектов, включая автоматизацию, мониторинг и управление.

На первом этапе работы мы рассмотрим основы цифровых систем передачи данных. Это позволит нам понять, как передаются сигналы, какие существуют методы и протоколы передачи информации, а также какие факторы влияют на качество и скорость передачи данных. Знание основ цифровых систем передачи данных является необходимым для успешной разработки и реализации системы взаимодействия между Python и Arduino.

Далее мы изучим протоколы связи, которые используются для организации взаимодействия между Python и Arduino.

Протокол передачи данных — это набор правил и соглашений, которые определяют способ обмена данными между устройствами и программами в сети. Протоколы связи играют важную роль в обеспечении надежного и эффективного обмена данными между устройствами. Мы рассмотрим наиболее популярные протоколы, такие как Serial, I2C, SPI и др., и проанализируем их преимущества и недостатки. Это знание поможет нам выбрать наиболее подходящий протокол для нашей системы передачи данных.

Подготовим среду для взаимодействия Python и Arduino, обсудим необходимые инструменты и библиотеки, которые облегчают процесс разработки, рассмотрим, как настроить Arduino и Python для совместной работы.

Напишем скрипты на Python для взаимодействия с Arduino. Запрограммируем Arduino для работы с данными от Python. Мы изучим, как писать код для Arduino, который будет обрабатывать входящие данные, отправлять ответы и выполнять определенные действия в зависимости от полученной информации. Программирование Arduino

требует знаний языка C/C++, который используется для разработки приложений на этой платформе.

После завершения этапов написания скриптов и программирования Arduino мы перейдем к тестированию и отладке нашей системы передачи данных. Этот процесс включает в себя проверку работоспособности системы, выявление и исправление возможных ошибок, а также оценку качества и надежности передачи данных. Тестирование является важным этапом, который позволяет убедиться в том, что разработанная система соответствует заявленным требованиям и работает корректно.

Наконец, мы проведем анализ результатов и сделаем выводы о проделанной работе. Мы обсудим, насколько успешно была реализована система передачи данных, какие проблемы были выявлены в процессе разработки и тестирования, а также какие перспективы открываются для дальнейшего развития проекта. Этот раздел станет итогом нашего исследования и позволит нам оценить достигнутые результаты.

Основы цифровых систем передачи данных

Основы цифровых систем передачи данных представляют собой ключевую область знаний, необходимую для понимания функционирования современных телекоммуникационных систем и цифровых устройств.

Цифровая передача данных — это процесс, при котором информация, представленная в виде двоичных данных, передается от одного устройства к другому. В отличие от аналоговых систем, где информация передается в виде непрерывных сигналов, цифровые системы используют дискретные значения, что позволяет значительно повысить устойчивость к помехам и искажениям. Основным элементом цифровой передачи данных является кодирование информации, которое позволяет преобразовать исходные данные в цифровую форму. Это кодирование

может быть реализовано различными способами, включая использование двоичных кодов, кодов с коррекцией ошибок и других методов.

Передача данных между устройствами, такими как компьютеры и микроконтроллеры, требует использования определенных протоколов и стандартов, которые описывают правила обмена данными. Протоколы передачи данных определяют, как данные упаковываются, как устанавливается соединение между устройствами, как осуществляется управление потоком данных и как обрабатываются ошибки. Наиболее распространенными протоколами являются UART (Universal Asynchronous Receiver-Transmitter), SPI (Serial Peripheral Interface) и I2C (Inter-Integrated Circuit). Каждый из этих протоколов имеет свои особенности и области применения, что делает их подходящими для различных задач.

UART — это асинхронный протокол, который используется для последовательной передачи данных. Он широко применяется в системах, где необходимо передавать данные между микроконтроллерами и компьютерами. Основное преимущество UART заключается в его простоте и низкой стоимости реализации. В отличие от других протоколов, UART не требует использования синхронизирующего сигнала, что упрощает подключение устройств. Однако, этот протокол имеет ограничения по скорости передачи данных и расстоянию, на которое может быть осуществлена передача.

SPI, с другой стороны, представляет собой синхронный протокол, который позволяет передавать данные с более высокой скоростью. Он использует несколько проводов для передачи данных, включая тактовый сигнал, что обеспечивает синхронизацию между передающим и принимающим устройствами. SPI подходит для приложений, где важна высокая скорость передачи данных, например, в системах, работающих с большими объемами информации, таких как камеры и сенсоры. Однако, использование SPI требует больше проводов, что может усложнить проектирование схемы.

I2C — это еще один популярный протокол, который позволяет подключать несколько устройств к одной шине. Он использует всего два провода для передачи данных, что делает его более экономичным в плане использования пространства на плате. I2C также поддерживает возможность адресации устройств, что позволяет нескольким устройствам обмениваться данными по одной линии. Этот протокол часто используется в системах, где необходимо подключить множество датчиков и других компонентов, таких как в мобильных комплексах на Arduino.

Кроме того, важным аспектом цифровых систем передачи данных является управление потоком. При передаче данных между устройствами может возникнуть ситуация, когда одно устройство отправляет данные быстрее, чем другое может их обрабатывать. Это может привести к переполнению буфера и потере данных. Для предотвращения таких ситуаций используются различные механизмы управления потоком, такие как стоп-биты, подтверждения получения данных и протоколы, обеспечивающие контроль за скоростью передачи.

В контексте разработки системы передачи данных между Python и Arduino, необходимо учитывать, что Python является высокоуровневым языком программирования, который предоставляет множество библиотек и инструментов для работы с данными и сетевыми соединениями. С другой стороны, Arduino представляет собой платформу для разработки аппаратных решений, где программирование осуществляется на уровне ниже, чем Python. Это различие в уровнях абстракции требует использования специальных библиотек и инструментов для обеспечения совместимости между двумя системами. Например, для работы с последовательным портом в Python можно использовать библиотеку PySerial, которая позволяет устанавливать соединение с Arduino и обмениваться данными через UART.

Таким образом, основы цифровых систем передачи данных охватывают широкий спектр тем, включая кодирование информации,

протоколы передачи, управление потоком. Понимание этих основ является необходимым для успешной разработки и реализации системы передачи данных между Python и мобильным комплексом на Arduino. Важно учитывать особенности каждого из используемых протоколов, а также возможности и ограничения используемых языков программирования и платформ, чтобы создать эффективную и надежную систему, способную решать поставленные задачи.

Необходимые компоненты для выполнения работы

Для реализации данного проекта используются следующие компоненты:

- Плата Arduino Nano – отладочная плата на базе 8-битного микроконтроллера ATmega328p семейства AVR.
- Ноутбук с веб-камерой
- USB-кабель для подключения Arduino
- Bluetooth модуль HC-05
- Провода и резисторы для подключения схемы
- Беспаечная макетная плата GSMIN MB-101

Выполним подключение Bluetooth модуля к Arduino. Bluetooth — это распространенный протокол радиосвязи для коротких дистанций, которым пользуются большинство современных электронных устройств. Главные достоинства Bluetooth: хорошая устойчивость к широкополосным помехам и простота реализации. Самыми популярным на сегодня Bluetooth модулем является HC-05. Он может работать как в режиме ведущего (slave), так и в режиме ведомого (master).

Вывод	Значение
EN	включение/выключение модуля
VCC	питание +5В
GND	земля

TXD, RXD	UART интерфейс для общения с контроллером
STATE	индикатор состояния
KEY	нога для входа в режим AT-команд

Таблица 1. Распиновка HC-05

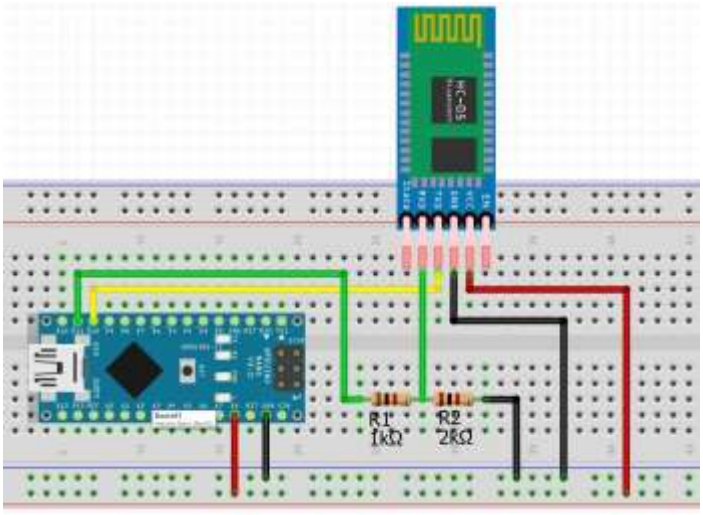

Рисунок 1. Схема подключения Bluetooth модуля HC-05 к Arduino Nano

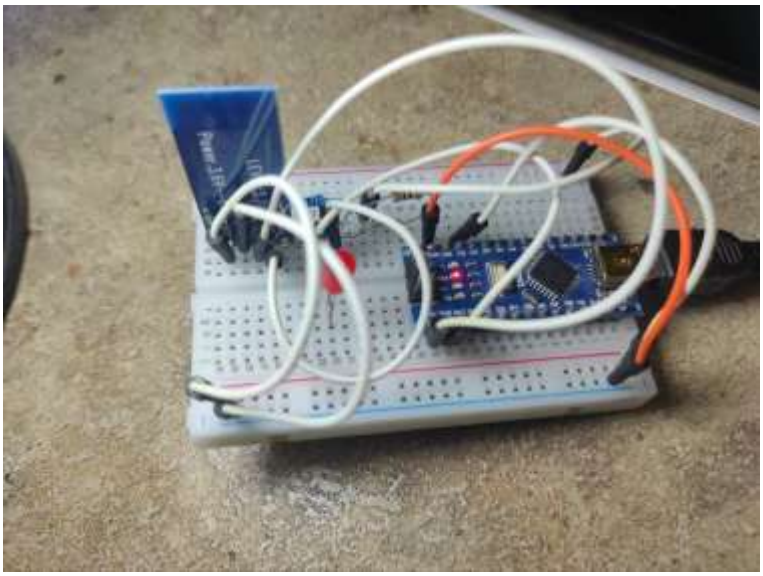

Рисунок 2. Фото подключенного Bluetooth модуля HC-05 к Arduino Nano

Программирование на Python

Программный код разработан в IDE PyCharm. Полная версия кода представлена в ПРИЛОЖЕНИИ А.

Скрипт на Python решает две основные задачи:

1. Взаимодействие с Arduino.
2. Обработка жестов рук с использованием веб-камеры.

Для реализации первой задачи подключаем необходимую библиотеку serial.tools.list_ports, которая позволяет работать с последовательным портом для передачи данных на Arduino или другие микроконтроллеры. Создаем переменную для работы с портом: ser = serial.Serial("COM3", 9600, timeout=1), где "COM3" - название порта, а 9600 бит/сек - скорость передачи данных с таймаутом в 1 секунду. Проверяем успешность подключения к порту. Если подключение не удалось, программа завершает свою работу. Теперь с помощью переменной ser мы можем отправлять сигналы на Arduino для дальнейшей обработки.

Для обработки жестов рук с веб-камеры участвует компьютерное зрение, которое занимается получением, обработкой и анализом визуальной информации. Для этой цели используются библиотеки mediapipe (для распознавания точек) и OpenCV (для визуализации изображения). OpenCV (Open Source Computer Vision Library) - это библиотека с открытым исходным кодом для работы с компьютерным зрением. Она включает в себя алгоритмы компьютерного зрения на основе машинного обучения для обработки изображений, видео, калибровки камеры и 3D-реконструкции, обнаружения объектов и других задач. MediaPipe - это кроссплатформенный фреймворк машинного обучения с открытым исходным кодом, предназначенный для создания сложных и мультимодальных конвейеров прикладного машинного обучения. Он может использоваться для разработки передовых моделей машинного обучения, таких как распознавание лиц, отслеживание рук, обнаружение объектов и их отслеживание и других задач.

С помощью библиотеки MediaPipe получим ключевые точи руки ПРИЛОЖЕНИЕ Б Именно эта структура и будет использоваться для распознования жестов

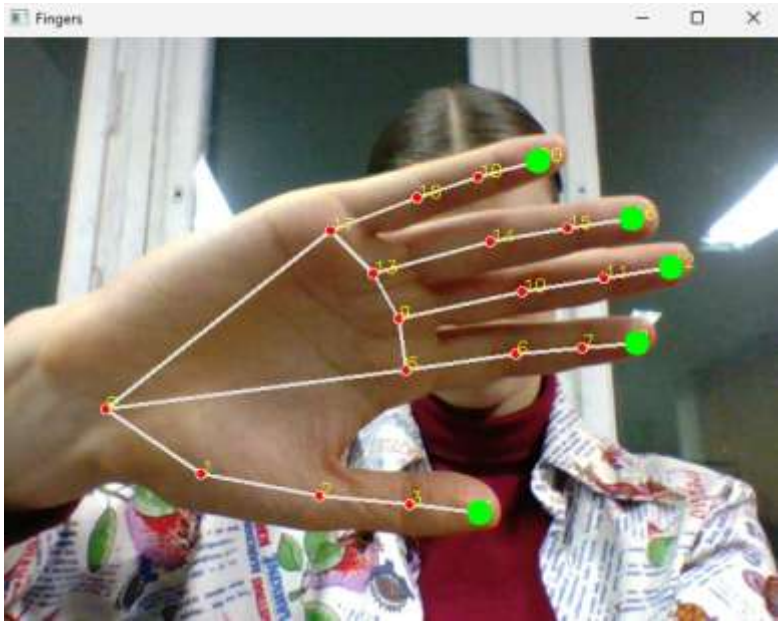

Рисунок 3. Распознование кисти руки и прорисовка веркотров, точек

В коде мы реализуем открытие камеру, считывание видеопотока и передачи его на обработку в библиотеку MediaPipe. Важные параметры:

- static_image_mode=False - гарантирует, что одна и та же рука будет непрерывно определяться в потоковом видео.
- max_num_hands=1 - исключает обработку других найденных рук в кадре.

Из-за сложности метода определения расстояния между крайними точками большого пальца, мы не будем рассматривать его в этом проекте. Мы выберем жесты руки для определения направления движения, основанные на согнутых и разогнутых пальцах. Для нас важно будет знать номер пальца и его положение относительно других пальцев. Присвоим каждому жесту номер, который мы будем отправлять на Arduino: Вперед – 1, Назад – 2, Налево – 3, Направо – 4, Стоп - 0.

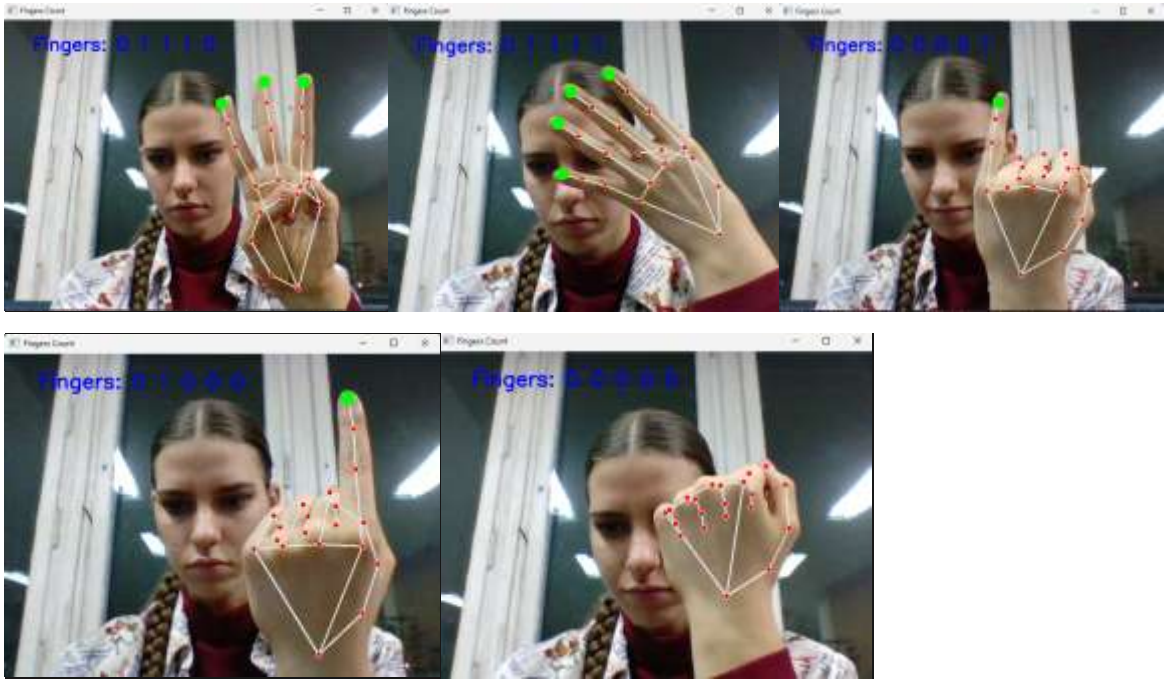

Рисунок 4. Жесты вперед-назад-налево-направо-стоп

Инициализируем следующие переменные:

- mp_hands - модель для обнаружения рук
- mp_drawing - утилита для рисования
- hands - экземпляр класса Hands для обнаружения рук на изображении
- tip_ids и base_ids - индексы кончиков пальцев и базы пальцев

Создадим объект VideoCapture с помощью библиотеки OpenCV (cv2), который будет использоваться для захвата видео с устройства по веб-каамере. Параметр 0 указывает, что мы хотим использовать первое доступное устройство захвата видео на компьютере: cap = cv2.VideoCapture(0).

extension_threshold используется для определения степени сгиба пальца.

Введём следующие функуии:

- get_vector(p1, p2) - возвращает вектор от точки p1 к точке p2
- is_finger_extended(base, tip, is_thumb=False) - определяет, разогнут ли палец, исходя из его вектора
- count_fingers(hand_landmarks) - подсчитывает количество разогнутых пальцев на изображении руки

Запустим бесконечный цикл обработки изображения:

1. Захватим изображения с веб-камеры
2. Обработаем изображения рук с использованием библиотеки mediapipe
3. Определим количество и состояния разогнутых пальцев на изображении
4. Подсветим кончики разогнутых пальцев на изображении
5. Выводим состояние каждого пальца на изображении
6. Передача значений состояний пальцев через последовательный порт в Arduino
7. Отображение обработанного изображения с количеством пальцев и их состоянием
8. Выход из цикла при нажатии на клавишу Esc

После завершения выполнения работы цикла необходимо освобождить ресурсы. Поэтому мы остановим захват изображения с веб-камеры и закроем все окона OpenCV.

Программирование на Arduino

Программный код реализован в среде программирования ArduinoIDE. Полная версия кода предложена а ПРИЛОЖЕНИЕ В.

Этот код на Arduino создает соединение между Arduino и Bluetooth модулем. Он использует библиотеку SoftwareSerial для создания вторичного последовательного порта на пинах 10 и 11, который подключен к Bluetooth модулю. Библиотека SoftwareSerial позволяет реализовать последовательный интерфейс на любых других цифровых выводах Arduino с помощью программных средств, дублирующих функциональность UART.

В функции setup() инициализируется основной последовательный порт с скоростью 9600 бит/сек и вторичный порт для Bluetooth с скоростью 38400 бит/сек.

В функции loop() проверяется наличие данных на основном последовательном порту. Если данные доступны, то символ считывается с

основного порта, отправляется обратно в монитор порта и затем передается в Bluetooth модуль.

Также проверяется наличие данных на вторичном порту от Bluetooth модуля. Если данные доступны, то символ считывается с Bluetooth модуля и выводится в монитор порта.

Таким образом, Arduino может получать данные через основной последовательный порт, и Bluetooth модуль может отправлять данные.

Тестирование системы передачи сигналов

Для проверки работоспособности системы, выполним следующие действия:

- Подключим плату Arduino к компьютеру с помощью USB-кабеля.
- В Arduino IDE выберите правильный порт в меню "Инструменты" -> "Порт".
- Загрузим скетч на плату Arduino.

Запись скетча на Arduino должен происходить без подключенного Bluetooth модуля, иначе возникает конфликт системы и выскакивает ошибка

Чтобы убедится, что Bluetooth модуль действительно получает и отправляет сигналы, подключим его к мобильному телефону. Установим на телефон любой доступный Bluetooth Terminal и выполним подключение. Находим в списке расстройств "HC-05" и подключаемся к нему. Телефон спросит пин-код, необходимо ввести "1234" или "0000". Устройство подключено.

Основной запуск системы происходит из среды програмирования Python.

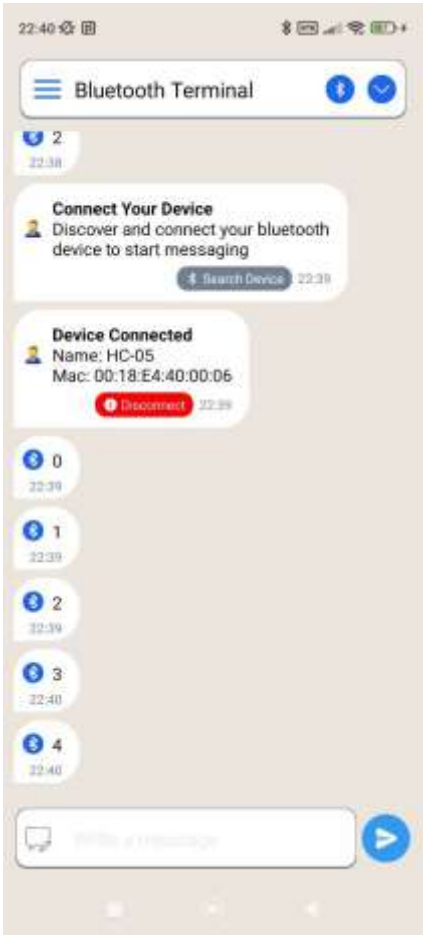

Рисунок 5. Вывод выполнения работы на Bluetooth Terminal

В результате мы получаем на мобильное устройство от Bluetooth модуля числа, которые соответствуют определенному направлению движения в зависимости от днмонстрируемого жеста. Значит, система выполняет поставленную перед нами задачу.

Заключение

В данной курсовой работе мы реализовали передачу данных компьютерного зрения с Python на Arduino. Мы изучили основные аспекты работы с Arduino, а также представили практическую реализацию обмена данными с использованием протокола Serial. Смогли передать данные дальше по Bluetooth модулю на мобильное устройство и убедились в корректности передаваемых данных.

Список использованной литературы

1. OpenCV Documentation. (n.d.). Retrieved from https://docs.opencv.org/4.x/
2. Arduino Documentation. (n.d.). Retrieved from https://www.arduino.cc/en/Reference/HomePage
3. pySerial Documentation. (n.d.). Retrieved from https://pyserial.readthedocs.io/en/latest/
4. Rifa, Tasfia., Zeratul, Izzah, Mohd, Yusoh., Adria, Binte, Habib., Tousif, Mohaimen. (2024). An overview of hand gesture recognition based on computer vision. International Journal of Electrical and Computer Engineering, doi: 10.11591/ijece.v14i4.pp4636-4645
5. Junjie, Ma., Yinqing, Xie., Hongyu, Jiang., Yihan, Wang. (2024). An algorithm for intelligent human-computer interaction gesture recognition. doi: 10.1117/12.3033276

ПРИЛОЖЕНИЕ А

```
import cv2
import mediapipe as mp
import numpy as np
import serial.tools.list_ports
import time

ser = serial.Serial("COM3", 9600, timeout=1)
if ser is None:
    print("Error")
    exit()  # Завершаем программу, если подключение не удалось
time.sleep(2)  # Ждем открытия порта

mp_hands = mp.solutions.hands
mp_drawing = mp.solutions.drawing_utils
hands = mp_hands.Hands(static_image_mode=False, max_num_hands=1,
min_detection_confidence=0.7)
tip_ids = [4, 8, 12, 16, 20]  # Индексы кончиков пальцев
base_ids = [0, 5, 9, 13, 17]  # Индексы баз пальцев

cap = cv2.VideoCapture(0)

extension_threshold = 0.17  # Общий порог для большинства пальцев

def get_vector(p1, p2):
    """ Возвращает вектор от точки p1 к точке p2 """
    return np.array([p2.x - p1.x, p2.y - p1.y, p2.z - p1.z])


def is_finger_extended(base, tip, is_thumb=False):
    """ Определяет, разогнут ли палец, исходя из его вектора """
    base_to_tip = get_vector(base, tip)
    # Нормализация вектора
    base_to_tip_norm = np.linalg.norm(base_to_tip)
    # Проверка на разгибание, учитывая, является ли это большим
пальцем
    if is_thumb:
        return base_to_tip_norm > thumb_extension_threshold
    else:
```

```
        return base_to_tip_norm > extension_threshold


def count_fingers(hand_landmarks):
    finger_count = 0
    extended_fingers = []
    finger_states = [0, 0, 0, 0, 0]  # Состояние пальцев: 0 - сжат, 1 - разогнут

    if hand_landmarks:
        landmarks = hand_landmarks.landmark
        # Проверка пальцев
        for i in range(1, 5):
            if is_finger_extended(landmarks[base_ids[i]], landmarks[tip_ids[i]]):
                finger_count += 1
                extended_fingers.append(tip_ids[i])
                finger_states[i] = 1

    return finger_count, extended_fingers, finger_states


while True:
    ret, frame = cap.read()
    if not ret:
        continue

    frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
    results = hands.process(frame)
    frame = cv2.cvtColor(frame, cv2.COLOR_RGB2BGR)

    if results.multi_hand_landmarks:
        for hand_landmarks in results.multi_hand_landmarks:
            mp_drawing.draw_landmarks(frame, hand_landmarks, mp_hands.HAND_CONNECTIONS)
            fingers_counted, extended_fingers, finger_states = count_fingers(hand_landmarks)

            # Подсветка кончиков разогнутых пальцев
            for tip_index in extended_fingers:
                tip_landmark = hand_landmarks.landmark[tip_index]
                x, y = int(tip_landmark.x * frame.shape[1]), int(tip_landmark.y *
```

```
frame.shape[0])
            cv2.circle(frame, (x, y), 10, (0, 255, 0), cv2.FILLED)

        # Вывод состояния каждого пальца
        finger_state_text = ' '.join(['1' if state else '0' for state in finger_states])
        cv2.putText(frame, f'Fingers: {finger_state_text}', (50, 50),
cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 0, 0), 2,
                    cv2.LINE_AA)

        # Передаем значение пальцев в Arduino
        match finger_state_text:
            case "0 0 0 0 0":           #Стоп
                ser.write("0".encode())
            case "0 1 1 1 0":           #Вперёд
                ser.write("1".encode())
            case "0 1 1 1 1":           #Назад
                ser.write("2".encode())
            case "0 0 0 0 1":           #Налево
                ser.write("3".encode())
            case "0 1 0 0 0":           #Направо
                ser.write("4".encode())


    cv2.imshow('Fingers Count', frame)

    if cv2.waitKey(10) & 0xFF == 27:
        break

cap.release()
cv2.destroyAllWindows()
```

ПРИЛОЖЕНИЕ Б

```
import cv2
import mediapipe as mp
import numpy as np

mp_hands = mp.solutions.hands
mp_drawing = mp.solutions.drawing_utils
hands = mp_hands.Hands(static_image_mode=False, max_num_hands=1)

cap = cv2.VideoCapture(0)

while True:
    ret, frame = cap.read()
    if not ret:
        continue

    frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
    results = hands.process(frame)
    frame = cv2.cvtColor(frame, cv2.COLOR_RGB2BGR)

    if results.multi_hand_landmarks:
        for hand_landmarks in results.multi_hand_landmarks:
            mp_drawing.draw_landmarks(frame, hand_landmarks,
mp_hands.HAND_CONNECTIONS)

    cv2.imshow('Fingers', frame)

    if cv2.waitKey(10) == 27:
        break

cap.release()
cv2.destroyAllWindows()
```

ПРИЛОЖЕНИЕ В

```
#include <SoftwareSerial.h>

SoftwareSerial bluetooth(10, 11);

void setup() {
  Serial.begin(9600);  // Инициализация последовательного порта
  bluetooth.begin(38400);
}

void loop() {
  if (Serial.available() > 0) {
    char receivedChar = Serial.read();  // Чтение полученного символа
    Serial.println(receivedChar);  // Отправка полученного значения обратно
    bluetooth.print(receivedChar);
  }
  if (bluetooth.available()) {
    char receivedChar = bluetooth.read();  // Читаем символ из Bluetooth модуля.
    Serial.print(receivedChar);  // Выводим символ в монитор порта.
  }
}
```