УДК 004.93

РАЗРАБОТКА ИГРОВОГО ВЕБ-ПРИЛОЖЕНИЯ С ИСПОЛЬЗОВАНИЕМ ТЕХНОЛОГИИ РАСПОЗНАВАНИЯ ЖЕСТОВ ДЛЯ УПРАВЛЕНИЯ В РЕАЛЬНОМ ВРЕМЕНИ

Кормин Н.А., студент гр.ПИб-222, IVкурс
Красных Г.С., студент гр.ПИб-222, IV курс
Научный руководитель: Глебова Е.А., старший преподаватель
Кузбасский государственный технический университет
имени Т.Ф. Горбачева, г. Кемерово

Современные технологии компьютерного зрения открывают новые возможности для создания естественных интерфейсов взаимодействия человека с компьютером. Одним из перспективных направлений является управление приложениями с помощью жестов рук, что особенно актуально для игровых систем, где требуется высокая скорость реакции и отсутствие физических контроллеров. В данной работе представлена система, позволяющая управлять двухмерной браузерной игрой с помощью движений указательного пальца, отслеживаемых через обычную веб-камеру.

Целью работы является разработка и демонстрация программного комплекса, который выполняет:

- захват видео с веб-камеры;
- детекцию руки и её ключевых точек с помощью MediaPipe;
- определение направления указательного пальца (вверх-вниз-влево-вправо);
- сглаживание получаемых данных для устранения дрожания;
- передачу команд через WebSocket в браузерную игру;
- визуализацию результатов на экране.

Система написана на языке Python с использованием асинхронного программирования и многопоточности для обеспечения высокой производительности. В качестве игрового клиента выступает любое веб-приложение, способное принимать команды по WebSocket.

OpenCV - библиотека компьютерного зрения, используемая для захвата видеопотока, обработки изображений и отрисовки аннотаций.

MediaPipe - фреймворк от Google, предоставляющий готовые модели для отслеживания ключевых точек тела, лица и рук. В проекте используется модель HandLandmarker, которая выдаёт 21 точку на каждой руке с высокой точностью.

WebSocket - протокол полнодуплексной связи поверх TCP, позволяющий обмениваться данными в реальном времени между сервером (на Python) и клиентом (браузерная игра).

Asyncio - библиотека для асинхронного программирования, обеспечивающая одновременную обработку WebSocket-соединений и потока видео без блокировок [2].

Tkinter - стандартная библиотека Python для получения размеров экрана, что позволяет разместить окно с видео в удобном месте.

Система состоит из трёх основных компонентов. Модуль захвата видео работает в отдельном потоке, постоянно считывает кадры с камеры и помещает их в очередь. Модуль обработки в основном асинхронном цикле извлекает кадры из очереди, передаёт их в детектор MediaPipe, анализирует результаты и формирует команды. WebSocket-сервер принимает подключения от браузерной игры и рассылает команды всем подключённым клиентам. Взаимодействие между потоком захвата и асинхронным циклом осуществляется через asyncio.Queue, что позволяет безопасно передавать данные между потоками.

Модель hand_landmarker.task загружается автоматически по URL при первом запуске с помощью функции download_model. Это позволяет избежать ручного скачивания и упрощает развёртывание. В коде сначала проверяется существование файла, и если его нет, выполняется скачивание [1].

```
def download_model(url, filename):
    if not os.path.exists(filename):
        print(f"Downloading model {filename}...")
        urllib.request.urlretrieve(url, filename)
        print("Model downloaded.")
```

Рис.1 Функция скачивания модели

Определение направления указательного пальца.

Для каждой обнаруженной руки вычисляются координаты основания указательного пальца (точка 5) и его кончика (точка 8). Разность координат даёт вектор направления. В коде учитывается зеркалирование по оси X, чтобы движения пользователя соответствовали естественным ожиданиям (поднятый палец вправо даёт команду «вправо»). Функция get_finger_direction_for_hand возвращает строковое обозначение одного из четырёх направлений: "right-up" (вправо-вверх), "right-down" (вправо-вниз), "left-up" (влево-вверх), "left-down" (влево-вниз) или None, если палец направлен не в одну из этих сторон.

На основе полученного направления формируется JSON-сообщение, содержащее тип команды ("button") и координаты направления (x и y). Это сообщение отправляется всем подключённым клиентам через WebSocket. Маппинг направлений в команды задаётся словарём DIRECTION_TO_COMMAND. Благодаря этому игра может интерпретировать полученные данные как нажатие виртуальной кнопки.

На каждом кадре отображается скелет руки с аннотациями MediaPipe, стрелка от основания до кончика указательного пальца, текстовое обозначение

текущего направления и статусная строка с последним переданным направлением. Окно с видео размещается в правом нижнем углу экрана, чтобы не мешать игровому процессу. Это достигается с помощью Tkinter, который определяет размеры экрана, и функции cv2.moveWindow.

Браузерная игра (написанная на JavaScript) подключается к WebSocket-серверу по адресу ws://localhost:8765. Получая JSON-сообщение, игра интерпретирует его как нажатие виртуальной кнопки, что позволяет управлять персонажем или объектами. Такой подход универсален и не требует модификации игрового движка, достаточно реализовать обработку входящих сообщений на стороне клиента.

Разработанная система была протестирована в помещении с различным освещением. Камера Logitech C920 работала на разрешении 640×480. Тестирование показало, что детекция руки происходит стабильно при расстоянии от 0,5 до 1,5 метров, а определение направления указательного пальца имеет задержку не более 100 мс.

На рисунках 2 и 3 представлены скриншоты игрового процесса и фотографии пользователя во время управления.

Рис.2 Скриншот игры

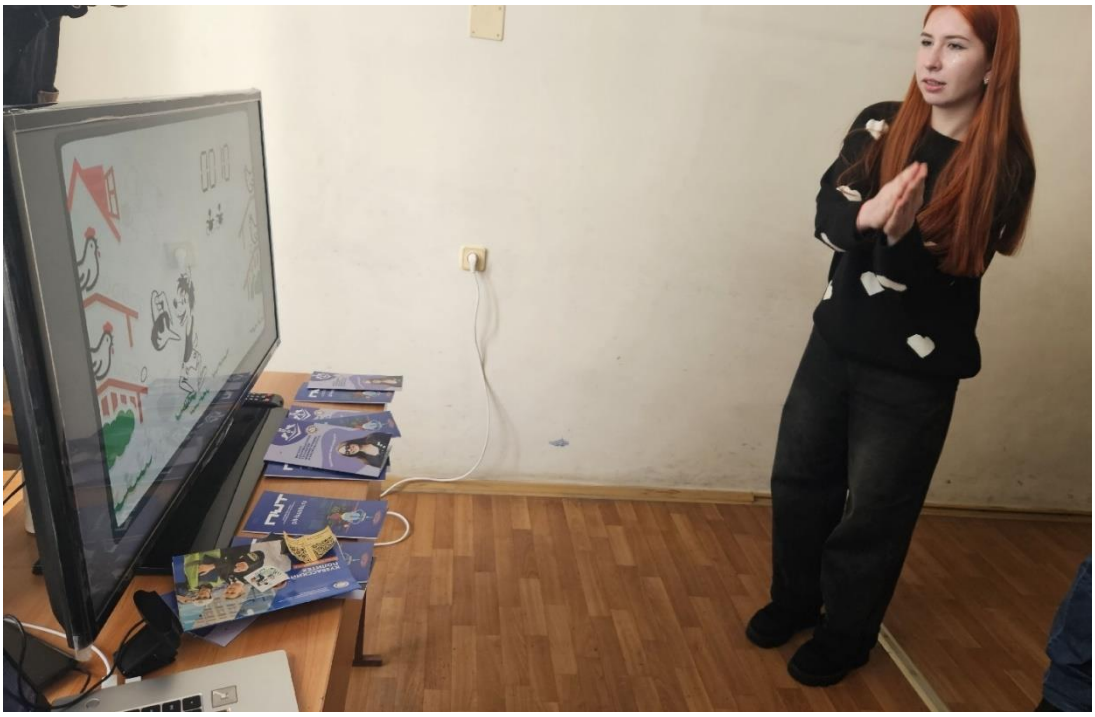

Рис.3 Игровой процесс

В работе представлена система управления игрой с помощью жестов рук, реализованная на Python с использованием современных библиотек компьютерного зрения. В качестве следующего этапа развития проекта планируется оптимизация производительности за счёт вынесения модели MediaPipe на серверную сторону, что позволит снизить вычислительную нагрузку на клиентское устройство. Кроме того, будет продолжена работа над повышением точности распознавания: предполагается использовать дополнительные методы фильтрации данных, а также экспериментировать с более сложными архитектурами нейронных сетей для устойчивой работы в различных условиях освещения и при частичном перекрытии руки.

Список литературы:

1. Google AI Edge. Hand landmarks detection guide [Электронный ресурс]. – Режим доступа: https://ai.google.dev/edge/mediapipe/solutions/vision/hand_landmarker (Дата обращения: 12.02.2026).
2. Python. Asyncio — Asynchronous I/O [Электронный ресурс] – Режим доступа: https://docs.python.org/3/library/asyncio.html (Дата обращения: 30.02.2026).