## Текст выступления

### Цель и задачи

Целью данной работы является разработка программной системы дополненной реальности с многопользовательским режимом. Главным отличием этой системы является то, что она работает с несколькими пользователями одновременно. В процессе разработки нам предстоит рассмотреть и решить следующие задачи:

- Проанализировать существующие системы дополненной реальности;
- Разработать собственный программный комплекс дополненной реальности;
- Выполнить тестирование и отладку разрабатываемой системы.

### Имеющиеся решения

В настоящее время известны несколько решений для программирования систем дополненной реальности. К ним относятся:

*ARCore*

Использует инерциальный измерительный модуль и характерные точки окружающего пространства, чтобы определить положение и ориентацию устройства в зависимости от его движения.

*IMU - инерциальный измерительный модуль или гиростабилизатор. Гиростабилизированная платформа обычно состоит из 1, 2 или 3 гироскопов, электронной системы обработки сигналов. Также может содержать акселерометры и другие датчики.*

ARCore обнаруживает горизонтальные поверхности, используя функции схожие с отслеживанием движения.

Определяет световое окружение устройства, тем самым улучшая внешний вид и делая изображение в реальном времени более точным, также обнаруживает пересечение объектов с лучами света.

*ARKit*

Способен точно отслеживать положение устройства в реальном мире. Используя визуальный инерционный одометр, ARKit объединяет данные камеры слежения и датчика движения, с помощью которых записывается положение устройства в реальном времени.

ARKit способен отследить поверхности в окружающей среде, например, пол, стены, столы, потолок.

*Vuforia*

Обладает теми же возможностями, что и ARKit и ARCore, а также новыми функциями, например, поддержкой внешних камер. Главной особенностью является то, что Vuforia позволяет распознавать объекты по форме с использованием уже существующих 3D-моделей.

### Принцип работы

Перед тем как разрабатывать свою систему имеет смысл рассмотреть общий принцип работы таких систем. Их алгоритм работы обычно состоит из следующих этапов: идет получение информации об окружающем мире с камеры, затем компьютер при помощи определённых алгоритмов распознаёт объект или маркер (в зависимости от реализации) и накладывает поверх полученной картинки новую информацию, например, 3д модель объекта или текст. Затем осуществляется вывод информации пользователю.

### Решение поставленной задачи

Для разработки собственной программной системы дополненной реальности использовались: среда программирования PyCharm, язык Python версии 3.7, для захвата камеры и

распознавания маркера использовалась библиотека OpenCV, маркеры, используемые в программе созданны на основе Кода Хэмминга.

**Архитектура системы**

В процессе проектирования системы было принято решение применить клиент-серверную архитектуру. Как видно на слайде система состоит из нескольких ключевых компонентов: сервера и клиента. В результате реализации клиент-серверной архитектуры были решены следующие проблемы:

- Необходимость централизованной обработки и хранения данных. Это позволило избавить клиентское устройство от некоторой нагрузки;

- Проблема синхронизации информации между несколькими клиентами.

**Архитектура клиентского приложения**

Клиентское приложение реализовано с применением модульной архитектуры, в результате чего имеет большую гибкость и масштабируемость. Например, при невозможности функционирования какого-либо модуля, вся система не потеряет работоспособность. Так, если откажет сетевой модуль, в результате отсутствия Интернет-соединения, то приложение продолжит свою работу, но перестанет получать данные (т.е. маркер будет распознан, но соответствующий ему объект получен не будет).

Кратко рассмотрим предназначение наиболее важных модулей:

- модуль сканирования, его основная задача – распознавание маркера и наложение дополнительной информации на кадр видеопотока, собственно в нём и реализован основной алгоритм системы;

- модуль камеры, главное предназначение этого модуля в получении видеопотока с камеры;

- модуль событий, по нажатию левой кнопки мыши сверяет позицию курсора с позицией маркера и при соблюдении этого условия открывает диалоговое окно для ввода сообщения.

**Код Хэмминга**

Стоит пояснить, что из себя представляет код Хэмминга. Итак, Код Хэ́мминга это алгоритм, который позволяет закодировать какое-либо информационное сообщение определённым образом и после передачи (например, по сети) определить появилась ли какая-то ошибка в этом сообщении (к примеру, из-за помех) и, при возможности, восстановить это сообщение. Из всего поля размером 7x7 для сообщения используется поле внутри черного контура размером 5x5.

Рассмотрим, что входит в 25 бит информации, содержащейся в маркере (цветная зона):

- 4 бита используются для определения ориентации (отмечены зелёным);

- 9 бит используются для контроля и коррекции ошибок (отмечены красным);

- остальные 12 бит — это сообщение (отмечены жёлтым).

при кодировке биты ориентации не используются, следовательно, закодированное сообщение это 01111001111111101001 (жёлтые + красные), а декодированное 110011110001 в десятичной 3313, это есть идентификатор маркера.

**Алгоритм обнаружения маркера**

1. Применяется порог для получения границ (рис. 1).

2. Затем происходит нахождение контуров. После этого обнаруживаются не только настоящие маркеры, но и множество ненужных нам границ. Остальная часть алгоритма состоит в том, чтобы отфильтровать нежелательные границы. Удаляются границы с небольшим количеством точек (рис. 2).

3. Происходит полигональная аппроксимация контура с сохранением замкнутых контуров с четырьмя углами (т.е. прямоугольников) (рис. 3).

4. Убираются слишком маленькие прямоугольники. На этом этапе сохраняется внешняя граница (рис. 4).

5. Происходит идентификация маркера. Если это маркер, то он содержит в себе код. Маркер поделён сеткой 7×7, из которой 5×5 это ячейки, в которых содержится информация об идентификаторе. Остальное относится к внешней чёрной границе. После считывания информации происходит её декодирование.

**Тестирование**

Всё что требуется, так это убедиться в том, что маркер распознаётся (т.е. он должен выделиться), появляется окно ввода сообщения, сообщение приходит на сервер, сообщение отображается поверх маркера (т.е. приходит информация об ассоциациях с конкретным идентификатором).

Тут мы можем видеть, что маркеры распознаются успешно, пробуем отправить сообщение.

Диалоговое окно открывается, значит события по нажатию работают.

(следующий слайд)

В итоге после отправки сообщений, можем увидеть, что они успешно были отправлены и приняты от сервера для соответствующего маркера, так как текст отображается поверх маркеров.

**Заключение**

Подводя итоги можно сказать, что в результате выполнения выпускной квалификационной работы была спроектирована и реализована программная система дополненной реальности с многопользовательским режимом, которая включает в себя два компонента: клиентское приложение для устройств на базе Windows 10 и серверное приложение для компьютеров с установленным интерпретатором Python.

Созданный программный продукт отвечает всем поставленным требованиям и решает все необходимые задачи. Тестирование данного продукта подтвердило его работоспособность и возможность дальнейшего использования для решения подобных задач.