МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»
Кафедра дискретной математики и информационных технологий

РАЗРАБОТКА ПЛАГИНА ДЕМОНСТРАЦИИ ЭКРАНА С ПРИМЕНЕНИЕМ АППАРАТНОГО УСКОРЕНИЯ ДЛЯ TEAMSPEAK

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 4 курса 421 группы
направления 09.03.01 — Информатика и вычислительная техника
факультета КНиИТ
Ширяева Дмитрия Михайловича

Научный руководитель
доцент _____ А. Д. Панфёров

Заведующий кафедрой
доцент, к. ф.-м. н. _____ Л. Б. Тяпаев

Саратов 2025

ВВЕДЕНИЕ

В современных условиях удалённого взаимодействия передача видеопотока в реальном времени становится всё более востребованной задачей, особенно в контексте трансляции игрового процесса или проведении совместной деятельности, как учебной, так и профессиональной. Наибольшую популярность среди программных решений в данной области получили Discord и Guilded за счёт своего пользовательского опыта, но обладают рядом недостатков.

Во-первых, многие из этих решений построены на фреймворке Electron, что приводит к избыточному потреблению ресурсов, снижая производительность на маломощных системах. Во-вторых, они являются централизованными и зачастую принадлежат иностранным компаниям, что ограничивает их использование. В-третьих, реализация кодирования видеопотока в большинстве подобных приложений построена на базе FFmpeg — библиотеки, которая, несмотря на свои широкие возможности, не всегда обеспечивает эффективную работу на всех видеокартах. Например, при использовании AMD Radeon RX580 аппаратное кодирование в формате, применяемом Windows, невозможно без дополнительной конвертации кадров.

В этих условиях актуальным становится создание автономного решения с поддержкой аппаратного ускорения и возможностью автономного (self-hosted) развёртывания.

Целью работы является создание плагина (расширения) для программного продукта TeamSpeak 3, реализующего функциональность демонстрации экрана посредством использования средств аппаратного ускорения сжатия видео компаний Nvidia и AMD.

Для достижения поставленной цели были сформулированы следующие задачи:

1. сравнить наиболее подходящие альтернативные платформы;
2. изучить существующие алгоритмы кодирования и декодирования видео и их реализации с использованием аппаратного ускорения на актуальных архитектурах видео-ускорителей (видеокарт) компаний AMD и Nvidia;
3. изучить особенности передачи видеопотока в реальном времени;
4. разработать и испытать программную реализацию клиент-серверного

приложения, осуществляющего передачу изображения экрана одного клиента другим в реальном времени.

В работе 4 главы:

1. Целевой функционал и существующие решения;
2. Кодирование и передача видео;
3. Использованный инструментарий и программная реализация;
4. Демонстрация и тестирование разработанного решения.

КРАТКОЕ СОДЕРЖАНИЕ РАБОТЫ

В первом разделе проведён сравнительный анализ популярных решений для голосовой связи и совместного взаимодействия: Discord, TeamSpeak, Mumble, Element, Revolt, Guilded и Tox. Рассматривались ключевые параметры: производительность (нагрузка на центральный процессор и потребление оперативной памяти), возможность автономного размещения серверов, наличие поддержки демонстрации экрана, а также расширяемость функциональности за счёт открытых API или SDK. По результатам был выбран TeamSpeak как наиболее подходящей платформы для разработки расширения. TeamSpeak отличается низким уровнем потребления системных ресурсов, наличием возможности развёртывания собственного сервера и открытым интерфейсом расширения клиентской части.

Также обозначена целевая аудитория разрабатываемого решения — игровые, технические и образовательные сообщества и группы пользователей, использующие голосовые платформы для неформального общения и взаимодействия в реальном времени. Отмечено, что основная причина широкого распространения ряда решений, таких как Discord, заключается в сочетании функциональности с интуитивно понятным пользовательским опытом (UX). Пользователь может свободно перемещаться между каналами и серверами, без необходимости участия в формализованных сессиях, в отличие от решений, ориентированных на проведение деловых или учебных конференций и встреч (например, Zoom, Телемост, BigBlueButton).

Во втором разделе подробно рассмотрены современные подходы к кодированию потокового видео и передаче сжатого потока в условиях реального времени. Раздел состоит из двух подразделов.

В первом подразделе даны определения кодека и кодирования видео, рассмотрены кодеки MPEG2, H.264 и H.265 (HEVC). Особое внимание уделено сравнению различных алгоритмов сжатия. Сравнительный анализ проведён с использованием двух методов оценки: численного и визуального. Тестовые видео-фрагменты генерировались инструментами FFmpeg, далее численное сравнение проводилось на основе полученного битрейта, а визуальное сравнение проводилось при помощи предложенного в работе графического фильтра, предназначенного для выделения артефактов сжатия изображения. Фильтр позволяет локализовать участки, подверженные искажениям,

и оценить отклонение формы и цвета от эталонного (несжатого) изображения. По итогам анализа в качестве основного кодека был выбран HEVC как обеспечивающий наилучшее соотношение качества и степени сжатия. Также в подразделе приведены два варианта постобработки, направленные на повышение читаемости текста после прохождения через этап сжатия.

Дополнительно кратко изложены принципы работы кодека HEVC, включая механизм разбиения изображения на блоки, пространственного и временного предсказания и трансформации кадров. Приведено различие между аппаратными решениями на базе специализированных блоков и реализациями, использующими универсальные графические процессоры с возможностью применения вычислительных шейдеров. В качестве примеров архитектур графических чипов, где впервые были введены отдельные схемотехнические решения для аппаратного ускорения сжатия видео, указаны Tahiti для AMD и Kepler для Nvidia, с которых начинается аппаратная поддержка кодека H.264, а так же актуальные архитектуры графических ускорителей потребительского сегмента — Ampere для Nvidia и Vega для AMD, уже включающие в себя поддержку аппаратного ускорения работы кодека HEVC.

Во втором подразделе уделено внимание сетевым аспектах передачи закодированного видеопотока. Рассмотрены особенности и ограничения протоколов Ethernet, IP, TCP и UDP применительно к задаче потоковой передачи видео. Для реализации был выбран более лёгкий и менее требовательный к задержкам протокол UDP. Для обеспечения визуальной целостности передаваемого видеопотока на принимающей стороне, в качестве транспортного контейнера был выбран формат MPEG-TS, обеспечивающий возможность синхронизации и восстановления структуры потока при частичной потере пакетов. Такой подход позволяет компенсировать недостаточную надёжность UDP при передаче фрагментированных кадров кодека.

Также в подразделе обосновано введение верхней границы битрейта, которую разрабатываемой реализации не следует превышать. Это позволяет учитывать ограничения на пропускную способность, характерные для бытовых интернет-соединений, а также оставлять достаточный запас пропускной способности для других параллельно выполняемых сетевых задач пользователя. Подобное ограничение делает реализацию более пригодной для практического применения в разнообразных условиях эксплуатации.

В третьем разделе обоснован выбор инструментария, использованного при реализации решения, а также описаны технические детали реализации.

Одной из значимых технических проблем при реализации программного решения стало обеспечение совместной работы с аппаратными кодеками двух крупных производителей графических процессоров — Nvidia и AMD, предлагающих собственные и несовместимые между собой программные интерфейсы для аппаратного кодирования видео. Каждая из платформ использует собственные механизмы инициализации кодеков и соглашения при работе в многопоточном режиме. Дополнительно, из-за особенностей библиотеки NvEnc (Nvida Encoder), работа приложения, использующего её напрямую, была бы невозможна в системах без оборудования Nvidia. В рамках работы был реализован унифицированный интерфейс, позволивший единым набором методов взаимодействовать с двумя различными API.

Для формирования, передачи и декодирования видеопотока в формате контейнера MPEG-TS была использована библиотека FFmpeg. Дополнительно она используется при декодировании видеопотока для сохранения работоспособности функции просмотра на несовместимом оборудовании.

Вся реализация выполнена на языке C++. Для плагина и серверного модуля использовалась библиотека Qt. В работе рассмотрены и возможные альтернативы альтернативны, которые были бы применимы для разработки серверного модуля, однако Qt был выбран как обеспечивающий оптимальное сочетание функциональности и удобства разработки. Подробно описана как общая архитектура программного решения, так и структура взаимодействия между основными компонентами, включая модули кодирования, модуль DXGI (графическая инфраструктура DirectX), серверный модуль и непосредственно плагин, реализующий взаимодействие пользователя и других компонентов между собой.

Для получения информации о видеокарте, формирования и отправки управляющих ею команд используются возможности DirectX и Windows Runtime Library. Захват изображения рабочего стола реализован с использованием механизма DXGI Desktop Duplication. Подробно описаны процессы инициализации и управления сессиями кодирования как в AMF (AMD Media Foundation), так и в NvEnc. Реализованные абстракции учитывают особенно-

сти каждого API и позволяют динамически выбирать реализацию в зависимости от доступного оборудования.

Важной составной частью реализации стало обеспечение взаимодействия с пользователем непосредственно через графический интерфейс клиента TeamSpeak. Поскольку официальные средства для расширения интерфейса клиента отсутствуют, а исходный код недоступен, интеграция собственных элементов управления потребовала нестандартного подхода, основанного на анализе структуры и поведения интерфейса программы. Клиент TeamSpeak представляет собой приложение, построенное с использованием фреймворка Qt 5. За счёт этого через общую память, используемую библиотекой, было возможно изучить часть структуры Qt-объектов, используемых программой, некоторые из которых могли отвечать как за логику работы приложения, так и за графический интерфейс. Результатом данного этапа стало внедрение необходимых пользовательских элементов управления непосредственно в уже существующее окно клиента TeamSpeak. Это обеспечило более интуитивно понятный пользовательский опыт, аналогично другим популярным решениям.

Кроме того, описан и реализован протокол аутентификации пользователей, реализованный в серверной части. Это было необходимо так как TeamSpeak SDK не предоставляет плагинам возможность проверить подлинность пользователя вне клиентского плагина. Для обеспечения базовой безопасности применяется схема удостоверения с открытой передачей ключа при подключении клиента к каналу запросов и упрощённая схема аутентификации при обращении к каналу данных за счёт уже подтверждённого канала запросов.

В завершении главы приведена структура файлов разработанного решения. Для каждого файла указано краткое описание его зоны ответственности и реализуемый функционал содержащимися в них классами и функциями.

В четвёртом разделе представлена демонстрация разработанного программного решения. Описана последовательность действий пользователя при запуске трансляции и подключении к просмотру демонстрации экрана другого участника. Описаны элементы интерфейса, взаимодействие с которыми требуется для выполнения соответствующих действий. Представлены резуль-

таты тестирования, выполненного на различных аппаратных конфигурациях, включая перечень использованных видеокарт и процессоров. В том числе проведено тестирование функции просмотра трансляции на несовместимом оборудовании с использованием встроенной графики Intel. Представлены снимки экранов, на которых зафиксирован внешний вид пользовательского интерфейса во время работы плагина и использование пропускной способности основного сетевого адаптера в диспетчере задач Windows во время просмотра трансляции.

ЗАКЛЮЧЕНИЕ

В результате выполнения работы была достигнута поставленная цель — разработан плагин для TeamSpeak 3, реализующий функциональность демонстрации экрана с использованием аппаратного кодирования видеопотока средствами графических ускорителей Nvidia и AMD.

Проведён анализ существующих платформ голосовой связи и трансляции экрана, рассмотрены технологии кодирования видео и особенности использования аппаратного ускорения. Для оценки визуального качества сжатого изображения предложен метод визуализации артефактов, позволяющий более наглядно выявлять искажения при кодировании.

Разработано программное расширение, реализующее клиент-серверную модель передачи изображения рабочего стола в реальном времени. Решение протестировано, интегрировано в интерфейс TeamSpeak 3 и продемонстрировало близкое к теоретическим ожиданиям потребление системных ресурсов.

Разработанное решение имеет потенциал для дальнейшего развития и может служить как основой для дальнейшего развития текущего плагина — например, путём расширения поддержки графических ускорителей, улучшения интерфейса и т. д., — так и основой для создания самостоятельного продукта с более широким функционалом.

Основные источники информации:

1. HWAccelIntro – FFmpeg [Электронный ресурс] URL: https://trac.ffmpeg.org/wiki/HWAccelIntro (дата обращения: 29.12.2024).
2. Desktop Duplication API - Win32 apps | Microsoft Learn [Электронный ресурс] URL: https://learn.microsoft.com/en-us/windows/win32/direct3ddxgi/desktop-dup-api (дата обращения: 20.11.2024).
3. Guide for Video CODEC Encoder App Developers · GPUOpen-LibrariesAnd SDKs/AMF Wiki [Электронный ресурс] URL: https://github.com/GPUOpen-LibrariesAndSDKs/AMF/wiki/Guide-for-Video-CODEC-Encoder-App-Developers (дата обращения: 18.11.2024).
4. NVENC Video Encoder API Programming Guide - NVIDIA Docs [Электронный ресурс] URL: https://docs.nvidia.com/video-technologies/video-codec-sdk/13.0/nvenc-video-encoder-api-prog-guide/index.html (дата обращения: 18.11.2024).
5. TeamSpeak Systems GmbH. TeamSpeak SDK's documentation (Client &

Server Integration TeamSpeak 3 SDK 3.3.1) [Электронный ресурс] URL: https://teamspeak.com/en/downloads/#sdk (дата обращения: 11.12.2024).