МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский Нижегородский государственный университет им. Н.И. Лобачевского»

Физический факультет

Кафедра информационных технологий в физических исследованиях

Масштабируемая система для обработки видео на базе ZeroMQ

Описание программы

Нижний Новгород
2026

Оглавление

1. Аннотация 4
2. Общие сведения 5
  2.1. Обозначение и наименование программы 5
  2.2. Необходимое программное обеспечение 5
  2.3. Языки программирования 5
3. Функциональное назначение 6
  3.1. Функциональные характеристики 6
    3.1.1. Характеристики системы захвата кадров 6
    3.1.2. Характеристики системы масштабирования нагрузки 6
    3.1.3. Характеристики системы обработки изображений 6
    3.1.4. Характеристики системы сборки результатов 7
  3.2. Ограничивающие характеристики 7
    3.2.1. Ограничения производительности 7
    3.2.2. Ограничения системы сборки результатов 7
4. Описание логической структуры 8
  4.1. Алгоритм программы 8
    4.1.1. Программа Capturer 8
    4.1.2. Программа Worker 9
    4.1.3. Программа Composer 10
  4.2. Используемые методы 11
    4.2.1. Программа Capturer 11
    4.2.2. Программа Worker 11
    4.2.3. Программа Composer 12
  4.3. Структура системы 12
  4.4. Связи программы с другими программами 13
5. Используемые технические средства 14
  5.1. Аппаратные средства 14
  5.2. Среда разработки и сборки 14
6. Вызов и загрузка 15
  6.1. Способ вызова программы 15

6.2. Входные точки в программу 15
6.3. Порядок остановки 16
7. Входные данные 17
7.1. Характер, организация и формат входных данных 17
7.2. Описание входных данных 17
8. Выходные данные 21
8.1. Характер, организация и формат выходных данных 21
8.2. Описание выходных данных 21
9. Приложения 22
Приложение А. Файл формата сообщений video_processing.proto 22
10. Источники 23

1. Аннотация

Документ предназначен для разработчиков распределенных систем, студентов и преподавателей, изучающих системную инженерию, а также для специалистов, осуществляющих эксплуатацию систем обработки данных в реальном времени.

Программа разработана в рамках образовательного задания по дисциплине «Системная инженерия» для изучения принципов масштабируемой обработки данных в многокомпьютерных распределенных средах. Целью разработки является создание масштабируемого программного комплекса для потоковой обработки изображений, поступающих с веб-камеры, на платформе ZeroMQ [1].

Актуальность работы обусловлена необходимостью эффективной балансировки нагрузки в условиях разнородных вычислительных ресурсов при обработке большого объема видеоданных. Система состоит из трех основных компонентов: Capturer, Worker и Composer, взаимодействующих через очереди сообщений ZeroMQ и использующих для сериализации данных Protocol Buffers [2].

Разработанный комплекс служит учебной моделью, демонстрирующей принципы проектирования распределенных систем на основе очередей сообщений между вычислительными узлами.

Документ разработан согласно ГОСТ 19.402-78, структура и оформление документа соответствуют ГОСТ 19.105-78.

2. Общие сведения

2.1. Обозначение и наименование программы:

- **Обозначение:** ZeroMQCameraSystem [3];
- **Наименование:** Масштабируемая система обработки видеопотока на базе ZeroMQ.

2.2. Необходимое программное обеспечение:

- **ZeroMQ (libzmq)** версия 4.3.4 – высокопроизводительная библиотека асинхронного обмена сообщениями, обеспечивающая межпроцессное взаимодействие компонентов системы;
- **OpenCV** версия 4.12.0 – библиотека компьютерного зрения, используемая для захвата потока кадров, обработки изображений и кодирования/декодирования графических данных [4];
- **Protocol Buffers** версия 3.7.1 – кроссплатформенная система сериализации структурированных данных, применяемая для формализации сообщений между компонентами.

2.3. Языки программирования:

Основные языки:

- **C++** (стандарт C++17) – основной язык реализации всех трёх компонентов системы (Capturer, Worker, Composer);
- **C** – используется в подключаемых библиотеках (ZeroMQ, OpenCV, libsodium).

Скриптовые языки:

- **PowerShell** – для скриптов автоматического запуска компонентов системы.

Языки описания данных:

- **Protocol Buffers (.proto)** – язык описания интерфейсов для сериализации структурированных данных. Определяет формат сообщений, передаваемых между компонентами.

3. Функциональное назначение

3.1. Функциональные характеристики:

3.1.1. Характеристики системы захвата кадров

- **Обнаружение и инициализация камер:** система обнаруживает и подключается к указанной в конфигурационном файле видеокамере;
- **Захват с настраиваемыми параметрами:** система поддерживает настройку разрешения, частоты кадров и качества сжатия потока кадров.

3.1.2. Характеристики системы масштабирования нагрузки

- **Подключение по заданным адресам:** система требует явного указания сетевых адресов для подключения вычислительных узлов;
- **Управление очередью кадров:** система поддерживает кольцевую очередь фиксированного размера с автоматическим удалением самых старых кадров при достижении максимальной емкости;
- **Масштабирование производительности:** добавление новых вычислительных узлов уменьшает заполнение очереди и снижает количество пропускаемых кадров.

3.1.3. Характеристики системы обработки изображений

- **Применение визуальных эффектов:** система применяет эффект "Scanner Darkly", включающий:
    - Квантование цвета с уменьшением градаций до указанного количества уровней на канал RGB;
    - Детекцию границ объектов с настраиваемыми порогами чувствительности;
    - Комбинирование квантованного изображения с выделенными контурами;
- **Параллельная обработка:** система поддерживает произвольное количество одновременно работающих вычислительных узлов.

3.1.4. Характеристики системы сборки результатов

- **Восстановление последовательности:** система восстанавливает исходный порядок кадров на основе их порядковых номеров;
- **Компенсация пропусков:** система автоматически вставляет черные кадры с пометками для пропущенных позиций;
- **Запись видео:** программа создает два видеофайла (исходный и обработанный) из полученных кадров;
- **Контроль целостности:** система предоставляет статистику работы и потерь кадров.

3.2. Ограничивающие характеристики:

3.2.1. Ограничения производительности

- **Ограничение размера очереди:** система поддерживает максимальный размер очереди, определяемый параметром, указанным в конфигурационном файле;
- **Зависимость от пропускной способности сети:** производительность системы ограничена пропускной способностью сети.

3.2.2. Ограничения системы сборки результатов

- **Формат выходных файлов:** программа сохраняет видео исключительно в формате AVI с кодеком MJPG;
- **Ограничение выбора пути сохранения:** отсутствие возможности выбора пути сохранения видеофайлов в системе.

4. Описание логической структуры

4.1. Алгоритм программы:

4.1.1. Программа Capturer: блок-схема алгоритма работы программы Capturer представлена на рисунке 1.

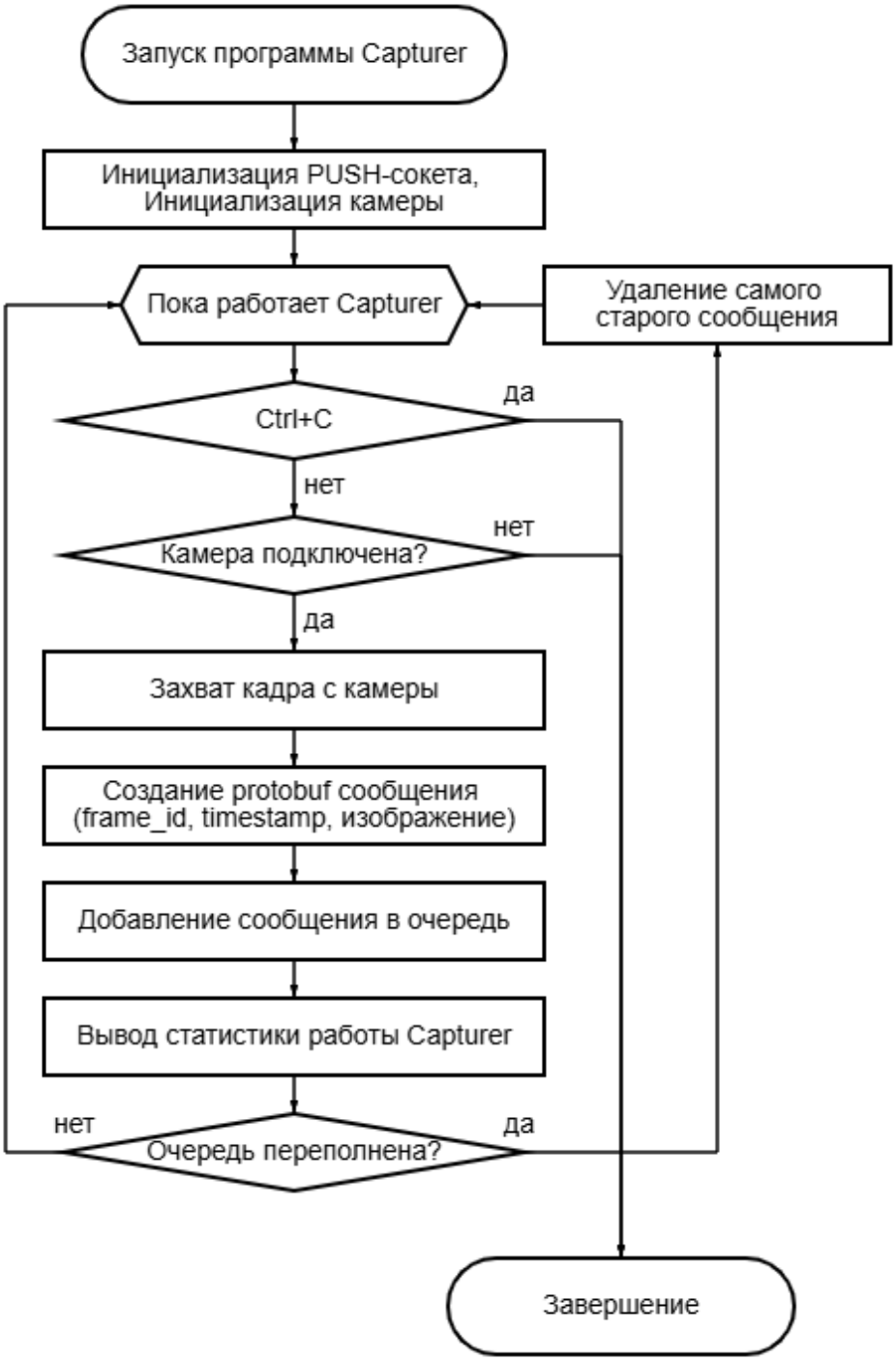

Рисунок 1. Блок-схема алгоритма работы программы Capturer

4.1.2. Программа Worker: блок-схема алгоритма работы программы Worker представлена на рисунке 2.

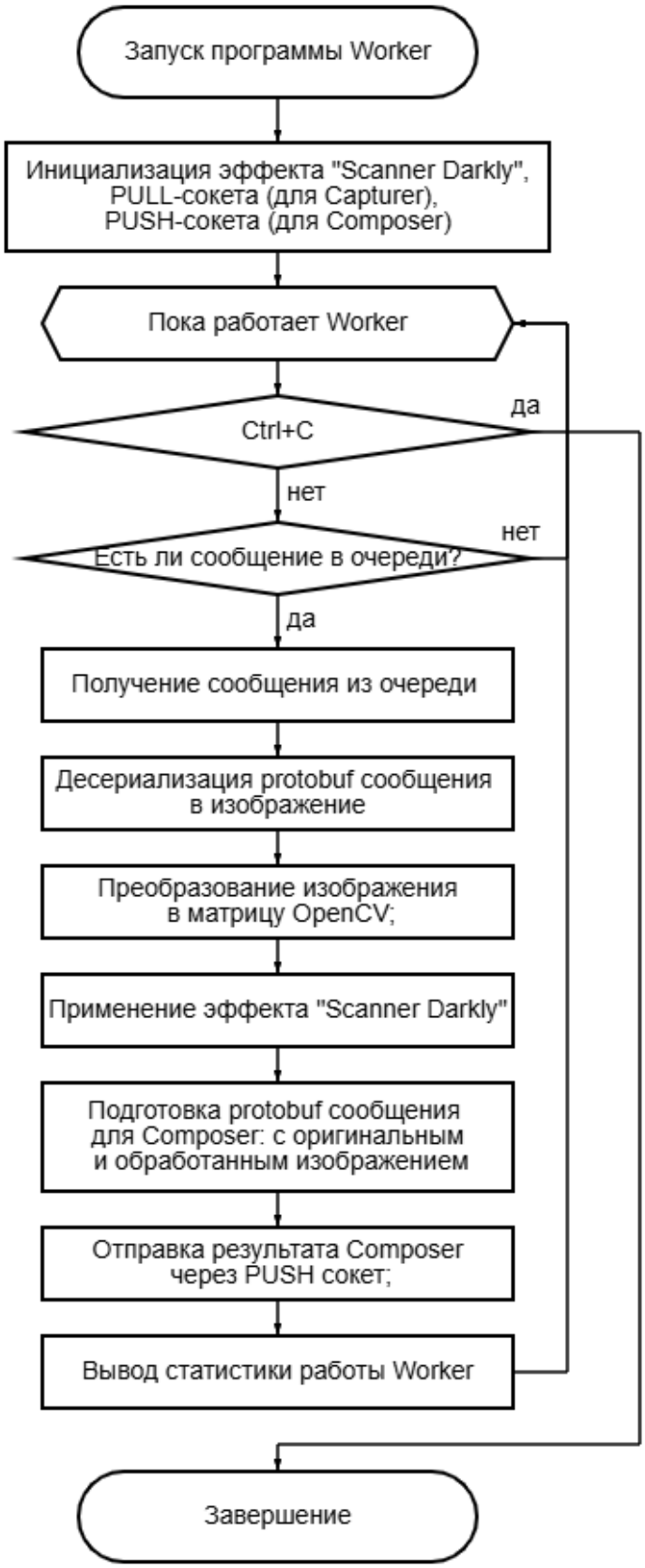

Рисунок 2. Блок-схема алгоритма работы программы Worker

4.1.3. Программа Composer: блок-схема алгоритма работы программы Composer представлена на рисунке 3.

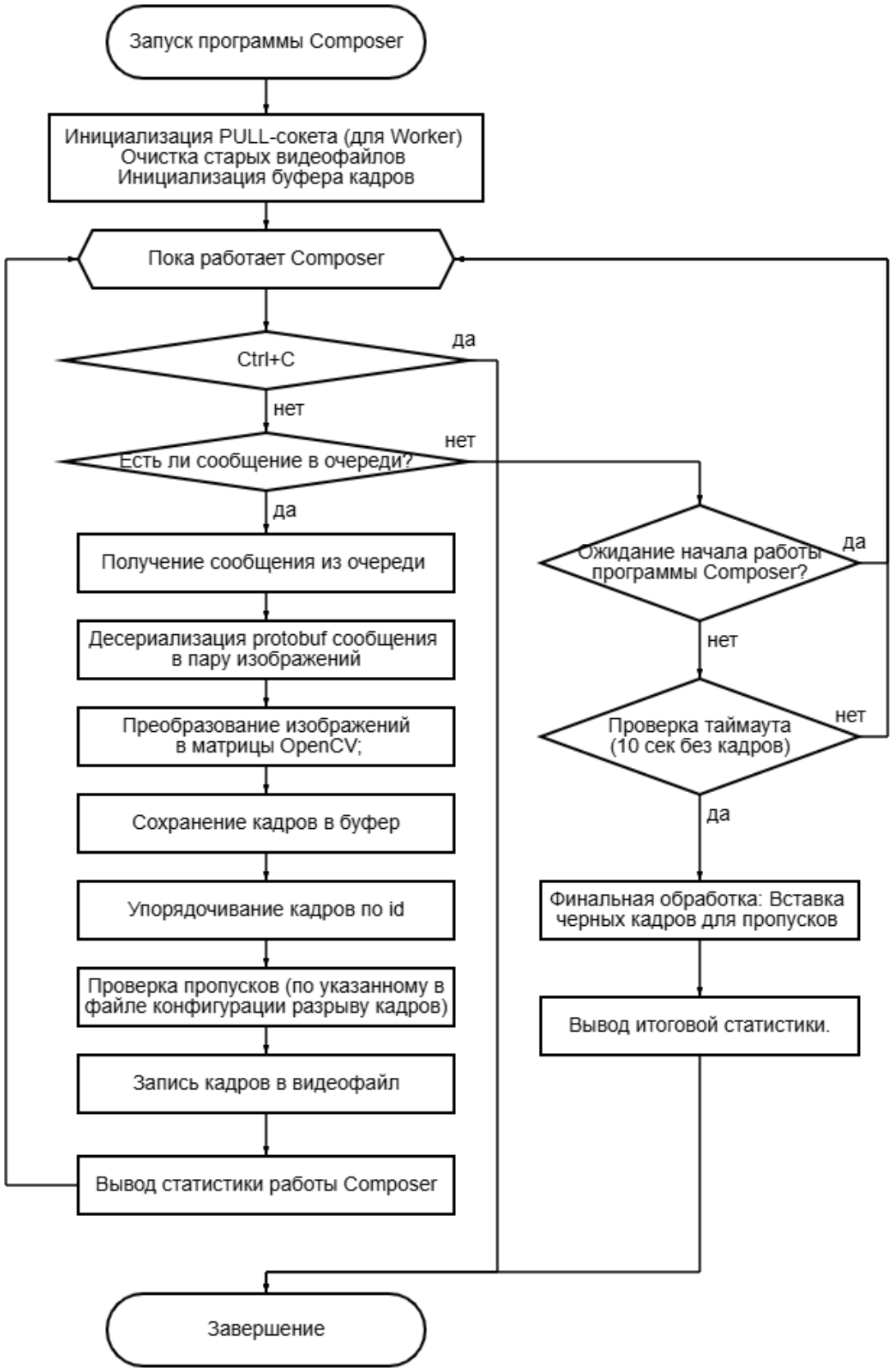

Рисунок 3. Блок-схема алгоритма работы программы Composer

4.2. Используемые методы:

4.2.1. Программа Capturer:

1. **init_camera()** — инициализация устройства захвата видео, выполняет установку параметров разрешения, FPS и формата;
2. **create_video_frame()** — преобразование кадра в protobuf-сообщение с добавлением данных (см. Приложение А);
3. **get_current_time()** — получение текущего системного времени для временных меток кадров;
4. **distribute_frames()** — помещение кадров в очередь для отправки Worker'ам с использованием паттерна PULL-PUSH;
5. **manage_queue_size()** — управление размером очереди кадров, удаление старых кадров при переполнении;
6. **run()** — основной цикл работы программы, реализующий полный цикл: захват → сериализация → отправка.

4.2.2. Программа Worker:

7. **extract_image()** — декодирование protobuf-сообщения в матрицу OpenCV;
8. **create_image_data()** — кодирование матрицы OpenCV в protobuf-сообщение;
9. **applyEffect()** — основной метод применения эффекта "Scanner Darkly" к изображению;
10. **request_frame()** — метод взятия кадра из очереди.
11. **send_to_composer()** — помещение кадров в очередь для отправки Composer с использованием паттерна PULL-PUSH;
12. **run()** — основной цикл работы программы, реализующий полный цикл: запрос → десериализация → обработка → сериализация → отправка.

Методы эффекта "Scanner Darkly":

13. **colorQuantization()** — квантование цветов изображения с использованием k-means кластеризации;
14. **extractEdges()** — выделение контуров изображения с помощью детектора Кэнни;
15. **combineEffect()** — комбинирование квантованного изображения с контурами.

4.2.3. Программа Composer:

16. **initialize_video_writers()** — инициализация объектов записи видеофайлов на основе параметров первого полученного кадра;
17. **insert_black_frame()** — создание и вставка черного кадра с текстовой меткой на место пропущенных кадров;
18. **write_available_frames()** — запись кадров из буфера в видеофайлы в правильном порядке;
19. **process_frame_for_video()** — обработка кадра: декодирование, буферизация и подготовка к записи;
20. **final_processing()** — финальная обработка всех пропусков при завершении работы;
21. **cleanup_old_video_files()** — удаление старых видеофайлов перед началом новой записи;
22. **file_exists()** — проверка существования файла на диске;
23. **should_stop()** — проверка условий завершения работы;
24. **show_statistics()** — вывод статистики работы в реальном времени;
25. **run()** — основной цикл работы, управляющий приемом, буферизацией и записью кадров.

4.3. Структура системы

Система состоит из трех независимых программ, взаимодействующих через ZeroMQ по архитектуре PUSH-PULL. Каждый компонент выполняет строго определенную роль в конвейере обработки видео. Общая схема взаимодействия представлена на рисунках 4-5.

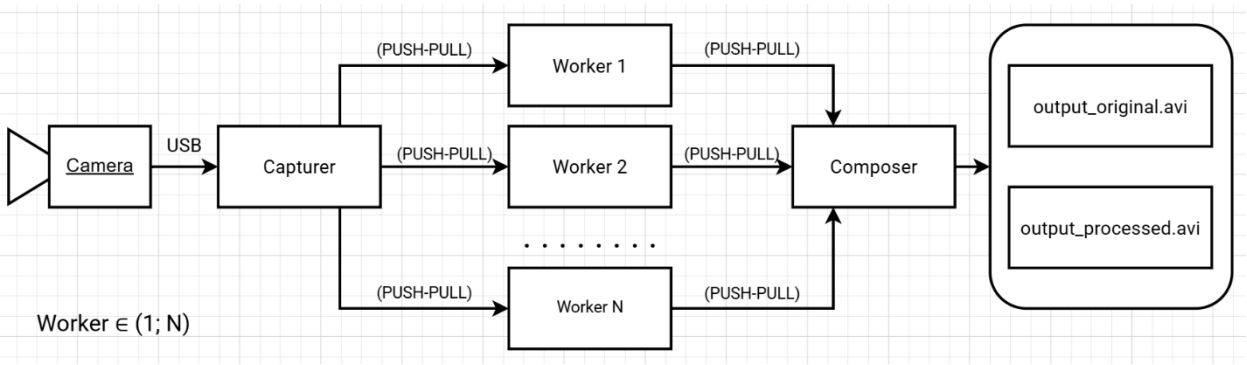

Рисунок 4. Схема взаимодействия компонентов системы (стрелками указано направление передачи данных между компонентами)

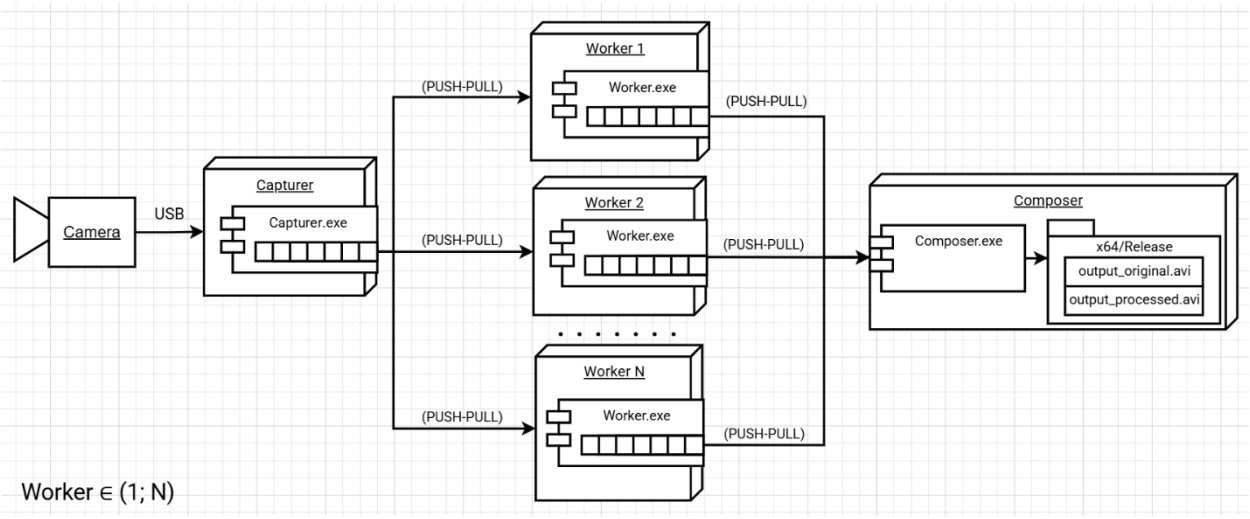

Рисунок 5. Диаграмма размещения (развёртывания) взаимодействия компонентов системы (стрелками указано направление передачи данных между компонентами)

4.4. Связи программы с другими программами

Взаимодействие с другими программами не предусмотрено.

Система не требует интеграции со сторонним программным обеспечением в процессе выполнения. Все необходимые библиотеки являются встроенными зависимостями и не выступают в роли внешних программ.

5. Используемые технические средства

Система разработана для работы в среде **Microsoft Windows** и использует следующие аппаратные средства и программное обеспечение:

5.1. Аппаратные средства

- **Процессор:** Intel(R) Xeon(R) CPU E5-2620 v3 @ 2.40GHz;
- **Оперативная память:** SK hynix HMA41GR7AFR4N-TF 8GB;
- **Свободное место на диске:** GIGABYTE GP-GSTFS31240GNTD 256Gb;
- **Видеокарта:** NVIDIA GeForce GTX 1650;
- **Сетевая карта:** Realtek PCIe GbE Family;
- **Устройство захвата видео:** USB веб-камера или встроенная камера.

5.2. Среда разработки и сборки

- **Операционная система:** Microsoft Windows 10/11 (64-bit);
- **Среда выполнения:** Microsoft Visual C++ Redistributable (версия, соответствующая используемому компилятору);
- **Система сборки:** MSBuild (входит в состав Visual Studio);
- **Средства разработки:**
- Microsoft Visual Studio 17 2022 с компонентами [5]:
    - "Разработка классических приложений на C++";
    - "Пакет SDK для Windows 10";
    - "MSVC версии 141 - средства сборки C++";
    - "MSVC версии 141 - библиотеки C++".

6. Вызов и загрузка

6.1. Способ вызова программы

Программа вызывается вручную путем последовательного запуска трех независимых исполняемых файлов в .\ZeroMQCameraSystem\x64\Release\ или .\ZeroMQCameraSystem\x64\Debug\ из командной строки или проводника Windows.

Последовательность вызова:

- **Запуск компонента Composer:** 3_Composer.exe
- **Запуск одного или нескольких компонентов Worker:** 2_Worker.exe
- **Запуск компонента Capturer:** 1_Capturer.exe

Дополнительно предоставляются скрипты PowerShell для вызова программ (в ZeroMQCameraSystem\x64\Release\ или ZeroMQCameraSystem\x64\Debug\) с записью работы в .txt файл (см. пп. 8.1-8.2):

- **Запуск компонента Composer с записью в .txt:** 3_Composer.ps1
- **Запуск компонента Worker с записью в .txt:** 2_Worker.ps1
- **Запуск компонента Capturer с записью в .txt:** 1_Capturer.ps1

6.2. Входные точки в программу

Каждый компонент системы представляет собой самостоятельное консольное приложение, которое запускается независимо от других компонентов. Все компоненты имеют единую архитектурную схему запуска.

Capturer:

- Исходный файл: Capturer.cpp
- Входная точка: функция int main()
    - Возвращаемое значение:
    - 0 — успешное завершение
    - -1 — ошибка инициализации или выполнения

Worker:

- Исходный файл: Worker.cpp
- Входная точка: функция int main()
    - Возвращаемое значение:
    - 0 — успешное завершение
    - -1 — ошибка подключения или выполнения

Composer:

- Исходный файл: Composer.cpp
- Входная точка: функция int main()
    - Возвращаемое значение:
    - 0 — успешное завершение
    - -1 — ошибка инициализации или выполнения

Параметры командной строки: Не поддерживаются. Все компоненты системы спроектированы для работы без параметров командной строки.

6.3. Порядок остановки

1. Нажать Ctrl+C в окне Capturer для остановки захвата
Capturer немедленно прекращает захват видеопотока с камеры, освобождает ресурсы OpenCV и корректно закрывает ZeroMQ сокеты.

2. Worker'ы автоматически завершатся при отсутствии кадров
После остановки Capturer'a, Worker'ы продолжают обработку оставшихся в их очередях кадров. По мере исчерпания очередей, Worker'ы автоматически завершают работу, корректно закрывая сетевые соединения с обоими компонентами системы.

3. Composer завершит запись видео и сохранит файлы
Composer ожидает поступления всех обработанных кадров от Worker'ов, после истечения 10-секундного тайм-аута на ожидание кадров выполняет финальную обработку: упорядочивает кадры, вставляет черные кадры на места пропусков, и сохраняет два итоговых видеофайла — output_original.avi (исходный), output_processed.avi (обработанный) — выводит статистику и завершает работу.

7. Входные данные

7.1. Характер, организация и формат входных данных

Система использует единый конфигурационный файл config.txt для всех трёх компонентов (Capturer, Worker, Composer), для обеспечения согласованности настроек между модулями.

Конфигурационные параметры устанавливаются перед запуском системы и не изменяются в течение всего времени её работы. Логически входные данные организованы в виде конфигурационного файла в формате ключ=значение. Пользователь должен:

1. Создать копию шаблонного конфигурационного файла (.\ZeroMQCameraSystem\ZeroMQCameraSystem\config.txt);
2. Отредактировать значения параметров в соответствии с требованиями конкретной инсталляции;
3. Сохранить файл config.txt в директории исполняемых файлов системы (в .\ZeroMQCameraSystem\x64\Release\ или .\ZeroMQCameraSystem\x64\Debug\).

Формат конфигурационного файла — текстовый файл в кодировке UTF-8 со следующей структурой:

1. **Комментарии** — строки, начинающиеся с символа **#;**
2. **Параметры** — строки в формате **ключ=значение**;
3. **Разделы** — логическая группировка параметров (см. пп. 7.2).

Система использует прямое чтение и парсинг файла без дополнительного шифрования или кодирования.

7.2. Описание входных данных

Входные данные представляют собой набор конфигурационных параметров, организованных в следующие группы:

А. Сетевые параметры: Определяют адреса и порты для межпроцессного взаимодействия:

1. **capturer_bind_addresses** — адреса, на которых Capturer ожидает подключений от Worker'ов
   - Тип: строковый (массив);
   - Пример: tcp://*:5555,tcp://192.168.1.100:5555.

2. **worker_to_capturer_connect_addresses** — адреса, по которым Worker'ы подключаются к Capturer'у
    - Тип: строковый (массив);
    - Пример: tcp://localhost:5555.

3. **worker_to_composer_connect_addresses** — адреса, по которым Worker'ы отправляют результаты в Composer
    - Тип: строковый (массив);
    - Пример: tcp://localhost:5556.

4. **composer_bind_addresses** — адреса, на которых Composer ожидает данных от Worker'ов
    - Тип: строковый (массив);
    - Пример: tcp://*:5556.

Б. Параметры захвата видео (Capturer): Определяют характеристики входного видеопотока:

5. **camera_id** — ID камеры
    - Тип: целочисленный (только положительные);
    - Диапазон: 0-9;
    - Пример: 0.

6. **queue_size** — максимальный размер очереди кадров в Capturer'е
    - Тип: целочисленный (только положительные);
    - Единицы: количество кадров;
    - Пример: 100.

7. **cap_frame_width** — ширина захватываемого кадра
    - Тип: целочисленный (только положительные);
    - Единицы: пиксели;
    - Пример: 640.

8. **cap_frame_height** — высота захватываемого кадра
    - Тип: целочисленный (только положительные);
    - Единицы: пиксели;
    - Пример: 480.

9. **cap_fps** — целевая частота кадров захвата
    - Тип: целочисленный (только положительные);
    - Единицы: кадры в секунду (FPS);
    - Пример: 30.

10. **cap_quality** — качество JPEG-сжатия захваченных кадров
    - Тип: целочисленный (от 1 до 100);
    - Единицы: проценты (100% — наилучшее качество);
    - Пример: 80.

В. Параметры обработки (Worker): Определяют настройки визуального эффекта "Scanner Darkly":

11. **effect_canny_low_threshold** — нижний порог для детектора границ Кэнни
    - Тип: целочисленный (только положительные);
    - Диапазон: 0-255;
    - Пример: 60.

12. **effect_canny_high_threshold** — верхний порог для детектора границ Кэнни
    - Тип: целочисленный (только положительные);
    - Диапазон: 0-255;
    - Пример: 160.

13. **effect_gaussian_kernel_size** — размер ядра размытия Гаусса
    - Тип: целочисленный (только положительные и нечётные);
    - Единицы: пиксели (ядро размером N×N);
    - Пример: 3.

14. **effect_dilation_kernel_size** — размер ядра операции дилатации контуров
    - Тип: целочисленный (только положительные);
    - Единицы: пиксели;
    - Пример: 0 (операция не применяется).

15. **effect_color_quantization_levels** — количество уровней квантования цвета
    - Тип: целочисленный (только положительные);
    - Единицы: количество цветовых кластеров;
    - Пример: 8.

16. **effect_black_contours** — использование черных контуров

- Тип: логический;
- Формат: true/false;
- Пример: true.

Г. Параметры компоновки (Composer): Определяют настройки записи итогового видео:

17. **frame_gap** — максимальный допустимый разрыв между кадрами

- Тип: целочисленный (только положительные);
- Единицы: количество кадров;
- Пример: 500.

18. **buffer_size** — максимальный размер буфера кадров в Composer'е

- Тип: целочисленный (только положительные);
- Единицы: количество кадров;
- Пример: 2000.

Д. Форматы данных

Определяют кодирование и форматы передаваемых данных:

19. **proto_pixel_format** — цветовой формат пикселей в protobuf-сообщениях

- Тип: строковый;
- Допустимые значения: BGR (формат OpenCV);
- Пример: BGR.

20. **proto_image_encoding** — алгоритм кодирования изображений

- Тип: строковый;
- Допустимые значения: JPEG;
- Пример: JPEG.

8. Выходные данные

8.1. Характер, организация и формат выходных данных

Данные формируются в строгой временной последовательности, соответствующей захвату, обработке и сборке видеокадров. Выходные данные подразделяются на:

1. Консольный вывод:

- Статистика работы компонентов системы;
- Сообщения о состоянии и ошибках;
- Информация о переданных и потерянных кадрах.

2. Файлы журналов (логи) работы компонентов: сохраняют консольные выводы (при запуске скриптов Powershell) в той же папке, где находится исполняемый файл соответствующего компонента (как правило в .\ZeroMQCameraSystem\x64\Release\ или .\ZeroMQCameraSystem\x64\Debug\). Консольные выводы хранятся в виде открытого текста в кодировке UTF-8.

3. Файлы видеозаписей: сохраняются в той же папке, где находится исполняемый файл Composer (в .\ZeroMQCameraSystem\x64\Release\ или .\ZeroMQCameraSystem\x64\Debug\). Формат видеофайлов:

- Контейнер: AVI (Audio Video Interleave);
- Кодек: MJPG (Motion JPEG);
- Разрешение: соответствует cap_frame_width × cap_frame_height;
- Частота кадров (FPS): соответствует настройке cap_fps (см. пп. 7.2);
- Цветовое пространство: BGR (Blue-Green-Red) 24 бита на пиксель.

8.2. Описание выходных данных

А. Файлы видеозаписей:

1. output_original.avi — видеофайл, собранный из исходных кадров;
2. output_processed.avi — видеофайл, собранный из обработанных кадров.

Б. Файлы журналов (логи):

1. Capturer.txt — журнал работы компонента Capturer;
2. Composer.txt — журнал работы компонента Composer;
3. Worker_<$id>.txt — журнал работы компонента Worker (имя формируется динамически).

9. Приложения

Приложение А. Файл формата сообщений video_processing.proto

Файл video_processing.proto определяет структуру сообщений для взаимодействия между компонентами системы. На основе .proto файла автоматически генерируется программный код для целевых языков программирования. Например, для генерации кода на C++ используется следующая команда (из директории .\ZeroMQCameraSystem\packages\ protobuf-v141.3.7.1\build\native\include\google\protobuf):

```
..\..\..\bin\protoc.exe --cpp_out=. .\video_processing.proto
```

которая генерирует video_processing.pb.cc и video_processing.pb.h. Этот код отвечает за корректную сериализацию и десериализацию данных, которые включают в себя кадры с идентификатором, временной меткой и их пиксельными данными, что обеспечивает надежный и типизированный обмен сообщениями между компонентами.

Файл video_processing.proto прилагается к настоящей документации в электронном виде и располагается в директории проекта .\ZeroMQCameraSystem\.

10. Источники

1. ZeroMQ: Распределённая система обмена сообщениями [Электронный ресурс]. – URL: https://zeromq.org/ (дата обращения: 16.12.2025).

2. Protocol Buffers: Механизм сериализации структурированных данных [Электронный ресурс]. – URL: https://protobuf.dev/ (дата обращения: 16.12.2025).

3. Github репозиторий проекта ZeroMQCameraSystem [Электронный ресурс].– URL: https://github.com/Volshebnik-Wisard/6nd-UNN__ZeroMQCameraSystem (дата обращения: 16.12.2025).

4. OpenCV: Библиотека компьютерного зрения и обработки изображений [Электронный ресурс]. – URL: https://opencv.org/ (дата обращения: 16.12.2025).

5. Visual Studio 2022: Интегрированная среда разработки Microsoft [Электронный ресурс]. – URL: https://visualstudio.microsoft.com/ru/vs/ (дата обращения: 16.12.2025).