УДК 004.896

# РАЗРАБОТКА ВЫСОКОПРОИЗВОДИТЕЛЬНОЙ БЭКЕНД-СИСТЕМЫ ДЛЯ ИНТЕРАКТИВНОЙ 3D-ВИЗУАЛИЗАЦИИ В ЗАДАЧАХ РОБОТОТЕХНИКИ НА ОСНОВЕ ORB-SLAM3 И ROS

Усенко Е.Р. – студент 2 курса направления подготовки
“Программная инженерия”
Багров И.О. – студент 2 курса направления подготовки
“Программная инженерия”
Научный руководитель: Гагарин Ю.Е., к.т.н., доцент, заведующий кафедрой
Калужский филиал Московского Государственного Технического
Университета имени Н.Э. Баумана
Россия, г. Калуга

Современные исследования в области автономной навигации и мобильной робототехники в значительной степени опираются на алгоритмы SLAM. Эти алгоритмы позволяют роботу одновременно строить карту неизвестной среды и определять своё местоположение в ней. Однако исходные данные, записанные с сенсоров робота (камер, лидаров, IMU) в формате ROS bag [1], представляют собой сложные для непосредственного анализа бинарные файлы. Существующие инструменты для работы с ними, такие как rviz и rqt, требуют глубокого погружения в экосистему ROS и не предоставляют удобных средств для дистанционного совместного анализа, архивации результатов и их интеграции в современные веб-приложения.

В связи с этим возникает задача создания специализированного программного обеспечения, которое бы абстрагировало пользователя от сложностей низкоуровневой работы с ROS и алгоритмами SLAM, предоставляя интуитивно понятный веб-интерфейс для загрузки данных, управления процессом обработки и визуализации результатов. Данная работа посвящена разработке и реализации такой бэкенд-системы, ключевыми особенностями которой являются высокая производительность, способность работать с большими данными и предоставление результатов через стандартизированное API.

Целью работы являлась разработка масштабируемой и отказоустойчивой бэкенд-системы, предназначенной для автоматического построения 3D-карты и траектории по данным ROS bag с использованием алгоритма ORB-SLAM3 [2] и предоставления этих данных через REST API для веб-визуализации.

**Задачи исследования:**

1. Провести анализ современных SLAM-алгоритмов и обосновать выбор ORB-SLAM3 в качестве ключевого вычислительного ядра системы.

2. Разработать модульную архитектуру бэкенда, способную к асинхронной обработке больших файлов (до 10 ГБ).

3. Реализовать механизм интеграции с ORB-SLAM3, обеспечивающий надёжный запуск, выполнение и получение результатов от нативного C++ кода в Python-окружении.

4. Разработать RESTful API, обеспечивающий полный цикл взаимодействия с клиентом: от загрузки файла до получения структурированных данных для визуализации.

5. Оценить производительность системы на реальных данных и подтвердить её практическую применимость.

В качестве вычислительного ядра системы был выбран алгоритм ORB-SLAM3 как на сегодняшний день одна из наиболее точных и универсальных открытых библиотек SLAM, поддерживающая монокулярные, стерео и RGB-D камеры, а также инерциальные данные. Для интеграции этого C++ компонента в Python-окружение был разработан специализированный адаптер. Данный адаптер, используя системные вызовы, запускает скомпилированный бинарный файл ORB-SLAM3 в отдельном процессе, передавая ему путь к конфигурационным файлам и целевому ROS bag-файлу. Стандартные выходные потоки процесса (stdout/stderr) перехватываются и логируются для отладки.

Система реализована на языке Python с использованием микрофреймворка Flask. Архитектура спроектирована как набор слабосвязанных модулей:

1. Модуль приёма файлов отвечает за валидацию и сохранение загружаемых файлов. Поддерживаются форматы ROS1 (.bag) и ROS2 (.db3, .mcap).

2. Диспетчер задач: реализует асинхронную модель обработки на основе многопоточности. Каждая задача на обработку получает уникальный идентификатор сессии (session_id), что позволяет обрабатывать запросы от нескольких пользователей без блокировки веб-интерфейса.

3. ORB-SLAM3-обработчик: Ядро системы, отвечающее за запуск алгоритма, мониторинг его выполнения и обработку результатов.

4. REST API модуль предоставляет следующие ключевые endpoint'ы: POST /api/upload – загрузка ROS bag-файла; GET /api/status/<session_id> – получение статуса задачи (в очереди, в процессе, завершена, ошибка); GET /api/data/<session_id> – получение результатов обработки (траектория, облако точек, метаданные) в формате JSON; GET /api/download/<session_id> – скачивание архивного файла с полными результатами.

Результатом работы системы являются три JSON-файла:

1. trajectory.json: Массив, содержащий последовательность поз робота (координаты x, y, z и кватернион ориентации) с временными метками.

2. pointcloud.json: Массив точек трёхмерной карты, где каждая точка представлена своими координатами (x, y, z).

3. metadata.json: Информация о сессии (исходный файл, время обработки, версия алгоритма, статистика).

Для проверки работоспособности и оценки производительности система была протестирована на наборе реальных данных, содержащем 12896 сообщений с сенсоров. Объём обрабатываемого ROS bag-файла составил приблизительно 8.5 ГБ.

Система успешно выполнила задачу: траектория движения робота и разреженное облако точек окружающей среды были корректно извлечены. Время обработки напрямую зависело от производительности вычислительного узла и сложности сцены, что подтверждает адекватность выбранной асинхронной архитектуры.

Главным достижением работы является успешная инкапсуляция сложного, нативного алгоритма SLAM в удобную веб-ориентированную среду. Это позволяет исследователям, не являющимся экспертами в ROS или C++, легко получать визуализируемые результаты SLAM. Асинхронность обработки и использование потоков обеспечивают конкурентность работы с несколькими пользователями.

В ходе работы была разработана и реализована высокопроизводительная бэкенд-система для интерактивной 3D-визуализации данных SLAM. Доказана практическая целесообразность интеграции алгоритма ORB-SLAM3 в веб-ориентированную архитектуру на Python. Система решает актуальную задачу автоматизации анализа больших данных робототехники, предоставляя результаты через REST API в структурированном формате, пригодном для визуализации в веб-браузере.

Предложенное решение открывает возможности для применения в научных лабораториях, для отладки навигационных алгоритмов, а также в качестве образовательного инструмента, наглядно демонстрирующего принципы работы современных SLAM-систем.

Список литературы:

1. Macenski, S. ROS 2 SLAM Toolbox: Official Repository and Tutorials / S. Macenski // GitHub. – 2025. [Электронный ресурс] // URL: https://github.com/SteveMacenski/slam_toolbox (дата обращения: 15.11.2025).

2. Rapid Deployment of Model Predictive Control for Robotic Systems // IEEE Transactions on Robotics. – 2024. – Vol. 40. [Электронный ресурс] // URL: https://ieeexplore.ieee.org/document/10505632 (дата обращения: 20.11.2025).

3. Qiao, J. SLAM-based Robot Localization and Navigation Algorithm / J. Qiao, J. Guo, Y. Li // Applied Water Science. – 2024. – Vol. 14, № 3. – P. 45–59. [Электронный ресурс] // URL: https://link.springer.com/article/10.1007/s13201-024-02183-6 (дата обращения: 23.11.2025).

4. Addison, A. Building a Map of the Environment Using SLAM – ROS 2 Jazzy / A. Addison. – 2025. [Электронный ресурс] // URL: https://automaticaddison.com/building-a-map-of-the-environment-using-slam-ros-2-jazzy (дата обращения: 17.11.2025).

5. Kehoe, B. Cloud Robotics and Automation / B. Kehoe, S. Patil, P. Abbeel, K. Goldberg // IEEE Transactions on Automation Science and Engineering. – 2015. – Vol. 12, № 2. [Электронный ресурс] // URL: https://ieeexplore.ieee.org/document/7006734 (дата обращения: 23.11.2025).

6. Macenski, S. Robot Operating System 2: Design, architecture, and uses in the wild / S. Macenski, T. Foote, B. Gerkey, C. Lalancette, W. Woodall // Science Robotics. – 2022. – Vol. 7, № 66. – P. eabm6074. – [Электронный ресурс] // URL: https://www.science.org/doi/10.1126/scirobotics.abm6074 (дата обращения: 22.11.2025).