

УДК 004.9

Автономная система воздушного мониторинга и трекинга на базе платформы Clover 4

Кузнецова Полина Александровна

Волосатова Тамара Михайловна (научный руководитель)

МГТУ им. Н.Э. Баумана, Москва, Россия

Представлено описание разработки автономной системы воздушного мониторинга и трекинга на базе платформы Clover 4. Система включает сценарий поведения БПЛА с автоматическим взлетом, стабилизацией, локализацией объекта, мониторингом подвижных целей с помощью ArUco-маркеров и возвратом на исходную позицию. Основные компоненты системы: Raspberry Pi 4, автопилот Pixhawk с прошивкой PX4, ROS Noetic и OpenCV для обработки видеопотока. Рассмотрены методы цветовой сегментации, детекции контуров и трекинга объектов.

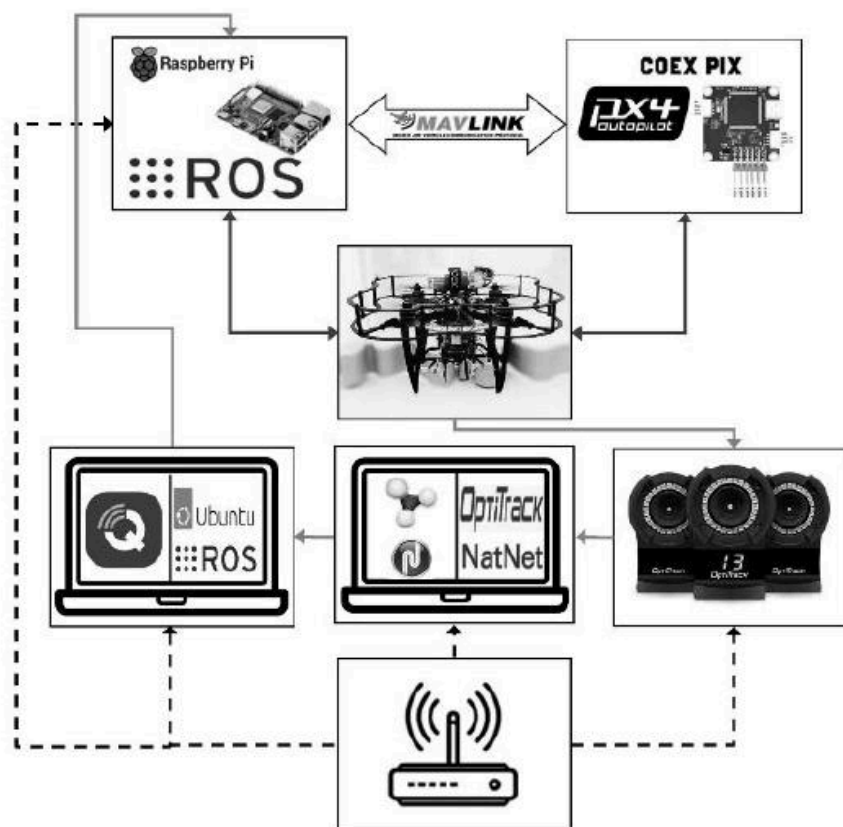
Ключевые слова: БПЛА, автономный мониторинг, компьютерное зрение, ROS, OpenCV, ArUco-маркеры

Введение. Актуальность автономных систем визуального наблюдения возрастает в связи с необходимостью эффективного мониторинга подвижных объектов в различных сферах: от соревнований до промышленного контроля. Одним из подходов к решению данной задачи является использование программируемых беспилотных летательных аппаратов, таких как Clover 4, в сочетании с открытым программным обеспечением [1].

Аппаратная платформа. Clover 4 представляет собой модульную платформу, включающую в себя одноплатный компьютер Raspberry Pi 4, полетный контроллер Pixhawk с прошивкой PX4, а также камеру, средства беспроводной связи, телеметрию и набор сенсоров. Raspberry Pi управляет высокоуровневой логикой, в то время как Pixhawk отвечает за стабилизацию, выполнение низкоуровневых команд и управление полетом [2]. Передача команд от Raspberry Pi к Pixhawk осуществляется через MAVLink — бинарный протокол, предназначенный для обмена сообщениями между станцией управления и автопилотом. Для интеграции MAVLink в среду ROS используется пакет MAVROS, который предоставляет стандартные ROS-интерфейсы для работы с командами навигации, телеметрией и переключением режимов работы автопилота [3] (см. рисунок).

Программная архитектура. Программная реализация построена в среде ROS Noetic — версии операционной системы для роботов, совместимой с Ubuntu 20.04. Все компоненты оформлены в виде ROS-нод и запущены в рамках пользовательского пакета. ROS предоставляет унифицированный механизм подписки и публикации сообщений, а также службы и действия, позволяющие организовать модульную и масштабируемую структуру системы [4, 5]. Код написан на языке Python, что обеспечивает высокую скорость

разработки, читаемость и легкость отладки. В программной логике реализован сценарий автоматического взлета. При старте системы дрон поднимается на заданную высоту в локальной системе координат и переходит в режим наблюдения.



Архитектура автономной системы на базе платформы Clover 4 [1]

Для работы с изображениями используется библиотека OpenCV, предоставляющая инструментарий для обработки видеопотока в реальном времени [6]. Благодаря подключению к ROS-ноде через библиотеку `cv_bridge` видеопоток с камеры преобразуется в формат, совместимый с OpenCV, и используется для анализа сцены.

Ключевым элементом визуальной логики является обнаружение физической стартовой/финишной линии, размещенной на поверхности. Сначала изображение преобразуется из формата BGR в HSV, после чего применяется цветовая сегментация [6]. Далее выделяются контуры, и из них выбирается самый крупный — предполагаемый контур стартовой линии. На его основе вычисляется вертикальная координата линии и ее горизонтальный диапазон, в пределах которого возможно пересечение.

Движущиеся объекты снабжены ArUco-маркерами — устойчивыми двумерными визуальными идентификаторами, каждый из которых имеет уникальный ID. Распознавание и извлечение координат маркеров осуществляется при помощи модуля `cv2.aruco` из OpenCV [6]. После идентификации метки

вычисляется ее центр, и система анализирует положение этого центра относительно линии. При первом пересечении статус участника меняется с `waiting` на `racing`, при втором — на `finished`. Завершение гонки всех участников отслеживается автоматически. После фиксации финиша последнего объекта активируется процедура посадки. Это завершает полный автономный цикл от взлета до посадки без участия оператора.

Состояние системы визуализируется путем наложения графических элементов на видеопоток. Каждая метка отображается с рамкой, ID и текущим статусом. Состояния гонки и порядок финиша выводятся в боковой панели изображения. Обработанное изображение публикуется в ROS-топик, откуда оно может быть захвачено утилитой `web_video_server`. Эта утилита преобразует поток изображений в формат, доступный для отображения в браузере по протоколу `WebSocket`. Таким образом, пользователь получает возможность отслеживать процесс в реальном времени с любого устройства, подключенного к сети, без необходимости подключения к терминалу или запуска специальных утилит [3].

ROS-пакет оформлен с соблюдением стандартной структуры `catkin`-проекта, включает `CMakeLists.txt`, `package.xml`, директорию `scripts` и при необходимости `launch`-файлы. Для запуска системы достаточно одной команды `roslaunch`, которая активирует все необходимые ROS-ноды и службы. Такой подход делает систему удобной в использовании и воспроизводимой на других экземплярах платформы [1].

Заключение. Таким образом, разработанная система успешно сочетает аппаратные средства и программные инструменты, демонстрируя эффективность интеграции для решения задач автономного трекинга и визуального анализа движения объектов. В будущем возможно расширение функциональности системы путем добавления трехмерной локализации маркеров, построения траектории движения и использования методов машинного обучения для анализа поведения объектов.

Литература

- [1] Clover: *программируемая платформа для беспилотников*. URL: <https://clover.coex.tech> (дата обращения 19.01.2025).
- [2] Мартынов В.Г. *Робототехнические комплексы: программное обеспечение и управление*. Санкт-Петербург, БХВ-Петербург, 2020, 416 с.
- [3] MAVROS. URL: <https://wiki.ros.org/mavros> (дата обращения 15.02.2025).
- [4] Брагин А.В., Сергеев А.Р. *Программирование в ROS. Практическое руководство*. Москва, ДМК Пресс, 2021, 320 с.
- [5] Quigley M., Gerkey B., Smart W. *Programming Robots with ROS: A Practical Introduction to the Robot Operating System*. Boston, O'Reilly Media, 2015, 448 p.
- [6] Buratowski T. *Mastering OpenCV 4 with Python*. Birmingham, Packt Publishing, 2019, 350 p.

Autonomous aerial monitoring and tracking system based on the Clover 4 platform

Kuznetsova Polina Alexandrovna

Volosatova Tamara Mikhailovna (*scientific advisor*)

BMSTU, Moscow, Russia

This work presents the development of an autonomous aerial monitoring and tracking system based on the Clover 4 platform. The system features an automated UAV behavior script including takeoff, stabilization, object localization, target tracking via ArUco markers, and return to home. Key components include Raspberry Pi 4, Pixhawk autopilot with PX4 firmware, ROS Noetic, and OpenCV. Techniques such as color segmentation, contour detection, and object tracking are discussed.

Keywords: UAV, autonomous monitoring, computer vision, ROS, OpenCV, ArUco