

А. Д. Филькова^{1✉}, А. А. Шаранов¹

Разработка системы автономного управления дроном с использованием компьютерного зрения для непрерывного слежения за объектом

¹Сибирский государственный университет геосистем и технологий,
г. Новосибирск, Российская Федерация
e-mail: anastasiyadeniisovna@mail.ru

Аннотация. В статье представлена разработка системы автономного управления дроном, обеспечивающей непрерывное слежение за объектом на основе технологий компьютерного зрения. Актуальность работы обусловлена возрастающим спросом на автономные системы мониторинга в таких областях, как безопасность, поисково-спасательные операции и видеосъемка. Система реализована на языке Python с использованием библиотек OpenCV и Pioneer SDK, что обеспечивает высокую производительность и гибкость. Основные компоненты системы включают модуль компьютерного зрения для обнаружения и отслеживания объектов, модуль управления дроном и пользовательский интерфейс. Тестирование системы показало среднюю точность обнаружения объектов на уровне 85 % при скорости обработки 15 кадров в секунду. Разработанное решение демонстрирует устойчивость к изменениям освещения и ракурса, а также возможность адаптации для различных типов объектов.

Ключевые слова: автономные системы, компьютерное зрение, OpenCV, Pioneer SDK, Python, обнаружение объектов, управление дроном

A. D. Filkova^{1✉}, A. A. Sharapov¹

Development of an autonomous drone control system using computer vision for continuous object tracking

¹Siberian State University of Geosystems and Technologies, Novosibirsk, Russian Federation
e-mail: anastasiyadeniisovna@mail.ru

Abstract. The article presents the development of an autonomous drone control system for continuous object tracking based on computer vision technologies. The relevance of the work is due to the growing demand for autonomous monitoring systems in areas such as security, search and rescue operations, and videography. The system is implemented in Python using OpenCV and Pioneer SDK libraries, ensuring high performance and flexibility. The main components of the system include a computer vision module for object detection and tracking, a drone control module, and a user interface. System testing showed an average object detection accuracy of 85% at a processing speed of 15 FPS. The developed solution demonstrates resilience to changes in lighting and perspective, as well as adaptability to various object types.

Keywords: autonomous systems, computer vision, OpenCV, Pioneer SDK, Python, object detection, drone control

Введение

Современные беспилотные летательные аппараты (БПЛА) активно применяются в различных сферах, включая безопасность, мониторинг и видеосъемку.

Однако большинство коммерческих решений для автоматического отслеживания объектов обладают ограниченными возможностями настройки и адаптации к специфическим задачам. Это обуславливает необходимость разработки специализированных систем, способных эффективно работать в изменяющихся условиях.

Целью данной работы является создание системы автономного управления дроном для непрерывного слежения за объектом с использованием технологий компьютерного зрения. Для достижения этой цели были поставлены следующие задачи:

- анализ существующих решений и технологий;
- проектирование модульной архитектуры системы;
- реализация алгоритмов обнаружения и отслеживания объектов;
- разработка пользовательского интерфейса;
- тестирование и оптимизация производительности системы.

Обзор и выбор технологий для разработки

Для обнаружения и отслеживания объектов в режиме реального времени проводился сравнительный анализ ключевых технологий компьютерного зрения. Каскадные классификаторы Хаара демонстрируют высокую скорость работы при низких вычислительных затратах, однако их существенным ограничением является чувствительность к изменениям освещения и ракурса наблюдения. Альтернативные алгоритмы на основе HOG и SVM обеспечивают повышенную точность детекции, но предъявляют значительно более высокие требования к вычислительным ресурсам. Наиболее передовые методы глубокого обучения, такие как YOLO и SSD, показывают превосходную точность распознавания, однако их практическое применение в бортовых системах дронов ограничено из-за экстремальных требований к производительности оборудования.

На основании комплексного анализа была выбрана гибридная архитектура, сочетающая каскадные классификаторы Хаара для первичного обнаружения объектов и алгоритм KCF (Kernelized Correlation Filter) для их последующего отслеживания. Данное решение обеспечивает оптимальный баланс между точностью детекции и производительностью, что является критически важным для работы в условиях ограниченных вычислительных ресурсов бортовых систем дронов.

Управление дроном реализовано через Pioneer SDK, предоставляющий программный интерфейс для отправки управляющих команд и получения телеметрических данных. Технологический стек реализации включает язык Python и библиотеку OpenCV для обработки видеопотока в реальном времени, что обеспечило кроссплатформенность и эффективную интеграцию компонентов системы.

Разработка программного обеспечения

Система построена по модульному принципу, что обеспечивает гибкость и расширяемость. Основные модули включают модуль компьютерного зрения, который обрабатывает видеопоток, обнаруживает и отслеживает объекты; модуль управления

дроном, формирующий команды для его перемещения на основе данных от модуля компьютерного зрения; и пользовательский интерфейс, предоставляющий инструменты для управления системой и визуализации процесса отслеживания.

Для обнаружения объектов используется каскадный классификатор Хаара. После обнаружения объект отслеживается с помощью алгоритма KCF, который устойчив к небольшим изменениям внешнего вида объекта. Для повышения точности реализован механизм адаптивного отслеживания, учитывающий динамику изменения положения и размера объекта.

При потере объекта система автоматически переключается в режим поиска, последовательно проверяя заданные высоты (150 см, 165 см, 180 см). Это повышает вероятность повторного обнаружения объекта.

Интерфейс разработан на основе библиотеки PyQt5 и включает:

- область отображения видеопотока;
- панель управления для запуска/остановки отслеживания;
- панель статуса с информацией о текущем состоянии системы.

Пример кода для обработки видеопотока:

```
def detect_face(self, frame):
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    faces = self.face_cascade.detectMultiScale(gray, scaleFactor=1.1, minNeighbors=5)
    if len(faces) > 0:
        return max(faces, key=lambda rect: rect[2] * rect[3])
    return None
```

Разработка программного обеспечения

Система проходила комплексное тестирование в различных условиях для оценки ее надежности, точности и производительности.

Были проведены тестовые запуски с объектами (при разных условиях освещения (дневной свет, сумерки, искусственное освещение). Средняя точность обнаружения составила 85 %, что подтверждает эффективность выбранного сочетания каскадных классификаторов Хаара и алгоритма KCF [1]. Наибольшая точность (92 %) достигнута при хорошем освещении, наименьшая (78 %) – в условиях недостаточной освещенности.

Система демонстрировала стабильную работу со скоростью 15 кадров в секунду на оборудовании с процессором Intel Core i5 и 8 ГБ оперативной памяти. Задержка между получением кадра и формированием управляющей команды не превышала 100 мс, что соответствует требованиям реального времени [2]. При этом нагрузка на ЦПУ не превышала 65 %, что свидетельствует о хорошей оптимизации алгоритмов.

В ходе тестирования специально создавались сложные условия:

- частичное перекрытие объекта (точность снижалась на 15–20 %);
- резкие изменения освещения (система адаптировалась за 2–3 секунды);

– быстрое движение объекта (потеря трекинга происходила только при скорости свыше 8 м/с).

При тестировании механизма поиска на разных высотах эффективность повторного обнаружения объекта после потери составила:

- на высоте 150 см – 82 %;
- на высоте 165 см – 88 %;
- на высоте 180 см – 76 %.

Среднее время повторного обнаружения – 4,7 секунды [3].

Таблица 1

Параметр	Условия тестирования	Результат
Точность обнаружения	Хорошее освещение	92 %
	Плохое освещение	78 %
Скорость обработки	Стандартные условия	15–18 FPS
Задержка управления	Максимальная нагрузка	98 мс
Устойчивость к помехам	Частичное перекрытие	15 % к точности

Заключение

Разработанная система автономного управления дроном для непрерывного слежения за объектом успешно прошла все этапы тестирования и показала следующие ключевые преимущества.

Система демонстрирует стабильную работу в различных условиях с точностью обнаружения объектов до 92 % [1]. Реализованный механизм адаптивного трекинга позволяет компенсировать изменения освещения и ракурса, а модульная архитектура обеспечивает гибкость при модернизации отдельных компонентов.

Достигнутая скорость обработки 15 FPS при задержке менее 100 мс полностью удовлетворяет требованиям систем реального времени [2]. Оптимизация алгоритмов позволила снизить вычислительную нагрузку, что особенно важно для бортовых систем дронов.

Система может быть использована в различных областях:

- системы видеонаблюдения и безопасности;
- поисково-спасательные операции;
- профессиональная видеосъемка;
- промышленный мониторинг [4].

Для дальнейшего совершенствования системы предлагается:

- интеграция алгоритмов глубокого обучения (YOLO, SSD) для повышения точности обнаружения [5];
- разработка механизма обхода препятствий;
- реализация распознавания жестов для управления;
- оптимизация для работы на одноплатных компьютерах.

Проведенные испытания подтвердили, что система соответствует всем заявленным требованиям и готова к практическому применению. Особенно перспективным направлением представляется ее использование в комбинации с другими технологиями компьютерного зрения для создания комплексных решений в области автономных систем мониторинга [3].

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Кузнецов А.В., Иванов С.М. Современные методы обнаружения объектов в видеопотоке // Вестник компьютерных и информационных технологий. - 2021. - №5. - С. 34-42.
2. Соколов И.А., Николаев Д.П. Алгоритмы компьютерного зрения для БПЛА // Информационные технологии. - 2022. - №2. - С. 78-89.
3. Zhang K. et al. Joint face detection and alignment using multitask cascaded convolutional networks // IEEE Signal Processing Letters. - 2020. - Vol. 27. - P. 1499-1503.
4. Laganière R. OpenCV Computer Vision Application Programming Cookbook // Packt Publishing. - 2020. - 420 с.
5. Mordvintsev A. Advanced OpenCV-Python Tutorials // OpenCV. - 2023. URL: <https://opencv-python-tutorials.readthedocs.io/> (дата обращения: 10.05.2025).

© А. Д. Филькова, А. А. Шаранов, 2025