ОШ МАМЛЕКЕТТИК УНИВЕРСИТЕТИНИН ЖАРЧЫСЫ

ВЕСТНИК ОШСКОГО ГОСУДАРСТВЕННОГО УНИВЕРСИТЕТА

BULLETIN OF OSH STATE UNIVERSITY

ISSN 1694-7452 e-ISSN: 1694-8610

№2/2026, 341-364

ИНФОРМАТИКА

УДК: 004.8:629.735.05

DOI: 10.52754/16948610_2026_2_25

АДАПТИВНАЯ СИСТЕМА АВТОНОМНОЙ НАВИГАЦИИ И ЦЕЛЕВОГО СОПРОВОЖДЕНИЯ ДЛЯ БПЛА НА ОСНОВЕ ГЛУБОКОГО ОБУЧЕНИЯ

ТЕРЕҢ ОКУТУУНУН НЕГИЗИНДЕ БПЛА ҮЧҮН АВТОНОМДУУ НАВИГАЦИЯНЫН ЖАНА МАКСАТТУУ КОШТООНУН АДАПТИВДҮҮ СИСТЕМАСЫ

ADAPTIVE AUTONOMOUS NAVIGATION AND TARGET TRACKING SYSTEM FOR DEEP-LEARNING BASED UAVS

Камалов Султанбек Садырбекович
Камалов Султанбек Садырбекович
Kamalov Sultanbek Sadyrbekovich

Преподователь, Ошский государственный университет
окутуучу, Ош мамлекеттик университети
lecturer, Osh State University
skamalov@oshsu.kg
ORCID: 0009-0002-9944-9782

Азимов Бектур Абдырахманович
Азимов Бектур Абдырахманович
Azimov Bektur Abdrahmonovich

к.ф.-м.н,доцент, Ошский государственный университет
ф.м.и.к,доцент, Ош мамлекеттик университети
Candidate of Physical and Mathematical Sciences, Associate Professor, Osh State University
azimov@oshsu.kg
ORCID:0000-0001-5849-8583

Ракишева Диляра Советовна
Ракишева Диляра Советовна
Rakishva Dilyara Sovetovna

PhD, доцент, Ошский государственный университет
PhD, доцент, Ош мамлекеттик университети
PhD, Associate Professor, Osh State University
dilya784@mail.ru
ORCID: 0000-0001-8434-9469

АДАПТИВНАЯ СИСТЕМА АВТОНОМНОЙ НАВИГАЦИИ И ЦЕЛЕВОГО СОПРОВОЖДЕНИЯ ДЛЯ БПЛА НА ОСНОВЕ ГЛУБОКОГО ОБУЧЕНИЯ

Аннотация

Актуальность. В статье представлена система автономного управления беспилотными летательными аппаратами (БПЛА) на базе Robot Operating System (ROS) для эффективного обнаружения и отслеживания объектов в режиме реального времени при ограниченных вычислительных ресурсах. Система интегрирует модифицированную архитектуру YOLOv4 для высокоскоростной детекции объектов и алгоритм SiamMask для непрерывного трекинга целей. Управление траекторией полёта реализовано через ПИД-регулятор, обеспечивающий стабильное автоматическое сопровождение целевых объектов. Ключевая новизна исследования заключается в разработке методологии оптимизации нейросетевых моделей для встраиваемых систем с критическими ограничениями вычислительных мощностей. Предложенный подход позволяет адаптировать современные архитектуры глубокого обучения для работы на одноплатных компьютерах и встроенных GPU без существенной потери точности детекции.

В основе системы лежит комбинация методов компьютерного зрения и машинного обучения. Свёрточные нейронные сети, обученные на расширенных датасетах, обеспечивают робастное распознавание объектов при различных погодных и световых условиях. Модуль трекинга использует оптический поток и фильтр Калмана с предиктивными алгоритмами для прогнозирования траектории движения. Система устойчива к шуму, помехам и частичной окклюзии целей.

Экспериментальная валидация проведена в помещениях и на открытой местности с использованием реальных видеоданных и симуляций. Тестовые сценарии включали сопровождение транспортных средств, людей и динамических объектов в различных условиях.

Разработанная архитектура расширяет возможности применения автономных БПЛА в задачах мониторинга, поисково-спасательных операций и промышленной инспекции, где критичны массогабаритные и энергетические характеристики бортового оборудования.

Ключевые слова: беспилотные летательные аппараты, глубокое обучение, ROS, сверточные нейронные сети, компрессия нейросетевых моделей, трекинг целей, ПИД-регулятор, детекция объектов, компьютерное зрение, машинное обучение, автономные системы.

Терең окутуунун негизинде бпла үчүн автономдуу навигациянын жана максаттуу коштоонун адаптивдүү системасы

Adaptive autonomous navigation and target tracking system for deep-learning based uavs

Аннотация

Маанилүүлүк. Макалада чектелген эсептөө ресурстары менен реалдуу убакыт режиминде объекттерди эффективдүү аныктоо жана көзөмөлдөө үчүн Robot Operating System (ROS) негизиндеги учкучсуз учуучу аппараттарды автономдуу башкаруу системасы (UAV) берилген. Система жогорку ылдамдыктагы объектти аныктоо үчүн өзгөртүлгөн YOLOv4 архитектурасын жана үзгүлтүксүз бутага көз салуу үчүн SiamMask алгоритмин бириктирет. Учуу жолу максаттуу объекттерге туруктуу автоматтык байкоо жүргүзүүнү камсыз кылган PID контроллери аркылуу башкарылат. Изилдөөнүн негизги жаңылыгы - эсептөө кубаттуулугунун критикалык чектөөлөрү менен орнотулган системалар үчүн нейрондук тармак моделдерин оптималдаштыруу методологиясын иштеп чыгуу. Сунушталган ыкма заманбап терең үйрөнүү архитектураларын аныктоо тактыгын олуттуу жоготпостон, бир такталуу компьютерлерде жана орнотулган GPUларда иштөөгө ылайыкташтырууга мүмкүндүк берет.

Система компьютердик көрүү жана машина үйрөнүү ыкмаларынын айкалышына негизделген. Кеңейтилген маалымат топтомдорунда үйрөтүлгөн конволюциялык нейрон тармактары ар кандай аба ырайы жана жарык шарттарында объекттерди күчтүү таанууну камсыз кылат. Көз салуу модулу кыймылдын траекториясын болжолдоо үчүн оптикалык агымды жана болжолдуу алгоритмдери бар Калман чыпкасын колдонот. Система ызы-чууга, тоскоолдуктарга жана буталардын жарым-жартылай окклюзиясына туруктуу.

Эксперименталдык валидация реалдуу видео маалыматтарды жана симуляцияларды колдонуу менен имараттын ичинде жана ачык жерлерде жүргүзүлдү. Сыноо сценарийлери ар кандай шарттарда унааларды, адамдарды жана динамикалык объекттерди коштоону камтыган.

Иштелип чыккан архитектура борттогу жабдуулардын салмагын жана көлөмүн жана энергетикалык мүнөздөмөлөрү маанилүү болгон мониторинг, издөө-куткаруу жана өнөр жай инспекциясынын тапшырмаларында автономдуу УУАларды колдонуу мүмкүнчүлүктөрүн кеңейтет.

Ачкыч сөздөр: учкучсуз учуучу аппараттар, терең үйрөнүү, ROS, конволюциялык нейрон тармактары, нейрондук тармак моделдерин кысуу, бутага көз салуу, PID контроллери, объектти аныктоо, компьютердик көрүү, машина үйрөнүү, автономдуу системалар.

Abstract

Relevance. This paper presents an autonomous control system for unmanned aerial vehicles (UAVs) based on Robot Operating System (ROS) for efficient real-time object detection and tracking under limited computational resources. The system integrates a modified YOLOv4 architecture for high-speed object detection and the SiamMask algorithm for continuous target tracking. Flight trajectory control is implemented through a PID controller ensuring stable automatic target following. The key novelty of the research lies in developing a methodology for optimizing neural network models for embedded systems with critical computational power constraints. The proposed approach enables adaptation of modern deep learning architectures for operation on single-board computers and embedded GPUs without significant loss of detection accuracy.

The system is based on a combination of computer vision methods and machine learning algorithms. Convolutional neural networks trained on extended datasets provide robust object recognition under various weather and lighting conditions. The tracking module employs optical flow and Kalman filter with predictive algorithms for trajectory forecasting. The system is resistant to noise, interference, and partial target occlusion.

Experimental validation was conducted in indoor and outdoor environments using real video data and simulations. Test scenarios included tracking of vehicles, people, and dynamic objects under various conditions.

The developed architecture expands the application capabilities of autonomous UAVs in monitoring tasks, search and rescue operations, and industrial inspection, where weight-size and energy characteristics of onboard equipment are critical.

Keywords: unmanned aerial vehicles, deep learning, ROS, convolutional neural networks, neural network model compression, target tracking, PID controller, object detection, computer vision, machine learning, autonomous systems.

Введение

В последние годы индустрия робототехники демонстрирует значительный рост. Среди различных направлений развития беспилотные летательные аппараты (БПЛА), также известные как дроны, стали одной из наиболее перспективных областей применения. БПЛА представляет собой автономный или дистанционно управляемый летательный аппарат, функционирующий без присутствия пилота на борту и способный транспортировать полезную нагрузку. Первоначально БПЛА нашли основное применение в военной сфере, где использовались для выполнения задач повышенного риска, таких как разведывательные операции, ударные миссии и материально-техническое обеспечение с целью минимизации рисков для персонала. Однако прогресс в области аэрокосмических материалов, инерциальных датчиков, навигационных технологий, обработки изображений и передачи данных существенно расширил спектр применения БПЛА, охватив гражданские сферы деятельности.

В настоящее время беспилотные летательные аппараты находят широкое применение в гражданских областях. Например, дистанционно управляемые аппараты активно используются в развлекательных и рекреационных целях, а появление гоночных дронов с видом от первого лица (FPV) способствовало развитию нового вида спорта. Кроме того, БПЛА применяются для аэрофотосъемки и видеосъемки, обеспечивая уникальные перспективы съемки с воздуха. Ряд компаний начали внедрять дроны в логистические службы доставки, что позволяет повысить скорость и эффективность транспортировки грузов. Вместе с тем широкое распространение беспилотных летательных аппаратов обусловило возникновение проблем регулирования и обеспечения безопасности полетов. Для решения данных проблем Федеральное авиационное управление США (FAA) разработало «Систему эксплуатации и сертификации малых беспилотных авиационных систем», которая классифицирует БПЛА-квадрокоптеры и устанавливает стандарты и требования к их эксплуатации и сертификации.

При разработке беспилотных летательных аппаратов часто возникают проблемы совместимости между различными системами. Для решения этих проблем была создана операционная система Robot Operating System (ROS) — инструмент с открытым исходным кодом для управления роботизированными системами. ROS предоставляет платформу и инструменты для управления межсистемными соединениями, что позволяет разработчикам упростить процессы установки, управления и мониторинга систем беспилотных летательных аппаратов. Система также предлагает множество программных модулей и библиотек для реализации различных функций, таких как обработка данных с датчиков, управление движением и планирование полетов. Целью настоящей работы является применение технологии ROS к беспилотным летательным аппаратам, что позволит создать модульную систему и упростить общий процесс разработки, сделав его более доступным и эффективным.

Многие современные беспилотные летательные аппараты оснащены функциями отслеживания и сопровождения целей. Как правило, эта функциональность реализуется посредством электронных устройств, размещаемых на объекте отслеживания, которые используют координаты GPS для определения местоположения и сопровождения цели. Однако в условиях недоступности сигналов GPS, например, в туннелях или подземных помещениях, GPS-позиционирование становится неэффективным или невозможным. В таких случаях отслеживание на основе анализа изображений служит альтернативным методом сопровождения цели. Одним из ключевых аспектов данного исследования является изучение

возможности управления системами беспилотных летательных аппаратов с помощью методов обнаружения и слежения на основе обработки изображений. Используя методы детекции и трекинга, дроны могут получать изображения целей посредством камер и выполнять анализ в режиме реального времени для достижения точного обнаружения и отслеживания положения и ориентации объекта.

С развитием методов глубокого обучения было предложено множество моделей для решения задачи обнаружения объектов на изображениях. R-CNN (Girshick, et al., 2014, pp. 580-587) использует высокопроизводительные сверточные нейронные сети для анализа регионов по принципу «снизу вверх» с целью локализации и сегментации объектов. SPP-Net (He, et al. 2014, pp. 580-587) применяет пространственный пирамидальный пулинг для исключения необходимости использования входных изображений фиксированного размера. Faster R-CNN (Dina, et al., 2025, pp. 1281-1308) совершенствует архитектуры R-CNN и SPP-Net, снижая время обучения и тестирования при одновременном повышении точности обнаружения. Детектор Single Shot MultiBox (SSD) использует многомасштабные сверточные выходы, подключенные к нескольким картам признаков в верхней части сети, для обнаружения объектов с использованием единой глубокой нейронной сети. В отличие от предыдущих работ, рассматривавших обнаружение объектов как задачу классификации, алгоритм You Only Look Once (YOLO) (Redmon, et al., 2016, pp. 779-788) формулирует обнаружение объектов как задачу регрессии к пространственное разделенным ограничивающим рамкам и связанным с ними вероятностям классов.

Единая нейронная сеть, оптимизируемая от начала до конца, используется для прогнозирования ограничивающих рамок и вероятностей классов непосредственно на основе полных изображений за одну итерацию. YOLO9000 и YOLOv2 (Redmon, et al., 2016, pp. 779-788) усовершенствовали оригинальный алгоритм YOLO путем внедрения концепций пакетной нормализации, классификатора с высоким разрешением, свертки с якорными рамками, кластеризации размерностей, прямого прогнозирования местоположения и многомасштабного обучения. YOLOv3 внес ряд модификаций для улучшения характеристик YOLO. YOLOv4 провел обширные эксперименты с методами взвешенных остаточных соединений, межэтапных частичных соединений, нормализации перекрестных мини-пакетов, самообучения, активации Mish, аугментации мозаичных данных, регуляризации DropBlock и функции потерь CIoU, объединив подмножество этих методов для достижения результатов, соответствующих современному уровню развития технологий.

Обнаружение людей представляет собой специализированную форму детекции объектов, предназначенную для идентификации класса «человек» на изображениях или видеокадрах. В связи с этим в данной работе используется архитектура YOLOv4 для выполнения задачи обнаружения людей беспилотным летательным аппаратом. Для снижения вычислительной сложности YOLOv4 с целью применения алгоритма в среде с существенно ограниченными вычислительными ресурсами выполняется оптимизация исходной модели посредством её сокращения.

Для уменьшения сложности сверточных нейронных сетей (CNN) были предложены различные методы прунинга (сокращения) (https://github.com/TNTWEN/Pruned-OpenVINO-YOLO 10.05.2023). Сокращение каналов направлено на использование избыточности карт признаков между каналами и удаление каналов с минимальной потерей производительности (Li, et al., 2023, pp. 1208-1219) предложили сокращать модели глубокого обучения, используя методы

компрессии как на уровне каналов, так и на уровне слоев разработали метод прунинга, который может быть непосредственно применен к существующим современным архитектурам CNN путем обеспечения разреженности каналов в сети для уменьшения размера модели, снижения объема используемой памяти во время выполнения и сокращения количества вычислительных операций при сохранении точности модели. В работе (https://github.com/TNTWEN/Pruned-OpenVINO-YOLO 10.05.2023) авторы демонстрируют методику сокращения моделей YOLOv3 и YOLOv4 с последующим развертыванием на платформе OpenVINO с увеличенной частотой кадров и незначительной потерей точности. Поскольку в данной работе используется YOLOv4 для обнаружения объектов, применяются методы сокращения, описанные в (https://github.com/TNTWEN/Pruned-OpenVINO-YOLO 10.05.2023).

Алгоритмы отслеживания на основе сиамских сетей получили широкое распространение в области визуального трекинга использовали полностью сверточную сиамскую сеть для задач отслеживания (Zhu, et al., 2018, pp. 101-117) разработали модуль, учитывающий отвлекающие факторы, для выполнения поэтапного обучения, способный эффективно адаптировать общее встраивание к текущему видеодомену. предложили трекер на основе сети предложений сиамского региона (Siamese Region Proposal Network), обучаемый в автономном режиме с использованием крупномасштабных пар изображений.

В работе используется ROS (Robot Operating System) (Quigley, et al., 2009, pp.1-6) для реализации обнаружения и отслеживания объектов на изображениях с целью управления беспилотными летательными аппаратами. Ввиду аппаратных ограничений бортового компьютера требуется использование облегченных моделей. Для детектора объектов обучается сверточная нейронная сеть на основе архитектуры YOLOv4 с последующим её сокращением. В данной работе целевым объектом для обнаружения является человек. Используется сокращенная версия детектора объектов YOLOv4 и монокулярный трекер SiamMask для обнаружения и отслеживания человека, фиксируемого камерой беспилотника. Разработанная система состоит из четырех основных компонентов: (1) модуль обнаружения объектов, (2) модуль отслеживания целей, (3) пропорционально-интегрально-дифференцирующий (ПИД) регулятор и (4) пакет драйверов для беспилотного летательного аппарата. Для реализации системы обнаружения и слежения за объектами используется беспилотник Tello. В процессе отслеживания параметры управления БПЛА включают крен, тангаж, рыскание и высоту, которые регулируются посредством ПИД-регуляторов. Эти ПИД-регуляторы определяют положение и расстояние до целевого объекта на основе входных данных. Местоположение и расстояние рассчитываются с использованием монокулярной фронтальной камеры беспилотника.

2. Методология исследования

В разделе подробно описаны методы, используемые в предлагаемой системе, включая обнаружение объектов, оптимизацию модели и визуальное отслеживание. На рисунке 1 представлена структура системы. Для связи с дроном Tello по протоколу Wi-Fi используется портативный компьютер. Беспилотник передает изображения с частотой 30 Гц, предварительно настроенной в программном обеспечении драйвера. Полученные изображения обрабатываются на компьютере с использованием усовершенствованной версии алгоритма обнаружения объектов YOLOv4. Пользователи имеют возможность выбирать

ограничивающие рамки в соответствии с требуемыми параметрами. Система использует сиамскую сеть SiamMask для отслеживания объектов. На основе положения и расстояния до отслеживаемого объекта применяется алгоритм на базе ПИД-регулятора, который вычисляет значения крена, тангажа, рыскания и высоты. Рассчитанные параметры передаются обратно на беспилотник для инициирования процесса отслеживания с использованием информации о текстуре фона для улучшения результатов.

Общая архитектура системы представлена на рисунке 1. Дрон отправляет изображение на компьютер, где полученные данные обрабатываются с помощью оптимизированной модели Pruned-YOLOv4 для распознавания людей. При обнаружении человека на изображении на экране отображается соответствующая ограничивающая рамка. Зеленые прямоугольники на рисунке 1 представляют результаты детекции. Если пользователь выбирает конкретный интересующий объект, нажимая на его ограничивающую рамку, система извлекает изображение человека внутри рамки в качестве шаблона для сети SiamMask, инициируя последующее отслеживание. Алгоритм трекинга вычисляет отклонение между положением цели и центром кадра. Данное отклонение служит входным сигналом для ПИД-регулятора, который генерирует команды управления полетом для корректировки курса, крена и высоты. Четвертая команда управления — тангаж — рассчитывается на основе относительного расстояния до отслеживаемого объекта с использованием данных о его местоположении. При отсутствии цели на изображении беспилотник сохраняет текущее положение до момента повторного обнаружения объекта.

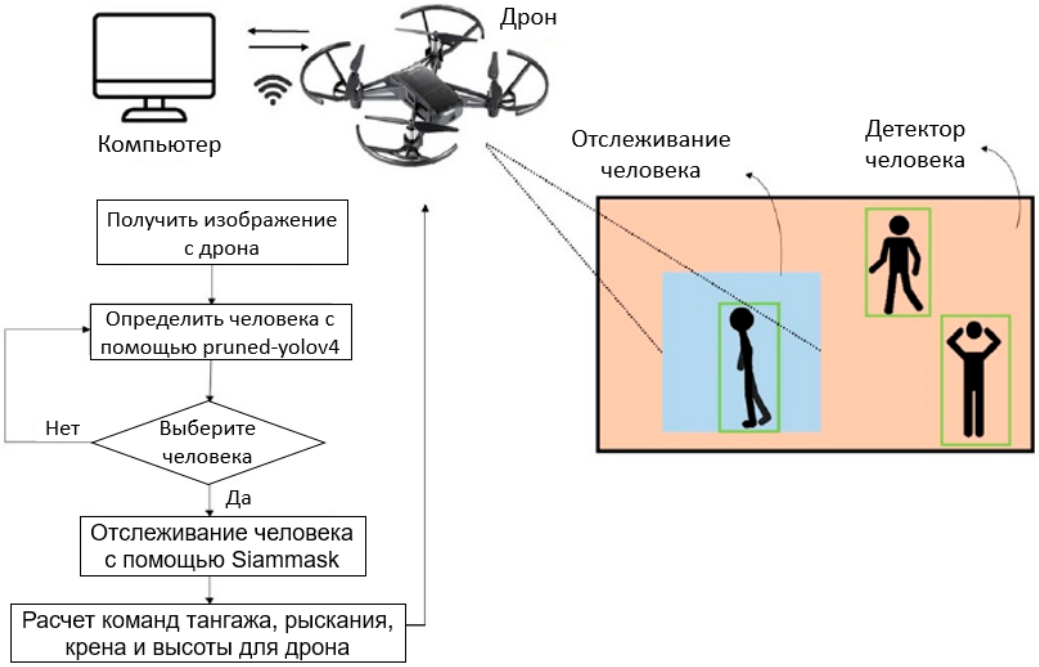

Рисунок 1. Структура системы.

2.1. Технические характеристики оборудования

В качестве экспериментальной платформы используется беспилотный летательный аппарат DJI Tello (https://www.ryzerobotics.com/zhtw/telloedu?site=brandsite&from=landing_page 18.11.2023) компактный БПЛА потребительского класса с простым управлением. Габаритные размеры устройства составляют приблизительно 98 × 92 мм при массе около 80 г. Конструкция данного беспилотника позволяет эксплуатировать его как в закрытых

помещениях, так и на открытых пространствах. Аппарат оснащен фронтальной камерой, 3-осевым гироскопом, 3-осевым акселерометром, 3-осевым магнитометром, датчиком давления и ультразвуковым высотомером. Разрешение фронтальной камеры составляет 1280 × 720 пикселей с возможностью видеосъемки со скоростью 30 кадров в секунду. Дрон Tello взаимодействует с внешними устройствами, такими как смартфоны или портативные компьютеры, посредством беспроводной сети Wi-Fi. В данном исследовании для связи с беспилотником использовался персональный компьютер.

2.2. Оптимизация модели обнаружения и детекция объектов

На этапе обнаружения объектов необходимо предварительное обучение модели. Процесс обучения представлен на рисунке 2. Желтая часть диаграммы соответствует модулям платформы Darknet framework, светло-голубая — модулям этапа оптимизации модели. На первом этапе с использованием фреймворка Darknet обучается базовая модель YOLOv4. Затем на этапе оптимизации последовательно выполняются разреженное обучение (sparse training), сокращение каналов (channel pruning), дообучение после сокращения и тонкая настройка (fine-tuning) базовой модели с использованием фреймворка Darknet. После завершения этапа прунинга модель подвергается финальной тонкой настройке на платформе Darknet. В заключение оптимизированная модель развертывается на портативном компьютере для выполнения задачи обнаружения объектов в режиме реального времени.

А. Обучение в Даркнете

Мы используем фреймворк Darknet для обучения модели YOLOv4 (https://github.com/AlexeyAB/Darknet (1.06. 2023) и корректируем несколько гиперпараметров на начальном этапе обучения, чтобы повысить точность и производительность модели. Одним из первых гиперпараметров, который необходимо настроить, является размер входных данных сети. Увеличение размера входных данных помогает обнаруживать небольшие объекты, хотя это также может замедлить скорость вывода модели и увеличить объем памяти графического процессора. Важно отметить, что сеть YOLOv4 уменьшает размер входных данных в 32 раза как по вертикали, так и по горизонтали, поэтому ширина и высота входных данных должны быть кратны 32. Чтобы добиться этого, мы решили использовать 416 × 416 в качестве входного размера.

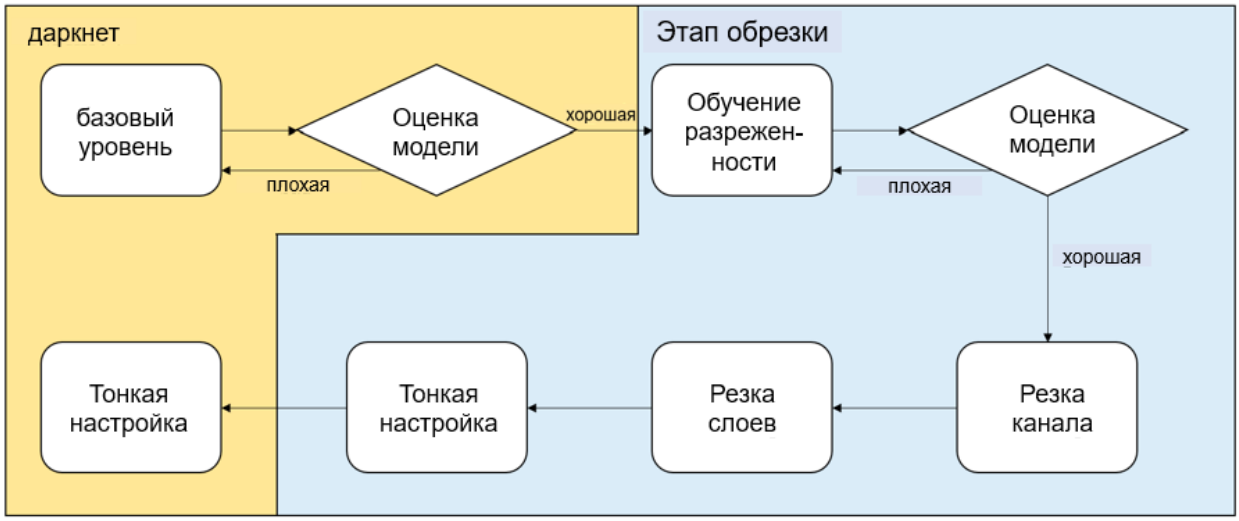

Рисунок 2. Этапы обучения модели обнаружения.

Вторым и третьим гиперпараметрами, которые необходимо настроить, являются размер пакета и его подразделения. Эти параметры настраиваются в зависимости от производительности графического процессора. Гиперпараметр размера пакета представляет собой количество изображений, загружаемых во время обучения, значение по умолчанию равно 64. Однако, если объем памяти графического процессора недостаточен, он не сможет загрузить 64 изображения одновременно. Чтобы устранить эту проблему, каждый пакет разбивается на несколько подпакетов. Каждый подпакет загружается в графический процессор один за другим, пока пакет не будет завершен. В этом исследовании мы установили размер партии и подразделения на 64 и 8 единиц соответственно. Четвертый гиперпараметр, который необходимо скорректировать, - это количество итераций (обратите внимание, что в Darknet framework обучение измеряется в итерациях, а не в эпохах). Согласно рекомендациям Darknet framework, каждый класс объектов должен пройти не менее 2000 итераций. Поскольку у нас только один класс, количество итераций должно превышать 2000. Мы установили число итераций равным 2200 для достижения более высокой точности.

B. Этап сокращения

Из-за аппаратных ограничений ноутбука необходимо использовать облегченную модель. Поэтому после обучения модели с использованием платформы Darknet ее необходимо сократить, чтобы достичь цели уменьшения веса. Мы используем показатели точности (mAP@0.5) и скорости логического вывода (Flops) для оценки сокращенной модели. Однако важно отметить, что существует компромисс между точностью и скоростью логического вывода. Предполагая, что конфигурация оборудования фиксирована, когда модель сокращается до очень малого размера, скорость ее вывода может увеличиться, но ее точность, как правило, снижается.

Перед сокращением веса из фреймворка Darknet проходят базовый процесс обучения. Как только базовое обучение завершено, полученная модель сокращается с использованием стратегии сокращения. Эта стратегия предполагает сначала проведение разреженного обучения модели, где разреженность каналов в глубоких моделях помогает при сокращении каналов. Для облегчения сокращения каналов каждый канал в сверточных слоях связан с коэффициентом масштабирования. Во время обучения к этим коэффициентам масштабирования применяется регуляризация для автоматического определения неважных каналов. Каналы с меньшими значениями коэффициента масштабирования (оранжевый цвет) отсекаются (слева). После отсечения мы получаем компактную модель (справа), которая затем подвергается точной настройке для достижения точности, сравнимой (или даже более высокой) с точностью полностью обученной сети. Процесс обрезки проиллюстрирован на рисунке 3.

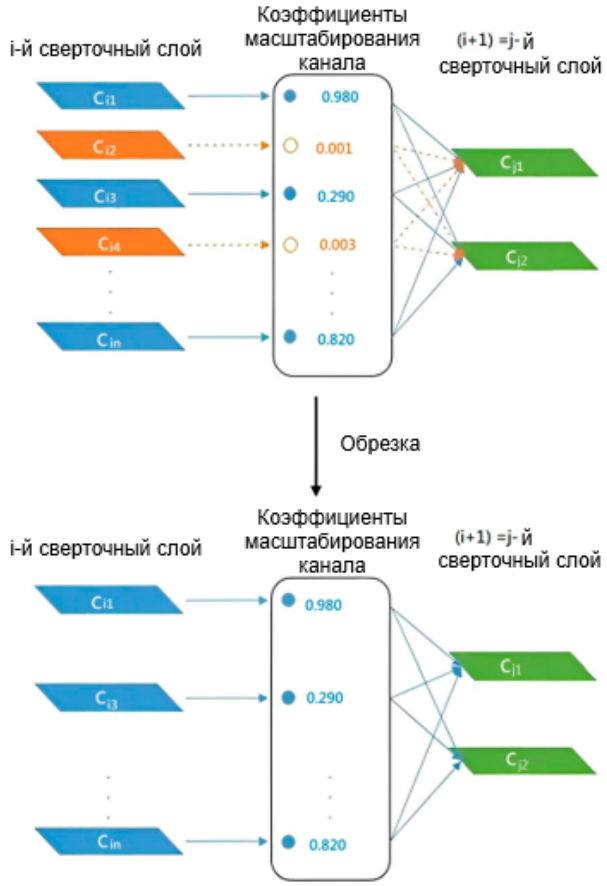

Рисунок 3. Метод сокращения.

C. Разреженное обучение

Мы добавляем слой пакетной нормализации (BN) после каждого сверточного слоя в YOLOv4, чтобы ускорить конвергенцию и улучшить обобщение. Слой BN использует пакетную статистику для нормализации сверточных объектов в виде:

$$y = \gamma \times \frac{x - \bar{x}}{\sqrt{\sigma^2 + \varepsilon}} \quad (1)$$

Здесь $\overline{X}$ и σ^2 представляют среднее значение и дисперсию входных характеристик в мини-пакете, соответственно. γ и β представляют масштабный коэффициент, поддающийся обучению, и смещение в слое BN. В этом исследовании мы непосредственно используем масштабный коэффициент в слое BN в качестве показателя важности канала. Чтобы эффективно различать важные и неважные каналы, мы применяем регуляризацию L1 к γ, обеспечивая разреженное обучение на уровне канала. Функция потерь для разреженного обучения представлена в виде:

$$L = loss_{yolo} + \alpha \sum_{\gamma \epsilon \Gamma} f(\gamma) \quad (2)$$

Функция f (γ) представляет собой норму L1, применяемую к γ, которая широко используется на этапе разрежения. α представляет собой штрафной коэффициент, который уравновешивает два условия потерь.

Эффективность сокращения зависит от разреженности модели. Ожидается, что до проведения разреженного обучения распределение γ в BN-слое YOLOv4 будет равномерным.

После разреженного обучения большинство значений γ в BN-слое будут сведены к нулю. Это дает два преимущества:

(1) Сокращение и сжатие сети для повышения эффективности модели: значения весов на уровне BN обычно используются для стандартизации и масштабирования каждой входной выборки в сети. Когда значения весов близки к нулю, соответствующие операции стандартизации и масштабирования сокращаются, что снижает вычислительную сложность.

(2) За счет сокращения весов уровня BN, близкого к нулю, становится возможным определить параметры, которые оказывают минимальное влияние на производительность сети, и сократить их.

D. Обрезка каналов

После завершения разреженного обучения можно выполнять обрезку каналов. Ниже приводится объяснение того, как приступить к обрезке каналов. Сначала вычисляется общее количество каналов в магистрали. Как только количество каналов определено, соответствующее значения γ сохраняются в переменной и сортируются в порядке возрастания. Следующим шагом является принятие решения о том, какие каналы сохранить, а какие обрезать. Этого можно достичь, установив коэффициент обрезки, который представляет собой долю каналов, подлежащих обрезке. Скорость обрезки обычно составляет значение от 0 до 1, где более высокое значение указывает на большую степень обрезки. Следуя этим шагам, можно выполнить процесс обрезки каналов для выборочного сохранения или удаления каналов в зависимости от заданной скорости обрезки.

E. Вырезание слоев

В основе YOLOv4 имеется несколько модулей CSPX, где каждый модуль CSPX состоит из трех слоев CBL и модулей X ResUnit. Результирующие функции
этих модулей объединены вместе, как показано на рисунке 4a. Для лазерной резки мы, в основном, обрезаем повторное соединение в YOLOv4. Архитектура результата показана на рисунке 4b, который состоит из двух слоев CBL и короткого соединения. Слой CBL содержит слой Conv, слой BN и функцию активации негерметичного ReLU, как показано на рисунке Рис. 4c. При обрезке слоев сначала сортируются средние значения γ для каждого слоя, и, оценивая предыдущий слой CBL каждого сокращения, можно выбрать минимальное значение для обрезки слоев. Чтобы обеспечить структурную целостность YOLOv4, при обрезке одного повторного соединения одновременно обрезаются как слой shortcut, так и предыдущий слой CBL, в результате чего в общей сложности обрезаются три слоя.

F. Точная настройка

Различные стратегии сокращения и пороговые настройки по-разному влияют на сокращенную модель. Иногда точность сокращенной модели может даже возрасти, хотя в большинстве случаев сокращение может отрицательно сказаться на точности модели. В таких случаях необходимо выполнить точную настройку сокращенной модели, чтобы компенсировать потерю точности, вызванную сокращением. Точная настройка имеет решающее значение для восстановления точности сокращенной модели. В наших экспериментах мы непосредственно переобучили Pruned-YOLOv4, используя тот же метод обучения гиперпараметры как обычный тренировочный процесс для YOLOv4.

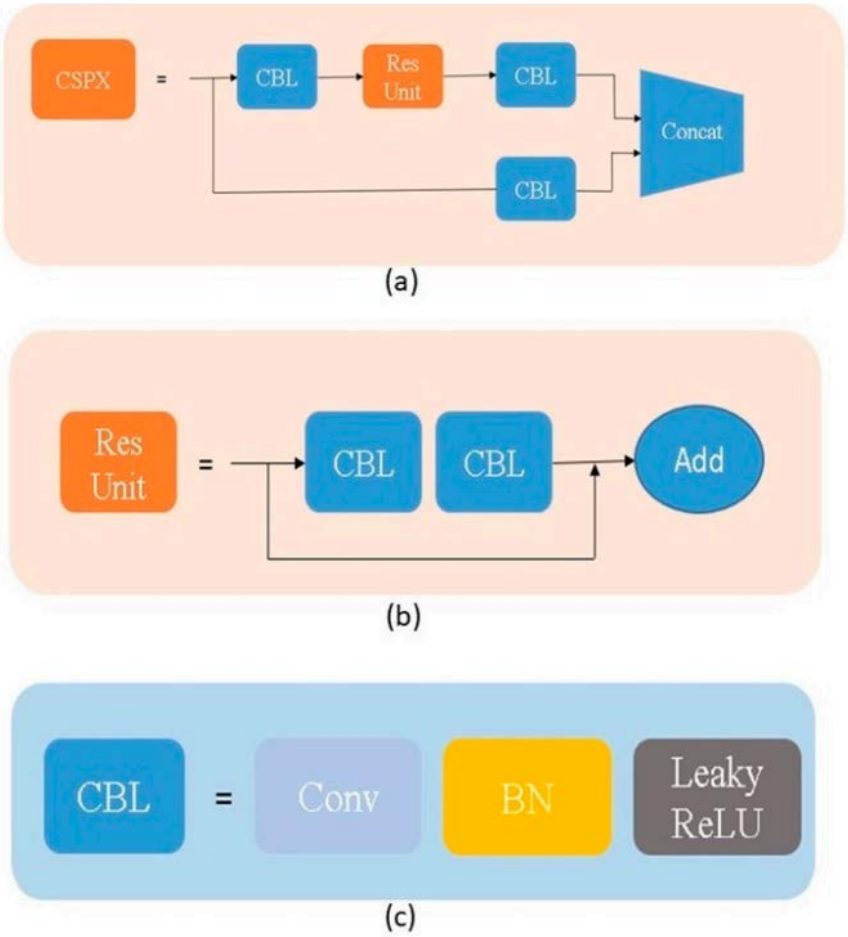

Рисунок 4. (а) CSPX, (б) модуль Res и (в) слой CBL в YOLOv4.

2.3. Отслеживание объектов и управление беспилотными летательными аппаратами.

Ноутбук выполняет обнаружение дронов в режиме реального времени, позволяя пользователям отслеживать обнаруженные цели до завершения отслеживания. Объекты, обнаруженные с помощью Pruned-YOLOv4, представлены ограничивающими рамками, каждая из которых содержит четыре координаты (x, y, w, h). Здесь x и y представляют координаты верхнего левого угла ограничивающего прямоугольника, в то время как w и h представляют ширину и высоту ограничивающего прямоугольника соответственно. Как только координаты получены, мы непрерывно определяем положение мыши пользователя. Если щелчок мыши попадает в ограничивающую рамку, четыре координаты ограничивающей рамки передаются в модуль отслеживания объекта, который использует SiamMask. SiamMask - это алгоритм отслеживания цели, основанный на нейронных сетях Siamese (Koch, et al., 2015, pp. 1-8). Сиамские нейронные сети были первоначально предложены Бромли и Лекуном для решения задач проверки сигнатур и с тех пор широко применяются в различных областях, таких как сопоставление изображений и отслеживание целей.

В задаче отслеживания цели сиамские нейронные сети используют две идентичные подсети с общими параметрами и весами. Шаблон отслеживания загружается в сеть, и получаются выходные значения весов. Затем эти значения сравниваются с выходными значениями области поиска для вычисления показателя сходства. Местоположение цели,

которую необходимо отследить, определяется путем вычисления карты оценки отклика. Основанная на традиционной сиамской сети, SiamMask включает в себя вычисление сегментации цели, что позволяет определить контур цели. Это помогает смягчить последствия изменений целевого объекта, вызванных вращением и деформацией.

При выполнении отслеживания SiamMask одновременно возвращает изображение с ограничивающей рамкой отслеживаемого объекта. Это ограничивающее поле содержит информацию о положении объекта на изображении. Ограничительная рамка отслеживаемого объекта состоит из четырех точек:(x_{min}, y_{min}), (x_{max}, y_{min}), (x_{min}, y_{max}), и (x_{max}, y_{max}).

Мы можем использовать эти четыре точки для вычисления центра положения объекта, который вычисляется как:

$$(x_{center}, y_{center}) = \frac{x_{min} + x_{max}}{2}, \frac{y_{min} + y_{max}}{2} \quad (3)$$

Для точного отслеживания объекта необходимо знать точное положение центра экрана дрона. Это связано с тем, что центр обнаруженного объекта всегда должен совпадать с центром экрана дрона для корректного отслеживания. Расчет разницы между центром экрана дрона и центром объекта выполняется следующим образом.

$$e_x imgx_{center} - x_{center} \quad (5)$$

$$e_y = imgy_{center} - y_{center} \quad (6)$$

Для эффективного отслеживания e_x и e_y всегда должны быть равны или близки к нулю. Дрон имеет четыре параметра управления: крен, рыскание, высота и тангаж. Крен-управляет боковым движением дрона, рыскание - вращением по часовой стрелке или против часовой стрелки, высота -вертикальным движением дрона, а тангаж - движением дрона вперёд или назад. На рисунке 5 показаны основные маневры дрона.

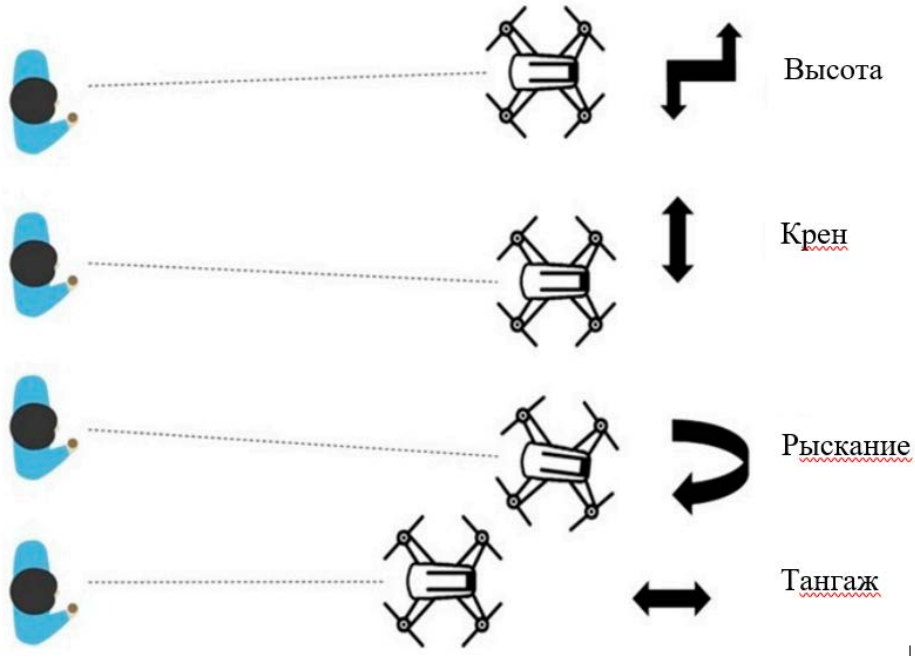

Рисунок 5. Основные маневры дрона.

Далее мы объясним, как ПИД-регулятор управляет полётом дрона. Очевидно, что, используя центральную точку отслеживаемого объекта и центральную точку экрана, мы можем получить погрешность по оси X. Эта погрешность связана с креном дрона при боковом движении и рысканием при вращении по часовой стрелке или против часовой стрелки. Если дрон обнаруживает, что объект движется влево или вправо, мы можем скорректировать курс дрона, чтобы развернуться к объекту, или выполнить боковые движения, чтобы не отставать от него. Кроме того, существует ось тангажа, которая включает движения вперёд и назад. Вычитая расстояние между дроном и реальным объектом из желаемого идеального расстояния, мы можем рассчитать погрешность расстояния и соответствующим образом управлять движением дрона вперёд или назад. Наконец, что касается высоты, вычитая координату Y отслеживаемого объекта из координаты Y центра экрана, мы можем получить погрешность по оси Y. Это позволяет нам рассчитать необходимую высоту для вертикального подъёма или спуска дрона. Конкретные методы контроля проиллюстрированы на рисунке 6.

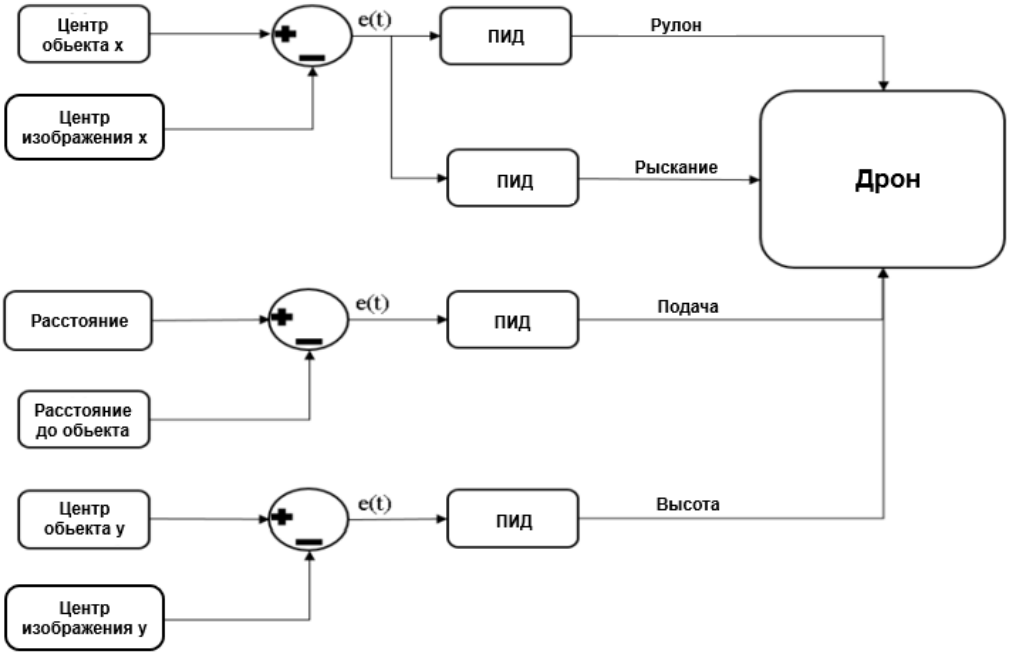

Рисунок 6. Схема управления дроном.

Что касается ошибки по оси X, выбор между креном и рысканием зависит от производной (D) ПИД-регулятора. D-член представляет собой скорость изменения ошибки. Если ошибка быстро меняется, дрону необходимо увеличить мощность, чтобы успевать за движением объекта. Однако, если объект продолжает двигаться вдоль оси X, как показано на рисунке 7a, для быстрого отслеживания цели требуется более сильное управление. Красный прямоугольник представляет собой ограничивающую рамку объекта. Поэтому мы выбираем вариант крена, что означает выполнение боковых движений вправо для следования за объектом, представленного зеленой стрелкой на рисунке 7a. С другой стороны, если отслеживаемый объект не демонстрирует значительного движения, выбирается вариант рыскания, который требует только корректировки курса дрона для следования за целью, представленного зеленой стрелкой на рисунке 7b.

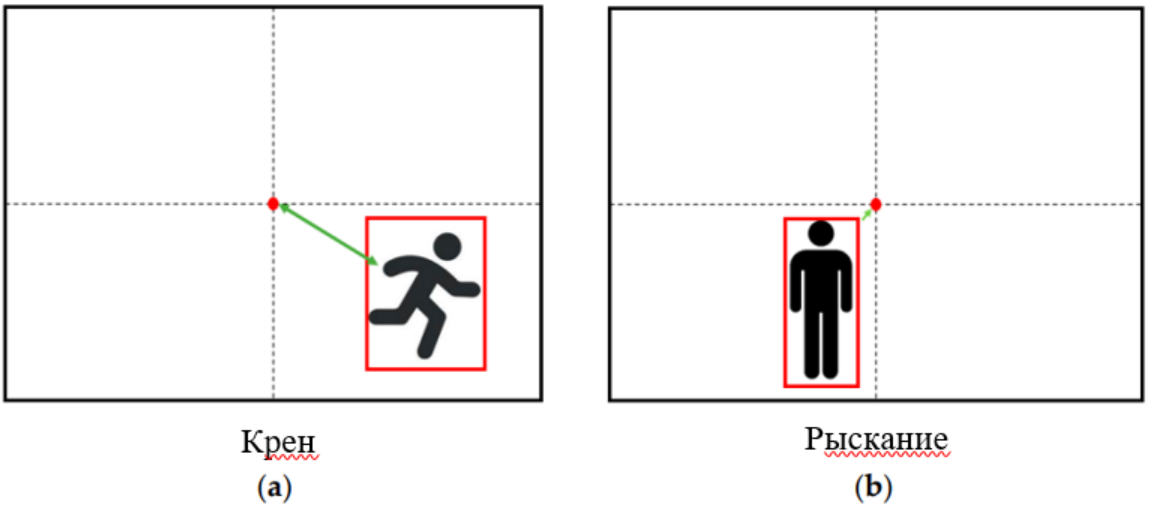

Рисунок 7. Примеры движения объекта (а) Крен (б) Рыскание.

Экспериментальные результаты

В этом разделе мы демонстрируем и анализируем эффективность обрезки модели обнаружения, обнаружения объектов, отслеживания и управления дронами. Отсечение модели обнаружения и обнаружение объектов. Мы обучили базовую модель YOLOv4, используя набор данных coco2014. Применив методы отсечения к базовой модели YOLOv4, мы получили облегченную модель, известную как Pruned-YOLOv4. В нашем сравнении мы рассматривали не только базовую модель YOLOv4, но и Tiny-YOLOv4. Tiny-версия YOLOv4 специально разработана как облегченный вариант для устройств с меньшими вычислительными ресурсами. Для оценки производительности детектора объектов мы использовали следующие четыре метрики:

1) Точность: Точность — широко используемая метрика в области обнаружения объектов. Она измеряет долю истинно положительных результатов среди всех обнаружений, выполненных системой. Более высокая точность указывает на то, что система может точно идентифицировать целевые объекты, снижая вероятность ложных срабатываний.

2) Полнота: Полнота — ещё одна часто используемая метрика в области обнаружения объектов. Она измеряет долю истинно положительных результатов среди всех фактически обнаруженных целевых объектов. Более высокая полнота указывает на то, что система может успешно обнаружить большую долю целевых объектов, снижая риск пропуска обнаружения.

3) Бифлопс: Бифлопс — метрика, используемая для измерения вычислительной эффективности компьютерной системы или модели машинного обучения. Бифлопс представляет собой количество миллиардов операций с плавающей запятой, необходимых для выполнения конкретной операции свертки. Суммируя бифлопс, затраченные на множественную свёртку и другие операции, можно количественно оценить сложность алгоритмической модели. Это широко используемая метрика для оценки вычислительной эффективности и скорости систем или моделей.

4) mAP@0,5 (средняя точность при IoU 0,5): mAP@0,5 — это широко используемая метрика оценки обнаружения объектов. Она измеряет среднюю точность при пороге пересечения по объединению (IoU) 0,5 для разных классов.

А. Базовое обучение в Darknet

Результаты производительности YOLOv4 и Tiny-YOLOv4 после базового обучения с использованием Darknet показаны в таблицах 1 и 2. Из таблицы 1 видно, что между двумя моделями существует значительная разница в точности модели. YOLOv4 достигает mAP@0.5, равного 0.749, в то время как Tiny-YOLOv4 достигает 0.55. Однако таблица 2 показывает, что хотя YOLOv4 достигает более высокой точности, для обучения модели требуется большее количество параметров. Размер обученного веса YOLOv4 в 10.6 раз больше, чем у YOLOv4-Tiny, а BFLOPs в 7.64 раза больше. Большой размер модели также приводит к более длительному времени вывода, что подчеркивает важность обрезки YOLOv4 в этом исследовании.

Таблица 1. Оценка базовой подготовки Darknet.

Модель	Точность	Полната	mAP@0.5
YOLOv4	0,818	0,605	0,749
Tiny-YOLOv4	0,65	0,41	0,55

Таблица 2. Размер параметра базового обучения Darknet

Модель	Параметры	Размер файла	Операции
YOLOv4	63,9M	250,2МБ	59,563
Tiny-YOLOv4	5,87M	23,6M	7,79

B. Результаты обучения с разрежением

Во время обучения с разрежением мы скорректировали баланс между двумя потерями, обозначенный как α. По мере обучения количество меньших масштабирующих факторов увеличивалось, в то время как количество больших масштабирующих факторов уменьшалось. Обучение с разрежением эффективно снижает масштабирующие факторы, что приводит к разреженности каналов в сверточных слоях YOLOv4. Это позволяет идентифицировать и отсеивать параметры с меньшим влиянием на производительность сети. Однако при использовании большего штрафного фактора (т. е. α = 0,01) для обучения с разрежением масштабирующие факторы уменьшаются слишком агрессивно, что приводит к недообучению и отказу модели. Это можно наблюдать по значительному падению полноты и mAP@0,5 в таблице 3, что указывает на то, что эта модель не подходит для отсечения. С другой стороны, использование очень малого штрафного коэффициента (например, α = 0,0001) приводит к недостаточной разреженности масштабирующих коэффициентов, что приводит к низкой производительности обрезки. Хотя эта модель демонстрирует отличную производительность в Таблице 3, становится сложно определить параметры, оказывающие меньшее влияние на производительность сети при последующих операциях обрезки. В наших экспериментах мы обучали модель YOLOv4 со штрафным коэффициентом α = 0,001 для обрезки каналов. Масштабирующие коэффициенты были сжаты практически до нуля, и производительность оставалась относительно стабильной, как показано в Таблице 3.

Таблица 3. Оценка штрафного коэффициента

Модель	Точность	Полната	mAP@0.5
α = 0,0001	0,822	0,664	0,75
α = 0,001	0,851	0,626	0,741
α = 0,01	0,981	0,059	0,26

C. Результаты обрезки каналов

В процессе обрезки каналов мы можем регулировать скорость обрезки, которая находится в диапазоне от 0 до 1. Более высокая скорость обрезки указывает на более высокий уровень обрезки. Из Таблицы 4 можно заметить, что при установке скорости обрезки на 0,74 наблюдается значительное снижение mAP@0,5 и полноты модели. Однако при установке скорости обрезки на 0,73 снижение mAP@0,5 и полноты не так значительно, как при скорости обрезки 0,74. Поэтому нам следует исследовать более узкий диапазон ниже скорости обрезки 0,73, чтобы найти оптимальное значение. Из Таблицы 5 можно сделать вывод, что более высокая скорость обрезки теоретически приводит к уменьшению параметров модели и увеличению скорости вывода. Если значения mAP@0,5 схожи при разных скоростях обрезки, предпочтительнее выбрать модель обрезки с более высокой скоростью, чтобы добиться более высокой скорости. Поэтому в конечном итоге мы выбрали скорость обрезки 0,735.

Таблица 4. Точность, полнота и mAP при различных скоростях обрезки.

Скорость обрезки	Точность	Полната	mAP@0.5
0,5	0,86	0,56	0,7423
0,6	0,865	0,559	0,741
0,7	0,865	0,559	0,741
0,71	0,87	0,55	0,742
0,72	0,883	0,515	0,728
0,73	0,935	0,266	0,649
0,735	0,94	0,21	0,51
0,74	0,92	0,00013	0,38

Таблица 5. Размер параметра и BFLOP при различных скоростях обрезки.

Скорость обрезки	Параметры	Размер файла	Операции
0,5	15,3M	58,8 МБ	28,564
0,6	9,134M	35,7 МБ	24,886
0,7	6,42M	25,2 МБ	21,922
0,71	6,25M	24,5 МБ	21,592
0,72	6,02M	23,6 МБ	21,270
0,73	5,7M	22,4 МБ	20,999
0,735	5,57M	21,8 МБ	20,858
0,74	5,4M	21,2 МБ	20,703

D. Результаты обрезки слоёв

В процессе обрезки слоёв мы можем определить количество ResUnits, подлежащих отсечению. Из Таблицы 6 видно, что при установке количества ResUnits равным 15 наблюдается значительное снижение mAP@0.5 и Recall модели. Однако при установке количества ResUnits равным 14, mAP@0.5 и Recall модели снижаются не так резко, как при 15 ResUnits. Кроме того, из Таблицы 7 можно сделать вывод, что чем больше единиц отсечено, тем теоретически меньше параметров модели и тем выше скорость вывода. Если значения mAP@0.5 одинаковы для разного количества ResUnits, предпочтительнее выбрать модель

обрезки с большим количеством единиц для достижения более высокой скорости. Поэтому в конечном итоге мы решили отсечь 14 ResUnits.

Таблица 6. Оценка точности обнаружения для сокращения числа ResUnits.

Числа Cut ResUnit	Точность	Полната	mAP@0.5
11	0,914	0,175	0,5336
12	0,91	0,132	0,494
13	0,91	0,131	0,483
14	0,912	0,116	0,463
15	0,929	0,0188	0,335

Таблица 7. Размеры параметров обрезки количества ResUnits.

Числа Cut ResUnit	Параметры	Размер файла	Операции
11	4,6 M	18 M	19,940
12	4,4 M	17,2 M	19,220
13	4,3 M	17,1 M	18,991
14	4,2 M	16,8 M	18,610
15	4,15 M	16,2 M	17,869

E. Финальная настройка модели

После обрезки модели, обрезка часто отрицательно влияет на точность модели. В таких случаях тонкая настройка сокращенной модели становится критически важной для восстановления потерянной точности. В нашем эксперименте мы напрямую использовали те же обучающие гиперпараметры и набор данных, что и при обычном обучении YOLOv4, чтобы переобучить модель Pruned-YOLOv4. Производительность после тонкой настройки показана в Таблице 8. По сравнению с базовым YOLOv4, есть небольшая потеря 0,013 в mAP@0.5. Однако из Таблицы 9 мы можем видеть, что Pruned-YOLOv4 достигает более высокой скорости вывода и значительно сокращает количество параметров модели и размер файла .weights. Это позволяет развернуть модель на встраиваемых устройствах с ограниченным объемом памяти. По сравнению с Tiny-YOLOv4 наблюдается значительное улучшение mAP@0.5, как показано в Таблице 8. Хотя и наблюдается небольшая потеря в BFLOP, как видно в Таблице 9, этот компромисс оправдан, поскольку приводит к более точной модели.

Таблица 8. Оценка различных моделей обнаружения

Модель	Точность	Полната	mAP@0.5
SSD	0,607	0,398	0,504
YOLOv3	0,661	0,436	0,579
YOLOv4	0,818	0,605	0,749
Tiny-YOLOv4	0,652	0,411	0,551
	0,784	0,619	0,736

Таблица 9. Размер параметра различных моделей обнаружения

Модель	Параметры	Размер файла	Операции
SSD	34,3 M	135,3 МБ	70,40
YOLOv3	61,6 M	242,9 МБ	65,86
YOLOv4	63,9 M	250,2 МБ	59,563

Tiny-YOLOv4	5,87 M	23,6 МБ	7,79
Pruned-YOLOv4	4,28 M	16,8 МБ	18,61

Отслеживание объектов и управление дронами

Цель исследования - изучить возможность и эффективность автоматизированного управления в реальных условиях. Для достижения этой цели мы разрабатываем серию экспериментов, имитирующих разведывательные задачи дронов в реальных условиях, и требуем, чтобы дроны успешно отслеживали целевые объекты автоматически. Для обеспечения достоверности экспериментальных результатов мы проводим эксперименты как в помещении, так и на открытом воздухе. Проводя эксперименты в этих местах, мы можем лучше оценить адаптивность и эффективность автоматизированного управления в различных реальных условиях. На рисунках 8 и 9 представлены выбранные сцены на открытом воздухе и в помещении из экспериментальных видео соответственно. В число отслеживаемых объектов входят десять разных людей. Каждый человек отслеживается в течение 50–90 секунд пять раз на открытом воздухе и в помещении. В процессе отслеживания случайным образом появляется от 0 до 7 других людей.

Рисунок 8. Избранные сцены на открытом воздухе

Рисунок 9. Избранные сцены в помещениях.

A. Параметры ПИД-регулятора.

После получения значений ошибок фактического состояния дрона можно определить маневры дрона в полете, позволяющие ему отслеживать цель. Для решения этой проблемы мы используем систему ПИД-регулирования. Важно отметить, что ПИД-регулятор формирует выходные сигналы по всем трём осям: x, y и z, представленным зелёной, красной и синей стрелками на рисунке 10. Таким образом, всего имеется девять параметров. Установка более высоких значений Kp позволяет контроллеру быстрее реагировать на ошибки управления. Следовательно, параметры Kp для всех трёх осей устанавливаются на максимальное значение. И наоборот, параметры Ki устанавливаются на минимальное значение, чтобы снизить влияние накопленных ошибок, тем самым избегая чрезмерной настройки или нестабильности системы. Что касается конфигурации параметров Kd, соответствующая настройка обеспечивает реакцию на скорость изменения ошибок, тем самым повышая скорость и стабильность реакции системы. Начальные значения для настроек параметров выбираются в соответствии с (https://github.com/murtazahassan/Drone-Face-Tracking 12.03.2023), после чего выполняется поиск по сетке для определения оптимальных значений.

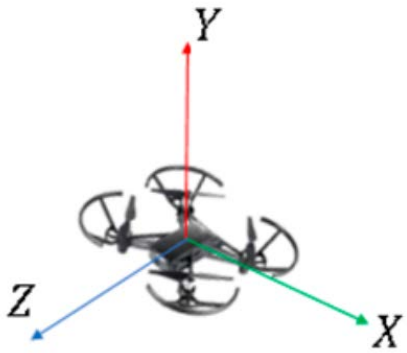

Рисунок 10. Трехосная ориентация дрона.

Б. Анализ ПИД-регулятора для дронов

В этом подразделе мы демонстрируем, как беспилотник непрерывно отслеживает целевой объект в полете и корректирует свои действия в полете по мере перемещения целевого объекта. Для демонстрации процессов отслеживания и управления выбраны два экспериментальных видеоролика. На видео 1 объект, за которым ведется наблюдение, движется по плоской поверхности, как показано на рисунке 11a. Четыре направления, в которых движется объект, обозначены красными, синими, желтыми и фиолетовыми стрелками на рисунке 11a. Реакция ПИД-регулятора на ошибку в положении отслеживаемого объекта по оси x на видео 1 показана на рисунке 11b. На рисунке 11b показано, что по мере перемещения целевого объекта погрешность увеличивается, и ПИД-регулятор быстро исправляет ошибку, сводя ее к нулю. Беспилотник непрерывно отслеживает целевой объект, в то время как ПИД-регулятор пытается уменьшить погрешность по оси x целевого объекта. Что касается погрешности по оси y, то, поскольку объект на видео 1 не претерпевает существенных изменений в росте, на рисунке 11c мы можем наблюдать, что погрешность по оси y существенно не меняется. Что касается расстояния в положении отслеживаемого объекта по оси z, то расстояние между дроном и целевым объектом фиксируется на исходном значении 150, что соответствует расстоянию в 1,5 м на земле между целевым объектом и дроном. Когда целевой объект перемещается вперед и назад, беспилотник должен непрерывно отслеживать целевой объект. На рисунке 11d показана реакция ПИД-регулятора на ошибку в положении отслеживаемого объекта по оси z в выбранном видео. Это демонстрирует, что погрешность изменяется в зависимости от расстояния между целевым объектом и дроном, и ПИД-регулятор пытается уменьшить погрешность по оси z целевого объекта.

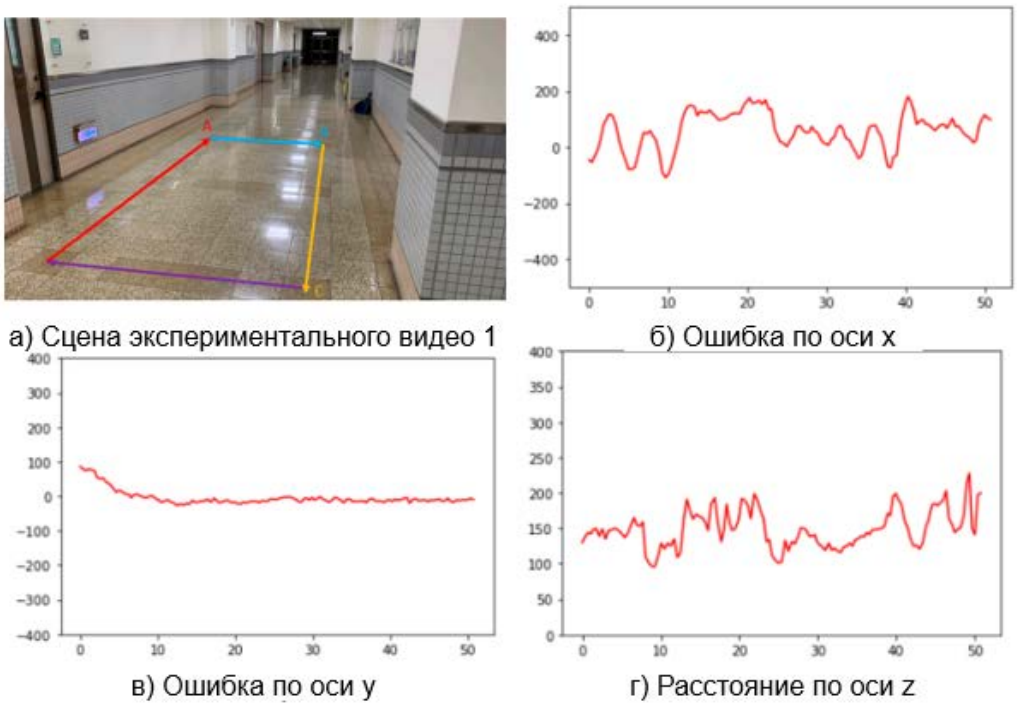

Рисунок 11. Реакция ПИД-регулятора на ошибки в разных направлениях для видео

На видео 2 отслеживаемый объект перемещается вверх и вниз по лестнице, как показано на рисунке 12a. Красная и синяя стрелки обозначают направления движения вверх и вниз по лестнице соответственно. На видео 2 беспилотник должен следовать за объектом и лететь прямо вверх или вниз по лестнице. Основное внимание уделяется проверке того, может ли беспилотник корректировать погрешность по оси y в режиме реального времени. На рисунках 12a–c показана реакция ПИД-регулятора на ошибку в позициях по осям x и y, а также на расстояние по оси z от отслеживаемого объекта на видео 2, соответственно. На рисунке 12 показано, что ПИД-регулятор непрерывно регулирует погрешности по осям x и y, приближая их к нулю, по мере того как объект перемещается вперед и назад при подъеме или спуске по лестнице.

C. Оценка точности отслеживания

Средние абсолютные погрешности между целью и центром кадра для отслеживания приведены в таблице 10. Погрешности по оси x немного превышают погрешности по оси y, поскольку в большинстве экспериментальных видеороликов объекты изменяют направление своего движения. Когда отслеживаемые объекты перемещаются по поверхности земли без подъема или спуска по лестнице, погрешности по оси y близки к нулю. Погрешности отслеживания на открытом воздухе выше, чем в помещении, из-за влияния ветра.

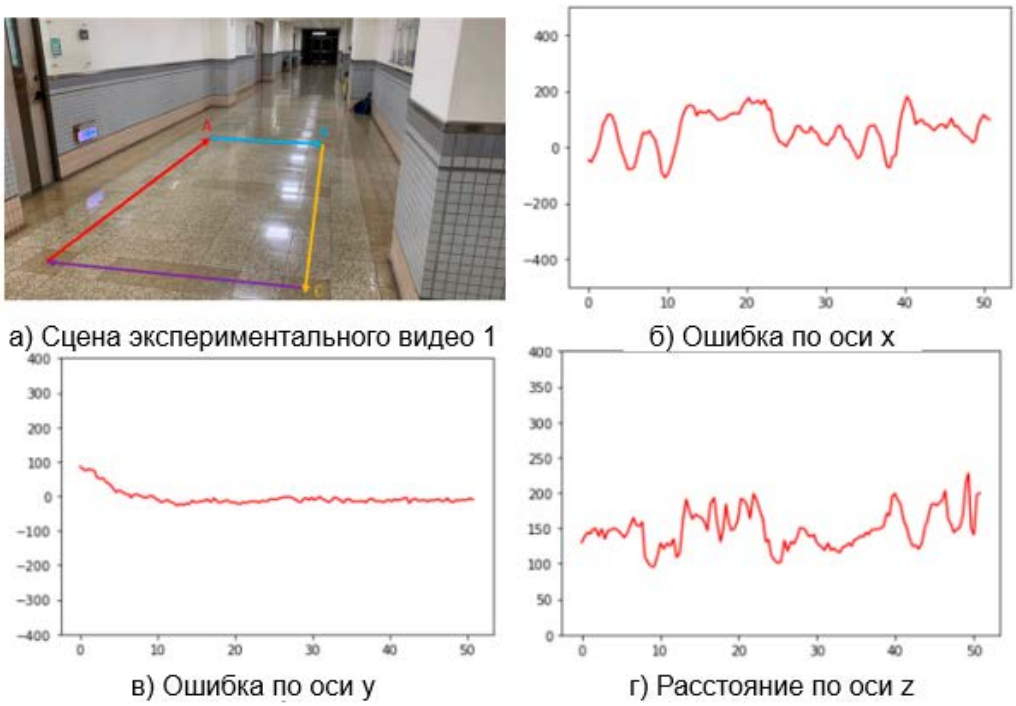

Рисунок 12. Реакция ПИД-регулятора на ошибки в разных направлениях для видео 2.

Таблица 10. Средняя абсолютная ошибка отслеживания

Модель	В помещении	На открытом воздухе
По оси X	128,95	137,62
По оси Y	98,73	116,98

Обсуждение с балансировать размер параметров модели и точность модели в процессе сокращения является сложной задачей. Анализируя точность, отзыв и отображение, а также размеры параметров при различных скоростях обрезки, мы можем определить подходящую скорость обрезки, чтобы сбалансировать компромиссы. Кроме того, тонкая настройка модели обрезки полезна для восстановления и повышения точности модели. Процесс ПИД-регулирования, который постоянно минимизирует погрешность между отслеживаемым объектом и центром кадра, позволяет выполнить задачу автоматического управления дроном и поддерживать стабильную траекторию полета в режиме реального времени. Это объясняется способностью автоматической системы управления оперативно подстраиваться под положение и перемещение целевого объекта, чтобы свести к минимуму погрешность между исходным положением и фактическим положением. Это позволяет беспилотнику точно отслеживать целевые объекты и быстро реагировать на изменения. Автоматизированное управление имеет огромное значение для взаимодействия человека и машины. Это расширяет когнитивные возможности людей-операторов. В то же время автоматизированное управление обеспечивает высокую эффективность и стабильность работы. Исходя из приведенных выше наблюдений и анализа, автоматическое управление дроном имеет очевидные преимущества. Оно может обеспечить точное и стабильное отслеживание с учетом эффективности автоматизированного управления. Эта модель сотрудничества позволяет исследователям и операторам участвовать в полетах беспилотных летательных аппаратов и использовать свой соответствующий опыт, одновременно повышая эффективность с помощью систем автоматического управления.

Выводы: В этой статье мы предлагаем метод реализации системы обнаружения объектов и сопровождения целей на основе операционной системы робота (ROS) и применяем его к беспилотнику Tello. Система обеспечивает эффективное обнаружение объектов и сопровождение целей в режиме реального времени. Мы используем сокращенную архитектуру YOLOv4 в качестве модели обнаружения и выбираем SiamMask в качестве модели отслеживания. Кроме того, мы внедряем модуль PID для расчета ошибок и определения направления полета и действий. Для модуля обнаружения мы выбрали усовершенствованную архитектуру YOLOv4, которая обеспечивает более высокую скорость выполнения при сохранении точности обнаружения. Сокращая избыточные параметры и вычисления в модели, мы достигаем упрощенной и ускоренной производительности. Это позволяет нашей системе эффективно выполнять задачи по обнаружению объектов в режиме реального времени. В качестве модуля отслеживания мы используем модель SiamMask. SiamMask – это метод отслеживания одиночных объектов, позволяющий отслеживать цели в режиме реального времени. В нашей системе SiamMask используется для отслеживания объектов, обнаруженных YOLOv4, что обеспечивает непрерывное отслеживание объектов и их позиционирование. Кроме того, мы внедрили модуль PID для расчета ошибок и корректировки траектории полета. PID – это классический алгоритм управления, который вычисляет управляющие сигналы на основе текущей ошибки, накопленной ошибки и скорости изменения ошибок, чтобы приблизить выходные данные системы к желаемому значению. В нашей системе модуль PID вычисляет ошибки на основе положения целевого объекта и текущего состояния дрона и корректирует сигналы управления ориентацией дрона для стабильного отслеживания целевого объекта. В ходе летных экспериментов мы подтверждаем возможность применения этой системы в повседневной жизни. Усовершенствованная модель YOLOv4 обеспечивает эффективные возможности обнаружения объектов, позволяя быстро обнаруживать цели в режиме реального времени. SiamMask используется для отслеживания целевого объекта, а модуль PID точно вычисляет ошибки и адаптируется к различным ситуациям полета, позволяя беспилотнику стабильно отслеживать целевой объект.

Список литературы

1. Drone-Face-Tracking. Available at: https://github.com/murtazahassan/Drone-Face-Tracking (accessed on 12 March 2023).
2. Girshick, R., Donahue, J., Darrell, T., Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2014), Columbus, OH, USA, pp. 580–587.
3. He, Y., Zhang, X., Sun, J. (2017). Channel pruning for accelerating very deep neural networks. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Venice, Italy, pp. 1398–1406.
4. Koch, G., Zemel, R., Salakhutdinov, R. (2015). Siamese neural networks for one-shot image recognition. In Proceedings of the International Conference on Machine Learning Deep Learning Workshop (ICML 2015), Lille, France, pp. 1–8.
5. Li, Q., Li, H., Meng, L. (2023). Deep learning architecture improvement based on dynamic pruning and layer fusion. Electronics, №12, p. 1208.
6. Pruned-OpenVINO-YOLO. TNTWEN. Available at: https://github.com/TNTWEN/Pruned-OpenVINO-YOLO (accessed on 10 May 2023).

7. Quigley, M., Gerkey, B., Conley, K., Faust, J., Foote, T., Leibs, J., Ng, A. (2009). ROS: An open-source robot operating system. In Proceedings of the IEEE International Conference on Robotics and Automation, Workshop on OpenSource Software (ICRA 2009), Kobe, Japan, pp. 1–6.
8. Redmon, J., Divvala, S., Girshick, R., Farhadi, A. (2016). You only look once: Unified, real-time object detection. In Proceedings of the 29th IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2016), Las Vegas, NV, USA, pp. 779–788.
9. Redmon, J., Farhadi, A. (2017). YOLO9000: Better, faster, stronger. In Proceedings of the 30th IEEE International Conference on Computer Vision (CVPR 2017), Venice, Italy, pp. 6517–6525.
10. Tello Edu. Ryze Robotics. Available at: https://www.ryzerobotics.com/tello-edu (accessed on 18 November 2023).
11. YOLOv4 Baseline Training. Available at: https://github.com/AlexeyAB/Darknet (accessed on 1 June 2023).
12. Zhu, Z., Wang, Q., Li, B., Wu, W., Yan, J., Hu, W. (2018). Distractor-aware siamese networks for visual object tracking. In Proceedings of the European Conference on Computer Vision (ECCV 2018), Munich, Germany, pp. 101–117.