УДК 004.93

Автономная навигация мобильного робота

И. В. Савицкая, А. П. Семёнова
Донецкий национальный технический университет
i.v.savitskaya@mail.ru, nastena-semenova19@rambler.ru

Аннотация
В работе представлено исследование и практическая реализация алгоритмов компьютерного зрения для автономного прохождения трассы роботом в рамках первого Кубка России по спортивному программированию в дисциплине "программирование робототехники". Основное внимание уделено разработке алгоритма навигации колесного робота по сложной трассе с различными типами покрытий (трава, лед, земля) и геометрическими особенностями (S-образные участки, "змейки").

Введение

В этом году Кубок России по спортивному программированию в дисциплине «программирование робототехники» проводился впервые. Отборочный этап соревнования проходил в формате онлайн на платформе симулятора «1Т МИР». Необходимо было создать решение на основе машинного зрения и нейронных сетей, обеспечивающее автономное прохождение роботизированной машиной трассы с различными препятствиями и типами покрытий.

Компания «1Т» более 20 лет выполняет ИТ-проекты для крупных государственных и бизнес-заказчиков, развивает образовательное направление и создаёт собственные цифровые продукты, включая симулятор «1Т МИР», функционирующий на импортонезависимом программном движке. С 2023 года на базе платформы «1Т МИР» прошли обучение свыше 10 000 слушателей, а также организованы масштабные соревнования, включая чемпионат «Новая высота» на площадке Калужского Технопарка профессионального образования и «Фестиваль пилотирования БЛА» в партнёрстве с УГТУ.

Для достижения цели проекта необходимо:

1. Изучить существующие полигоны и особенности их прохождения.
2. Выполнить анализ предоставленного организаторами полигона.
3. Проанализировать существующие алгоритмы автономного ориентирования мобильного робота.

Разработать ПО для автоматического прохождения трассы.

Анализ полигонов

На соревнованиях по робототехнике предлагаются полигоны (трассы) различных размеров и форм [1]:

– линейные трассы – представляют собой замкнутые или разомкнутые линии с поворотами и различными участками, по которым должен перемещаться робот;

– игровые поля – специальные площадки для тренировок и соревнований, где роботы выполняют конкретные задания;

– городские среды – имитируют дорожную инфраструктуру: движение по улицам, пересечение перекрёстков, реакцию на внешние факторы;

– специализированные сценарии – моделируют уникальные условия (например, зоопарк, ферму, космическое пространство);

– лабиринты – полигоны с трёхмерными бортиками, ограничивающими траекторию движения робота;

– лестница – робот должен пройти замкнутый путь по специальной лестнице;

– слалом по линии – задача: пройти s-образную трассу (чёрная линия) за минимальное время, избегая кеглей-препятствий;

– кегельринг и сумо – робот должен за наиболее короткое время вытолкнуть кегли или противника за пределы круга (ринга);

– дорога – робот должен преодолеть маршрут без столкновений с другими роботами-участниками.

Ключевой фактор для робота – не только ширина, но и форма линии. Среди линейных трасс выделяют [2]:

– «Simple» (см. рис. 1а) – простая трасса для демонстрации базовых алгоритмов;

– «Гантеля» (см. рис. 1б) – универсальное поле для тренировок и соревнований разного уровня;

– Полигон Политехнического музея (см. рис. 1в) – позволяет развивать высокие скорости (свыше 1 м/с);

– «Зародыш» (см. рис. 1г) – трасса эмбрионоподобной формы с поворотами малого радиуса;

– «Истукан» (см. рис. 1д) – содержит сложные участки, включая «прямые змейки».

Наиболее сложными для прохождения являются S-образные участки трассы («змейки») и «Прямые змейки» (например, на полигоне «Истукан»). При определённых углах риск потери трассы резко возрастает, а некоторые роботы на определённых скоростях неспособны их преодолеть [3].

Обзор платформы для соревнований

Стартовая точка при загрузке симулятора [4] имеет вид, представленный на рисунке 2. Управление роботом осуществляется в двух режимах: ручном и автоматическом. Вид управления указан в левом верхнем углу.

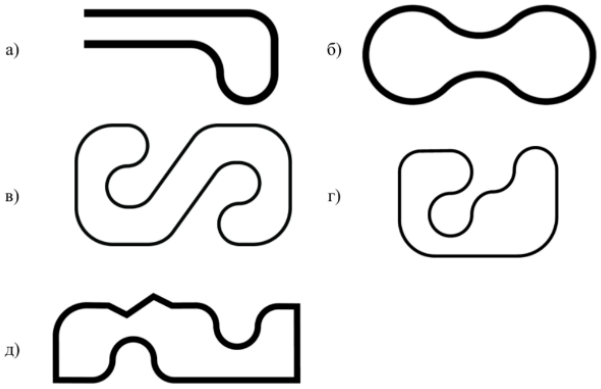


Рисунок 1 – Виды линейных трасс

Рисунок 2 – Режим автоматического управления роботом

Переключение между режимами происходит при нажатии клавиши «T». Перемещение машины на стартовую позицию осуществляется клавишей «R». Данная подсказка размещена в левом нижнем углу. По центру в верхней части экрана через наклонную черту отображается два показателя fps – количество кадров в секунду и fp – интенсивность светового потока. А в правом нижнем углу указано состояние подключения робота к автоматическому управлению.

При прохождении трассы в ручном режиме для изменения доступны множество параметров робота, появляющихся в правой части экрана при нажатии на клавишу «Q» (см. рис. 3). Приведение в движение роботизированной тележки происходит клавишами WASD. Кроме вышеперечисленных параметров в ручном режиме доступны две кнопки: «Скачать снимки с камер» и «Скачать autopilot.zip». Содержимое файлов при скачивании снимков с камер представлено на рисунке 4 и содержит изображение робота с 5 статичных камер и камеры, следящей за машинкой.

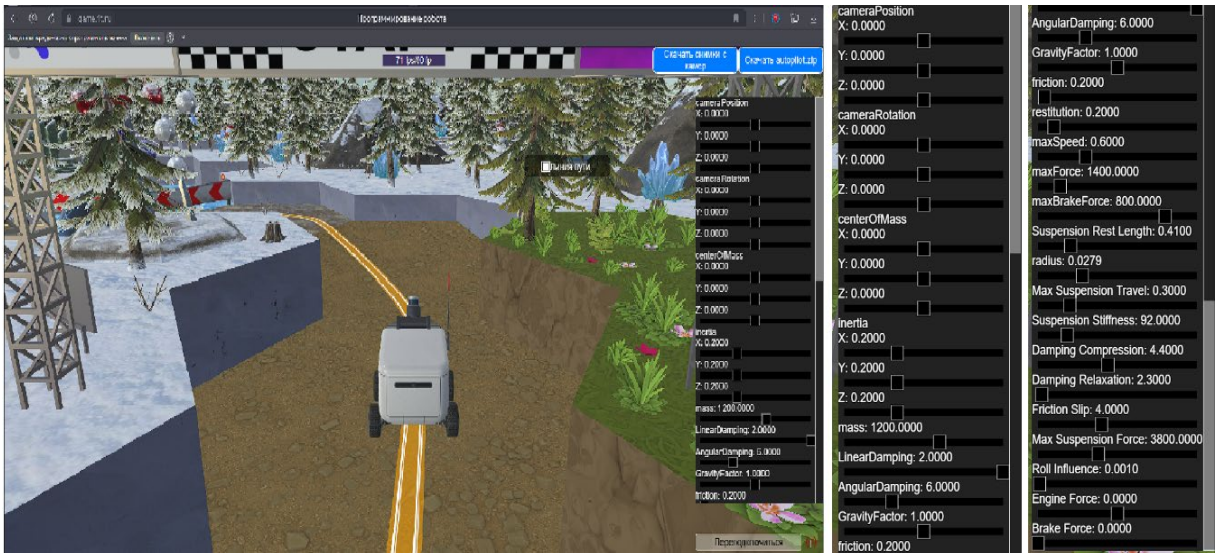


Рисунок 3 – Доступные для изменения параметры робота

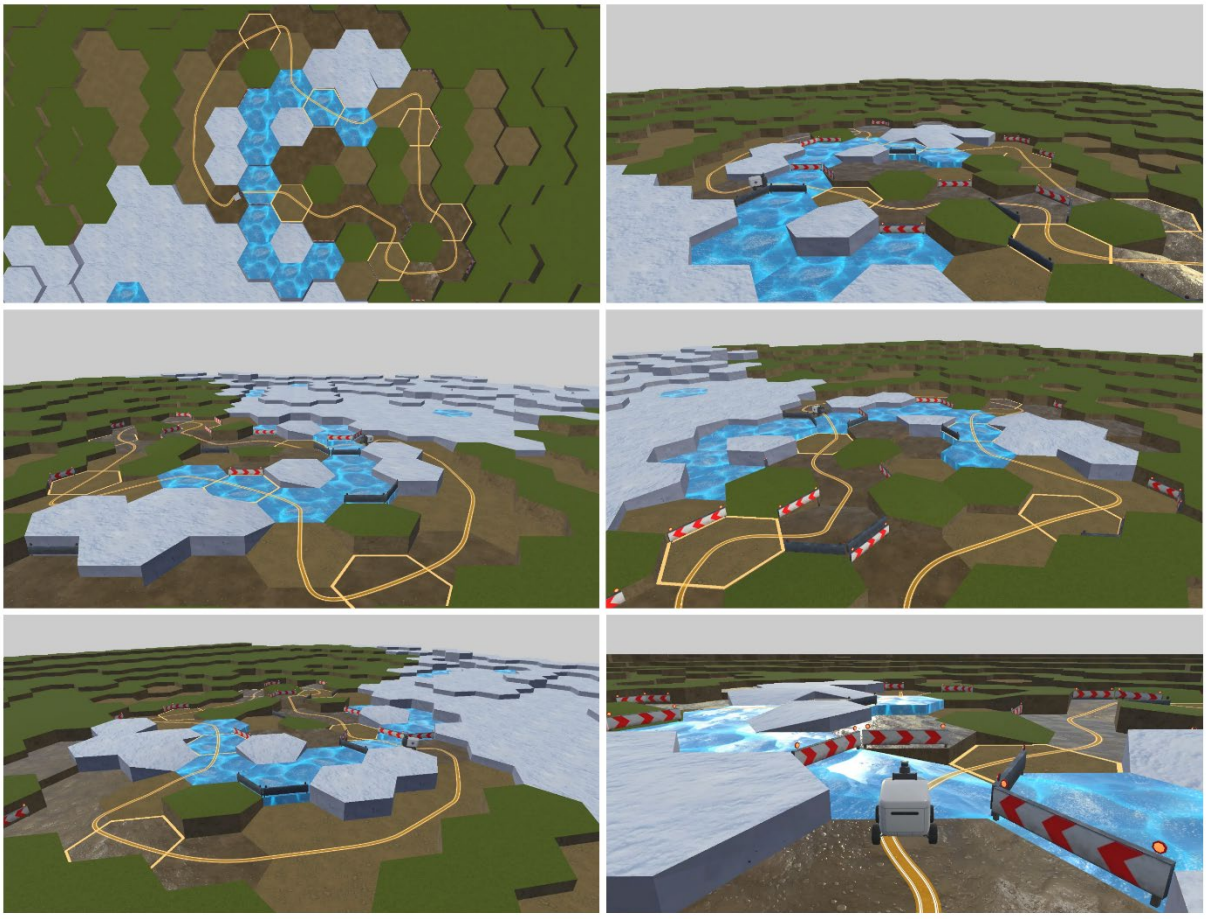


Рисунок 4 – Доступные снимки с камер

Обзор программы

При нажатии на кнопку «Скачать autopilot.zip» происходит загрузка проекта на языке python, реализующего примеры применения управляющих роботом команд. После распаковки архива autopilot.zip появляются 4 папки с функциями управления роботом и файлы. Содержание папок с файлами:

– algorithm – модули алгоритма работы робота;

– connection – модули, реализующие подключение к роботу;

– services – модули для создания log-файла;

– vehicle – модули непосредственного управления машинкой.

Отдельные файлы:

– requirements.txt – установка зависимостей;

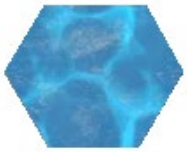
– readme.md – инструкция;

– config.py – модуль конфигурации;

– main.py – основная программа для запуска автоматического управления роботом.

Задача, стоявшая перед участниками соревнований, заключалась в том, чтобы роботизированная машинка проехала всю трассу без съезда с линии и посетила все контрольные точки (так называемые чек-поинты). Вид трассы с камер представлен на рисунке 4. Если внимательно рассмотреть виды с разных камер, то можно заметить, что трасса представляла собой шестиугольные ячейки в виде сот. Кроме того, ячейки могли располагаться под различными углами, имитирующими горку. Ячейки с ярко желтой обводкой представляли собой контрольные точки. На каждую ячейку-соту была наложена текстура. Всего на трассе представлено 5 текстур (таблица 1).

Таблица 1 - Виды текстур

Внешний вид	Описание
	зеленая трава; проблем в движении «тележки» нет
	сине-голубой лед; возможен занос «тележки» на поворотах
	темно коричневая земля, покрытая льдом; у «тележки» возникают проблемы с подъемом на горку
	белый снег; проблем в движении «тележки» нет, т.к. частью трассы не является
	светло коричневая земля; проблем в движении «тележки» нет

Непосредственное управление движущими элементами машинки-робота в автоматическом режиме осуществляется в файле vehicle_control.py, находящемся в папке vehicle. Организаторами Кубка было предоставлено всего две управляющие команды:

1. setMotorPower(right, left) – подает управляющие сигналы на сервоприводы колес и имеет два параметра. Первый параметр – мощность вращения правых колес в процентах, второй – мощность вращения левых колес в процентах. Если значение фактического параметра положительное, то колесо вращается по часовой стрелке, если отрицательное – то против часовой стрелки.
2. rotate(angle) – метод поворота робота относительно текущего положения. Отрицательные значения параметра angle будут поворачивать робота по часовой стрелке, положительные – против часовой стрелки.

Анализ современных алгоритмов компьютерного зрения

Компьютерное зрение является междисциплинарной областью исследований, объединяющую методы цифровой обработки изображений, машинного обучения и искусственного интеллекта. Оно представляет собой ключевую технологию современной цифровой трансформации, находящую широкое применение в таких стратегически важных областях, как робототехника, промышленная автоматизация, медицинская диагностика и интеллектуальные системы безопасности. Его внедрение позволяет существенно расширить функциональные возможности автоматизированных систем за счет реализации сложных визуально-аналитических функций.

Основная задача компьютерного зрения заключается в разработке алгоритмов автоматического извлечения и интерпретации значимой информации из визуальных данных. Фундаментальные задачи включают:

– распознавание и классификацию объектов в статических изображениях и видеопотоках;

– детектирование и точную пространственную локализацию объектов интереса;

– семантическую и инстанс-сегментацию изображений;

– трекинг объектов в динамических сценах;

– оценку пространственной ориентации и положения объектов;

– оптическое распознавание текстовой информации;

– анализ и интерпретацию динамики сцены.

Особое значение в рамках компьютерного зрения занимают задачи трехмерной реконструкции сцены по двумерным изображениям и семантической интерпретации визуального контента. Решение этих задач требует комплексного подхода, сочетающего

методы цифровой обработки изображений, машинного обучения и вычислительной геометрии. Современные алгоритмы позволяют не только идентифицировать объекты, но и восстанавливать их пространственные характеристики, анализировать взаимное расположение и динамику изменений, что открывает новые перспективы для создания интеллектуальных систем принятия решений.

Современная система компьютерного зрения включает в себя несколько ключевых компонентов:

1. Оптическая подсистема – состоит из одной или нескольких видеокамер, системы освещения и оптических фильтров.
2. Аппаратная платформа обработки данных – включает центральный процессор (CPU), графический процессор (GPU), специализированные ускорители (FPGA, ASIC).
3. Программное обеспечение – состоит из алгоритмов предварительной обработки и методов анализа изображений, решающих правил.
4. Интерфейс передачи данных – включает проводные и беспроводные каналы связи и протоколы обмена данными.

В рамках решения задачи визуальной идентификации объектов существует множество различных подходов. Применительно к конкретной задаче следования по линии, ключевым аспектом становится надежное выделение целевой линии на контрастном фоне, где может использоваться как классическая схема черной линии на белом фоне, так и инверсный вариант с белой линией на черном фоне.

Типичный алгоритм компьютерного зрения для решения подобных задач предполагает последовательную обработку изображения через несколько взаимосвязанных этапов (см. рис. 2).

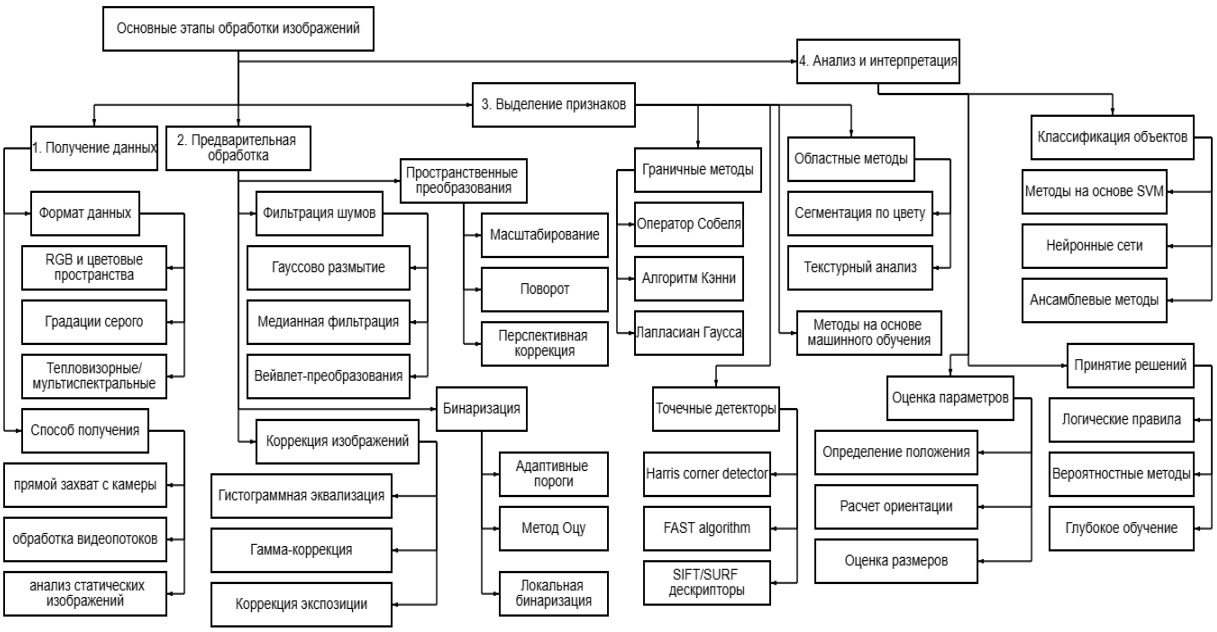


Рисунок 5 – Основные этапы обработки изображений

Первоначально система получает исходное изображение, после чего выполняет его предварительную обработку для улучшения качества и выделения значимых признаков. Затем следует этап детектирования ключевых элементов изображения, таких как границы, линии или характерные точки. На заключительной стадии происходит анализ и интерпретация выделенных признаков для принятия решенияКаждый из этих этапов может реализовываться различными методами в зависимости от конкретных условий задачи, характеристик оборудования и требований к точности и быстродействию системы. Важно отметить, что эффективность работы алгоритма в значительной степени зависит от корректного выбора и настройки методов обработки на каждом этапе, а также от их согласованного взаимодействия в рамках единого конвейера обработки изображений.

Предлагаемое решение

В качестве средств реализации выбран язык Python и библиотека OpenCV. Python в сочетании с OpenCV представляет собой оптимальное решение для обработки изображений, обеспечивая высокую производительность за счёт C++-ядра и удобство разработки благодаря лаконичному синтаксису Python и интеграции с NumPy/SciPy. OpenCV

поддерживает широкий спектр методов - от классических алгоритмов фильтрации и детектирования до современных нейросетевых моделей (YOLO, SSD) через DNN-модуль, сохраняя кроссплатформенность (Windows/Linux/macOS, CPU/GPU) и масштабируемость (OpenCL/CUDA). Ключевыми преимуществами являются быстрое прототипирование, обширная документация, активное сообщество и регулярные обновления, что делает эту связку универсальным инструментом как для исследований, так и для промышленных решений. Таким образом, OpenCV предоставляет оптимальный баланс между производительностью, удобством разработки и доступностью ресурсов для обучения.

Робот определяет направление движения ориентируясь на снимки с камер. Всего предоставлено 5 камер стационарных и 1 следующая за роботом. Для автономной работы робота необходимо, чтобы на используемом изображение осталась видна только линия пути. Рассмотрим алгоритм детектирования линии трассы с использованием OpenCV:

Шаг 1. Сбор входные данных. На вход алгоритма подается изображение с камеры (цветное). Также учитываются параметры линии, такие как цвет, ширина и контрастность относительно фона.

Шаг 2. Предварительная обработка изображения. Изображение конвертируется в цветовое пространство HSV с помощью cvtColor (images, cv2.COLOR_BGR2HSV). Применяется фильтр-маска для отсечения цветов окружающего пространства. Для фильтрации изображения используется функция inRange (hsv_ images, lower_line, upper_line). Первый параметр – изменяемое изображение, а второй и третий – левая и правая граница пропускаемого цвета.

Затем переводим полученное изображение в градации серого для лучшего выделения линии с помощью cvtColor(img, cv2.COLOR_BGR2GRAY). Далее выполняется бинаризация изображения с помощью threshold(gray_images, 1, 255, cv2.THRESH_BINARY), после чего применяется размытие Гаусса для уменьшения шумов и артефактов изображения GaussianBlur(result_line, (3, 3), 0).

Шаг 3. Выделение контуров линии. С помощью функции поиска контуров findContours(blurred, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE) находятся все контуры на бинаризованном изображении. Для устранения мелких шумов выполняется фильтрация контуров по минимальной площади, что позволяет оставить только значимые элементы, соответствующие линии трассы.

Шаг 4. Определение центральной линии. Из оставшихся контуров выбираем наибольший по площади, который считается основной линией трассы с помощью функции max(filtered_contours, key=cv2.contourArea). Для этого контура выполняется аппроксимация для упрощения его формы arcLength(largest_contour, True), строится ограничивающий прямоугольник минимальной площади и вычисляется центр масс контура, который принимается за центральную точку линии.

Шаг 5. Определение угла отклонения. На основе положения центра линии относительно центра робота вычисляется угол отклонения «тежки» от трассы. Этот угол используется в системе управления: если отклонение превышает пороговое значение в 5 градусов вправо или влево, роботу подается команда на соответствующий поворот, в противном случае - команда движения прямо.

Шаг 6. Визуализация (опционально). Для отладки и демонстрации работы алгоритма может выполняться визуализация: на исходное изображение наносятся контуры обнаруженной линии и маркер центра робота. Результат отображается в отдельном окне до нажатия клавиши (см. рис. 6).

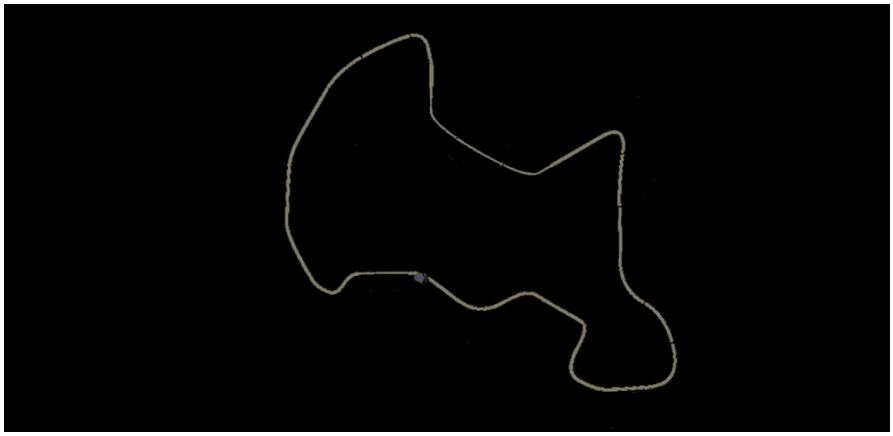


Рисунок 6 – Обработанный рисунок

Заключение

Проведенное исследование и практическая реализация алгоритмов компьютерного зрения для автономного прохождения трассы роботом в рамках Кубка России по спортивному программированию позволило убедиться в эффективности выбранного подхода. Использование связки Python + OpenCV доказало свою эффективность для задач автономной навигации. Предложенный алгоритм детектирования линии трассы, основанный на цветовой сегментации, бинаризации и контурном анализе, обеспечил устойчивость к различным типам покрытий (трава, лед, снег), точное определение центра линии даже при сложной геометрии трассы (S-образные участки, "змейки"), реальное время обработки (~15-20 FPS).

Литература

1. Трассы соревнований. [Электронный ресурс]. – Режим доступа: https://myrobot.ru/sport/index.php?n=Reglaments.Fields

2. Мартыненко Ю.Г. Динамика мобильных роботов. // Соросовский образовательный журнал. – 2000. – Т.6. – №5. – С. 110-116.

3. Мартыненко, Ю.Г. Управление движением мобильных колесных роботов // Фундаментальная и прикладная математика. - 2005. – №8. – С. 29-80.

4. Ларкин Е.В. Трассировка движения мобильного робота по пересеченной местности / Е.В. Ларкин, Ву Зуй Нгхиа // Известия ТулГУ. Технические науки. – 2012. – №8. – С.257-260.

5. Рядчиков И.В. Создание робота автономного движения по линии / И.В. Рядчиков, С.Г. Синица, Б.О. Брагин [и др.]. // Технические науки: проблемы и перспективы : материалы III Междунар. науч. конф. (г. Санкт-Петербург, июль 2015 г.). – Санкт-Петербург : Свое издательство, 2015. — С. 19-25.

6. Лазарев М.В. Управление движением мобильного робота по различным трассам на основе принципа редукции теории нечеткости / М.В. Лазарев, В.Г. Ежкова // Вестник Белгородского государственного технологического университета им. В.Г. Шухова. – 2024. – №.1. – С. 95-103.

7. Орехов С.Ю. Исследование методики планирования движений мобильного робота в известной среде / С.Ю. Орехов, Р.А. Багдошвили, В.О. Копылов // StudNet. – 2022. – №6. – С.7073-7084.

8. Власов, С.М., Бойков В.И., Быстров С.В., Григорьев В.В. Бесконтактные средства локальной ориентации роботов. – СПб: Университет ИТМО, 2017. – 169с.

9. Чулкова, А.С. Разработка алгоритма автономного движения робота в условиях смоделированной городской среды / А.С. Чулкова, Я.А. Карташов, А.А. Шарапов // Сборник материалов Международной научной конференции «Интерэкспо ГЕО-Сибирь 2024». – Т.7. – №2. – *С. 208-212*

10. Шапиро, Л. Компьютерное зрение / Л. Шапиро, Д. Стокман. – 5-е изд. – Москва: Лаборатория знаний, 2024. – 762 с.

11. Компьютерное зрение с OpenCV и Python: практическое руководство [Электронный ресурс]. – Режим доступа: https://www.litres.ru/book/inzhener-33334081/komputernoe-zrenie-s-opencv-i-python-prakticheskoe-rukov-71881402/

Савицкая И.В., Семёнова А.П. Автономная навигация мобильного робота. *В работе представлено исследование и практическая реализация алгоритмов компьютерного зрения для автономного прохождения трассы роботом в рамках первого Кубка России по спортивному программированию в дисциплине "программирование робототехники". Основное внимание уделено разработке алгоритма навигации колесного робота по сложной трассе с различными типами покрытий (трава, лед, земля) и геометрическими особенностями (S-образные участки, "змейки").*

Ключевые слова: *робототехника, соревнования, OpenCV, компьютерное зрение, обработка изображений.*

Savitskaya I.V., Semenova A.P. Autonomous navigation of a mobile robot. *The paper presents a study and practical implementation of computer vision algorithms for autonomous track running by a robot within the framework of the first Russian Cup in sports programming in the discipline "robotics programming". The main attention is paid to the development of an algorithm for navigating a wheeled robot along a complex track with various types of surfaces (grass, ice, earth) and geometric features (S-shaped sections, "snakes").*

Key words*: robotics, competitions, OpenCV, computer vision, image processing.*

Статья поступила в редакцию 18.05.2025
Рекомендована к публикации профессором Павлышом В. Н.