СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 4](#_Toc41554441)

[1. СИСТЕМНЫЙ АНАЛИЗ И ПОСТАНОВКА ЗАДАЧИ](#_Toc41554442) 6

[2. ПРОЕКТИРОВАНИЕ СИСТЕМЫ](#_Toc41554443) 7

[2.1 Проектирование архитектуры ПО](#_Toc41554447) 7

[2.2 Структура программного, аппаратного и информационного обеспечения системы](#_Toc41554447) 13

[2.3 Макетирование пользовательского интерфейса](#_Toc41554447) 15

[3. РЕАЛИЗАЦИЯ СИСТЕМЫ](#_Toc41554446) 15

[4. ТЕСТИРОВАНИЕ](#_Toc41554449) 17

[4.1 Описание входных и выходных данных](#_Toc41554450) 17

4.2 Результаты тестирования…………………………………………………………..18

[ЗАКЛЮЧЕНИЕ](#_Toc41554452) 19

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ](#_Toc41554453) 20

ПРИЛОЖЕНИЕ А ТЕКСТ ПРОГРАММЫ

ПРИЛОЖЕНИЕ Б ГРАФИЧЕСКИЙ МАТЕРИАЛ

# Введение

Термин «искусственный интеллект» появился в 1956 году, но настоящей популярности технология ИИ достигла лишь сегодня на фоне увеличения объемов данных, усовершенствования алгоритмов, оптимизации вычислительных мощностей и средств хранения данных.

Первые исследования в области ИИ, стартовавшие в 50-х годах прошлого века, были направлены на решение проблем и разработку систем символьных вычислений. В 60-х годах это направление привлекло интерес Министерства обороны США: американские военные начали обучать компьютеры имитировать мыслительную деятельность человека. Например, Управление перспективных исследовательских проектов Министерства обороны США (DARPA) выполнило в 70-х годах ряд проектов по созданию виртуальных уличных карт. И специалистам DARPA удалось создать интеллектуальных личных помощников в 2003 году, задолго до того, как появились Siri, Alexa и Cortana.

Эти работы стали основой для принципов автоматизации и формальной логики рассуждений, которые используются в современных компьютерах, в частности, в системах для поддержки принятия решений и умных поисковых системах, призванных дополнять и приумножать возможности человека.

### **В чем заключается важность искусственного интеллекта?**

1. **ИИ позволяет автоматизировать повторяющиеся процессы обучения и поиска за счет использования данных.**

Цель ИИ — не автоматизация ручного труда, а надежное и непрерывное выполнение многочисленных крупномасштабных компьютеризированных задач. Такая автоматизация требует участия человека для первоначальной настройки системы и правильной постановки вопросов.

1. **ИИ делает существующие продукты интеллектуальными.**

Как правило, технология ИИ не реализуется как отдельное приложение. Функционал ИИ интегрируется в имеющиеся продукты, позволяя усовершенствовать их, точно так же, как технология Siri была добавлена в устройства Apple нового поколения.

1. **ИИ адаптируется благодаря алгоритмам прогрессивного обучения**.

ИИ обнаруживает в данных структуры и закономерности, которые позволяют алгоритму освоить определенный навык. Таким образом, по тому же принципу, по которому алгоритм осваивает игру в шахматы, он может научиться предлагать подходящие продукты онлайн.

1. **Глубинные нейросети позволяют ИИ достичь беспрецедентного уровня точности.**

К примеру, работа с Alexa, поисковой системой Google Search и сервисом Google Photos осуществляется на базе глубокого обучения, и чем чаще мы используем эти инструменты, тем эффективнее они становятся. В области здравоохранения диагностика раковых опухолей на снимках МРТ с помощью технологий ИИ (глубокое обучение, классификация изображений, распознавание объектов) по точности не уступает заключениям высококвалифицированных рентгенологов.

## **Где используется искусственный интеллект?**

Функционал ИИ широко востребован во всех отраслях.

Технологии ИИ могут применяться в персонализированной медицине и при расшифровке рентгеновских снимков. Персональные медицинские помощники могут напоминать пользователям, что нужно принять лекарство, выполнить физические упражнения или перейти на более здоровый режим питания.

В розничных продажах. ИИ помогает совершать покупки онлайн с индивидуально подобранными рекомендациями, а также дает возможность продавцам обсуждать покупки с клиентами. Кроме того, технологии ИИ могут оптимизировать процессы управления товарными запасами и размещения товара.

**А так–же ИИ могут использоваться в спорте.** Тренеры получают отчеты со снимками с камер и показателями датчиков о том, как лучше организовать игру, в том числе как оптимизировать расстановку игроков и стратегию.

# 1. СИСТЕМНЫЙ АНАЛИЗ И ПОСТАНОВКА ЗАДАЧИ

Трекинг движущихся объектов в видеопотоке является одной из важнейших проблем в системах компьютерного зрения. Трекинг нашел свое применение во многих областях, таких как видеонаблюдение, искусственный интеллект, робототехника, медицинские и биологические приложения и т. д. В последнее время появилось множество надежных алгоритмов для трекинга одного объекта, однако при наличии нескольких объектов, необходимо отслеживать соответствие объектов текущего фрейма объектам предыдущих фреймов. Отслеживание нескольких объектов является сложной задачей, особенно в случае их сходства.

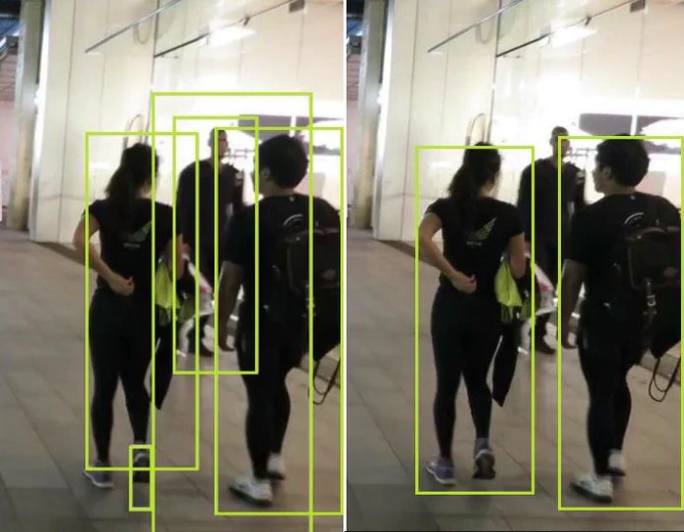
Отслеживание объектов на абстрактном уровне может выполняться с помощью любого из двух существующих в нем подходов. Один называется слежением за одним объектом (SOT), а другой - слежением за несколькими объектами (MOT). Как следует из самого названия, отслеживание одного объекта - это когда в видео или наборе кадров отслеживается только один конкретный объект. Точно так же отслеживание нескольких объектов – это когда различные объекты отслеживаются одновременно в одном видео или наборе кадров. Последний по понятным причинам намного сложнее первого. MOT представляет собой основную трудность при взаимодействии нескольких отслеживаемых объектов друг с другом. Следовательно, модели для SOT не могут быть напрямую применены к MOT и приводят к низкой точности.

В последнее время слежение за объектами широко используется в наблюдении, безопасности, мониторинге трафика, обнаружении аномалий, зрении роботов и визуальном отслеживании. Визуальное отслеживание – это захватывающее приложение, в котором будущее положение объекта в видео оценивается без ввода остальной части видео в алгоритм. Это можно рассматривать как взгляд в будущее.

**Трудности с отслеживанием объектов**

Несмотря на то, что это полезный метод, не каждый рынок и / или процесс может позволить себе выполнять отслеживание объектов из-за того, что одним из важнейших препятствий при обучении модели отслеживания объектов является скорость обучения и отслеживания. Ожидаемые алгоритмы слежения и необходимые для обнаружения и локализации объекта на видео за доли секунды и с высокой точностью. Эта скорость обнаружения может быть существенно нарушена непроизвольно из-за множества отвлекающих фонов в любом сценарии.

Другая проблема с отслеживанием объектов, которая также является важной проблемой при обнаружении и распознавании объектов, – это окклюзия. Окклюзия – это когда несколько объектов подходят так близко друг к другу, что кажется, что они слились. Это может сбить компьютер с толку и заставить его думать, что объединенный объект представляет собой единый объект, или просто ошибочно идентифицировать этот объект.



Пример окклюзии

Помимо этого, некоторые проблемы создают трудности при отслеживании объектов, такие как переключение идентичности после пересечения, размытие движения, изменение точки обзора, загромождение похожих объектов на заднем плане, низкое разрешение и изменение освещенности.

**Постановка задачи:**

Задачей данного курсового проекта является разработка оконного приложения, позволяющее производить трекинг движущегося объекта в видеопотоке.

В данном курсовом проекте будет реализован трекинг потока машин в видеопотоке, а именно на автомагистрали. А так подсчёт движущихся транспортных средств и уведомление о выходе объекта из кадра.

# 2. ПРОЕКТИРОВАНИЕ СИСТЕМЫ

### **2.1 Проектирование архитектуры ПО**

Для отслеживания объекта существуют множество инструментов, я остановлюсь на OpenCV. OpenCV имеет несколько встроенных алгоритмов, разработанных исключительно с целью отслеживания объектов. Мы можем использовать эти предварительно обученные алгоритмы для отслеживания объекта по нашему собственному выбору. У каждого алгоритма есть свои плюсы и минусы.

OpenCV на данный момент одна из самых популярных библиотек алгоритмов компьютерного зрения, обработки изображений и численных алгоритмов общего назначения с открытым кодом. Реализована на C/C++, также разрабатывается для Python, Java, Ruby, Matlab, Lua и других языков.

Алгоритм работы программного продукта будет следующим:

**Рассмотрим пример алгоритма на синем шарике.**

**Шаг 1**: Для начала мы получаем на вход фото/видеоизображение:

**Шаг 2**: Накладываем на наше видеоизображение маску с помощью функции из

библиотеки OpenCV, функция под названием createBackgroundSubtractorMOG().

Маска сделает нахождение контуров вокруг обнаруженного объекта более эффективным. Белая область – это интересующая область в указанном выше цветовом диапазоне. Я могу использовать различные фильтры изображений, чтобы улучшить маску изображения. Однако применение фильтров для получения идеальной маски может быть дорогостоящим с точки зрения вычислительной мощности.

Рисунок 2.1 – Первоначальный видеопоток без изменений.

Рисунок 2.2 – Результат функции функция createBackgroundSubtractorMOG().

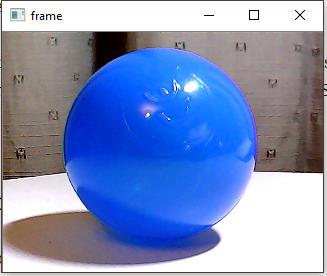


Рисунок 2.1

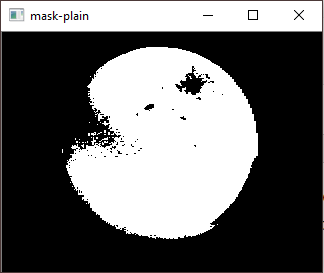


Рисунок 2.2

**Шаг 3:** Находим контур нашего объекта.

Теперь, когда имеется маска изображения, я могу продолжить поиск контуров. Все белые пятна на маске будут закрашены. Список массивов найденных контуров будет в переменной контура. Если качество маски зашумленное, будет много контуров - много мелких контуров. Чтобы получить полезное обнаружение объекта, мне нужно удалить ненужные контуры. Поэтому мы находим самый большой контур среди всех контуров нашего объекта.

Рисунок 2.3 – Объект с самым большим ограничивающим контуром

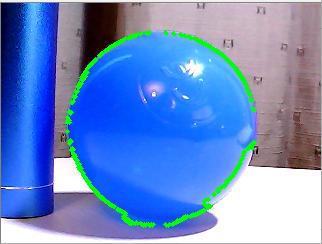


Рисунок 2.3

**Шаг 4:** Создаём ограничивающий прямоугольник.

Далее мы будем создавать ограничивающий прямоугольник исходя из нашего ограничивающего контура. Для этого мы будем использоваться функцию из библиотеки CV под названием cv2.rectangle.

Рисунок 2.4 – Объект с самым большим ограничивающим контуром

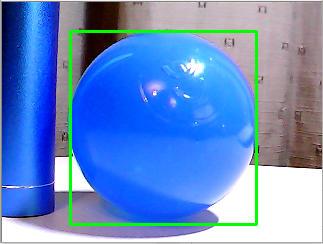


Рисунок 2.4

**Шаг 5**: Находим центр нашего объекта.

Конкретно в курсовом проекте потребуется центр нашего объекта, а именно красная точка.

А так же мы создадим горизонтальную линию на нашем фрейме в следующем шаге.

Для нахождения центра мы будем использовать функцию cv2.circle.

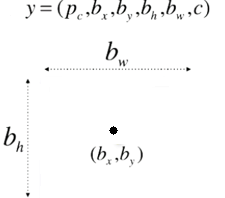


Рисунок 2.5 – Найденный центр объекта

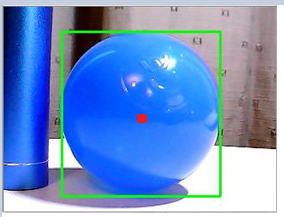


Рисунок 2.5

**Шаг 6:** Накладываем горизонтальную линию на видеопоток.

Красная точка и горизонтальная линия потребуется для подсчёта транспортных средств проезжающих на автомагистрали при пересечении красной точки и горизонтальной линии. Всё это используется для минимизации непредвиденных ситуациях.

Для построения линии мы будем использовать функцию cv2.line.

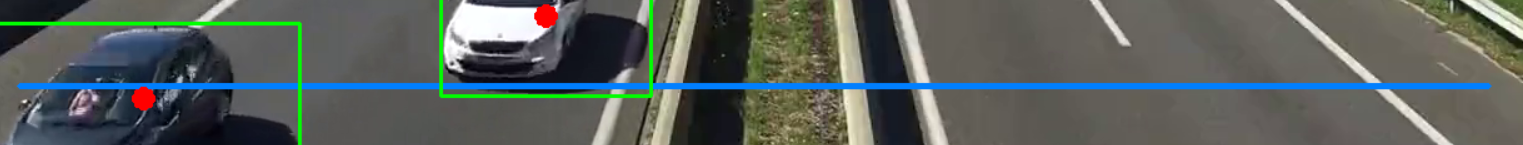
 Рисунок 2.6 – конкретная ситуация из программы курсового проекта.

Рисунок 2.6

### **2.2 Структура программного, аппаратного, информационного обеспечения системы**

Для решения поставленной задачи в данном проекте будет использован язык Python, так как он кроссплатформенный – хорошо работает на всех популярных ОС и переносится на них без каких-либо доработок, а также он обладает такими качествами как простота, лаконичность и выразительность, позволяющие с минимальными затратами времени и сил разрабатывать сложные алгоритмы.

Для трекинга за объектом будет использована библиотека OpenCV.

Что касается среды разработки, то среди большого количества инструментального ПО была выбрана среда разработки JetBrains PyCharm. PyCharm делает разработку максимально продуктивной благодаря функциям автодополнения и анализа кода, мгновенной подсветке ошибок и быстрым исправлениям. а также работать с другими библиотеками для научных вычислений и анализа данных, а именно NumPy. А так же PyCharm – настраиваемая кросс-платформенная IDE, а значит PyCharm можно установить на нескольких компьютерах в операционных системах Windows, Mac OS или Linux.

## Также для реализации поставленной задачи будут использованы следующие библиотеки:

OpenCV — это open source библиотека компьютерного зрения, которая предназначена для анализа, классификации и обработки изображений. Широко используется в таких языках как C, C++, Python и Java.

NumPy — это библиотека языка Python, добавляющая поддержку больших многомерных массивов и матриц, вместе с большой библиотекой высокоуровневых (и очень быстрых) математических функций для операций с этими массивами.

Imutils — этот пакет включает ряд удобных функций OpenCV +, которые выполняют базовые задачи, такие как перевод, поворот, изменение размера и скелетонизация.

## Для обучения и тестирования также будет применен Jupyter-ноутбук, позволяющий хранить вместе код, изображения, комментарии, формулы и графики. С его помощью будет сформировано наглядное представление результатов проведенной работы.

Что касается аппаратного обеспечения, то курсовой будет выполнен на ПК, который имеет процессор Intel Core i5-8300H CPU с частотой 2.3 ГГц, ОЗУ 16 ГБ, а также тип операционной системы: 64-разрядная OC Windows10.

### **2.3 Макетирование пользовательского интерфейса**

Для разработки макетов к курсовому проекту, была выбрана программа под названием Figma.

Figma — онлайн-сервис для разработки интерфейсов и прототипирования с возможностью организации совместной работы в режиме реального времени. Сервис доступен по подписке, предусмотрен бесплатный тарифный план для одного пользователя. Имеются офлайн-версии для Windows, macOS.

Конкретно в моём курсовом проекте будет использована офлайн версия Figma.

Рисунок 2.7 – макет разработанной программы

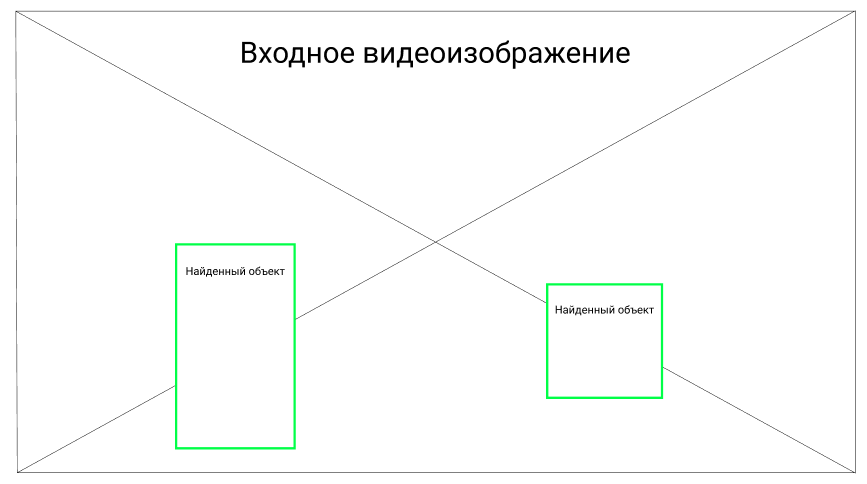
****

Рисунок 2.7

# 3. РеАЛИЗАЦИЯ СИСТЕМЫ

Для решения задачи обнаружения объектов в данном проекте будем использовать язык Python, так как он кроссплатформенный – хорошо работает на всех популярных ОС и переносится на них без каких-либо доработок.

Среди большого количества инструментального ПО была выбрана среда разработки JetBrains PyCharm она удобна в работе и имеет бесплатную версию с функционалом, которого вполне достаточно. Кроме интерпретатора python был использован OpenCV.

OpenCV — это open source библиотека компьютерного зрения, которая предназначена для анализа, классификации и обработки изображений. Широко используется в таких языках как C, C++, Python и Java.

|  |  |
| --- | --- |
| **Название метода** | **Описание метода** |
| p-centro | Указывает на середину отслежеваемого объекта. |
| cv2.circle | Накладывает в видеопотоке круг в центре остлежеваемого объекта |
| subtracao = cv2.bgsegm.  create  BackgroundSubtractorMOG() | Создаём фоновый вычитатель. Модуль определяет по видеовходу, какие объекты находятся на переднем плане, а какие-на заднем плане |
| cv2.line | Метод накладывает линию на видеопоток, для того чтобы: для точного вычисления количества проезжающих т/с по автомагистрали. |
| detec | Вычисляет количество машин, которые пересекают введенную нами линию |
| cv2.putText | Выводит на экран количество проезжающих т/с |
| cap.read() | Используется для чтения нашего видеопотока. |
| t = float(1 / delay)  sleep(t) | Используется для небольшого замедлевания видео, для более точных вычислений. |
| cv2.cvtColor(frame1, cv2.COLOR\_BGR2GRAY) | Преобразуем цветовое пространство в серый цвет |
| blur = cv2.GaussianBlur(grey, (3, 3), 5) | Используется для сглаживания и фильтрации видеопотока. А именно используется для улучшения качества видеопотока. |
| cv2.dilate | Основная функция метода является постоянное увеличение границ областей пикселей переднего плана (например, обычно белых пикселей). Таким образом, области пикселей переднего плана увеличиваются в размере, а дыры в этих областях становятся меньше. Что приводит к более точной и быстрой детекции. |
| cv2.findContours | Данный метод используется для нахождение контура остлежеваемого объекта. |
| Car\_detected | Данный метод обнаруживает движущиеся т/с |

# ТЕСТИРОВАНИЕ

**4.1 Описание входных и выходных данных**

В данном программном обеспечении разработанного специально для данного

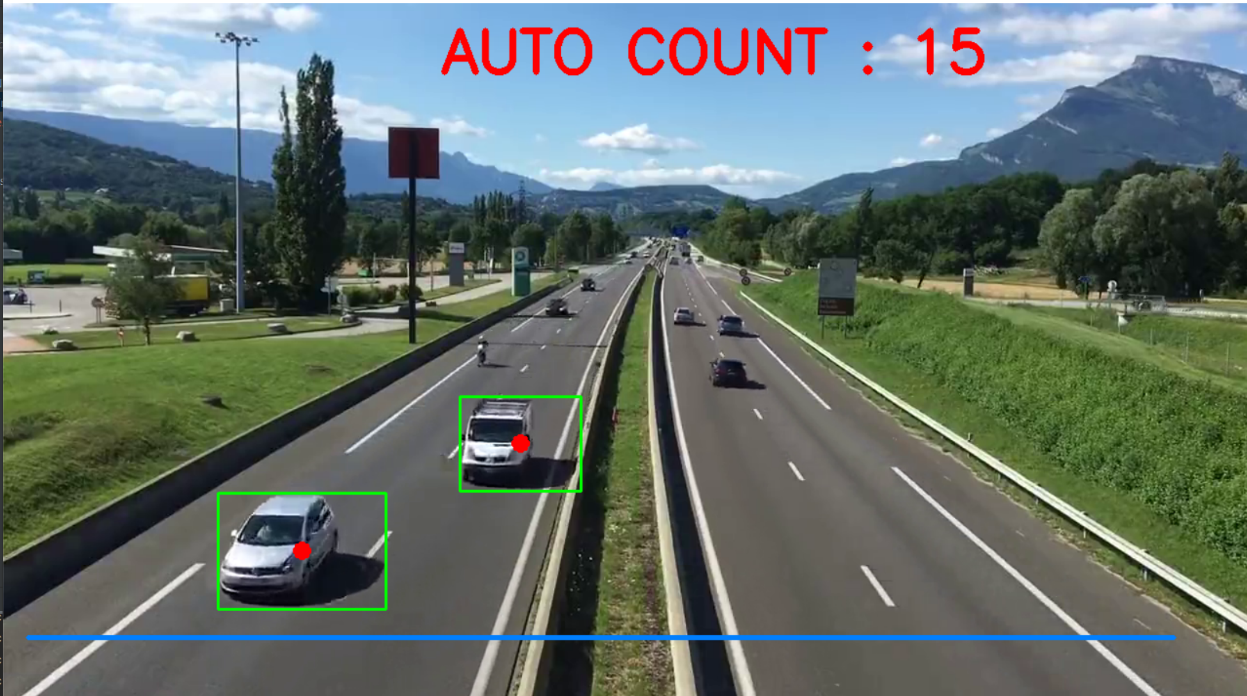
курсового проекта, в качестве, **входных данных**, был использован видеопоток с камеры видеонаблюдения расположенного прямо на автомагистралью.

В качестве **выходных данных,** мы получаем всё тот же видеопоток. После чего данное программное обеспечение производит трекинг движущихся объектов и производит подсчёт количества проезжающих автотранспортных средств.

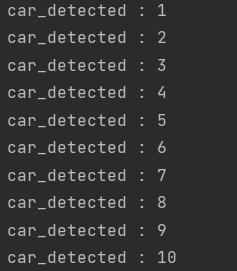
**4.2 Результаты тестированя**

Среда тестирования – ПК Hp Pavilion, процессор Intel Core i5-8300H с частотой 2.4 ГГц, ОЗУ 16 ГБ, тип системы: 64-разрядная OC Windows 10.

Обучение и тестирование производилось на базе IDE PyCharm.

** Тест:** «Тестирование программного обеспечение на выбранном видеопоток»

Видим, что данное программное обеспечение, производит трекинг движущихся объектов, которые помечаются на видеопотоке, как зелёный квадрат, приблежающихся к горизонтальной линии.

При пересечении cо статической линией и центра объекта, где центр у нас красная точка. Происходит подсчёт транспортного средства.

# Заключение

В результате курсового проекта было разработанно программное обеспеченеие которое позволяет производить трекинг движущихся объектов в видеопотоке. А именно отслежевание транспортных средств на автомагистрали и подсчёта количества.

Подводя итоги можно сказать: трекинг нашел свое применение во многих областях, таких как видеонаблюдение, искусственный интеллект, робототехника, медицинские и биологические приложения и т. д. В последнее время появилось множество надежных алгоритмов для трекинга одного объекта, однако при наличии нескольких объектов, необходимо отслеживать соответствие объектов текущего фрейма объектам предыдущих фреймов. Отслеживание нескольких объектов является сложной задачей, особенно в случае их сходства. А так же нужно не забывать про Окклюзию.

Окклюзия – это когда несколько объектов подходят так близко друг к другу, что кажется, что они слились. Это может сбить компьютер с толку и заставить его думать, что объединенный объект представляет собой единый объект, или просто ошибочно идентифицировать этот объект.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. ГОСТ 2.105-95. Единая система конструкторской документации (ЕСКД). Общие требования к текстовым документам.
2. ГОСТ 19.504-79. Единая система программной документации ЕСПД. Руководство программиста. Требования к содержанию и оформлению.
3. ГОСТ 19.701-90. ЕСПД. Схемы алгоритмов, программ, данных и систем. Обозначения условные и правила выполнения.
4. ГОСТ 19.005-85. ЕСПД.

Р-схемы алгоритмов и программ. Обозначения условные графические и правила выполнения.

1. ГОСТ 19.101-77. ЕСПД. Виды программ и программных документов.
2. ГОСТ 19.102-77. ЕСПД. Стадии разработки.
3. ГОСТ 19.103-77. ЕСПД. Обозначения программ и программных документов.
4. ГОСТ 19.401-78. ЕСПД. Текст программы. Требования к содержанию и оформлению.
5. ГОСТ 19.402-78. ЕСПД. Описание программы.
6. ГОСТ 7.1-2003. Система стандартов по информации, библиотечному и издательскому делу. Библиографическая запись. Библиографическое описание. Общие требования и правила составления.